UNIVERSIDADE DE SANTIAGO DE COMPOSTELA
DEPARTAMENTO DE ELETRÔNICA E COMPUTAÇÃO

TESE DE DOUTORADO

REDES ART COM CATEGORIAS INTERNAS DE GEOMETRIA IRREGULAR

Apresentada por:

Dinani Gomes Amorim

Dirigida por:

Dr. Manuel Fernández Delgado
Dr. Senén Barro Ameneiro

Santiago de Compostela, Dezembro 2006

MANUEL FERNÁNDEZ DELGADO
Professor Contratado Doutor
Departamento de Eletrônica e Computação
Universidade de Santiago de Compostela

SENÉN BARRO AMENEIRO
Catedrático de Universidade
Departamento de Eletrônica e Computação
Universidade de Santiago de Compostela

FAZEM CONSTAR

Que a memória entitulada "**Redes ART com categorias internas de geometria irregular**" foi realizada por Da. **Dinani Gomes Amorim** sob nossa direção no Departamento de Eletrônica e Computação da Universidade de Santiago de Compostela, e constitui a Tese que apresenta para obter o grau de Doutora em Física (Área de Ciência da Computação e Inteligência Artificial).

Santiago de Compostela, 5 de Dezembro de 2006.

Assinado: Manuel Fernández Delgado
Co-diretor da Tese

Assinado: Senén Barro Ameneiro
Co-diretor da Tese
Mgfico. e Exmo. Sr. Reitor da
Universidade de Santiago de Compostela

Assinado: Dinani Gomes Amorim
Doutoranda

A meu Pai Celestial e Santiago de Compostela, com Eles tudo é possível.

A meus filhos, Natália e Tiago, e meu marido, Ricardo,
por serem a razão do meu viver,

A meus pais terrenos, José (in memorian) e Neuza,
por terem me encaminhado na busca do saber.

Às minhas irmãs, Denize (in memorian), Dayse e Jayse,
por serem minhas melhores amigas.

Agradecimentos

Meus sinceros agradecimentos aos meus co-diretores D. Manuel Delgado e D. Senén Barro, por terem me orientado na realização desta tese.

Ao Senén pela confiança e apoio que dedicou a mim desde o primeiro momento que nos conhecemos, e pela amizade desenvolvida por todos esses anos, juntamente com sua esposa e seus filhos, que de forma indireta, também contribuiram para tornarem meus dias em Santiago mais agradáveis e felizes.

Ao Manuel Delgado, por toda a sua dedicação à nossa investigação e ajuda constante, pela confiança que sempre demonstrou ter em mim, e pelo contínuo bom-humor inteligente e pontual. Ao apoio direto e indireto que obtive de Eva Cernadas, com suas observações e sua disponibilidade constante.

Ao prof. José Antonio Vila, pela atenção que me dedicou, resolvendo junto comigo questões burocráticas e urgentes, ou bem me dando preciosos conselhos e orientações, muitas vezes, através de momentos de conversas informais.

A Victor Brea, pelos memoráveis, e não poucos, momentos de *relax* que proporcionou a mim e à minha família, sempre programando algo para nos alegrar e amar cada vez mais a Galícia, com seus maravilhosos passeios a Finisterre e à toda costa galega. Por ser simples, autêntico, sensível, divertido, e por ter se tornado nosso melhor amigo, praticamente o irmão que nunca tive.

A todos os companheiros do Departamento, pelo ambiente agradável, descontraído que pude desfrutar todos estes anos e por terem me auxiliado tantas vezes.

Aos companheiros do grupo de Visão Artificial, especialmente à Natália Fernández, a David García, à Raquel Dosil, a Leboran, e Pili, pela amizade e carinho que sempre recebi de todos eles e pelos momentos divertidos e descontraídos que me proporcionaram.

À minha família, em especial aos meus filhos e ao meu marido – Natália, Tiago Tito e Ricardo – por terem me acompanhado e apoiado nesta "louca aventura da busca do saber", por terem me suportado nos momentos difíceis, e tornado meus dias mais felizes, coloridos, nunca monótonos e mais fáceis de serem vividos.

À minha mãe, Neuza, pelo apoio incondicional que sempre me deu, e às minhas queridas irmãs, Dayse e Jayse, aos meus sobrinhos, Pedro e Annyela, e meus amigos

Sidney e André por serem meus fãs incondicionais.

À Gícia, Juracy, Sizenice e Rita, pelo apoio e pelas orações, que nos fortaleceram a prosseguir em nosso trabalho.

Aos amigos Marquinhos, Vilani, Ednalva, Luzia Pia e Nádia Carvalho, que sempre me ajudaram a resolver meus "probleminhas" brasileiros tanto em Petrolina como em Juazeiro.

A Benjamin, Marta, Elis, Ellen, Vijay, Dora, Alcides, Nathaniel, Irina e aos companheiros do EC, pelos momentos agradáveis, divertidos, e relaxados que me proporcionaram, cada um com seu "toque especial", mas enfim, todos maravilhosos.

Aos do IKOS, especialmente, a Juana, Maika, Belém, Mônica, Ramón e Eduardo, pela paciência, carinho e dedicação que sempre demonstraram com meus filhos e por todos os momentos de lazer e diversão que ofereceram a eles, integrando-os, dessa forma, ainda mais na Galícia.

Por último, e com um carinho especial, à minha amiga-irmã e segunda mãe de Tito e Natália – Luz Divina– que iluminou minha vida e da minha família, com seu apoio, amizade, dedicação e enorme carinho, juntamente com suas filhas Eva, Rocio, Natália e Tamara.

À Universidade do Estado da Bahia (UNEB) e à Xunta da Galícia, pelo suporte ecônomico fornecido para a realização deste trabalho.

Sumário

Lista de Tabelas

Lista de Figuras

Parte I

Tese de Doutorado

Prefácio

O trabalho de investigação descrito na presente tese de doutorado se enquadra no âmbito das redes neurais ART (*Adaptive Resonance Theory*) [24, 25], e emerge como continuação de trabalhos prévios realizados no nosso grupo de investigação, na segunda metade dos anos 90. Nestes trabalhos, empregaram-se as redes ART para a aprendizagem não supervisionada (*clustering*) de morfologias de batimentos cardíacos sobre os sinais multi-canais de ECG [8], em tempo real, através do modelo MART [53, 51, 52]. Uma das características inovadoras deste modelo é o uso das categorias internas com distintos tamanhos máximos (dados pelo parâmetro de vigilância, que nesta proposta, é distinto para cada categoria). Outro aspecto destacável é o caráter adaptativo deste tamanho máximo da categoria, incrementando-se em presença de ruído, quando aumenta a variabilidade nos padrões de entrada, e reduzindo-se, em caso contrário. Esta adaptabilidade, nas vigilâncias alcança o objetivo de adaptar o volume ocupado por cada categoria interna às características dos padrões de entrada. Entretanto, esta aproximação é relativamente limitada, porque ainda que o tamanho da categoria evoluciona no tempo, a sua forma geométrica permanece invariável, o qual limita a capacidade de adaptação das redes aos dados de entrada.

O presente trabalho pretende ampliar a capacidade de adaptação das redes ART existentes, que possuem as categorias internas com geometrias predefinidas (hiper-retangulares ou hiper-elipsóides). Com este objetivo, esta tese de doutorado propõe novos modelos ART com a aprendizagem supervisionada, orientados aos problemas de classificação, e baseados em categorias internas com geometrias genéricas ou irregulares. Cada categoria adapta a sua forma geométrica em função dos padrões de entrada, ao longo da etapa de treinamento, cobrindo o volume do espaço de entrada definido pelos padrões com a sua mesma predição de saída. Poderemos nos referir, diferentemente das redes ART clássicas, de categorias com geometria

genérica. Como veremos ao longo do texto, esta característica introduz mudanças fundamentais em comparação com as redes ART existentes. Por exemplo, cada categoria interna não está definida por um único vetor representante, como ocorre nas redes ART, senão por uma série de padrões de treinamento selecionados entre os que foram codificados pela categoria. Por outro lado, dado que cada categoria interna pode adotar a forma geométrica dos agrupamentos presentes nos dados, é possível construir categorias disjuntas, quer dizer, não sobrepostas. Este fato tem uma conseqüência importante sobre a operação da rede, uma vez que permite que as categorias internas limitem entre si a sua expansão, suprimindo a necessidade de tamanhos máximos de categorias, típica das redes ART. Finalmente, uma conseqüência indireta do uso de categorias com geometria genérica é a supressão do parâmetro de vigilância, o qual permite uma operação livre de parâmetros a serem ajustados.

A tese de doutorado está organizada em quatro capítulos e vários apêndices, que descrevem as técnicas adicionais, não desenvolvidas nesta investigação mas utilizadas no trabalho experimental. O capítulo 1 é uma contextualização do âmbito das redes ART, descrevendo os seus princípios de funcionamento e características básicas. Também realizamos uma breve análise dos principais modelos ART desenvolvidos ao longo dos anos, fazendo uma divisão entre os modelos iniciais, auto-organizados, e os modelos mais recentes, inspirados nos anteriores, mas com a aprendizagem supervisionada. Também descrevemos, com especial detalhe, aqueles modelos que usaremos na comparação com as nossas propostas. O capítulo conclui com uma discussão dos aspectos geométricos destas redes, e da possibilidade de usar geometrias irregulares para as suas categorias internas, enlaçando diretamente com os modelos que propomos nos capítulos seguintes.

A nossa primeira aproximação a este objetivo, *Simplex* ARTMAP (SAM), é proposta no capítulo 2, e baseia-se em categorias com forma de simplex, de modo que as suas geometrias não são completamente genéricas. As suas principais características, a função de escolha que define as categorias internas, o processo de aprendizagem das categorias simplex e a sobreposição entre as predições, constituem aspectos fundamentais desta proposta. A simulação experimental desta rede sobre os problemas artificiais de classificação em duas dimensões, junto com a discussão dos resultados, com os seus aspectos positivos e negativos, leva à proposta do modelo *PolyTope* ARTMAP (PTAM) no capítulo 3, no qual as categorias têm uma forma geométrica de politopo, definido por um número variável de padrões de treinamento seleciona-

dos. As características próprias do treinamento e processamento de PTAM (função de escolha, aprendizagem, sobreposição de categorias, vigilância, ...), junto com a análise dos resultados sobre uma completa coleção de problemas de classificação, completam o capítulo. O capítulo 4 complementa o anterior, analisando o interesse da sobreposição entre as categorias de forma geométrica politopo e propondo a rede *Overlapping PolyTope* ARTMAP (OPTAM). Trata-se de uma alternativa ao PTAM, que permite a sobreposição entre as categorias, e portanto utiliza o parâmetro de vigilância como as redes ART existentes. Neste capítulo é apresentado uma análise comparativa entre os resultados obtidos por ambas alternativas, discutindo-se as suas contribuições positivas e negativas. Finalmente, são discutidas globalmente as principais contribuições deste trabalho de investigação, e as principais linhas de avanço futuras.

Capítulo 1

Introdução

1.1 A Teoria da Ressonância Adaptativa (ART)

Podemos considerar que os antecendentes do modelo ART remontam desde Von der Malsburg, 1973, que foi um dos primeiros a apresentar uma rede neural artificial de mapa de características auto-organizável para a modelagem do córtex visual das vértebras superiores. A investigação de Von der Malsburg influenciou os trabalhos posteriores de Teuvo Kohonen, que deram lugar aos Mapas Auto-Organizativos (*Self-Organizing Maps*, SOM), mas também serviram como base às aportações de Gail Carpenter e Steven Grossberg, criadores da Teoria da Ressonância Adaptativa. Estes trabalhos se orientaram em direcionar o *dilema da plasticidade-estabilidade* [75, 76], que têm como suas principais questões: *i)* como um sistema de aprendizagem pode manter sua plasticidade (ser adaptável) em resposta às informações novas, ainda não conhecidas, e manter sua estabilidade diante da apresentação de informação irrelevante?; *ii)* como pode um sistema preservar seu conhecimento adquirido e ao mesmo tempo ser suficientemente flexível para armazenar uma nova informação? e *iii)* como o sistema pode decidir quando alternar do estado de estabilidade à plasticidade e vice-versa? A resposta a este dilema foi a Teoria da Ressonância Adaptativa (*Adaptive Resonance Theory*, ART) [79, 78, 23, 80]. Todas as redes neurais baseadas em ART compartilham um conjunto de propriedades que permitem o processamento *"on-line"*, portanto, proporcionam uma solução ao *dilema de estabilidade-plasticidade*. As redes ART permitem a aprendizagem incremental [54] em ambientes que variam no tempo (não-estacionários), aprendizagem estável rápida, escalas de generalização múltipla e convergência rápida com um

número relativamente pequeno de padrões de treinamento. Estas redes modelam a predição, busca, aprendizagem, e reconhecimento em tempo-real, funcionando como modelos de processamento de informação cognitiva [20, 27, 77, 81, 82, 119], porém também como sistemas neurais aplicados no âmbito das engenharias [1, 69, 72, 106].

A denominação ART advém do modo como os processos de aprendizagem e de recuperação de informações interagem dinamicamente no modelo original ART1 [23]. O conceito de ressonância corresponde a "oscilações" que ocorrem em um sistema físico quando o mesmo é submetido a uma entrada de freqüência específica (freqüência de ressonância), usualmente definida pelas propriedades físicas do sistema (materiais, dimensões, disposição geométrica, etc.). Em uma rede ART, a aprendizagem ocorre quando as informações, na forma de saída das unidades de processamento (neurônios), "oscilam" entre as camadas de unidades da rede, desenvolvendo um estado de ativação de equilíbrio. Essas "oscilações" seriam o equivalente neural à ressonância em um sistema físico.

O propósito de uma rede ART é incorporar um modelo de aprendizagem competitiva dentro de uma estrutura de controle auto-organizável, cujo reconhecimento e aprendizado autonômos continuam estáveis em resposta a uma seqüência arbitrária de padrões de entrada. Esta teoria foi desenvolvida para resolver problemas de instabilidade de sistemas de realimentação, particularmente o dilema estabilidade/plasticidade. A estabilidade está relacionada com a garantia de agrupamento de todos os elementos nas categorias criadas pelo sistema, tendo em vista que os pesos da rede possuem a característica somente de decrescimento, ou seja, à medida que as adaptações dos pesos são realizadas, os novos valores tendem sempre a diminuir até a estabilização. A plasticidade é a característica que a rede possui de aprender um novo padrão, em qualquer tempo de sua operação, sem perder o aprendizado adquirido anteriormente. Na atividade de reconhecimento de padrões, quando uma certa entrada não se assemelha a nenhum dos grupos já existentes, um novo grupo é criado para a referida entrada. Essa característica pode, em alguns sistemas neurais, comprometer a estabilidade da rede, ou seja, quando uma nova aprendizagem ocorre, os padrões já treinados anteriormente são prejudicados podendo a rede perder a capacidade de reconhecer os padrões antigos. Nas redes ART esse problema é resolvido pelo mecanismo de vigilância que administra a inclusão de novas entradas em cada grupo.

Através de muitas variações destes modelos, o centro computacional neural para ambas as análises, científicas e tecnológicas, é a "regra de coincidência ART" [23],

Tabela 1.1: Marcos históricos relacionados com ART na literatura.

Ano	Arquitetura	Referência
1976	GN	Grossberg
1987	**ART1**	**Carpenter**
	ART2	**Carpenter**
1990	ART3	Carpenter
1991	ART 2-A	Carpenter
	ARTMAP	**Carpenter**
	***Fuzzy*-ART**	**Carpenter**
1992	***Fuzzy*-ARTMAP**	**Carpenter**
	AFCL	Newton
	Fuzzy* Min-Max *Neural Network(FMMNN)	**Simpson**
1993	LAPART	Healy
	Simplified Fuzzy-ARTMAP (SFAM)	Kasuba
	Fusion ARTMAP	Asfour
1994	IAFC	Kim
1995	ART-EMAP	Carpenter
	Adaptive Hamming Net (AHN)	Hung
	ARAM	Tan
	PROBART	Marriot
1996	**Gaussian-ARTMAP**	**Williamson**
	FasArt	**Cano-Izquierdo**
	hART-J	Barfai
1997	*Supervised* AHN (SAHN)	Hung
	hART-S	Barfai
	mART	Kim
	ART-LD	Zhou-J.
	FasBack	Cano-Izquierdo
	dART, dARTMAP	**carpenter**
	Cascade-ARTMAP	Tan
	PNN-ART	Lim
1998	ARTMAP-IC	Carpenter
	LAPART2	Healy
	Multi-channel ART (MART)	Fernández-Delgado
	Boosted ARTMAP	Verzi
1999	*Fuzzy* (*Supervised*) AHN	Hung
	ARTEX	Grossberg
	FA *Variant*	Georgiopoulos
	oFAM	Dagher
	dFasArt	Parrado-Hernández
2000	MicroARTMAP (μARTMAP)	Gómez-Sánchez
	HART & HARTMAP	Anagnostopoulos
	FANNC	Zhou-Z.
2001	***Ellipsoid* ART (EA) & *Ellipsoid* ARTMAP (EAM)**	**Anagnostopoulos**
	Safe μARTMAP	Gómez-Sánchez
2002	***semi-supervised Ellipsoid* ARTMAP (ssEAM)**	**Anagnostopoulos**
	ART-C	He
2003	AFC	Sapozhnikova
	Default ARTMAP	Carpenter
2004	***Simplex* ARTMAP (SAM)**	**Gomes**
	Analog-ART1	Rajasekaran
	ARTSTREAM	Grossberg
2005	GreyART	Yeh
	***PolyTope* ARTMAP (PTAM)**	**Gomes**
	***Overlapping PolyTope* ARTMAP (OPTAM)**	**Gomes**

que representa a interação entre a expectativa aprendida descendente (*top-down*) e a entrada sensorial ascendente (*bottom-up*). Esta interação gera um ponto de atenção, o qual, sucessivamente, determina a natureza da memória armazenada. A aprendizagem nestas redes é do tipo competitivo, e na sua maioria, tanto supervisionadas como auto-organizadas, utilizam as representações de código *WTA* (*Winner Take All*), mesmo que algumas das mais recentes, como *Distributed* ARTMAP [36] e *Distributed* FasArt [121], incorporam representações de código e aprendizagem distribuída.

Ao longo dos anos foram propostos distintos modelos ART pertencentes aos dois principais paradigmas de aprendizagem neural [84]: supervisionado e não supervisionado. As primeiras redes ART usavam aprendizagem não supervisionada ou auto-organizativa, e entre elas tem que se mencionar ART1 [23, 80] que foi desenvolvida para executar agrupamentos de padrões de valores binários, e ART2 [24] desenvolvida para detectar regularidades nas seqüências randômicas analógicas, emprega uma arquitetura computacionalmente cara que apresenta dificuldades na seleção de parâmetros. Para superar estas dificuldades, *Fuzzy* ART [32, 31] foi desenvolvido como uma generalização de ART1, também capaz de agrupar padrões de valores reais. Em ART3 [26] temos a utilização de sistemas *feedback* não-lineares de controle de processos neurotransmissores locais na rede. Enquanto em ART2-A [30] tem-se uma arquitetura paralela e apropriada para o tratamento de dados multidimensionais com valores reais, em *Adaptive Hamming Net* (AHN) [92] tem-se uma rede funcionalmente equivalente a ART1 que melhora a convergência do algoritmo ART, convertendo o problema de busca a um problema de otimização e; sua versão supervisionada *supervised*-AHN (SAHN) [93] é adequada para fins de classificação, e mais tarde estende esta versão para duas outras mais, *Fuzzy Adaptive System* ART(FAHN) e *Supervised Fuzzy* AHN (SFAHN) [94] acrescentando elementos de lógica difusa e módulos supervisionados.

Enquanto que em *Analog*-ART1 [126] é uma variação de ART1, onde permite a possibilidade de manipular também valores de entrada analógicos e que melhora a capacidade de reconhecimento de ART1 pela adição de um extrator de características baseado em momento, duas redes ART para agrupamento hierárquico foram propostas, *Hierarchical* ART (hART-J) [9] e hART-S [10], baseadas, respectivamente, na união (*joining*) e na divisão (*splitting*) no algoritmo de agrupamento hierárquico. *Modified* ART (mART) [97] é uma rede neural com a capacidade de fusão de agrupamentos.

Nos anos posteriores foram propostos os modelos que usavam aprendizagem supervisionada, ainda que também permitiam a aprendizagem auto-organizativa. Entre estes modelos cabe destacar ARTMAP [29], que atua de fundamento para todas as outras redes ART supervisionadas, *Fuzzy* ARTMAP (FAM) [28] que associa os padrões de E/S com os valores difusos e, a partir dele, várias extensões foram desenvolvidas. Entre eles figura ART-EMAP [37] que difere sensivelmente na sua fase de teste pela introdução de um método *Q-max* na seleção das categorias. FasArt [15] e FasBack [16] são sistemas híbridos que combinam os conceitos da teoria dos conjuntos difusos, e possuem a arquitetura similar à FAM. *Distributed* FasArt (dFasArt) [121] é a versão distribuída de FasArt.

Em *Gaussian* ART/ARTMAP [144] seguem o paradigma ART e utiliza núcleos gaussianos para descrever as categorias internas. Carpenter modificou o mecanismo *Winner-Take-All (WTA)* ("o vencedor leva tudo") e o empregou em ART/ARTMAP, melhorando seu desempenho de classificação através da implementação da aprendizagem distribuída em *Distributed* ART [20] e *Distributed* ARTMAP [36]. Temos ARTMAP-IC [33], que estende ART-EMAP, como outro esquema de memória distribuída, que também faz uso do contador de instâncias para a predição e, está orientado para solucionar certos problemas no âmbito médico.

Em outras extensões, foram propostos melhoramentos para simplificar a arquitetura FAM, como em *Simplified Fuzzy* ARTMAP (SFAM) [96], que é uma modificação de FAM, na qual a redundância foi removida, mas que ainda sofre com os altos números de neurônios categorias em ART_a. *Fuzzy*-ART *Variant* (FA Variant) [58] é uma versão modificada e eficiente da arquitetura FA, com aumento de velocidade de aprendizagem. *Boosted* ARTMAP (BARTMAP) [138] tenta direcionar o problema de proliferação de categorias, e aumenta o desempenho de classificação dos dados ruidosos. Em μARTMAP [66, 67], que é baseada em FAM, é utilizado o critério de informação mútua para reduzir o número de categorias geradas durante o treinamento numa rede FAM. Logo em *Safe* μARTMAP [65] encontramos uma generalização de μARTMAP, que limita o crescimento de uma categoria, em resposta a um padrão único, e dessa forma impede a criação de grandes hiper-retângulos sob estas condições. Em *Hypersphere* ART/ARTMAP (HA/HAM) encontramos redes baseadas nas idéias principais de FA e FAM, diferindo, entre outras inovações, na forma de representar suas categorias internas, por hiper-esferas. Em *Ellipsoid* ART/ARTMAP (EA/EAM) [3], que são generalizações de HA e HAM, respectivamente, a representação das categorias de FAM foi modificada, utilizando a função

de distância euclidiana e moldes, representados por elipsóides num espaço multi-dimensional. A *semi-supervised Ellipsoid* ARTMAP (ssEAM) [6] consiste de uma generalização de EAM e de EA, que permite agrupar numa categoria única, padrões de treinamento não necessariamente da mesma categoria, através de um teste de predição modificado. *Fast Adaptive Neural Network Classifier* (FANNC) [151] está especialmente orientada às tarefas de aprendizagem *on-line* em tempo real.

Além de *Fuzzy* ART/ARTMAP, outras contribuições baseadas na teoria de conjuntos difusos são: *Fuzzy* Min-Max *Neural Network* (FMMNN) [130], que combina a função membro dos pontos min-max do hiper-retângulo, e o algoritmo *Adaptive Fuzzy Leader Clustering* (AFLC) [117], que demonstrou potencial para análise de conjunto de dados complexos. Também se tem que mencionar o algoritmo *Integrated Adaptive Fuzzy Clustering* [99] que utiliza uma estrutura similar à ART1, mas incorpora uma nova regra de aprendizagem e medida de similaridade que elimina alguns problemas estruturais inerentes às outras redes baseadas em *Fuzzy* ART.

Em *Cascade* ARTMAP [133] temos uma rede híbrida com ARTMAP para lidar com a classificação de conhecimento simbólico. ART-LD [150] capacita a rede *Fuzzy* ART com discriminação logística para classificação de padrões. Enquanto que PNN-ART [104] combina as redes ART e probabilísticas (PNN) [131] para estimativa de probabilidade. Adicionalmente, Grossberg introduziu ARTEX [74] para classificação de regiões de imagens com texturas. Em *ordered Fuzzy* ARTMAP (oFAM) [44] um procedimento de ordenamento de padrões de treinamento é combinado com FAM. A rede LAPART [87] e LAPART2 [86] são muito similares em funcionalidade a ARTMAP, tendo a última arquitetura uma melhora na velocidade de convergência, durante a fase de treinamento e não sofre do problema de proliferação de categorias. Enquanto em *Adaptive Ressonance Associative Map* (ARAM) [132] temos uma rede que desempenha tarefas similares a ARTMAP, mas está implementada por uma arquitetura mais simples computacionalmente. A rede PROBART [107] foi proposta como uma solução possível para a potencial super-aprendizagem de FAM. Já o modelo ART-*based Fuzzy Classifier* (AFC) [128] surge também da necessidade de superar o problema de proliferação de categorias e utiliza para isso uma função de escolha assimétrica não-plana.

Algumas variações foram propostas com a propriedade de múltiplos-canais, com o fim de fusão de informações. Nesta linha temos *Fusion* ARTMAP [7] que é uma derivação de FAM para classificação supervisionada de problemas multicanais, tanto do tipo "votação", como do tipo "combinação". Enquanto que *Multi-channel* ART

(MART) [51] baseia-se em FA e utiliza uma aproximação modular para a aprendizagem não supervisada de padrões multicanais.

Em *Default* ARTMAP [122] encontramos uma rede que se comporta igual a FAM durante o treinamento, mas utiliza uma representação de código distribuído durante a fase de teste. ART-C [85] executa agrupamentos de seqüências de padrões *on-line* sujeitas às restrições na representação da categoria. Finalmente, destacamos também um modelo de rede para "análise de cena de auditoria"(em inglês, *auditory scene analysis* - ASA), ARTSTREAM [83], que propõe como o cérebro realiza esta tarefa e a GreyART [147] que incorpora análise relacional *grey* à rede ART2 para agrupamento de dados.

As referências que mencionamos não pretendem ser exaustivas em ART ou em seus campos relacionados. Existem várias outras referências quanto a ART híbridas, extensões, implementações de *hardware* e aplicações de redes baseadas em ART em diversas áreas. Entretanto, o material bibliográfico apresentado acima descreve a pesquisa ativa na área e fornece uma visão do potencial da teoria da ressonância adaptativa. Na tabela 1.1 destacamos essas contribuições selecionadas na área de forma cronológica.

Não aprofundamos em detalhes todas arquiteturas ART aqui mencionadas, apenas as mais relevantes para a nossa investigação. Assim, dividimos em dois grupos, em função do paradigma de aprendizagem empregado por cada rede e também por ordem cronológica (na tabela 1.1 destacamos, em negrito, as redes que descreveremos nas seções a seguir).

1.2 Redes ART não supervisionadas

1.2.1 ART1

Utilizando como base a *Grossberg Network* (GN) [76], foi desenvolvido ART1 [23, 25] para aprendizagem não supervisionada de padrões de entrada binários. Sua topologia básica encontra-se na fig. 1.1. Em termos rigorosos a rede ART1 é caracterizada por um conjunto de equações diferenciais não-lineares, que implementam as propriedades de estabilidade-plasticidade, objetivando a aprendizagem incremental. Entretanto, para efeito da implementação computacional, esse modelo dinâmico de equações diferenciais pode ser resumido a um algoritmo seqüencial

comum, utilizando-se dos valores do regime estacionário das ativações da rede [54].

Arquitetura

A rede é constituída por um **Subsistema Atencional** e um **Subsistema de Orientação**. O Subsistema Atencional é formado por duas camadas de unidades denominadas F_1 e F_2, e pelo **Nó de Controle de Ganhos**, como se encontra na fig. 1.1. O padrão binário de entrada **I** (cujas componentes $I_j \in \{0, 1\}$) se apresentam na camada F_1, são de natureza binária e de dimensão M. Os padrões de ativação desenvolvidos nestas unidades da rede são denominados traços de memória de curta duração (**STM** -*Short-Term Memory*), pois somente existem em decorrência da aplicação de uma entrada às unidades. Enquanto que na camada F_2, com N unidades de processamento (neurônios) associadas às distintas categorias internas (agrupamentos ou *clusters*) aprendidas pela rede, e seus pesos de conexão com F_1, integram os representantes ou valores esperados destas categorias. O fluxo de informações entre F_1 e F_2 é bidirecional, envolvendo pesos sinápticos *bottom-up* (de F_1 para F_2) e *top-down* (de F_2 para F_1), sendo portanto uma rede com topologia *feedback*. Os pesos dessas conexões sinápticas entre as camadas F_1 e F_2 são denominadas de traços de memória de longa duração (**LTM** - *Long-Term Memory*), pois são responsáveis pela codificação das informações pela rede e representam a sua memória, que permanece quando se retira o padrão de entrada em F_1. Por último, o Nó de Controle de Ganhos, que direciona o fluxo de informações entre as camadas, além de possuir um efeito excitatório sobre as unidades de F_1, recebe também um sinal inibitório proveniente de F_2.

O **Subsistema de Orientação** (circuito de *reset*) é caracterizado pelo parâmetro ρ (de vigilância), e controla o processo de "busca por coincidência" de padrões e de aprendizagem da rede, atuando como inibidor das unidades em F_2. Somente no caso em que a atividade em F_1 seja suficientemente elevada, o Subsistema de Orientação se encontra desativado, e as unidades de F_2 podem se ativar.

Funcionamento

Quando a rede é inicializada para o treinamento, os pesos *top-down* das unidades da camada F_2 ainda não participaram de qualquer aprendizagem, e por isso as

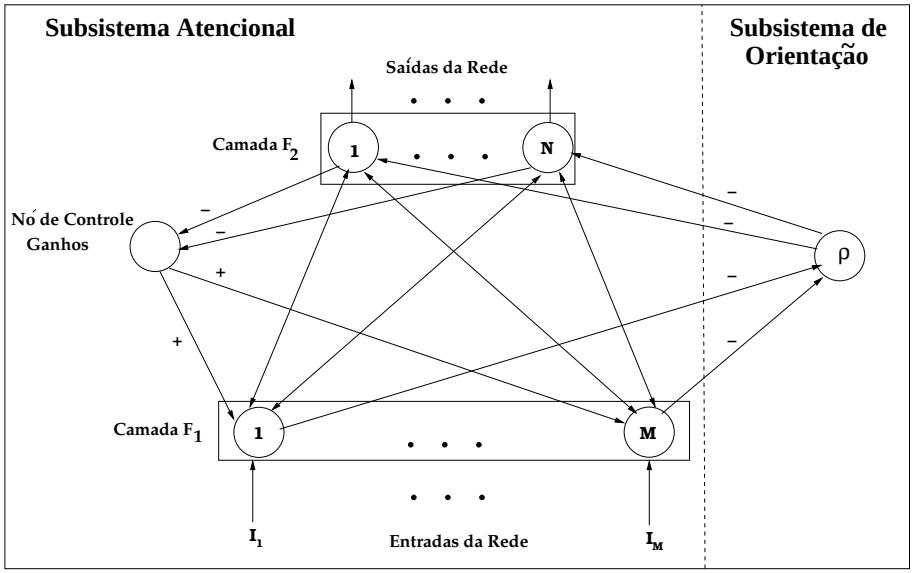


Figura 1.1: Estrutura da rede ART1.

unidades são ditas **não-rotuladas** ou desativadas (*uncommitted*), e o Nó de Controle de Ganhos está ativo. À medida que o treinamento prossegue, as unidades que já participaram de alguma aprendizagem, codificando os padrões em seus traços LTM *top-down*, passam a ser ditas **rotuladas** (*committed*) ou ativadas. As saídas de F_1 até o Subsistema de Orientação estão regidas pela **regra de 2/3**, ou seja, se ativando somente se 2 de suas 3 entradas se encontram ativas simultaneamente. O fato do Nó de Controle de Ganhos se encontrar ativo e existir um padrão na entrada, faz com que a atividade inibidora sobre o Subsistema de Orientação seja suficiente para sua desativação. A propagação do padrão de entrada de F_1 a F_2 (propagação ascendente) determina as funções de escolha de categoria (*Category Choice Function*, FEC) $T_j(\mathbf{I})$, que representa a similaridade entre o padrão de entrada e o vetor de pesos $\mathbf{w}_j$ da unidade j em F_2:

$$T_j(\mathbf{I}) = \frac{|\mathbf{I} \cap \mathbf{w}_j|}{|\mathbf{w}_j|} \tag{1.1}$$

onde $\cap$ designa o operador AND binário e $\mathbf{I} = (I_1, \ldots, I_M)$ é o padrão de entrada (M é a dimensão do espaço de entrada). O $\mathbf{w}_j$ é o vetor representante (valor esperado ou protótipo) da categoria interna j associada à unidade j de F_2, e $|\mathbf{x}|$ designa a soma dos componentes do vetor $\mathbf{x} = (x_1, \ldots, x_M)$:

$$|\mathbf{x}| = \sum_{i=1}^{M} x_i \tag{1.2}$$

A camada F_2 possui um mecanismo de ativação competitivo (competição "*Winner-Takes-All*", WTA), onde somente a unidade J com entrada máxima T_J é a ganhadora (em caso de empate, ganha a categoria J com $T_J = max_j\{T_j\}$ e índice J mínimo). A ativação exclusiva desta unidade, associada à categoria que apresenta uma função de escolha maior com o padrão de entrada, ocasiona a propagação descendente de seu representante $\mathbf{w}_J$ até a camada F_1.

Por outro lado, a ativação da unidade ganhadora J em F_2 inibe o nó de Controle de Aquisição, de forma que a saída inibidora desde F_1 ao Sistema de Orientação é uma medida de coincidência ("*matching*") $m_J(\mathbf{I})$ entre os dois vetores de entrada a F_1: $\mathbf{I}$ ascendente, e $\mathbf{w}_J$ (representante (LTM) da classe ganhadora J), descendente. Esta coincidência está determinada pela expressão:

$$m_J(\mathbf{I}) = \frac{|\mathbf{I} \cap \mathbf{w}_J|}{|\mathbf{I}|} \tag{1.3}$$

Se $m_J(\mathbf{I})$ é superior ao parâmetro de vigilância ρ ($0 \leq \rho < 1$), se alcança **ressonância** entre o padrão de entrada e o valor esperado para a unidade ganhadora, situação caracterizada por uma coincidência $m_J(\mathbf{I})$ entre ambos os vetores ($\mathbf{I}$ e $\mathbf{w}_J$) superior ao mínimo ρ (vigilância) exigido pela rede. O estado de ressonância se prolonga até a apresentação do padrão de entrada seguinte. Durante este intervalo de tempo, o valor esperado $\mathbf{w}_J$ da unidade ressonante J atualiza-se mediante um AND binário, componente a componente, com o padrão de entrada $\mathbf{I}$, conforme eq. 1.4.

$$\mathbf{w}_J(n+1) = (1-\beta)\mathbf{w}_J(n) + \beta(\mathbf{I} \cap \mathbf{w}_J(n)) \tag{1.4}$$

onde $0 < \beta \leq 1$ é a velocidade de aprendizagem; no caso especial de $\beta = 1$, chamamos de **aprendizagem rápida** (eq. 1.5).

$$\mathbf{w}_J(n+1) = \mathbf{I} \cap \mathbf{w}_J(n) \tag{1.5}$$

Se, ao contrário, o *matching* $m_J(\mathbf{I})$ é inferior a vigilância ρ, o Subsistema de Orientação se ativa, inibindo a unidade ganhadora em F_2 (rejeição ou *reset*). O mecanismo competitivo determina uma nova unidade ganhadora e, repete-se o processo

até que se alcance a ressonância com alguma das unidades associadas às categorias, já aprendidas pela rede. Se isso não ocorre, porque todas as unidades estão rejeitadas, a rede cria uma categoria nova e associa-se a uma unidade livre (quer dizer, não associada a nenhuma categoria) J' em F_2, igualando-se seu representante (ou valor esperado) $\mathbf{w}_{J'}$ ao padrão de entrada atual, $\mathbf{w}_{J'} = \mathbf{I}$.

Assim, desde a sua criação, a rede ART1 tem sido popular e muito estudada por vários pesquisadores, em termos de suas capacidades gerais e em termos de seus usos, em diversas aplicações; em [38] foi utilizado na aeronáutica em construção, especificamente para o reconhecimento de peças de aviões e a sua recuperação a partir de uma base de dados; nesta linha também em [47] ART1 foi aplicado na área de produção, utilizado com sucesso para classificar e agrupar vetores similares de uma matriz máquina-peça, e em[48] ART1 foi adaptado e implementado em um neuro-computador de 256 processadores, permitindo assim analisar a vantagem do paralelismo inerente ao seu algoritmo, resultando em uma melhora significativa na otimização da matriz máquina-peça. Em [137], com a filosofia de manufatura GT, foi aplicado ART1 modificado para solucionar o problema de formação da célula fracionária minimizando significantemente o número de elementos excepcionais. Enquanto que em [108, 109, 110] foram destacadas as propriedades interessantes que ART1 possui para resolução de aplicações do mundo real, através dos estudos das propriedades de agrupamento de ART1 e, um estudo comparativo dos resultados obtidos através de agrupamentos sob várias situações características de aplicações da vida real, e outras de processamento em lote de um ambiente estático. E, finalmente, em [55] foi aplicado como rede base da proposta do mecanismo de categorização adaptativa para o estudo de comportamento em robôs.

Mesmo que em nosso trabalho de investigação não usamos ART1, o seu estudo em primeiro lugar responde a um critério cronológico, e também se faz necessário estudar os seus princípios, para compreender o funcionamento dos modelos ART posteriores.

1.2.2 ART2

Uma das limitações da rede ART1 é o fato de não operar com dados contínuos, levando, assim, ao desenvolvimento das redes ART2 e *Fuzzy* ART; as quais são aptas ao processamento desse tipo de dados. A rede ART2 [24] está planejada para categorizar seqüências arbitrárias de dados de entrada analógicos ou digitais.

Esta possui mecanismos de normalização e filtragem que proporcionam uma boa capacidade de eliminação de ruídos. Cada subcamada realiza um processamento de similaridade e aprendizagem dessas seqüências de padrões em um modo estável.

Arquitetura

A rede ART2 consiste basicamente da mesma arquitetura geral ART1, conforme a fig. 1.2[1]. Possui um subsistema Atencional e um subsistema de Orientação. O subsistema Atencional está dividido no bloco F_1 (detector de características) e a camada F_2 (representação das categorias internas, agrupamentos ou *clusters*), enquanto que o subsistema de orientação (similar à ART1) possui o controle dos sinais de rejeição (*reset*). Os pesos de conexões entre a F_1 e a F_2 formam uma memória de longo-termo (LTM) adaptativa, que são utilizados para controlar a similaridade dos padrões de uma mesma unidade (mecanismo de rejeição).

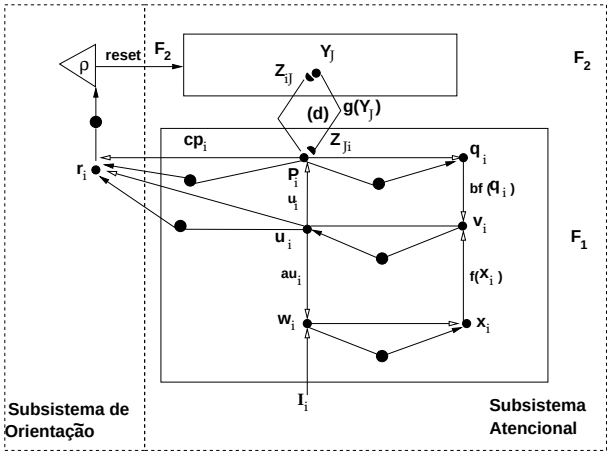


Figura 1.2: Estrutura típica da rede ART2.

O bloco F_1 é constituído por um conjunto de seis camadas possuindo conexões *feedforward* e *feedback* entre si. As unidades de cada camada são expressas como $\mathbf{w}_i$, $\mathbf{x}_i$, $\mathbf{u}_i$, $\mathbf{v}_i$, $\mathbf{p}_i$ e $\mathbf{q}_i$, enquanto que as unidades da camada F_2 são expressas como $\mathbf{y}_j$. Cada camada realiza um processamento específico, não servindo apenas como simples transmissoras do sinal de entrada, como ocorre na rede ART1.

[1]Nesta fig. 1.2 os círculos maiores representam as operações de normalização executadas pela rede, e as flechas brancas indicam as entradas padronizadas específicas para as unidades destinos, enquanto que as flechas negras indicam as entradas dos controle de ganhos não-específicas.

Funcionamento

Assim como em ART1, o subsistema de Orientação é o mecanismo de rejeição que controla o estado de cada unidade na camada F_1. No entanto em ART2, os padrões de entrada descendentes (*bottom-up*) e os sinais ascendentes (*top-down*) são recebidos em diferentes localizações em F_1. Os laços (*loops*) positivos de retorno na camada F_1 realçam as características e suprimem o ruído. As equações 1.6 descrevem as atividades das subcamadas de F_1. A aprendizagem ocorre se houver suficiente similaridade entre os pesos da unidade ganhadora em F_2 e o vetor de entrada.

$$\begin{gathered} w_i = I_i + au_i \qquad x_i = \frac{w_i}{e+\|\mathbf{w}\|} \qquad p_i = u_i + \sum_{j=1}^{N} g(y_j) Z_{ji} \\ q_i = \frac{p_i}{e+\|\mathbf{p}\|} \qquad v_i = f(x_i) + bf(q_i) \qquad u_i = \frac{v_i}{e+\|\mathbf{v}\|} \end{gathered} \tag{1.6}$$

onde a e b são valores constantes positivos. A função $g(y_j)$ refere-se a saída do elemento j-ésimo da camada F_2, que é definida pela eq. 1.8.

A camada F_2 consiste de uma camada de unidades, associadas às categorias internas criadas pela rede no espaço de entrada, que está completamente conectada com a subcamada $\mathbf{p}$ em F_1, ambos em termos de pesos de entrada e saída. O número de unidades de F_2 é arbitrário, tão grande quanto necessária no caso ideal, mas na prática é imposto um limite. Cada unidade realiza uma soma ponderada simples dos vetores-$\mathbf{p}$ de entrada e então compete com as outras unidades de F_2[2] para determinar a unidade (a categoria) de maior ativação, utilizando as equações 1.7 e 1.8.

$$T_j = \sum_{i=1}^{M} p_i Z_{ij} \tag{1.7}$$

$$g(y_j) = \begin{cases} d \quad se & T_J = max(T_j : j \notin \Re) \\ 0 & \text{senão} \end{cases} \tag{1.8}$$

onde $\Re$ é o conjunto de unidades que foram rejeitadas e $0 < d \leq 1$. Assim, o propósito da camada F_1 é o de filtrar os vários sinais LTM e STM, numa forma composta, que permita fazer uma comparação útil dos dois. Enquanto que o propósito

[2]Em [24] refere-se a esta competição como "aumento de contraste" através do qual até mesmo as pequenas diferenças de uma unidade, pode torná-la ganhadora sobre outra.

da camada F_2 é o de buscar e selecionar uma categoria interna, não rejeitada, que tenha mais coincidência com o padrão STM.

O mecanismo de orientação consiste de uma subcamada **r**, e uma unidade integrada. A subcamada **r** possui o mesmo número de unidades que o padrão de entrada, e está completamente conectada à unidade integrada. O vetor **r** monitora o grau de coincidência entre o padrão STM na camada F_1 e o padrão ativo LTM na camada F_2, de acordo com a eq. 1.9.

$$\mathbf{r} = \frac{\mathbf{u} + c\mathbf{p}}{e + \|\mathbf{u}\| + \|c\mathbf{p}\|} \tag{1.9}$$

onde c é o parâmetro de ativação da unidade ganhadora em F_2 ($c > 0$). Se $\|\mathbf{r}\| < \rho$ o sistema de rejeição é ativado (ρ é o parâmetro de vigilância e $0 \leq \rho < 1$), e em caso contrário, ocorre a ressonância. Este estado se caracteriza pela repetição alternada dos processos de atualização incrementalmente, baseados no vetor-**p**, dos pesos de duplo sentido $F_1 - F_2$, através das equações 1.10.

$$Z_{ij}(n+1) = Z_{ij}(n) + \beta d(p_i - Z_{ij}(n)); \quad Z_{ji}(n+1) = Z_{ji}(n) + \beta d(p_i - Z_{ji}(n)) \tag{1.10}$$

onde β é a velocidade da aprendizagem. Quando $\beta = 1$, é o modo de aprendizagem rápida em ART2, e neste modo fornece vetores de peso equivalentes aos vetores obtidos em ART1.

Podemos citar alguns exemplos de aplicações com a rede ART2, como [56] que utilizou ART2 no reconhecimento de caracteres chineses; em [139] com a proposta de melhorar o reconhecimento de padrões, usou ART2 com vários vetores de características; em [111] utilizou-a para classificar imagens ASTER para o mapeamento de uso e cobertura da terra em uma área de floresta tropical; em [100] explorou sua potencialidade em manipular padrões de entrada analógicas, para classificar parâmetros de simulação; em [50] foi usada em conjunto com aprendizagem por reforço no sentido de impedir colisões entre os robôs móveis e obstáculos.

1.2.3 *Fuzzy* ART

A rede *Fuzzy* ART (FA) [32] constitui uma evolução da rede ART1, orientada a categorizar de forma estável padrões de entrada analógicos com componentes compreendidos entre 0 e 1. Por isso FA substitui os operadores de intersecção ($\cap$) e de união ($\cup$) de ART1 pelos operadores difusos $MIN(\wedge)$ e $MAX(\vee)$, respectivamente, da teoria de lógica difusa [148]. Esta mudança, com a ajuda da codificação em complemento (*complement coding*), preserva a amplitude da informação ao mesmo tempo que normaliza os vetores de entrada, permite implementar o algoritmo de classificação não supervisionado com grande rapidez de aprendizagem. Por outro lado, em [91] foi apresentada uma série de resultados teóricos sobre a rede FA que demostraram interessantes propriedades na sua aprendizagem.

Em comparação com a rede ART1, a rede *Fuzzy* ART possui duas diferenças: (1) No que se refere aos traços LTM, o fluxo bidirecional de informações entre as camadas F_1 e F_2 é realizado por um único conjunto de pesos sinápticos, $\mathbf{w}_j$; (2) a rede *Fuzzy* ART necessita de um tipo específico de pré-processamento das entradas, de modo a evitar um problema inerente de proliferação de categorias [113]. A solução proposta foi a utilização de padrões com norma constante, através da codificação em complemento. Significa que se duplica o número de entradas, transformando o padrão de entrada $\mathbf{a} = (a_1, ..., a_M)$ no $\mathbf{I} = (\mathbf{a}, \mathbf{a}^c) = (a_1, ..., a_M, 1 - a_1, ..., 1 - a_M)$, garantindo assim que $|\mathbf{I}| = M$, o qual evita o decrescimento nos valores dos representantes e a proliferação de categorias derivado do contínuo decrescimento na norma $|\mathbf{w}_j|$ dos pesos de conexão por usar o operador MIN difuso $\wedge$ na sua atualização.

FA pode ser considerado um "método baseado a exemplos" para o agrupamento (*clustering*) de padrões de entrada. Estes padrões agregam-se em categorias ou agrupamentos com forma de hiper-retângulos (em geral, sobrepôndo-se), que cobrem distintas regiões do espaço de entrada. A formação de grupos é uma maneira de compressão, na que se formam regras abstratas sobre a distribuição dos dados que permitem uma certa generalização. Por outro lado, FA usa a aprendizagem baseada em coincidência (*matching*) para incorporar padrões a uma categoria.

Arquitetura

O estudo precedente sobre a rede ART1 (subseção 1.2.1) permite um entendimento mais direto da rede *Fuzzy* ART. Na fig. 1.3 é apresentada a arquitetura básica desta rede. FA possui como padrão de entrada $\mathbf{a}$ um vetor M-dimensional $(a_1, \ldots, a_M)$. A cada categoria interna lhe corresponde um vetor $\mathbf{w}_j = (\mathbf{w}_{j1}, \ldots, \mathbf{w}_{j(2M)})$ de pesos adaptativos, onde $j = 1, \ldots, N$ e N é o número de possíveis categorias internas (ou unidades em F_2). O vetor de pesos de *Fuzzy* ART é o equivalente, em um único vetor, aos vetores de pesos *top-down* e *bottom-up* de ART1.

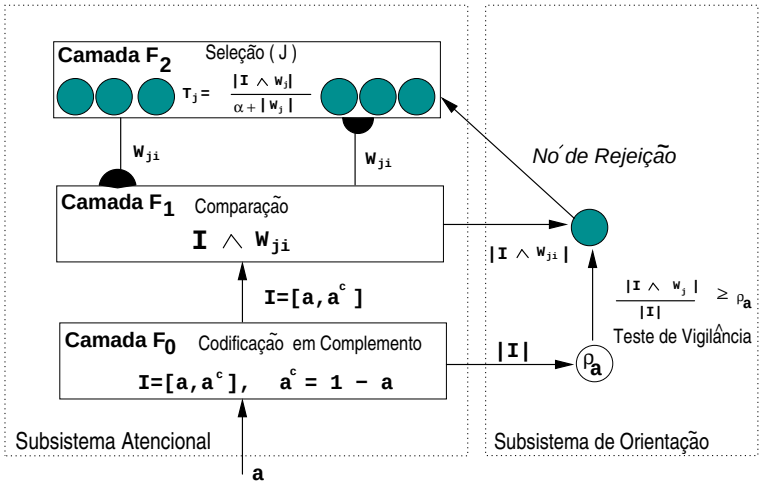


Figura 1.3: Estrutura de uma rede *Fuzzy* ART.

Assim, FA possui dois subsistemas, o atencional e o de orientação, semelhante a ART1. O atencional consiste em três camadas de unidades. Se a dimensionalidade do padrão de entrada é M, a camada do módulo F_0 tem M unidades e é uma etapa de pré-processamento a qual codifica o complemento do padrão de entrada que é a entrada da camada F_1. A camada F_1 então possui $2M$ unidades. A camada F_2 é do tipo competitiva, e as suas unidades estão completamente interconectadas via conexões laterais e cada unidade caracteriza um retorno inibitório (negativo), esta também é chamada "camada de representação de categoria", já que cada unidade está associada a uma categoria interna; enquanto que o subsistema de orientação consiste de apenas uma unidade, a de rejeição (*reset*), que recebe as entradas de F_0. Sua função é inibir as unidades de F_2 durante a busca de categorias.

Funcionamento

A rede *Fuzzy* ART possui além da estrutura, também o mecanismo de funcionamento muito semelhante a ART1. A fig. 1.2 apresenta um resumo das operações de ART1 e a sua correspondência em *Fuzzy* ART.

Tabela 1.2: Equivalências entre ART1 e *Fuzzy* ART.

Características	ART1	*Fuzzy*-ART
Natureza dos Dados	Binário	Contínuo (em $[0,1]$)
Operador	*AND* lógico ($\cap$)	*MIN fuzzy* ($\wedge$)
Função de Escolha de Categoria	$\frac{\lvert\mathbf{I}\cap\mathbf{w}_j\rvert}{\lvert\mathbf{w}_j\rvert}$	$\frac{\lvert\mathbf{I}\wedge\mathbf{w}_j\rvert}{\alpha+\lvert\mathbf{w}_j\rvert}$
Critério de coincidência	$\frac{\lvert\mathbf{I}\cap\mathbf{w}_j\rvert}{\lvert\mathbf{I}\rvert} \geq \rho$	$\frac{\lvert\mathbf{I}\wedge\mathbf{w}_j\rvert}{\lvert\mathbf{I}\rvert} \geq \rho$
Aprendizagem *fast-learning*	$\mathbf{w}_J(n+1) = \mathbf{I} \cap \mathbf{w}_J(n)$	$\mathbf{w}_J(n+1) = \mathbf{I} \wedge \mathbf{w}_J(n)$

Durante a etapa de aprendizagem, assume-se que um novo padrão $\mathbf{a}$ é apresentado a rede. Depois de ser codificado seu complemento em F_0 de modo que $\mathbf{I} = (\mathbf{a}, \mathbf{a}^c)$, este é propagado através da camada F_1 às unidades de F_2. Na aprendizagem, F_2 consistirá de unidades rotuladas (*committed*) e não-rotuladas, e sendo uma camada competitiva, todas as unidades competirão nos termos da função de escolha de categoria (FEC), definida pela equação 1.11.

$$T_j(\mathbf{I}) = \frac{|\mathbf{I} \wedge \mathbf{w}_j|}{\alpha + |\mathbf{w}_j|} \tag{1.11}$$

onde $\wedge$ designa o operador *fuzzy* AND que é definido como $(\mathbf{x} \wedge \mathbf{y})_i \equiv min(x_i, y_i)$. O α é o parâmetro de escolha ($\alpha \gtrsim 0$). Esta função de escolha de categoria (FEC) é a denominada Lei de *Weber*. Após o cálculo de todas as FEC das unidades de F_2, seleciona-se a categoria (unidade de F_2) J com maior $T_j(\mathbf{I})$:

$$J = argmax_j\{T_j(\mathbf{I})\} \tag{1.12}$$

Caso exista mais de um T_J máximo, a categoria j com menor índice é escolhida. A ressonância ocorre se a FEC da categoria ganhadora J satisfaz o teste de vigilância:

$$m_J(\mathbf{I}) = \frac{|\mathbf{I} \wedge \mathbf{w}_J|}{|\mathbf{I}|} \geq \rho \Rightarrow |\mathbf{I} \wedge \mathbf{w}_J| \geq \rho M \tag{1.13}$$

Onde se considera que $|\mathbf{I}| = M$ pela codificação em complemento. Em caso

de superar o teste de vigilância, o processo de aprendizagem é efetuado de forma análoga a ART1, mas com o operador $\wedge$, gerando o decrescimento na norma $|\mathbf{w}_J|$ do vetor de pesos[3]:

$$\mathbf{w}_J(n+1) = \beta(\mathbf{I} \wedge \mathbf{w}_J(n)) + (1-\beta)\mathbf{w}_J(n) \tag{1.14}$$

onde $\beta \in (0, 1]$ é a velocidade da aprendizagem e é especificada na etapa de aprendizagem de FA. A aprendizagem rápida ocorre quando $\beta = 1$ (conforme eq. 1.15).

$$\mathbf{w}_J(n+1) = \mathbf{I} \wedge \mathbf{w}_J(n) \tag{1.15}$$

Caso a categoria vencedora não satisfaça o teste de vigilância, ela recebe um sinal de *reset* (rejeição) procedente do Subsistema de Orientação (fig. 1.3), permanecendo inativa, enquanto o padrão de entrada atual estiver presente. Neste caso, o processo de competição seleciona uma nova categoria ganhadora, com a que se repete o teste de vigilância na (eq. 1.11). De modo similar à ART1, este processo continua até que a categoria ganhadora selecionada satisfaça o dito teste ou, se nenhuma categoria o satisfaz, FA cria uma categoria nova J', associada a uma unidade não rotulada (*uncommitted*) em F_2, cujo representante se iguala ao padrão de entrada: $\mathbf{w}_{J'} = \mathbf{I}$.

A estrutura das unidades rotuladas possui uma interessante representação geométrica. O representante $\mathbf{w}_j = (w_{j1}, \ldots, w_{j(2M)})$ da categoria j se pode escrever como $\mathbf{w}_j = (\mathbf{u}_j, \mathbf{v}_j^c)$, de modo que $\mathbf{u}_j$ e $\mathbf{v}_j$ definem um hiper-retângulo no espaço de entrada com os seus lados paralelos aos eixos de coordenadas. Este hiper-retângulo cobre a região associada à dita categoria, uma vez que $u_{jk} = min_l\{a_{lk}\}$ e $v_{jk} = max_l\{a_{lk}\}$, $k = 1, \ldots, M$, sendo $\mathbf{a}_l$ o l-ésimo padrão de entrada que alcançou a ressonância com a categoria j. Com a codificação em complemento, a primeira metade das componentes de $\mathbf{w}$ são os mínimos dos padrões codificados por essa categoria, enquanto que a segunda metade são os máximos desses padrões. Portanto, o valor esperado define o intervalo de variação dos diferentes componentes dos padrões de entrada codificados por uma categoria. Cada categoria resume os dados que caem dentro de seu hiper-retângulo correspondente (ou nas suas fronteiras) apenas através do armazenamento destes dois vetores. A fig. 1.4 descreve

[3]Se $|\mathbf{w}_J|$ alcança um valor muito baixo, o teste de vigilância poderia não se cumprir nunca, por ser $|\mathbf{I} \wedge \mathbf{w}_J| \leq |\mathbf{w}_J|$, de modo que $|\mathbf{I} \wedge \mathbf{w}_J|$ nunca superaria ρM, nenhuma categoria superaria o teste de vigilância e, se produziria a criação de uma categoria nova para cada padrão de entrada. Esta é a origem do problema da proliferação de categorias [113].

graficamente uma categoria em um espaço de características bi-dimensional. A área do hiper-retângulo R_j definido por u_j e v_j na fig. 1.4 é a "região de representação de categoria" (*Category Representation Region*, CRR) da categoria j. Assim, uma característica geométrica de cada categoria é o seu tamanho R_j, que é definido como (perímetro do hiper-retângulo) a soma dos intervalos do hiper-retângulo em todas as dimensões de R_j, conforme 1.16.

$$|R_j| \equiv |\mathbf{v}_j - \mathbf{u}_j| = \sum_{k=1}^{M} (v_{jk} - u_{jk}) \tag{1.16}$$

onde $\mathbf{w}_j$ é o vetor peso correspondente e M a dimensão de entrada. Pode-se demonstrar que:

$$|R_j| = M - |\mathbf{w}_j| \tag{1.17}$$

de modo que $0 \leq R_j \leq M$.

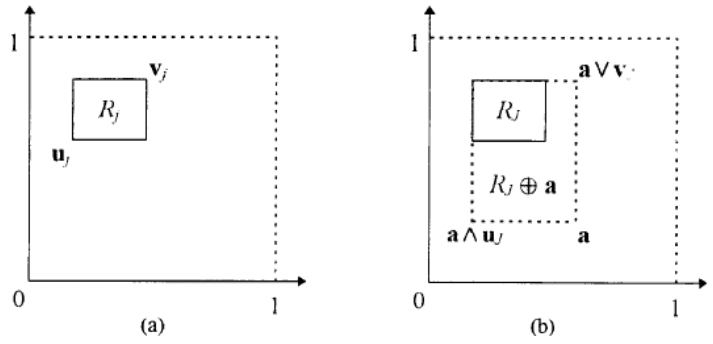


Figura 1.4: (a) Interpretação geométrica da categoria em *Fuzzy* ART em um espaço de entrada bi-dimensional, na forma de codificação em complemento. (b) Representação geométrica do treinamento rápido.

A categoria representa um conjunto de padrões de treinamento similares, usando a norma $|\mathbf{x}|$ de um vetor como métrica de classificação para avaliar a similaridade entre os padrões de entrada e os representantes das categorias. A distância de um padrão $\mathbf{a}$ à categoria j com peso $\mathbf{w}_j$, denominada por $dist(\mathbf{a}, \mathbf{w}_j)$ é definida como a distância mínima entre $\mathbf{a}$ e a região de representação da categoria j. Se o padrão $\mathbf{a}$ está localizado dentro do hiper-retângulo, então $dist(\mathbf{a}, \mathbf{w}_j) = 0$ e o padrão é

codificado pela categoria. Em caso contrário, a distância de um padrão $\mathbf{a}$ de uma categoria com peso $\mathbf{w}_j$ é calculado pela equação 1.18.

$$dist(\mathbf{a}, \mathbf{w}_j) = |\mathbf{w}_j| - |\mathbf{I} \wedge \mathbf{w}_j| = \sum_{k=1}^{M} [\{(a_k \vee v_{jk}) - v_{jk}\} + \{u_{jk} - (a_k \wedge u_{jk})\}] \quad (1.18)$$

onde $\mathbf{a}$ é o padrão de entrada e $\mathbf{I}$ é o padrão de entrada com seu código em complemento $\mathbf{I} = (\mathbf{a}, \mathbf{a}^c)$. Esta é a distância *city-block* ou *Manhattan* entre $\mathbf{a}$ e o $\mathbf{w}_j$, e verifica $0 \leqslant dist(\mathbf{a}, \mathbf{w}_j) \leqslant M$.

A aprendizagem em FA é obtida de duas maneiras: pela expansão da categoria ressonante (painel (b) na fig. 1.4) ou através da criação de categorias novas. No primeiro caso, a categoria ressonante incrementa o seu tamanho e, simultaneamente, reduz a distância entre o padrão e a categoria (eq. 1.14). A criação de uma categoria nova envolve a rotulação de uma unidade J' não-rotulada na camada F_2, igualando o seu representante ao padrão de entrada $\mathbf{w}_{J'} = \mathbf{I}$.

A aprendizagem aumenta o tamanho do hiper-retângulo R_J, ou seja, cresce monotonicamente em uma ou várias das suas dimensões, e o valor de $\mathbf{w}_j$ reduz (eq. 1.17). O tamanho máximo de R_j é condicionado ao parâmetro de vigilância ρ. Durante cada processamento da aprendizagem rápida, R_j expande para $R_j \oplus \mathbf{a}$, sendo $\oplus$ o operador de expansão, o qual é o retângulo mínimo contendo R_j e $\mathbf{a}$ (conforme painel (b) na fig. 1.4). Os vértices de $R_j \oplus \mathbf{a}$ são dados por $\mathbf{a} \wedge \mathbf{u}_J$ e $\mathbf{a} \vee \mathbf{v}_J$. O tamanho de $R_j \oplus \mathbf{a}$ pode ser representado por:

$$|R_J \oplus \mathbf{a}| \equiv |(\mathbf{a} \vee \mathbf{v}_J) - (\mathbf{a} \wedge \mathbf{u}_J)| \quad (1.19)$$

Com a aprendizagem rápida, cada R_j iguala-se ao menor retângulo que engloba todos os vetores $\mathbf{a}$ escolhidos para a dita categoria J. O teste de vigilância estabelece que seja escolhida outra categoria se $R_J \oplus \mathbf{a}$ for muito grande. Concretamente, isto implica que:

$$|R_J| \leq M(1 - \rho) \quad (1.20)$$

Em FA, não pode existir sobreposição entre um hiper-retângulo de uma categoria dentro do hiper-retângulo de outra categoria. Ou seja, se um padrão $\mathbf{a}$ cai dentro

de um hiper-retângulo de uma única categoria, alcança a ressonância com a mesma. Para demonstrá-lo, observemos o seguinte:

$$|\mathbf{I} \wedge \mathbf{w}_J| = \sum_{i=1}^{2M} I_i \wedge w_{Ji} = \sum_{i=1}^{M} a_i \wedge u_{Ji} + \sum_{i=1}^{M} a_i^c \wedge v_{Ji}^c = \tag{1.21}$$
$$= \sum_{i=1}^{M} a_i \wedge u_{Ji} + \sum_{i=1}^{M} a_i \vee v_{Ji} = \sum_{i=1}^{M} u_{Ji} + \sum_{i=1}^{M} v_{Ji} = |\mathbf{w}|$$

Entretanto como $|R_J|$ (o tamanho da categoria J antes da ressonância) verifica $|R_J| = M - |\mathbf{w}| < M(1-\rho)$ (em caso contrário, a categoria J não poderia alcançar este tamanho), chegamos a que $|\mathbf{I} \wedge \mathbf{w}_J| = |\mathbf{w}_J| > \rho M$, de modo que a categoria J supera o teste de vigilância e alcança a ressonância. As categorias com hiper-retângulos menores são mais específicas (cobrem uma menor área do espaço de entrada), e as de hiper-retângulos grandes são mais genéricas.

Outra propriedade de FA é que a sua resposta às entradas específicas é facilmente explicada, já que existe uma representação de categorias internas que constituem uma partição do espaço de entrada. Ao contrário de outras redes neurais, onde, em geral, é difícil explicar porque um padrão de entrada produz uma saída. Devido a esta propriedade, é direto extrair um subconjunto significante de regras de uma FA treinada, o qual é importante em problemas de extração de conhecimento, mediante aprendizagem automática, onde não apenas a resposta do sistema é importante, mas também o conhecimento que levou a esta resposta.

Encontramos na literatura várias aplicações desta rede, em diferentes áreas como em [98], onde foi utilizado como base para um modelo de reconhecimento de padrões "robusto e invariável"[4]. Em [118], uma rede FA foi utilizada para melhorar os resultados de segmentação de imagens e foi testado a sua eficiência para uma aplicação com robôs móveis; enquanto que em [116] foi utilizada, em combinação com um algoritmo genético (AG), para classificação de dados de radar para mapeamento de cobertura de terra. Em [14] temos a utilização (e de *Fuzzy* ARTMAP [28]) como classificador de frutas segundo seus diferentes estados de maturação com o uso de um nariz eletrônico; ou em [149] onde FA foi aplicado para melhorar a detecção e classificação de falhas em linhas de transmissão.

[4]Estas propriedades significam, respectivamente: que esse modelo reconhece os objetos que são transladados, escalados e rotacionados, e que o sistema tem uma forte resistência ao ruído.

1.3 Redes ART supervisionadas

Os modelos ART mais recentes permitem a aprendizagem supervisionada se baseando nos princípios e conceitos (categorias internas, representantes de categorias, funções de escolha, aprendizagem competitiva, teste de vigilância, aprendizagem *on-line*) que aparecem nas redes ART não supervisionadas. Em geral, os modelos ART supervisionados, que veremos nas seguintes seções, permitem também a aprendizagem não supervisionada, mediante a desativação de certos módulos da rede, recuperando assim a funcionalidade dos modelos já vistos.

1.3.1 ARTMAP

Os modelos ART foram originalmente desenhados para o agrupamento ou categorização não supervisionada de padrões, quer dizer, para o estabelecimento de uma correspondência entre um padrão de entrada e uma saída ativa (categoria ressonante). Porém, a rede ARTMAP [29] se orienta à construção de correspondências entre as múltiplas entradas e saídas, convertendo-se num associador de padrões. No caso particular em que um dos padrões seja a predição de saída desejada para o padrão de entrada, a rede permite o tratamento de problemas de classificação supervisionada de padrões. Por isso, baseia-se em duas redes ART1 (ART_a e ART_b) e um módulo F^{ab} (no módulo Inter-ART) entre as camadas F_2 de ambas as redes, como observa-se na fig. 1.5. Desativando os módulos ART_b e F^{ab} recuperamos o funcionamento não supervisionado da rede ART1 original.

ARTMAP utiliza uma aprendizagem supervisionada onde se apresentam padrões nas camadas F_1^a e F_1^b. Estes padrões ressonam com as unidades em F_2^a e F_2^b, respectivamente, e F^{ab} estabelece correspondências entre elas. Durante o treinamento, ARTMAP permite a extração automática, a partir de um conjunto de amostra, de regras de decisão com a seguinte estrutura:

IF (A_1 AND ... AND A_N) THEN (B_1 AND ... AND B_M)

Os antecedentes $A_1, ..., A_N$ estão associados às unidades de F_1^a (entradas de ARTMAP), cada uma das quais representa uma condição, e os conseqüentes $B_1, ..., B_M$ se correspondem com as unidades de F_1^b (saídas de ARTMAP) e representam conseqüências inferidas a partir das condições. Por sua vez, durante o processamento ou teste, ARTMAP opera no modo de sistema especialista, usando as correspondências

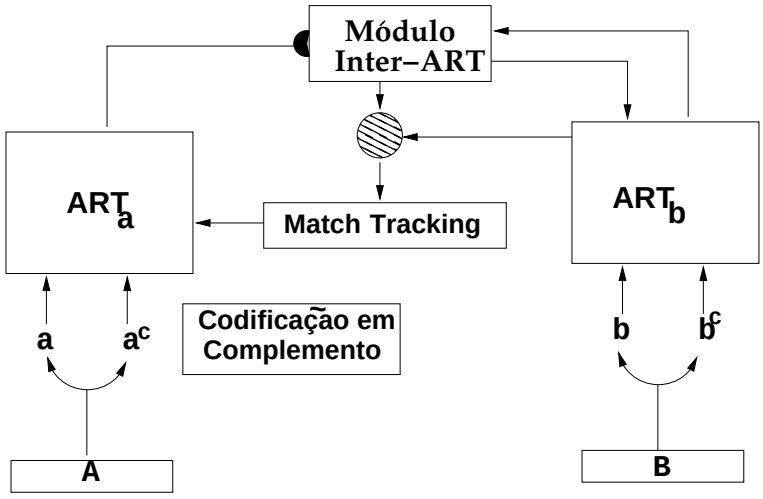


Figura 1.5: Estrutura da rede ARTMAP.

aprendidas, durante o treinamento, para codificar os padrões de entrada pelos padrões de saída. Os antecedentes em F_1^a provocam a ressonância com uma unidade em F_2^a. Esta, por sua vez, determina a unidade ganhadora em F_2^b, cujo representante se propaga à saída F_1^b e, determina os conseqüentes $B_1, ..., B_M$. Estas regras proporcionam uma certa flexibilidade tanto que, distintamente dos sistemas expertos ou especialistas tradicionais, não requerem o cumprimento exato de todos os antecedentes, senão um cumprimento global aproximado com um nível de aproximação determinado pela vigilância ρ. Por último, esta arquitetura permite a alternância entre as aprendizagens supervisionada e não supervisionada, proporcionando a qualidade na aprendizagem associada aos sistemas supervisionados, juntamente com a autonomia dos sistemas auto-organizáveis.

Assim a rede ARTMAP, durante a aprendizagem supervisionada, opera da seguinte forma: a rede ART_a recebe os padrões **a** e a ART_b recebe os padrões **b**, que atua como o padrão que ARTMAP deve aprender a associar ao padrão **a**[5]. Uma rede de aprendizagem associativa e um controlador interno une estes módulos para fazer o sistema ARTMAP operar em tempo real. O controlador cria o número mínimo de categorias de reconhecimento em ART_a, ou "unidades ocultas", necessárias para

[5]Em problemas de classificação, **b** é um vetor de dimensão igual ao número de predições, com todos os seus componentes nulos, exceto ao associado à predição desejada para o padrão de entrada **a**, que vale 1. Deste modo, **b** atua como a saída desejada para o padrão **a**. Em todo caso, a rede ARTMAP admite uma simplificação importante para problemas de classificação, nos quais todo o módulo ART_b pode ser suprimido.

atingir o critério de precisão. Uma regra de aprendizagem mínimo-máximo (*min-max*) permite a ARTMAP aprender rápido e eficientemente, minimizando o erro de treinamento e maximizando a compressão do código.

O módulo Inter-ART em ARTMAP forma associações entre as categorias através da saída da aprendizagem e disparos de buscas através de uma regra de "*Match-Tracking*". Quando a categoria ganhadora J em F_2^a está associada a um padrão de saída $\mathbf{b}'$ distinto do padrão desejado $\mathbf{b}$, ARTMAP incrementa o parâmetro de vigilância ρ_a a mínima quantidade necessária para que a categoria J seja rejeitada em ART_a e uma nova categoria saia ganhadora. A base do parâmetro de vigilância $\overline{\rho}_a$ calibra um nível mínimo de confiança em que ART_a aceitará uma categoria selecionada. Os valores baixos de $\overline{\rho}_a$ permitem formar menos categorias, maximizando a compressão do código. Inicialmente $\rho = \overline{\rho}_a$. Durante a aprendizagem, uma falha na predição em ART_b provoca o *Match-Tracking*, que incrementa ρ_a somente o suficiente para disparar uma busca em ART_a assim, corrigindo o erro de predição. O módulo ART_a seleciona uma nova categoria, que enfoca a atenção a um agrupamento de características do padrão de entrada que seja mais capaz de predizer $\mathbf{b}$.

Na literatura existem várias aplicações de ARTMAP. Em [22], ARTMAP foi utilizada para melhorar a eficiência da produção de mapas de vegetação a partir de dados de sensoriamento remoto[6], e em [129] utilizando como dados uma seqüência de imagens coletados por satélite, que tinha por objetivo analisar automaticamente as mudanças no uso da terra, especificamente com as imagens da região do delta do rio de Nilo e as áreas circunvizinhas. Enquanto que em [35, 34] utilizam conjuntos de dados de sensoriamento remoto com um sistema de fusão[7] de ARTMAP para extrair o conhecimento hierárquico, resultando em regras hierárquicas multi-níveis, que descrevem relações entre as categorias.

1.3.2 *Fuzzy* ARTMAP

O sistema *Fuzzy* ARTMAP (FAM) mais geral [28] aprende a associar padrões de entrada e saída com valores difusos, quer dizer, no intervalo $[0, 1]$ e que portanto

[6]Com as imagens *Landsat Thematic Mapper* (TM) da floresta nacional da *Sierra*, Califórnia do Norte.

[7]Utiliza as respresentações distribuídas do código que exploram a capacidade da rede neural de aprendizagem um-para-muitos.

se podem interpretar como graus de pertinência aos conjuntos difusos. Esta generalização é alcançada pela substituição dos módulos ART1 do ARTMAP binário por módulos *Fuzzy* ART. A fig. 1.6 detalha a arquitetura de *Fuzzy* ARTMAP. A sua operação está baseada na similaridade entre o padrão de entrada $\mathbf{I}$ e os representantes $\mathbf{w}_j$ das categorias internas aprendidas pela rede. As funções de escolha, *Lei de Weber* (eq. 1.22) e *Choice-By-Difference* (eq. 1.23), são as mais usadas como FEC (Funções de Escolha de Categorias) $T_j(\mathbf{I})$ entre os padrões e a categoria j, associada à unidade j em F_2^a:

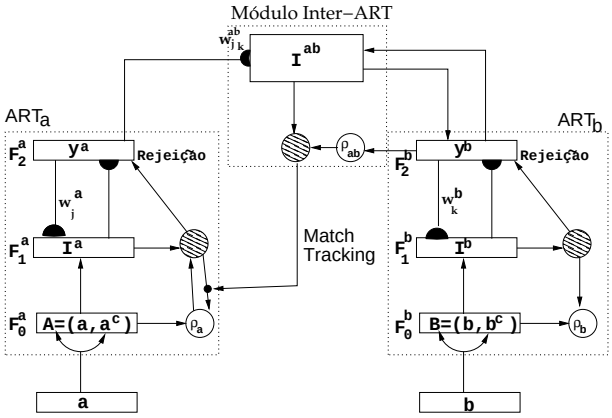


Figura 1.6: Estrutura de uma rede *Fuzzy* ARTMAP.

$$T_j^{WL} = \frac{|\mathbf{I} \wedge \mathbf{w}_j^a|}{\alpha + |\mathbf{w}_j^a|} \tag{1.22}$$

$$T_j^{CBD} = |\mathbf{I} \wedge \mathbf{w}_j^a| + (1-\alpha)(M - |\mathbf{w}_j^a|) \tag{1.23}$$

onde $\mathbf{I}$ é o padrão de entrada, codificada em complemento, $\mathbf{w}_j^a$ é o protótipo para a categoria interna j de ART_a, $\alpha \gtrsim 0$ e M é a dimensão do espaço de entrada. Ambas as funções definem implicitamente uma região geométrica hiper-retângular, cujas esquinas são definidas pelos componentes do vetor $\mathbf{w}_j$ (os valores máximo e mínimo dos componentes dos padrões de entrada codificados pela categoria j). Como em todos os modelos ART examinados, existe um processo de competição *WTA* em ART_a, no qual a categoria ganhadora J é aquela com maior $T_j(\mathbf{I})$: $J = argmax_j\{T_j(\mathbf{I})\}$. O mesmo ocorre em ART_b. O teste de vigilância determina se esta categoria ganhadora J alcança a ressonância com o padrão de entrada (eq. 1.24).

$$|\mathbf{I}^a \wedge \mathbf{w}_J^a| \geq \rho^a M \tag{1.24}$$

onde, como no caso de FA, se leva em consideração que $|\mathbf{I}^a| = M$, pela codificação em complemento. O teste de vigilância é igual em ART_b operando sobre $\mathbf{I}^b, \mathbf{w}_j^b$ e ρ_b. A aprendizagem em FAM é similar ao caso de FA. Em caso de ressonância, o representante $\mathbf{w}_J^a$ da categoria ressonante J se aproxima ao mínimo $\mathbf{I} \wedge \mathbf{w}_J^a$ de modo similar à FA (eq. 1.25).

$$\mathbf{w}_J^a(n+1) = (1-\beta)\mathbf{w}_J^a(n) + \beta(\mathbf{I}^a \wedge \mathbf{w}_J^a(n)) \tag{1.25}$$

onde $0 \leq \beta \leq 1$ é a velocidade de aprendizagem (o caso $\beta = 1$ é a aprendizagem rápida). A mesma expressão é válida para ART_b. Como no caso de FA, a codificação em complemento faz com que $\mathbf{w}_J^a$ (de dimensão $2M$, sendo M a dimensão do padrão de entrada antes da codificação) codifique, nos seus M primeiros componentes, os valores mínimos dos componentes dos padrões de entrada codificados pela categoria J. Analogamente, os últimos M componentes de $\mathbf{w}_J^a$ são os valores máximos dos componentes dos padrões de entrada codificados pela categoria j. Portanto, o representante de uma categoria interna de FAM, armazena os intervalos dos distintos componentes dos padrões codificados pela categoria.

Assim como em ARTMAP, o módulo Inter-ART forma associações entre as categorias de ART_a com as unidades em F^{ab}, através da saída da aprendizagem e disparos de busca através de uma regra de *Match-Tracking*. Estas unidades em F^{ab} representam as categorias em F_2^b, uma vez que as conexões entre F_2^b e F^{ab} possuem pesos igual a 1. As categorias (unidades em F_2^a) representam grupos (*clusters*) de padrões de treinamento no espaço de entrada. O valor esperado de cada categoria são os vetores máximo $\mathbf{v}$ e mínimo $\mathbf{u}$ dos padrões de treinamento ressonantes com essa categoria, como em *Fuzzy* ART.

A aprendizagem dos pesos $\mathbf{w}^{ab}$ segue a mesma regra que em FA, conforme eq. 1.26, mas empregando a ativação $\mathbf{y}_b$ em F_2^b ao invés de $\mathbf{I}$.

$$\mathbf{w}_J^{ab}(t+1) = (1-\beta)\mathbf{w}_J^{ab}(t) + \beta(\mathbf{y}_b \wedge \mathbf{w}_J^{ab}(t)) \tag{1.26}$$

onde, inicialmente $\mathbf{w}_J^{ab}(t=0) = 1$ e $\mathbf{y}_b$, em problemas de classificação, é a predição desejada para o padrão de entrada.

Se a aprendizagem é rápida $\beta = 1$, então $\mathbf{w}_J^{ab}$ somente varia no momento que se cria a categoria J. De fato, se supomos que a categoria J é a ganhadora e, se a predição desejada é K, então $\mathbf{y}_b = \mathbf{w}_J^{ab}$, ($w_{JK}^{ab} = 1$ e $w_{jk}^{ab} = 1, j \neq J$ ou bem $k \neq K$) e $\mathbf{w}_J^{ab}$ não varia. Se, ao contrário, a predição desejada não é K, então a categoria J é inibida pelo *Match-Tracking* e já não alcança a ressonância.

Quando a predição de ART_a é distinta de ART_b, a inibição do módulo Inter-ART F^{ab} ativa o processo de *Match-Tracking*, uma vez que satisfaz a condição de $|\mathbf{y}^b \wedge \mathbf{w}_J^{ab}| < \rho_{ab}|\mathbf{y}^b|$, e assim, incrementa-se o valor da vigilância (ρ_a) em ART_a através da equação:

$$\rho_a = \frac{|\mathbf{I} \wedge \mathbf{w}_J^a|}{M} + \epsilon \tag{1.27}$$

onde $\epsilon \gtrsim 0$. Este processo rejeita a categoria J ganhadora em F_1^a e força as futuras categorias ganhadoras em ART_a tenham um tamanho menor que $M(1 - \rho_a)$ para superar o teste de vigilância. Se não existir uma categoria com estas características, cria-se uma categoria nova que associa-se à predição correta. Em tal caso, a categoria nova teria o hiper-retângulo (inicialmente um único ponto, coincidente com o padrão de entrada) dentro do hiper-retângulo associado à primeira categoria, e as duas categorias terian predições distintas. Portanto, em FAM pode existir hiper-retângulos "ninhados" (sobrepostos completamente), e as categorias podem se sobrepôr parcialmente, quando tenham predições distintas. De fato, se um padrão cai dentro dos hiper-retângulos de várias categorias, codifica-se por a de menor tamanho, e portanto, a mais específica destas duas. Consideremos as categorias C_1 e C_2 em ART_a. Se $R_1 > R_2$, como $R_j = M - |\mathbf{w}_j|$, então $|\mathbf{w}_1| < |\mathbf{w}_2|$. Por outro lado, como $|\mathbf{I} \wedge \mathbf{w}_j| = |\mathbf{w}_j|$ com $j = 1, 2$ por cair **a** dentro dos dois hiper-retângulos, temos que:

$$T = \frac{|\mathbf{I} \wedge \mathbf{w}|}{\alpha + |\mathbf{w}|} = \frac{|\mathbf{w}|}{\alpha + |\mathbf{w}|} \tag{1.28}$$

Esta função é crescente com $|\mathbf{w}|$, e como $|\mathbf{w}_1| < |\mathbf{w}_2|$, chegamos a que $T_2 > T_1$ e a categoria com menor hiper-retângulo ganha a competição. Assim vai alcançar a ressonância, porque $|\mathbf{I} \wedge \mathbf{w}| = |\mathbf{w}| > \rho M$ uma vez que, como sempre se verifica $R = M - |\mathbf{w}| < M(1 - \rho)$, então $|\mathbf{w}| > \rho M$ para todas as categorias.

O fato de que um padrão caindo nos hiper-retângulos de várias categorias seja codificada por a de menor hiper-retângulo (a mais específica) garante que, se o

padrão se volta a apresentar imediatamente depois, essa categoria voltaria a alcançar a ressonância. No entanto, isto poderia não ocorrer se o mesmo padrão voltasse a ser apresentado depois de outros padrões, uma vez que, em tal caso, essa categoria poderia não ser a mais específica. Observa-se com este caso que o resultado da classificação depende fortemente da ordem de apresentação dos padrões. Este é um dos problemas mais criticados das redes *Fuzzy* ART/ARTMAP. Outro problema importante de FAM é a super-aprendizagem e a falha de generalização, especialmente em presença de ruído, por isso o algoritmo de aprendizagem de PTAM permite aprender o conjunto de treinamento com um erro muito baixo, mas sem garantir uma elevada capacidade de generalização.

Sobre as redes *Fuzzy* ART e *Fuzzy* ARTMAP foram demonstrados os teoremas de convergência que estabelecem limites superiores ao número de épocas necessárias para alcançar um erro nulo sobre qualquer conjunto de treinamento [91, 59], considerando a existência de um número suficiente de unidades nas camadas F_2. Porém, isto não garante que o erro seja nulo em um conjunto de teste, uma vez que pode ocorrer problemas de super-aprendizagem.

Em problemas de classificação supervisionada, FAM pode ser notavelmente simplificado, uma vez que neste caso não existe um padrão de saída que se deve associar ao padrão de entrada, mas sim a predição desejada para o mesmo. Portanto, não tem sentido categorizar os padrões $\mathbf{b}$ apresentados a ART_b, de modo que esta rede perde o seu sentido, e FAM se reduz a ART_a e F^{ab}, que atuaria como a camada onde as unidades codificam as distintas predições de saída. Além do mais, a camada F_2^a atuaria como uma camada oculta, análoga à camada oculta numa rede de funções de base radial (RBF) [90] ou numa MLP. Esta simplificação de FAM, para problemas de classificação supervisionada de padrões, constitui a base do modelo *Simplified* ARTMAP [96].

Entre as várias aplicações desta rede, temos em [70] que FAM teve avaliado seu desempenho conjuntamente com outras três variações de ARTMAP (ART-EMAP [37], ARTMAP-IC [33], e *Gaussian* ARTMAP [144]) na identificação de tipos de emissores de radar; em [71, 105] foi utilizado para classificação de dados incompletos de uma e mais formas. Também podemos destacar em [49] que FAM foi utilizado como base de uma ferramenta interativa que tinha como entrada os resultados de um sistema de fusão multi-sensor para imagem noturna, o qual destinava-se à busca e aprendizagem do destino.

1.3.3 *Fuzzy* Min-Max *Neural Network*

A rede neural de classificação *Fuzzy* Min-Max (FMMNN) [130] não se trata estritamente de uma rede ART, mas possui importantes similaridades com elas. Esta rede utiliza categorias com geometria hiper-retangular, e está estruturada em três camadas (fig. 1.7): a camada $F_A = (a_1, a_2, \ldots, a_n)$, associada ao padrão de entrada A_h; a camada $F_B = (b_1, b_2, \ldots, b_m)$, associada às categorias hiper-retangulares, e $F_C = (c_1, c_2, \ldots, c_p)$, associada às predições de saída (classes). As conexões entre as unidades de F_B e F_C são binárias e armazenam as associações entre as categorias internas e as classes.

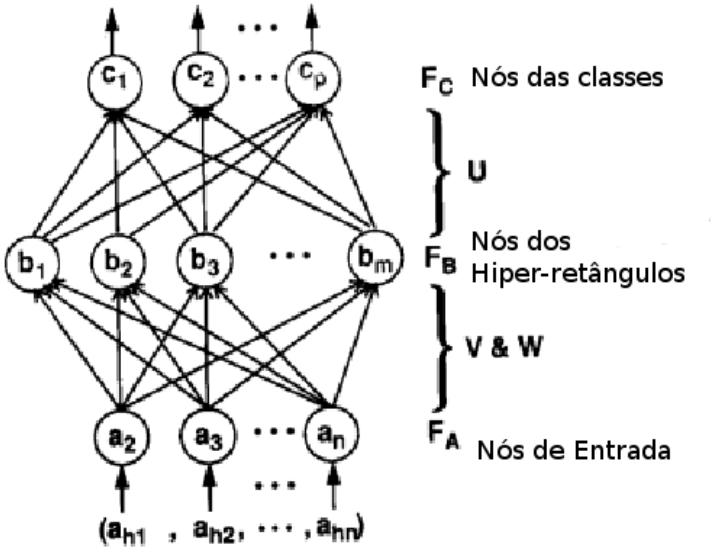


Figura 1.7: A estrutura da rede *Fuzzy* Min-Max *Neural Network*.

A aprendizagem *Fuzzy* Min-Max é um processo de expansão/contração incremental *on-line*. O conjunto de treinamento D consiste de M padrões $\{X_h, d_h\}$, onde $X_h = (x_{h1}, x_{h2}, \ldots, x_{hn}) \in [0,1]^n$ é o padrão de entrada e $d_h \in \{1, 2, \ldots, m\}$ é o índice de uma das m classes (saída desejada para o padrão de entrada)[8]. O processo de aprendizagem começa pela seleção de um padrão de treinamento e a busca de um hiper-retângulo para a mesma categoria que pode ser expandida (se necessário) para incluir o padrão. Se este hiper-retângulo não pode ser encontrado que corresponda ao critério de expansão, um novo hiper-retângulo é formado e adicionado à rede. Este processo de crescimento permite que as categorias formadas sejam

[8]Nesta subseção tentaremos seguir a anotação empregada em [130], ainda que seja distinta da empregada no resto das redes ART.

não-linearmente separáveis, permitindo a existência de categorias a ser refinadas todo tempo, e permitindo novas categorias serem adicionadas sem re-treinamento. A sobreposição entre hiper-retângulo pode ocorrer, caso estes representem a mesma categoria. Em caso contrário, a sobreposição é eliminada utilizando o processo de contração. Este processo somente elimina a sobreposição entre àquelas partes das categorias associadas às classes distintas.

O algoritmo de aprendizagem tem três passos:

- Expansão: Identifica o hiper-retângulo que pode ser expandido e estendê-lo. Se um hiper-retângulo expansível não puder ser encontrado, acrescenta um novo hiper-retângulo para aquela categoria.
- Teste de sobreposição: Determina se alguma sobreposição existe entre os hiper-retângulos de diferentes categorias.
- Contração: Se existe sobreposição entre os hiper-retângulos que representam diferentes categorias, elimina-se a sobreposição através do ajuste mínimo de cada um dos hiper-retângulos.

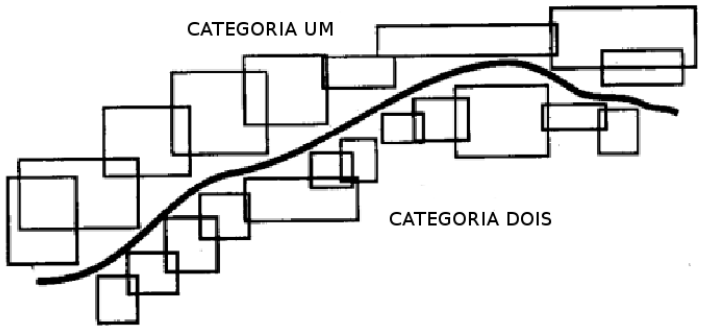


Figura 1.8: Um exemplo de hiper-retângulos *Fuzzy* Min-Max *Neural Network* dispostos ao longo da fronteira entre as categorias não-sobrepostas.

A etapa de *expansão* das distintas categorias hiper-retangulares em $\mathbb{R}^2$ está ilustrado na fig. 1.8. A classe C_k se define como:

$$C_k = \bigcup_{j \in K} B_j \tag{1.29}$$

onde K é o conjunto de índices das categorias hiper-retangulares B_j associadas à classe C_k. A operação de união dos conjuntos difusos é definida como o máximo de todas as funções de pertinência dos conjuntos difusos associados.

A *etapa de expansão* do hiper-retângulo ocorre quando, dado um padrão e a sua saída desejada $\{X_h, d_h\} \in D$, se encontra o hiper-retângulo B_j em relação ao qual o padrão de entrada apresenta o maior grau de pertinência (i.e., a maior função de escolha) entre os seus membros, que permite a expansão (se necessário), e que representa a mesma categoria que d_h. O grau de pertinência do padrão X_h a B_j (função de escolha) é calculado pela eq. 1.31.

$$b_j(X_h) = \frac{1}{2n}\sum_{i=1}^{n}[max(0, 1 - max(0, \gamma min(1, x_{hi} - w_{ji}))) + max(0, 1 - max(0, \gamma min(1, v_{ji} - x_{hi})))] \tag{1.30}$$

onde $X_h \in [0,1]^n$ é o h-ésimo padrão de entrada, $V_j = (v_{j1}, ..., v_{jn})$ é o ponto de mínimo para a categoria B_j, $W_j = (w_{j1}, ..., w_{jn})$ é o ponto de máximo de B_j, e γ é o parâmetro de sensibilidade que regula a velocidade com que os valores de pertinência decrescem, assim como o aumento da distância entre o X_h e o B_j

A rede FMMNN estabelece que o tamanho máximo de uma categoria hiper-retangular B_j deve ser inferior a um parâmetro θ, com $0 \leq \theta \leq 1$ definido pelo usuário. Para um hiper-retângulo B_j ser expandido até incluir o padrão X_h, deve satisfazer a condição:

$$n\theta \geq \sum_{i=1}^{n}[max(w_{ji}, x_{hi}) - min(v_{ji}, x_{hi})] \tag{1.31}$$

Se o critério de expansão for satisfeito para B_j, o ponto de mínimo e o ponto de máximo do hiper-retângulo são ajustados, de modo similar a FAM com a aprendizagem rápida, pelas eqs. 1.32 e 1.33, respectivamente.

$$v_{ji}^{novo} = min(v_{ji}^{antigo}, x_{hi}) \qquad \forall i = 1, 2, \ldots, n \tag{1.32}$$

$$w_{ji}^{novo} = min(w_{ji}^{antigo}, x_{hi}) \qquad \forall i = 1, 2, \ldots, n \tag{1.33}$$

Para se determinar se esta expansão criou alguma sobreposição de hiper-retângulos (a qual não seria problema se representar a mesma classe), deve-se realizar uma comparação dimensão a dimensão entre os hiper-retângulos. Se, para cada dimensão, ao menos uma das quatro condições forem satisfeitas (casos de (1) a (4)), então existe sobreposição entre os dois hiper-retângulos. Durante este *teste de sobreposição*, a sobreposição mais pequena ao longo de qualquer dimensão e o índice da dimensão é armazenado para ser utilizada durante a contração dessa parte do processo de aprendizagem. Assumindo inicialmente que $\delta^{antigo} = 1$, os quatro testes de casos e o valor de sobreposição mínimo correspondente para a i-ésima dimensão são descritos como seguem:

caso (1): $v_{ji} < v_{ki} < w_{ji} < w_{ki}$,

$$\delta^{novo} = min(w_{ji} - v_{ki}, \delta^{antigo})$$

caso (2): $v_{ki} < v_{ji} < w_{ki} < w_{ji}$,

$$\delta^{novo} = min(w_{ki} - v_{ji}, \delta^{antigo})$$

caso (3): $v_{ji} < v_{ki} < w_{ki} < w_{ji}$,

$$\delta^{novo} = min(min(w_{ki} - v_{ji}, w_{ji} - v_{ki}), \delta^{antigo})$$

caso (4): $v_{ki} < v_{ji} < w_{ji} < w_{ki}$,

$$\delta^{novo} = min(min(w_{ji} - v_{ki}, w_{ki} - v_{ji}), \delta^{antigo})$$

Se $\delta^{antigo} - \delta^{novo} > 0$, então $\Delta = i$ e $\delta^{antigo} = \delta^{novo}$, significando que existia sobreposição para a Δ-ésima dimensão e o teste de sobreposição procederá para a próxima dimensão. Em caso contrário, o teste pára e a variável do índice de sobreposição mínima é definida para indicar que o próximo passo de contração não é necessário, ou seja, $\Delta = -1$.

Logo, no caso de $\Delta > 0$ (*etapa de contração*), as Δ-ésimas dimensões dos dois hiper-retângulos são ajustadas. Somente uma das n dimensões são ajustadas em cada um dos hiper-retângulos para manter o tamanho de hiper-retângulo o maior possível e, comprimir minimamente a forma dos hiper-retângulos que são formados. Para determinar o ajuste próprio a fazer, quatro casos são examinados:

caso (1): $v_{j\Delta} < v_{k\Delta} < w_{j\Delta} < w_{k\Delta}$,

$$w_{j\Delta}^{novo} = v_{k\Delta}^{novo} = \frac{w_{j\Delta}^{antigo} + v_{k\Delta}^{antigo}}{2}$$

caso (2): $v_{k\Delta} < v_{j\Delta} < w_{k\Delta} < w_{j\Delta}$,

$$w_{k\Delta}^{novo} = v_{j\Delta}^{novo} = \frac{w_{k\Delta}^{antigo} + v_{j\Delta}^{antigo}}{2}$$

caso (3a): $v_{j\Delta} < v_{k\Delta} < w_{k\Delta} < w_{j\Delta}$ e $(w_{k\Delta} - v_{j\Delta}) < (w_{j\Delta} - v_{k\Delta})$,

$$v_{j\Delta}^{novo} = w_{k\Delta}^{antigo}$$

caso (3b): $v_{j\Delta} < v_{k\Delta} < w_{k\Delta} < w_{j\Delta}$ e $(w_{k\Delta} - v_{j\Delta}) > (w_{j\Delta} - v_{k\Delta})$,

$$w_{j\Delta}^{novo} = v_{k\Delta}^{antigo}$$

caso (4a): $v_{k\Delta} < v_{j\Delta} < w_{j\Delta} < w_{k\Delta}$ e $(w_{k\Delta} - v_{j\Delta}) < (w_{j\Delta} - v_{k\Delta})$,

$$w_{k\Delta}^{novo} = v_{j\Delta}^{antigo}$$

caso (4b): $v_{k\Delta} < v_{j\Delta} < w_{j\Delta} < w_{k\Delta}$ e $(w_{k\Delta} - v_{j\Delta}) > (w_{j\Delta} - v_{k\Delta})$,

$$v_{k\Delta}^{novo} = w_{j\Delta}^{antigo}$$

A rede FMMNN foi aplicada em [140] na segmentação de imagens de faces, onde realiza a segmentação automática de regiões da pele por cores. Também destacamos o trabalho apresentado em [115], onde esta rede foi utilizada para reconhecimento de caracteres invariáveis em escala, rotação e translação.

1.3.4 *Gaussian* ARTMAP

Gaussian ARTMAP [144] (GAM) foi criada com o objetivo de solucionar algumas deficiências identificadas da rede *Fuzzy* ARTMAP (FAM), como sensibilidade ao ruído, e ineficiência na representação que as categorias internas realizam da geometria das predições de saída presentes nos dados. Esta rede é uma síntese de um classificador gaussiano e uma rede ART, obtida definindo a função de escolha

$T_j(\mathbf{I})$ como a função discriminante de um classificador gaussiano com distribuições separáveis, e a função de coincidência $m_J(\mathbf{I})$ da mesma forma, mas com as distribuições normalizadas pela unidade altura [134].

A representação geométrica da categoria muda de hiper-retângulo para hiper-elipsóide. O benefício desta inovação é que as funções de escolha e de coincidência, definidas pelas gaussianas, crescem monotonicamente até o valor médio que representa o centro da categoria, que deste modo previnem a proliferação de categorias causadas pelo ruído. Adicionalmente, as distribuições gaussianas são as mais comuns na natureza e possuem as propriedades de generalização, especialmente úteis para espaços de alta dimensão [46, 124, 125]. Outra vantagem é que usa distribuições gaussianas separáveis, que requerem somentes operações simples implementáveis em paralelo.

Uma categoria GAM pode facilmente ajustar os dados que variam independentemente em cada dimensão, mas não pode facilmente ajustar os dados que cubram duas dimensões. Como GAM deve representar a covariância, então cada categoria teria de armazenar uma matriz de covariância, e a classificação necessitaria do cálculo do determinante e a inversa desta matriz. Utilizando somente gaussianas separáveis, por outro lado, GAM tem exigências computacionais e de armazenamento similares as de FAM. Uma implicação desta limitação é –também válido para FAM– que GAM represente mais eficientemente os dados que não tenham correlação através das dimensões.

Cada categoria interna j em GAM é definida por um vetor $\mathbf{w}_j$, de dimensão M, representando sua média. Assim mesmo, a categoria conta com um vetor σ_j, cujos componentes σ_{ji} representam o seu desvio padrão em cada dimensão do espaço de entrada. Finalmente, também conta com um escalar n_j que representa o número de padrões de treinamento codificados pela categoria j (contador de categoria j). Então, cada categoria interna de *Gaussian* ART requer $2M + 1$ componentes para representar uma entrada $\mathbf{I} = (I_1, \ldots, I_M)$ de dimensão M (no caso de GAM, a codificação em complemento não é necessário).

Função de escolha

A função de escolha de *Gaussian* ARTMAP seleciona a categoria mais provável para um padrão de entrada durante a etapa de treinamento. A probabilidade de uma categoria é determinada pela probabilidade a que entrada pertence a sua distri-

buição, assim como pela probabilidade a priori da categoria. Pelo teorema de *Bayes* temos que a probabilidade posterior da categoria j dado o padrão de entrada $\mathbf{I}$ é:

$$P(j|\mathbf{I}) = \frac{p(\mathbf{I}|j)P(j)}{p(\mathbf{I})} \tag{1.34}$$

onde $P(\mathbf{I}|j)$ é a probabilidade a priori de $\mathbf{I}$ dada a classe (predição) j, $P(j)$ é a probabilidade a priori da classe j, e $P(\mathbf{I})$ é a probabilidade do padrão $\mathbf{I}$. Considerando cada categoria interna de GAM como uma variável randômica gaussiana [103] a função densidade da probabilidade condicional do padrão de entrada $\mathbf{I}$ dado a categoria j é definida por:

$$p(\mathbf{I}|j) = \frac{1}{(2\pi)^{M/2}\prod_{i=1}^{M}\sigma_{ji}} exp\left\{-\frac{1}{2}\sum_{i=1}^{M}\left(\frac{w_{ji}-I_i}{\sigma_{ji}}\right)^2\right\} \tag{1.35}$$

onde $\mathbf{w}_j$ é a média da categoria j e σ_{ji} é o desvio padrão da categoria j na dimensão i-ésima. A probabilidade a priori da categoria j é dada por:

$$P(j) = \frac{N_j}{\sum_{j=1}^{N_c} N_j} \tag{1.36}$$

onde N_c é o número de categorias, N_j é o número de padrões da categoria j-ésima e $\sum_j N_j$ o total de padrões. Finalmente, a função de escolha $T_j(\mathbf{I})$ da categoria j define-se a partir do logaritmo da probabilidade posterior $p(\mathbf{I}|j)$, ignorando a densidade $p(\mathbf{I})$ (o denominador da eq. 1.34), que se supõe normalmente $p(\mathbf{I}) = 1$, por ser constante para todas as categorias.

$$\begin{aligned} T_j(\mathbf{I}) &= log[(2\pi)^{M/2}p(\mathbf{I}|j)P(j)] \\ &= -\frac{1}{2}\sum_{i=1}^{M}\frac{(w_{ji}-I_i)^2}{\sigma_{ji}^2} - log\left(\prod_{i=1}^{M}\sigma_{ji}\right) + log(P(j)) \end{aligned} \tag{1.37}$$

onde $(2\pi)^{M/2}$ é o fator de escala dimensional. A categoria J, com a função de escolha $T_J(\mathbf{I})$ máxima, é selecionada como a candidata.

$$J = arg \max_{j=1,...,N} \{T_j(\mathbf{I})\} \tag{1.38}$$

Função de coincidência

A função de coincidência $m_J(\mathbf{I})$ da categoria ganhadora J é baseada na similaridade do padrão de entrada $\mathbf{I}$ com a forma da distribuição da categoria J, a qual é normalizada por uma unidade altura, segundo a equação:

$$\begin{aligned} m_J(\mathbf{I}) &= log\left[(2\pi)^{M/2} p(\mathbf{I}|J) \prod_{i=1}^{M} \sigma_{Ji}\right] \\ &= -\frac{1}{2} \sum_{i=1}^{M} \frac{(w_{Ji} - I_i)^2}{\sigma_{Ji}^2} \\ &= T_J(\mathbf{I}) + log\left(\prod_{i=1}^{M} \sigma_{Ji}\right) - log(P(J)) \end{aligned} \tag{1.39}$$

Se o valor de coincidência $m_J(\mathbf{I})$ da categoria J é inferior ao parâmetro de vigilância de GAM (ρ), então a categoria candidata é rejeitada. Ocorre ressonância quando $m_J(\mathbf{I}) > \rho$. Uma vez que uma categoria é rejeitada, ela é mantida inativa até a apresentação do próximo padrão de entrada. Se nenhuma categoria rotulada satisfaz a condição da vigilância, então uma categoria não-rotulada J', com $n_{J'} = 0$ é selecionada.

Aprendizagem

Quando uma categoria J alcança a ressonância com o padrão de entrada $\mathbf{I}$, são atualizados os três parâmetros (contador de padrões, média e desvio padrão) da categoria:

$$n_J = n_J + 1 \tag{1.40}$$

$$\mathbf{w}_J(n+1) = \left(1 - \frac{1}{n_J}\right) \mathbf{w}_J(n) + \frac{1}{n_J}\mathbf{I} \tag{1.41}$$

$$\sigma_{Ji} = \begin{cases} \sqrt{\left(1 - \frac{1}{n_J}\right)\sigma_{Ji}^2 + \frac{1}{n_J}(w_{Ji} - I_i)^2} & n_J > 1 \\ \gamma & n_J = 0 \end{cases} \tag{1.42}$$

onde se pode observar que a velocidade de aprendizagem depende do n_J e, portanto, modifica-se ao longo do treinamento, sendo superior ao princípio e próximo a 0 quando $n_J \to \infty$. Por sua vez, o parâmetro γ é o valor inicial do desvio padrão σ_{Ji}, que determina a extensão, igual em todas as dimensões do espaço de entrada, de uma categoria interna nova sobre seu primeiro padrão. A diferença das redes ART anteriores, onde $I_i \in [0, 1]$, GAM permite utilizar qualquer valor para as entradas. No entanto, os padrões de entrada devem ser pré-processsados, de modo que tenham desvios padrões iguais em cada dimensão.

Processamento

Depois do treinamento, o espaço de características está dividido em N_c hiperelipsóides, cujos eixos principais são paralelos aos eixos das coordenadas. Cada predição de saída pode estar associada à uma ou à várias categorias internas (hiperelipsóides). Durante o teste (ou validação), a saída de GAM é um vetor $\mathbf{y}(\mathbf{I}) = (y_1(\mathbf{I}), \ldots, y_{N_p}(\mathbf{I}))$, onde N_p é o número de predições de saída, e $y_k(\mathbf{I})$ é a probabilidade $P(k|\mathbf{I})$ estimada por GAM de que $\mathbf{I}$ pertença a predição k. Finalmente, GAM codifica $\mathbf{I}$ para a predição mais provável. Esta probabilidade é estimada pela soma das ativações de todas as categorias associadas à essa predição. Se denotamos por P_j a predição associada à categoria interna j, a probabilidade $P(k|\mathbf{I})$ da predição k pode ser expressa como:

$$y_k(\mathbf{I}) = P(k|\mathbf{I}) = \sum_{j:P_j=k} P(j|\mathbf{I}) = \sum_{j:P_j=k} e^{T_j(\mathbf{I})} \tag{1.43}$$

Conseqüentemente, GAM codifica o padrão de entrada para a predição K com probabilidade estimada máxima:

$$K = arg \max_{k=1,\ldots,N_p} \{y_k(\mathbf{I})\} \tag{1.44}$$

Se temos um comitê de "votação" integrado por V redes GAM e $y_k^v(\mathbf{I})$ designa à probabilidade estimada pela rede v para a predição k, então o veredito deste comitê

é a soma de todas as predições das categorias através dos diferentes membros do comitê:

$$K = arg \max_{k=1,\ldots,N_p} \left\{ \sum_{v=1}^{V} y_k^v(\mathbf{I}) \right\} \tag{1.45}$$

É interessante observar que se o treinamento tem como resultado que cada categoria codifica apenas um padrão, então GAM torna-se idêntico ao método das funções potencial, utilizando as distribuições gaussianas. Se o treinamento gera em cada predição de saída corresponder a apenas uma categoria interna, por outro lado, então, GAM torna-se idêntico a um classificador gaussiano com gaussianas separáveis. Portanto os extremos de GAM da compressão do código mínimo e máximo correspondem, respectivamente, à classificação das funções potenciais e à classificação gaussiana.

Na literatura encontramos várias aplicações para GAM, entre as quais destacamos em [70], onde foi avaliado o desempenho das redes ARTMAP, entre elas GAM, para a identificação de tipos de emissores de radar; assim como em [101] que analisou seu desempenho, entre outras redes ARTMAP, para a tarefa de classificação de imagens de satélite. Encontramos também em [114], onde foi utilizado como um algoritmo de classificação supervisionado para o mapeamento de cobertura de terra, utilizando dados indexados de vegetação multi-temporais aplicados à América Central.

1.3.5 FasArt

O modelo FasArt foi proposto [15, 17] usando a arquitetura geral de *Fuzzy* ARTMAP como sistema lógico difuso. Com este objetivo se introduz uma equivalência entre a função de escolha de categoria e uma função de pertinência ao conjunto difuso definido pela mesma. A eq. 1.46 calcula a função de escolha-pertinência $T_j(\mathbf{I})$ para a categoria j em F_2^a.

$$T_j(\mathbf{I}) = \prod_{i=1}^{M} \mu_{ji}(I_i) \tag{1.46}$$

onde $\mu_{ji}(I_i)$ é a função de pertinência do conjunto difuso j na dimensão i, aplicada

ao componente i do padrão de entrada, determinada por:

$$\mu_{ji}(I_i) = \begin{cases} \max\left\{0, \dfrac{\gamma(I_i - w_{ji}) + 1}{\gamma(c_{ji} - w_{ji}) + 1}\right\} & I_i \leq c_{ji} \\ \max\left\{0, \dfrac{\gamma(1 - I_i - w_{j,i+M}) + 1}{\gamma(1 - c_{ji} - w_{j,i+M}) + 1}\right\} & I_i > c_{ji} \end{cases} \tag{1.47}$$

sendo $\mathbf{w}_j$ o representante e $\mathbf{c}_j^a$ o centróide (ponto onde $\mu_{ji}(I_i)$ é máximo) do suporte do conjunto difuso da categoria j em F_2^a (analogamente para F_2^b). O parâmetro γ determina a largura do suporte do conjunto difuso (a mesma em todas as dimensões). O cálculo de $\mu_{ji}^a(I_i^a)$ e $\mu_{ji}^b(I_i^b)$ corresponde a avaliação do antecedente de um conjunto de regras difusas utilizando o produto como uma $T-norma$, de modo similar à outros sistemas neuro-difusos. Adicionalmente obtém a propriedade de derivabilidade, permitindo assim acentuar a aprendizagem através da minimização do erro de predição, como ocorre em fasBack [16].

Durante a etapa de processamento, a escolha de uma categoria k em F_2^b é induzida por regras do tipo:

IF (I_1 IS R_{j1}) AND ... AND (I_n IS R_{jn})
THEN (y_1 IS R_{k1}) AND ... AND(y_N IS R_{kn})

onde $\mathbf{I} = (\mathbf{a}, \mathbf{a}^c) = (a_1, \ldots, a_M, a_1^c, \ldots, a_M^c)$ e $\mathbf{y}$ são os padrões de entrada de ART_a e ART_b, respectivamente. Por sua vez, R_j e R_k são os conjuntos difusos associados às categorias j e k em F_2^a e F_2^b, respectivamente. Deste modo, durante o processamento, FasArt associa cada padrão de entrada $\mathbf{I}$ a uma saída $\mathbf{y} = (y_1, \ldots, y_M)$ dada pelo seguinte processo de *defuzzificação*, baseado na média dos centros dos conjuntos difusos [141].

$$y_m(\mathbf{I}) = \frac{\displaystyle\sum_{k=1}^{N_b}\sum_{j=1}^{N_a} c_{km}^b w_{jk}^{ab} T_j^a(\mathbf{I})}{\displaystyle\sum_{k=1}^{N_b}\sum_{j=1}^{N_a} w_{jk}^{ab} T_j^a(\mathbf{I})} \qquad m = 1, \ldots, M \tag{1.48}$$

onde N_a e N_b são os números de categorias internas ou unidades de F_2^a e F_2^b, respectivamente. Durante esta fase de processamento, FasArt se comporta como um sistema lógico difuso pelos seguintes passos: *fuzzificação* por um ponto, inferência através do produto, associação de conjuntos difusos de F_2^a a F_2^b, mediante F^{ab} e,

defuzzificação através da média dos centros dos conjuntos difusos.

Na fase de aprendizagem os pesos $\mathbf{w}_J^a$ e $\mathbf{c}_J^a$ associados à categoria J ganhadora em F_2^a (e analogamente para F_2^b) são atualizados através das eqs.(1.49, 1.50).

$$\mathbf{w}_J^a(n+1) = \beta(\mathbf{I} \wedge \mathbf{w}_J^a(n)) + (1-\beta)\mathbf{w}_J^a(n) \tag{1.49}$$

$$\mathbf{c}_J^a(n+1) = \beta^c\mathbf{I} + (1-\beta^c)\mathbf{c}_J^a(n) \tag{1.50}$$

onde β e β^c são as taxas de aprendizagem de $\mathbf{w}_J$ e $\mathbf{c}_J$, respectivamente. FasART mantém a lei de aprendizagem para $\mathbf{w}_j$ de *Fuzzy* ART, assim como a rejeição, os mecanismos do módulo Inter-ART de *reset* e o *Match-Tracking*. Além de preservar a estabilidade e cumprir o dilema de estabilidade-plasticidade, que são algumas das principais características da arquitetura ARTMAP, esta também é adequada à aprendizagem incremental.

A rede FasArt foi aplicada em [68] para o agrupamento e classificação num sistema de reconhecimento *on-line* de caracteres escritos à mão. Existem também trabalhos [120] nos que se analisa o seu comportamento em comparação com EAM e DAM, e se propõe uma versão com aprendizagem distribuída (dFasArt)

1.3.6 *Ellipsoid* ARTMAP

A rede *Ellipsoid* ARTMAP (EAM) [3] consiste de dois módulos ART interconectados através de um módulo inter-ART (conhecido como campo de mapeamento). Cada módulo ART é na verdade uma rede Ellipsoid ART (EA). EAM e EA são como generalizações da *Hyper-sphere* ARTMAP (HAM) e *Hyper-sphere* ART (HA) [2], respectivamente. Estas arquiteturas são baseadas nas idéias principais de *Fuzzy* ART e *Fuzzy* ARTMAP.

Funcionamento e arquitetura

Enquanto EA_a agrupa os padrões do espaço de entrada, EA_b agrupa padrões de um espaço de saída relacionado. Além disso, o *clustering* em cada módulo é

executado pelo agrupamento dos padrões similares em categorias EA. A informação que descreve as associações entre as categorias de entrada e de saída é codificada nos pesos w_{jk} do campo de mapeamento. Cada módulo consiste de duas camadas: a F_1 e a camada de representação F_2. Em contraste com FAM, EA/EAM falta uma camada de codificação F_0, devido a estas não necessitarem executar a codificação em complemento nos seus padrões de entrada. Ambas as camadas F_1 e F_2, consistem de um vetor de unidades que são inter-conectadas através das camadas pelos pesos *bottom-up* W_j e os pesos *top-down* w_j.

Antes de qualquer treinamento, todas as unidades de F_2, em ambos os módulos ART_a e ART_b são não-rotuladas, refletindo o fato que o sistema inicia com a memória em branco. Quando o treinamento inicia, os pares $(\mathbf{x}, \mathbf{y})$ de entrada-saída de valores reais de padrões de treinamento são apresentados em um momento. No modo de aprendizagem *on-line* cada padrão é apresentado para a rede uma única vez. Entretanto, durante a aprendizagem *off-line* cada padrão é apresentado repetidamente. Uma apresentação única do conjunto de treinamento completo constitui uma lista de apresentação (época). Então, a aprendizagem *off-line* pode envolver várias listas de apresentações.

Durante o progresso da fase de treinamento, algumas das unidades de F_2 podem já estar rotuladas e associadas aos grupos aprendidos. Na apresentação de um padrão $\mathbf{x}$ ao módulo ART_a, todas as unidades, rotuladas e não-rotuladas, competirão por este padrão através da função de escolha de categoria (FEC). Para uma unidade j rotulada a FEC é calculada pela eq. 1.51.

$$T(\mathbf{w}_j|\mathbf{x}) = \frac{D - R_j - max\{R_j, \|\mathbf{x} - \mathbf{m}_j\|_{C_j}\}}{D - 2R_j + a} \tag{1.51}$$

onde R_j é o tamanho do semi-eixo principal da categoria j, $a > 0$ é o parâmetro de escolha, $\mathbf{m}_j$ é o centro do hiper-elipsóide e $D > 0$ é diâmetro efetivo do módulo de domínio de entrada, calculado pela eq. 1.52.

$$D \geq \frac{1}{\mu} \max_{p,q} \{\|\mathbf{x}_p - \mathbf{x}_q\|_2\} \tag{1.52}$$

onde D é definido pelo menos igual ao diâmetro Euclidiano do domínio de entrada dividido por μ, quociente entre o eixo maior e os eixos menores do hiper-elipsóide, denominado parâmetro de proporção da elipse. Este parâmetro deve ser prefixado pelo usuário, e não é adaptativo, de modo que as proporções (excentricidades) de

todas as categorias hiper-elipsóidais são iguais. A restrição 1.52 serve para assegurar que os valores da função de critério de coincidência (FCC) (eq. 1.56) permaneçam positivos. Por outro lado, $\|\mathbf{x} - \mathbf{m}_j\|_{C_j}$ é a distância de *Mahalanobis*, ponderada L_2 pela matriz de peso C_j, e é calculada pela eq. 1.53.

$$\|\mathbf{x} - \mathbf{m}_j\|_{C_j} = \frac{1}{\mu}\sqrt{\|\mathbf{x} - \mathbf{m}_j\|_2^2 - (1-\mu^2)[\mathbf{d}_j(\mathbf{x} - \mathbf{m}_j)]^2} \tag{1.53}$$

onde $\mathbf{d}_j$ é o vetor diretor da categoria, i. e., um vetor na direção do eixo maior do hiper-elipsóide. As unidades não-rotuladas têm uma função de escolha:

$$T^u(\mathbf{w}_j|\mathbf{x}) = \frac{D}{2D\omega + a} \tag{1.54}$$

onde o parâmetro ω, também definido pelo usuário, deve ser escolhido de modo que $\omega \geq 0.5$ para garantir a estabilidade da rede. A categoria ou unidade J com função de escolha máxima torna-se ganhadora da competição *WTA*:

$$J = arg\max_j \{T(\mathbf{w}_j|\mathbf{x})\} \tag{1.55}$$

Em seguida, EAM mede o grau em que $\mathbf{x}$ coincide com as características da categoria correspondente a unidade J. Isto é estabelecido através do Teste de Vigilância (conforme eq. 1.57), o qual é um componente principal da aprendizagem baseada na coincidência de EAM. Primeiramente, o valor da função de coincidência da categoria (FCC) da unidade J é calculado. Se J é uma unidade rotulada, a função de coincidência define-se como:

$$\rho(\mathbf{w}_j|\mathbf{x}) = 1 - \frac{R_j + max\{R_j, \|\mathbf{x} - \mathbf{m}_j\|_{C_j}\}}{D} \tag{1.56}$$

Se J é uma unidade não-rotulada, a sua coincidência é $\rho(\mathbf{w}_j|\mathbf{x}) = 1$, o qual implica que qualquer padrão coincidirá a forma da categoria "virtual" associada com uma unidade não-rotulada. Em seguida, o valor da FCC é comparado ao valor base do parâmetro de vigilância do módulo ($\rho \in [0, 1)$) usando o teste de vigilância:

$$\rho(\mathbf{w}_j|\mathbf{x}) \geq \rho \tag{1.57}$$

Se esta condição não for satisfeita, ocorre uma rejeição da categoria J, durante a qual o valor da FCC de J é, temporariamente, igual a zero, e a competição ao longo da camada F_2 para $\mathbf{x}$ continua sem a participação de J.

Em caso de ressonância, EAM aprende a associação entre as categorias J ganhadora em ART_a e K em ART_b, e atualiza os centros e os eixos dos hiper-elipsóides segundo:

$$\mathbf{m}_J^{novo} = \mathbf{m}_J^{antigo} + \frac{\gamma}{2}\left[1 - \frac{min\{R_j^{antigo}, \|\mathbf{x} - \mathbf{m}_J^{antigo}\|_{C_J^{antigo}}\}}{\|\mathbf{x} - \mathbf{m}_J^{antigo}\|_{C_J^{antigo}}}\right](\mathbf{x} - \mathbf{m}_J^{antigo}) \quad (1.58)$$

$$R_J^{novo} = R_J^{antigo} + \frac{\gamma}{2}\left[max\left[R_J^{antigo}, \|\mathbf{x} - \mathbf{m}_J^{antigo}\|_{C_J^{antigo}}\right] - R_J^{antigo}\right] \quad (1.59)$$

onde $\gamma \in (0, 1]$ é a velocidade de aprendizagem (a aprendizagem rápida significa $\gamma = 1$). O vetor de orientação $\mathbf{d}_J$ é inicializado com o segundo padrão $\mathbf{x}$ codificado pela categoria, de acordo com a eq. 1.60, somente se a categoria J aprendeu um único padrão $\mathbf{m}_J$ anteriormente.

$$\mathbf{d}_J = \frac{\mathbf{x} - \mathbf{m}_J}{\|\mathbf{x} - \mathbf{m}_J\|_2} \qquad \mathbf{x} \neq \mathbf{m}_J \quad (1.60)$$

Após esta inicialização, $\mathbf{d}_J$ se manterá fixo em todo o momento da fase de treinamento. Finalmente, se J é uma unidade não-rotulada, então esta se tornará em rotulada e seu vetor de pesos é inicializado a $\mathbf{w}_J = \mathbf{x}$. Estas operações de aprendizagem são similares para as unidades de F_2 em ART_b.

EAM está orientada fundamentalmente para tarefas de classificação. Para isto, o conjunto de rótulos de categorias deve ser o seu domínio de saída, e seu parâmetro de vigilância de ART_b deve ser $\rho^b = 1$. Destacamos que EAM se reduz a outros classificadores com uma escolha apropriada de parâmetros da rede. Quando o treinamento com $\mu = \rho = 1$ e no teste com $\mu = 1, \rho = 0$ e $\omega \to \infty$, EAM torna-se um classificador do vizinho mais próximo (*Nearest Neighbor classifier*) de norma L_2 [46]. Também, quando $\mu = 1$ é usado na fase de treinamento e na de teste, EAM torna-se equivalente a rede HAM.

EAM aplicou-se em [145], para o agrupamento e classificação de tecidos, através da análises de dados de expressões genéticas gerados por experimentos de micro-tabelas de DNA.

1.3.7 *Semi-supervised Ellipsoid* ARTMAP

A rede *semi-supervised Ellipsoid* ARTMAP (ssEAM) [6] está baseada em EAM e tem como principal propriedade a de tolerar uma proporção predeterminada de erros na etapa de treinamento, para evitar problemas de super-aprendizagem. Concretamente, ssEAM permite o agrupamento numa mesma categoria de padrões de treinamento, não necessariamente pertencentes à sua predição associada. Isto é realizado modificando o teste de predição de EAM da seguinte maneira: uma categoria ganhadora J pode ser atualizada por um padrão de treinamento $\mathbf{x}$, mesmo se o rótulo de J não seja igual ao rótulo da categoria de $\mathbf{x}$, enquanto mantenha a desigualdade (eq. 1.61):

$$\frac{W_{J,I(J)}}{1+\sum_{i=1}^{N_c} W_{Ji}} \geq 1-\varepsilon \tag{1.61}$$

onde $0 \leq \varepsilon \leq 1$ é o parâmetro de ssEAM chamado "tolerância do erro de predição". O N_c representa o número das classes do problema a resolver, $W_{J,I(J)}$ é o número de padrões codificados por ssEAM pela categoria J com a mesma predição associada à $I(J)$ que a predição desejada para o padrão, e W_{Ji} é o número de vezes que a categoria J foi atualizada por um padrão de treinamento com predição desejada i. A eq. 1.61 garante que a percentagem dos padrões de treinamento associados por ssEAM, à categoria J com uma predição desejada I' distinta da predição $I(J)$ associada à categoria J não possa exceder $100\varepsilon\%$. Para $\varepsilon = 1$ o teste de predição modificado permitirá as categorias serem formadas pelo agrupamento dos padrões de treinamento, baseado nas funções de escolha e não nas suas predições desejadas, como se faria em uma rede não-supervisionada (daí o termo "*semi-supervised*"). Ao contrário, com $\varepsilon = 0$ o teste de predição modificado só permitirá o agrupamento numa categoria de padrões de treinamento com a sua mesma predição desejada, de um modo completamente supervisionado. Sob estas circunstâncias ssEAM torna-se equivalente ao EAM. Para valores intermediários de ε, o processo de formação de categoria é executado de um modo semi-supervisionado.

A fase de teste de ssEAM é similar à de EAM com apenas uma exceção. Se o treinamento foi executado utilizando somente o valor de $\varepsilon > 0$, é necessário converter as associações de $n - n$ (do campo de mapeamento) para $n - 1$. Isto devido que, usualmente, se deseja que a rede prediga um rótulo de categoria única para cada padrão de teste não-rotulado. Então, para cada categoria j em ART_a é necessário extrair primeiro seu rótulo de categoria dominante $D(j)$ para executar a fase de teste de ssEAM. Se ocorre que $D(j)$ não seja único para alguma categoria j a categoria fortemente prediz mais de um rótulo de categoria - uma categoria "confusa"), usualmente será descartada.

Devido a sua estrutura, ssEAM possui muitas características atrativas da aprendizagem, bastante desejáveis em tarefas de agrupamentos e classificação. Utilizando a aprendizagem rápida em modo *off-line*, a fase de treinamento finaliza-se em um pequeno número de passos. Outra importante característica de ssEAM é a capacidade de detectar padrões atípicos durante sua fase de treinamento ou de teste. Adicionalmente, através da utilização das categorias hiper-elipsóides, ssEAM pode aprender fronteiras de decisão complexas, que surge, freqüentemente, em problemas de classificação de expressões genéticas. Na literatura, destacamos como exemplo em [146], onde foi demonstrado sua capacidade de discriminar múltiplas categorias cancerígenas, com base nos seus perfis de expressão genética, através da análise de bases de dados de disponibilidade pública.

1.4 Aprendizagem Distribuída: Distributed ARTMAP

A rede *Distributed* ARTMAP (DAM) [36] foi derivada da rede *Fuzzy* ARTMAP (FAM) para executar a aprendizagem e teste distribuídos, ao invés do processo original *WTA* de FAM. DAM incorpora à sua rede supervisionada um módulo não supervisionado *Distributed* ART (dART) [19, 20]. O processo competitivo numa rede Fuzzy ART estabelece que uma única unidade permanece ativa e atua como candidata à ressonância, em cujo caso, codifica exclusivamente o padrão de entrada. No entanto, em *Distributed* ART (dART) temos que depois da competição todos as unidades se mantêm ativas, e só então elas codificarão o padrão, na proporção de sua ativação relativa. Contudo, a aprendizagem distribuída causa o esquecimento catastrófico, se esta é implementada na arquitetura original de FAM. Para introdu-

zir a aprendizagem distribuída e, conservar a aprendizagem rápida estável, algumas inovações foram introduzidas em na arquitetura FAM para resultar em DAM. Estas inovações incluem: os pesos dinâmicos, uma nova regra de memória endereçável de conteúdo (*Content-Addressable Memory* - CAM), um contador de instâncias (*instance counting*), atribuição de crédito e as leis de aprendizagem distribuída.

Os *pesos dinâmicos* substituem os pesos LTM de FAM. São funções LTMs, representados por um limiar de ativação e uma memória LTM, a qual é representada pela ativação da categoria atual. Geometricamente, em DAM os hiper-retângulos tem uma parte da LTM, dada pelos padrões aprendidos previamente, e uma parte da STM, dada pelo padrão atual e a ativação das outras unidades presentes na competição. Parte da LTM do hiper-retângulo, igual que em FAM, é apenas modificado quando ocorre ressonância, enquanto que a parte da STM é um alargamento do hiper-retângulo da LTM que é calculada para cada padrão de entrada determinado.

A *regra CAM* usada no algoritmo DAM é designada para realçar as diferenças da entrada como representadas na codificação interna distribuída sem aumentar os componentes de entrada a valores de p altos (p é o parâmetro de realce de contraste de DAM). Isto é chamado a regra CAM de gradiente aumentado, que também se emprega para definir a função de ativação estável em outras redes neuronais. O papel de p é análogo ao papel da variância nas funções de ativação gaussianas [88, 112].

Cada unidade (categoria interna) possui um contador de instâncias (*instance counting*), que guarda o número de padrões codificados pela mesma. O estado constante da ativação da unidade é pesado por este contador. Este processo é necessário devido ao método utilizado por DAM para fazer a predição distribuída para a categoria de saída. Para encontrar a predição de saída, DAM soma as ativações de todas as unidades associadas para cada categoria de saída e prediz aquela que obtém o valor da soma mais alta. O contador de instâncias é introduzido pelo fato de que, se cada unidade representa um padrão adicionado para a soma da predição, uma categoria de saída representada por poucas unidades, onde cada um unidade foi codificada muitas instâncias, facilmente seria superada por outra categoria de saída representada por muitas unidades, que tenham a codificação de poucas instâncias. Assim, o contador de instâncias ajuda a equilibrar este efeito. No entanto, o uso de *instance counting* poderia ser inconveniente em certas aplicações, onde o número de padrões de certas predições fosse muito baixos.

Enquanto a *atribuição de crédito* permite a aprendizagem realçar somente àquela

parte de um código ativo que está associado com o resultado correto. Este procedimento é semelhante ao algoritmo de atribuição de crédito, largamente utilizado em outras redes neurais (por exemplo em [144]) e algoritmos genéticos (por exemplo em [13]).

As *leis de aprendizagem distribuídas* de *instar* [20] e *outstar* [18] usadas em dART repartem, dinamicamente, as modificações aprendidas, segundo o grau de ativação de cada unidade de codificação, com a aprendizagem rápida e também com a lenta. As regras de atualização de DAM representam as soluções, na forma exata e fechadas, das equações diferenciais do modelo. Estas soluções são válidas através da escala de tempo, com a aprendizagem rápida ou lenta. Quando a codificação é *WTA*, as leis da aprendizagem distribuída se reduzem às equações *instar* e *outstar*, e dART se reduz à *Fuzzy* ART. Similarmente, quando a codificação é *WTA* durante o treinamento, mas distribuída durante o teste, o algoritmo DAM se reduz à ARTMAP-IC, e, além disso, se reduz à FAM quando a codificação é *WTA* durante ambas as etapas, treinamento e teste.

Distributed ART

Esta rede, de natureza não supervisionada dART [19, 20] usa umas leis de aprendizagem distribuídas *instar* e *outstar*, empregando também pesos adaptativos dinâmicos τ_{ij}, concebidos como limiares ao invés de fatores multiplicativos, para estabilizar a aprendizagem rápida com os códigos distribuídos. Apesar de sua arquitetura diferente, dART se reduz a *Fuzzy* ART, quando a codificação é *WTA*.

Seu funcionamento, conforme na fig. 1.9, se dá da seguinte forma: A camada F_2 recebe os sinais diretamente de uma entrada da camada F_0. O sinal T_j de $F_0 \longrightarrow F_2$ é uma função de um componente fásico S_j, que depende do padrão de entrada atual, e um componente tônico Θ_j, que por sua vez, é independente da entrada. A regra CAM (*Content-addressable Memory*) define a transformação de sinais $T_j(\mathbf{y})$ para o código $\mathbf{y}$ da camada F_2, que pode ser distribuída arbitrariamente. A atividade $\mathbf{x}$ de F_1 reflete uma coincidência entre a entrada $\mathbf{I}$ (*bottom-up*) e a entrada σ (*top-down*). O código ativo é rejeitado quando $\mathbf{x}$ falha em atingir o critério de coincidência de vigilância ρ. A LTM é armazenada como limiares τ_{ij} de $F_0 \longrightarrow F_2$, que se adaptam segundo uma lei de aprendizagem *instar* distribuída, e limiares τ_{ji} de $F_2 \longrightarrow F_1$, que se adaptam segundo uma lei de aprendizagem *outstar* distribuída.

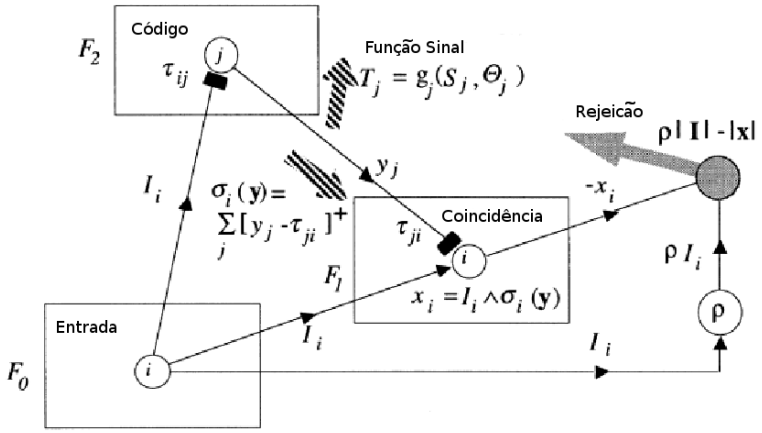


Figura 1.9: Arquitetura da rede *Distributed* ART (dART).

A arquitetura e funcionamento de DAM

DAM não é um sistema completamente distribuído, mas um sistema híbrido distribuído-*WTA*. Quando um padrão é apresentado, DAM tenta codificá-lo em modo distribuído. Se este código produz uma predição correta do padrão, procede a aprendizagem distribuída. No entanto, se a predição do padrão é incorreta, a rede comuta para o modo *WTA*, e sua operação passa a ser como a de FAM. Esta comutação para *WTA* é devido a falta de um método de inibição distribuído eficiente, ou seja, um mecanismo que penalize as unidades responsáveis da predição incorreta sem interromper a estabilidade do processo de aprendizagem.

Num caso geral, DAM aprende a predizer um vetor de saída arbitrário $\mathbf{b} = (b_1, \ldots, b_k, \ldots, b_L)$ dado um vetor de entrada $\mathbf{a} = (a_1, \ldots, a_i, \ldots, a_M)$. Considerando o componente $b_K = 1$, a entrada $\mathbf{a}$ e a categoria de saída como K. A simplificação do processo de busca, estabelecendo $\rho = 0$ converte o algoritmo ao tipo de sistema de aprendizagem competitiva distribuída.

Conforme a fig. 1.10, uma entrada $\mathbf{A}$ codificada em complemento ativa um código distribuído $\mathbf{y}$ em F_2, o qual por sua vez é filtrado através da contagem de pesos (*instance counting)* c_j para produzir a ativação $\mathbf{Y}$ em F_3. A competição *WTA* de F_0^{ab} ativa a unidade K' que recebe a maior entrada σ_k de F_3, representando a predição de saída proporcionada pela rede. Durante o treinamento, a ativação em F_1^{ab} determina se esta predição K' coincide com a predição desejada K, que se apresenta a DAM em F_0^b. Durante a aprendizagem dos pesos que conectam F_0^b e F_2, é realizada a atribuição de crédito. Um erro na camada F_1^{ab} gera um sinal de

Match-Tracking em $dART_a$ para aumentar a vigilância ρ somente o necessário para rejeitar o código ativo (a categoria escolhida).

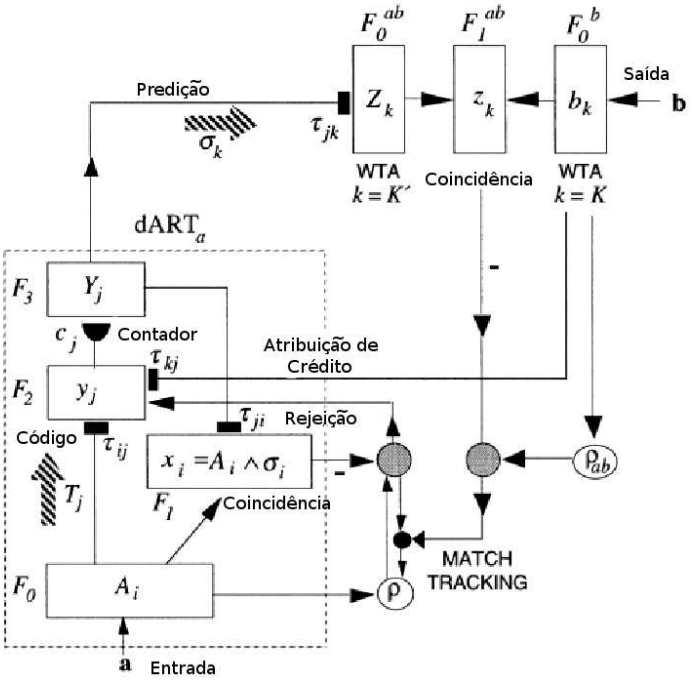


Figura 1.10: Arquitetura da rede *Distributed* ARTMAP (DAM).

Durante o treinamento de DAM, os pares de padrões de entrada $(\mathbf{a}_1, \mathbf{b}_1), \ldots, (\mathbf{a}_n, \mathbf{b}_n)$ são apresentados em igual intervalo de tempo. Ao início do treinamento, todas as variáveis LTM (limiar de $F_0 \longrightarrow F_2, \tau_{ij}$, contador de instâncias (*instance counting*) c_j, limiar de $F_3 \longrightarrow F_1, \tau_{ji}$) estão igual a zero. Na primeira iteração ($n = 1$) temos o vetor de entrada $A_i = a_i^1$ se $1 \leq i \leq M$ ou $A_i = 1 - a_i^1$ se $M + 1 \leq i \leq 2M$; ademais, $b_k = 0$, exceto $b_K^1 = 1$, onde K que é a predição desejada.

O valor de ativação que cada unidade alcança depois da apresentação de um padrão é calculado em duas etapas. A primeira, um valor de ativação temporária, $T_j(y_j = 1, \tau_{1j}, \tau_{2j}, \ldots, \tau_{2Mj})$ é calculada utilizando somente o limiar LTM e o padrão de entrada, conforme a eq. 1.63.

$$T_j(y_j) = S_j + (1 - \alpha)\Theta_j = \sum_{i=1}^{2M} A_i \wedge [y_j - \tau_{ij}]^+ + (1 - \alpha) \sum_{i=1}^{2M} \tau_{ij} \tag{1.62}$$

$$T_j(1) = \sum_{i=1}^{2M} A_i \wedge [1 - \tau_{ij}]^+ + (1 - \alpha) \sum_{i=1}^{2M} \tau_{ij} \tag{1.63}$$

onde $[x]^+ = x$ se $x > 0$ e, em caso contrário, $[x]^+ = 0$. O parâmetro α verifica $\alpha \in [0, 1]$ e $j = 1, \ldots, C$, onde C é o número de unidades rotuladas em F_2 (categorias internas). Inicialmente, $y_j = 1$ para todas as unidades em F_2. O S_j representa o componente fásico e Θ_j o componente tônico da função de escolha T_j. Então, as unidades de F_2 interagem de uma forma competitiva para encontrar o componente STM dos seus pesos dinâmicos (y_j). Esta competição determina o estado de equilíbrio da saída alcançada por cada unidade para o padrão atual, utilizando a regra CAM de gradiente aumentado (eq. 1.64).

$$y_j = f(T_1, \ldots, T_C) \equiv \frac{1}{1 + \sum_{\lambda \in \Lambda, \lambda \neq j} \left[\frac{(2-\alpha)M - T_j}{(2-\alpha)M - T_\lambda} \right]^p} \tag{1.64}$$

onde $p \geq 1$, e Λ, no modo distribuído, é o conjunto de índices das unidades ativadas ($T_j > T^u$, sendo $T^u = [T_j(1)]_{\tau_{ij}=0} = M$ pela eq. 1.63 a função de escolha das unidades não rotuladas) em F_2, pela regra CAM.

Se existe algum retângulo, cujo tamanho é exatamente um padrão (retângulo-ponto), que inclui o atual padrão de entrada, então $(2-\alpha)M - T_j = 0$, para algum j, e a saída será calculada pela eq.:

$$y_j = \begin{cases} \frac{1}{|\Lambda'|} & Se \quad j \in \Lambda' \\ 0 & \text{senão} \end{cases} \tag{1.65}$$

onde $\Lambda' = \{j \in \Lambda : (2-\alpha)M - T_j = 0\}$, ou seja, $|\Lambda'|$ é o número de retângulos ponto que incluem o padrão, e $|\Lambda'|$ é o seu cardinal (número de elementos).

DAM implementa uma estratégia de votação que considera a saída de cada unidade ativa após a competição. As saídas de todas as unidades ligadas à uma mesma categoria são somadas, e a rede classifica o padrão dentro da categoria com o somatório máximo. A camada F_3 em DAM é a responsável em equilibrar este processo. A sua saída é um vetor $\mathbf{Y}$ resultante da normalização da saída de F_2 ($\mathbf{y}$) pelo número de padrões previamente codificados por cada categoria, conforme eq. 1.66.

$$Y_j = \frac{c_j y_j}{\sum_{\lambda=1}^{C} y_\lambda c_\lambda} \qquad j = 1, \ldots, C \tag{1.66}$$

onde $c_j \in [0,1]$ é o crédito da unidade j em F_3, sua utilização favorece as categorias que captam mais padrões, que pode não ser adequado para certas aplicações. O vetor $\mathbf{Y}$ é utilizado para avaliar o critério de coincidência na camada F_1.

O representante da categoria é calculado como uma combinação dos pesos dinâmicos *top-down* das unidades ativas através da eq.:

$$\sigma_i = \sum_{j=1}^{C} [Y_j - \tau_{ji}]^+ \qquad i = 1, \ldots, 2M \tag{1.67}$$

A partir deste representante, o critério de coincidência (eq. 1.68) entre o padrão de entrada e o representante da categoria é o seguinte:

$$\frac{1}{M} \sum_{i=1}^{2M} A_i \wedge \sigma_i \geq \rho \tag{1.68}$$

Se este critério for satisfeito, a aprendizagem prossegue para o passo de predição. Em caso contrário, ocorre o disparo da rejeição, a operação da rede comuta para o modo *WTA* e a entrada de F_3 a F_1 será dada por σ_i, calculada pela eq.:

$$\sigma_i = (1 - \tau_{ji}) \tag{1.69}$$

Depois do padrão ser classificado corretamente no modo *WTA* (conforme a dinâmica FAM descrita na sec. 1.3.2, a rede comuta de volta para o modo distribuído e o próximo padrão é apresentado.

Se a rede está no modo distribuído, é calculado o sinal (σ_k) da camada $F_3 \longrightarrow F_0^{ab}$ conforme eq. 1.70:

$$\sigma_k = \begin{cases} \displaystyle\sum_{j=1,\kappa(j)=k}^{C} Y_j & Se \quad \kappa(j) = K \quad para \quad algum \quad j \\ 0 & \text{senão} \end{cases} \tag{1.70}$$

onde $\kappa(j)$ é a predição associada à categoria j, $k = 1, \ldots, L$, e L é o número de predições. A predição de saída será $K' = argmax_k\{\sigma_k\}$.

Em caso do modo *WTA*, o $K' = \kappa(J)$, onde J é a unidade ganhadora em F_3. Se $K \neq K'$ ocorre o *Match-Tracking*, a vigilância é modificada (eq. 1.71) e retorna ao modo *WTA* para recalcular y_j, Y_j, e σ_i.

$$\rho = \varepsilon + \frac{1}{M} \sum_{i=1}^{2M} A_i \wedge \sigma_i \tag{1.71}$$

Em caso contrário, se realiza a atribuição de crédito, o qual preserva a estabilidade da rede durante a aprendizagem. Neste caso, são normalizados novamente F_2 e F_3 (conforme eqs. 1.72, 1.73 e 1.66), além de recalcular o sinal σ_i (eq. 1.67). A vigilância atualiza-se segundo:

$$\overline{y}_j = \begin{cases} y_j & Se \quad \kappa(j) = K \\ 0 & \text{senão} \end{cases} \tag{1.72}$$

$$y_j = \frac{\overline{y}_j}{\sum_{\lambda=1}^{C} \overline{y}_\lambda} \tag{1.73}$$

Enfim, na ressonância são atualizados os pesos dinâmicos, ou seja, o limiar (*instar* distribuído) de $F_0 \longrightarrow F_2$ é incrementado através da eq. 1.74, e o limiar (*outstar* distribuído) de $F_3 \longrightarrow F_1$ é incrementado através da eq. 1.75. Além do mais, também atualiza-se o contador de instâncias (*instance counting*) de $F_2 \longrightarrow F_3$ através da eq. 1.76.

$$\tau_{ij}(n+1) = \tau_{ij}(n) + \beta[y_j - \tau_{ij}(n) - A_i]^+ \tag{1.74}$$

$$\tau_{ji}(n+1) = \tau_{ji}(n) + \beta \frac{[\sigma_i - A_i]^+}{\sigma_i} [Y_j - \tau_{ji}(n)]^+ \tag{1.75}$$

$$c_j(n+1) = c_j(n) + y_j \tag{1.76}$$

onde $i = 1, \ldots, 2M$ e $j = 1, \ldots, C$, β é a taxa de velocidade da aprendizagem, e igual que nas demais redes, quando $\beta = 1$ representa a aprendizagem rápida.

DAM foi empregado em [127], conjuntamente com FAM, para solucionar a tarefa de teste de qualidade na indústria de semi-condutores.

1.5 Geometria das categorias internas

Um dos principais problemas identificados na literatura nos modelos ART examinados é a "proliferação de categorias" [60, 65]. Este problema foi associado com a presença de ruído [107], e com a ineficiência da geometria das categorias internas [144], que requer um excessivo número de categorias para cobrir o espaço de entrada. A proliferação de categorias contribui à super-aprendizagem [138] nas redes ART, uma vez que a rede tende a criar um número de categorias comparável ao tamanho do conjunto de treinamento, e a validação cruzada foi sugerida como solução em [102]. De fato, uma das motivações do modelo DAM, e da sua aprendizagem distribuída, foi outra proposta para a atenuação deste problema. *Parrado-Hernández et al.* [120] analisou quantitativamente a influência da aprendizagem distribuída na proliferação de categorias, concluindo que a redução do número de categorias criadas por DAM em relação a FAM, depende da geometria das predições de saída dos conjuntos de dados. Especificamente, as geometrias não-retangulares permitem a DAM criar menos categorias que FAM, enquanto esta redução não acontece em conjuntos de dados com geometrias internas retangulares. Este fato sugere que o número de categorias criadas está relacionado, de certa forma, com a capacidade da rede para aprender as fronteiras entre as predições. A rede cria menos categorias, quando a forma geométrica das suas categorias internas se ajusta melhor às suas fronteiras entre as predições. Ao contrário, quando as categorias internas criadas pela rede não ajustam estas fronteiras tão bem, a rede necessita mais categorias. Se a forma da categoria é predefinida, como ocorre em todos os classificadores existentes baseados em ART, a taxa de erro e o número de categorias dependerá da forma geométrica das fronteiras.

A resposta de uma categoria interna C_i para um padrão de entrada $\mathbf{I}$ é dada por sua Função de Escolha de Categoria (*Category Choice Function*-FEC) T_i, que depende de $\mathbf{I}$ e também do representante $\mathbf{w}_i$ da categoria, e define a geometria da Região de Representação da Categoria (*Category Representation Region* - RRC). A forma geométrica, como vimos nas seções anteriores, difere entre os distintos modelos ART. GAM utiliza a FEC com simetria hiper-elipsoidal. FAM e DAM é hiper-retangular, enquanto HAM [2] e EAM utilizam hiper-esféricas e hiper-elipsóides, respectivamente. Apesar de utilizar RRCs não-hiper-retângular, a geometria da categoria em HAM e EAM é também pré-estabelecida (esférica ou elipsóide) pela FEC. A FEC é constante dentro da RRC, ainda que alguns modelos ART (FasART (sec. 1.3.5) e AFC [128]) usam FECs que variam dentro da RRC. A rede cria ca-

tegorias de maneira que suas fronteiras aproximam as fronteiras entre as predições de saída do conjunto de dados. Esta aproximação é mais precisa, e requer menos categorias, se as geometrias das categorias e as predições de saída são similares. Por exemplo, FAM e DAM, provavelmente, necessitam mais categorias que EAM ou HAM para alcançar um desempenho igual com os conjuntos de dados de geometria circular. Por outro lado, EAM e HAM usam categorias com geometrias circulares e elípticas, respectivamente, de modo que se adaptarão melhor à geometria circular dos dados. Portanto, a forma geométrica das categorias internas pode introduzir uma dependência nas redes ART tradicionais em relação à geometria dos dados.

A fim de suprimir esta dependência, o trabalho que apresentamos nesta tese de doutorado explora a utilização de categorias com forma geométrica não predefinida. O objetivo é que a própria rede ART seja capaz de aprender a sua geometria, durante a aprendizagem, a partir dos próprios padrões de treinamento. Deste modo, dota-se à rede ART de uma representação de categorias potencialmente mais rica, em potência expressiva, que à proporcionada pelas redes ART existentes. Seria esperável que este grau de flexibilidade adicional redundasse em uma maior capacidade para aprender fronteiras entre as predições com formas geométricas arbitrárias, e portanto, se obtivesse menos erro de classificação com um número também menor de categorias. Também seria esperável uma menor dependência dos resultados em relação às características geométricas do conjunto de dados, uma vez que, usando geometrias genéricas, não haveria conjuntos de dados com geometrias mais (ou menos) adequadas à aprendizagem.

Nas redes ART existentes, exceto DAM, e no caso da "aprendizagem rápida" (utilizada usualmente), a categoria ressonante aprende um padrão de entrada se expandindo até o mesmo, enquanto mantém a sua geometria predefinida. Uma categoria C_i com predição associada $P(C_i)$ expande-se não somente até o padrão de entrada, mas também em outras direções, porque não pode mudar a sua forma geométrica para ocupar somente a região entre a categoria e o padrão (fig. 1.11). Depois da expansão, a categoria C_i pode abranger regiões populadas por padrões com predições desejadas P_d distintas a $P(C_i)$. De fato, o volume das regiões cobertas pela categoria em direções distintas a do padrão de treinamento, incrementa com a dimensão do espaço de entrada (fig. 1.11, painéis (a) e (b)). Então, quando um padrão de treinamento com predição desejada $P_d \neq P(C_i)$ se encontra dentro destas regiões, o *Match-Tracking* é invocado, de modo que a categoria C_i é rejeitada. Além do mais, a rede seleciona outra categoria C_j que, no caso de superar o teste de vigilância e ter

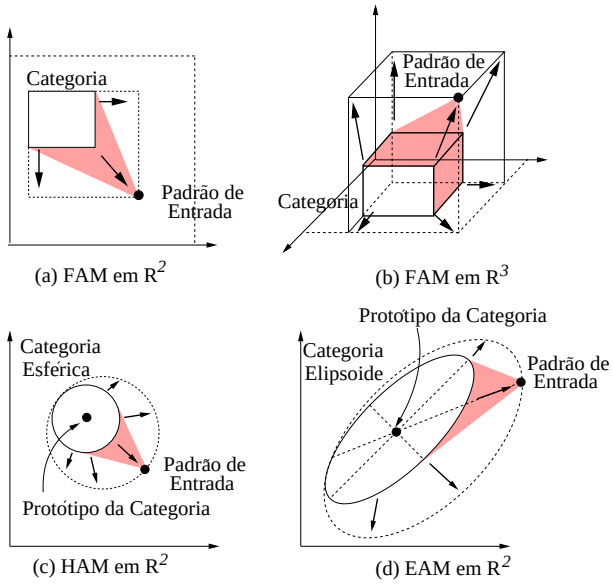


Figura 1.11: A aprendizagem rápida pela expansão de categoria por FAM em $\mathbb{R}^2$ (a) e $\mathbb{R}^3$ (b), HAM em $\mathbb{R}^2$ (c) e EAM em $\mathbb{R}^2$ (d). A linha tracejada é a categoria nova depois da aprendizagem. Esta categoria inclui regiões que não estão na direção (sombreada na figura) da categoria antiga ao padrão de entrada. A comparação de FAM em $\mathbb{R}^2$ e $\mathbb{R}^3$ mostra que o volume destas regiões crescem com a dimensão do espaço de entrada.

uma predição associada $P(C_j) = P_d$, obterá a ressonância com o padrão de entrada e, depois da aprendizagem, cobrirá aquela região. Alternativamente, pode ocorrer que se crie uma categoria nova C_j, no caso de que nenhuma categoria cumpra o teste de vigilância e tenha a predição desejada P_d. Porém, a categoria C_i não é atualizada, então C_i e C_j são sobrepostas. Em conseqüência, como as categorias ART podem cobrir regiões populadas por padrões com outras predições, elas necessitam se sobrepôr para serem expandidas até o padrão de entrada, enquanto mantém sua geometria predefinida (fig. 1.12). A rede *Fuzzy* Min-Max (FMMNN) [130] (sec. 1.3.3) é uma exceção notável, porque utiliza as categorias com a geometria predefinida (hiper-retângulo) que não realiza sobreposição se elas têm diferentes predições associadas. No entanto, FMMNN utiliza as técnicas geométricas para testar a sobreposição da categoria e, neste caso, a categoria expandida é contraída. Por isso, quando uma categoria em FMMNN tenta ser expandida até o padrão de entrada, a expansão pode não ser possível se a forma do hiper-retângulo obriga a cobrir regiões cobertas por caregorias com outra predição e, portanto, populadas por padrões de outras

predições.

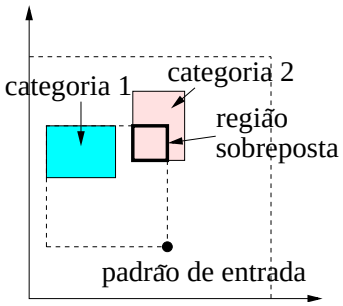


Figura 1.12: Aprendizagem rápida de FAM em $\mathbb{R}^2$. Quando a categoria 1 aprende o padrão de entrada, enquanto mantém sua forma de hiper-retângulo, ela sobrepõe-se com a categoria 2, que tem uma predição associada diferente.

Se, durante a etapa de processamento, um padrão de entrada cai dentro da região sobreposta, as categorias com menor tamanho (as mais específicas) são candidatas para ressonar em primeiro lugar, devido que nas redes ART existentes a FEC é decrescente com o tamanho da categoria (subsecção 1.3.2). Então, o tamanho da categoria é usada para selecioná-las entre as várias categorias sobrepostas. Como as categorias podem se sobrepôr, e podem cobrir regiões do espaço de entrada com predições desejadas diferentes, algumas categorias podem cobrir, completamente, o espaço de entrada durante o treinamento, reduzindo assim a capacidade da rede para discriminar entre as predições. Para impedir esta possibilidade, deve ser utilizado um limite superior para o tamanho da categoria, definido pelo parâmetro de vigilância $0 \leq \bar{\rho} < 1$, (Usando $\bar{\rho} = 0$ se permite que qualquer categoria possa cobrir todo o espaço de entrada)[9]. No caso de FAM, o tamanho R_i da categoria C_i deve satisfazer $R_i \leq M(1 - \rho)$ (Teste de vigilância). Se este tamanho, depois da expansão da categoria, é maior que este limite, então a categoria é rejeitada, e outra categoria é selecionada como a ganhadora. Os valores baixos para ρ permitem hiper-retângulos grandes (categorias muito genéricas) e os valores altos obriga a criar hiper-retângulos pequenos (categorias muito específicas). Infelizmente, não existe uma maneira automática de calcular o limite superior do tamanho das categorias internas num conjunto de dados determinado, exceto em exemplos muito simples. Portanto, o parâmetro de vigilância ρ deve ser otimizado, para cada conjunto de dados, por tentativa e erro. De modo similar à otimização dos outros parâmetros

[9]No entanto o valor $\bar{\rho} = 0$ é geralmente usado para tarefas de classificação, a vigilância é incrementada a valores positivos quando o *Match-Tracking* é invocado. Assim existe um tamanho de categoria máximo até a apresentação do padrão de treinamento seguinte.

ajustáveis nos outros algoritmos, geralmente se utilizam técnicas de validação cruzada para determinar o valor de ρ, de modo que x minimize o erro obtido sobre um certo conjunto de treinamento. No caso da rede FMMNN, o teste de sobreposição limita a expansão da categoria, mas também é usado um tamanho máximo de categoria, dado por um parâmetro similar à vigilância. Ambos os testes, de sobreposição e de tamanho máximo de categoria, parecem ter o mesmo objetivo, ou seja, limitar a expansão da categoria, portanto eles são, de alguma maneira, redundantes. Além do mais, FMMNN poderia trabalhar sem vigilância, utilizando somente o teste de sobreposição para limitar a expansão da categoria.

Podemos concluir, a partir das considerações anteriores, que a geometria predefinida está relacionada com outras características típicas das redes ART, tais como sobreposição entre as categorias, tamanho máximo de categoria e vigilância. As categorias com geometria predefinida devem sobrepôr-se, porque para evitar a sobreposição deveriam mudar a sua forma geométrica. Entretanto, na aprendizagem de um padrão de treinamento uma categoria genérica pode mudar a sua geometria para ser expandida somente na direção do padrão. Deste modo, uma categoria com geometria genérica pode ser expandida cobrindo somente as regiões do espaço de entrada populadas por padrões com a sua mesma predição de saída[10], e portanto não é necessário que se sobreponham entre elas. De fato, é, de certa forma, contraditório usar categorias com geometrias genéricas que possam se sobrepôr, porque a sobreposição reduz as suas vantagens geométricas para aproximar fronteiras arbitrárias entre as predições. Por estas razões, alguns modelos ART que propomos nesta tese de doutorado não permitem a sobreposição entre as categorias. Como veremos, a não sobreposição entre as categorias se pode empregar como critério limitador de sua expansão, evitando a necessidade de um tamanho máximo e, consequëntemente, de um parâmetro de vigilância. Este fato é de grande relevância, uma vez que suprime o principal parâmetro ajustaável das redes ART, com grande impacto nos seus resultados. Isto permite propôr redes ART sem parâmetros e com uma operação completamente automática.

[10]Se existem padrões com uma predição diferente entre a categoria e o padrão de entrada, a categoria não pode ser expandida.

1.6 Organização da tese

Os seguintes capítulos apresentam e discutem vários modelos baseados en ART que usan categorias internas con geometrias non pre-definidas. O primeiro deles, *Simplex* ARTMAP (SAM), descrito no capítulo 2, utiliza uma descrição de categorias baseada em simplexes definidos por padrões de treinamento selecionados, e constrói regiões com geometrias genéricas como conjuntos de simplexes adjacentes. As limitações na eficiência deste modelo nos levaram a propôr o modelo *Poly-Tope* ARTMAP (PTAM), discutido no capítulo 3, onde cada categoria tem forma geométrica de politopo irregular e, portanto, completamente genérico nas suas capacidades para aproximar as fronteiras arbitrárias entre as predições. Ambos os modelos usam categorias não sobrepostas, de modo que não incluem o tamanho da categoria nas funções de escolha, nem também o parâmetro de vigilância. Uma questão imediata surge neste ponto, associada à relação entre a geometria irregular para as categorias, e a sobreposição entre as categorias: que ocorre se as categorias são irregulares, porém podem sobrepôr-se? A resposta se encontra no capítulo 4, onde se propõe, discute e avalia o modelo *Overlapping Polytope* ARTMAP (OPTAM) como alternativa de PTAM, o qual permite a sobreposição entre as categorias e, portanto, usa o parâmetro de vigilância. Finalmente, o capítulo de conclusões recompila as principais contribuições da tese, e as linhas de trabalho futuras.

Capítulo 2

Simplex ARTMAP

A definição de uma forma geométrica genérica para as categorias internas não pode se basear num único vetor, como ocorre nas redes ART (valor esperado, representante da categoria, intervalos de padrões codificados pela categoria, etc.) que usam geometrias predefinidas. Ao contrário, para evitar uma forma geométrica predefinida, é necessário que os próprios padrões de treinamento, ou ao menos uma parte deles, definam o contorno (fronteira) da categoria, durante o processo de treinamento. Uma segunda questão, é relativa a forma dessa fronteira entre os dois vértices da categoria, que poderia ser linear– neste caso, a categoria seria um politopo– ou não linear (uma curva). No entanto, esta última possibilidade, que se poderia implementar usando hiper-superfícies não lineares (por exemplo, de tipo polinômico) para modelar a fronteira entre as categorias, exigiria o uso de funções complexas, e por simplicidade consideramos somente a primeira opção (fronteira linear por partes, e portanto, categorias com geometria politopo). Neste caso, estes padrões selecionados atuariam como vértices de fronteira de categoria.

Como se comentou no capítulo 1, as redes ART usam funções de escolha T_j que assumem o seu valor máximo[1] dentro do volume ocupado pela categoria. Este valor é decrescente com a distância à mesma. Como a nossa proposta deveria ser análoga, a função de escolha deve assumir valores elevados dentro do volume do politopo, e decrescentes com a distância fora dele. Entretanto, somente é factível avaliar diretamente se o padrão de entrada se encontra dentro ou fora de um politopo, no

[1]Dado que, geralmente, estas redes usam o tamanho da categoria como critério para selecionar a categoria ganhadora, quando o padrão cai dentro de várias categorias sobrepostas, o valor de T_j dependerá também deste tamanho.

caso de politopos convexos, e a sua complexidade computacional é elevada. Temos que considerar que todo politopo pode se decompôr em um conjunto de simplexes[2] adjacentes, não sobrepostos, o qual tem a vantagem de que a função de escolha para um simplex é menos complexa, já que o simplex é convexo. Por outro lado, a aprendizagem rápida nas redes ART consiste na expansão da categoria (hiper-retangular, hiper-elipsóide, ...) até o padrão de treinamento. No caso das categorias politopo, consideradas como conjuntos de simplexes, a expansão se pode realizar de modo simples adicionando um simplex novo entre o politopo e o padrão de treinamento. Deste modo, se garante que a categoria somente se expande na direção deste padrão, diferentemente das redes ART convencionais (seção 1.5).

Com base nestas considerações, a nossa primeira aproximação usa categorias com forma de simplex, e por esta razão, se denomina *Simplex* ARTMAP (SAM) [61]. SAM aproxima a região do espaço de entrada associada à uma predição de saída mediante categorias simplexes adjacentes, não sobrepostas. Esta aproximação não proporciona realmente categorias com geometria genérica, dado que estas categorias são simplexes ao invés de politopos. Entretanto, a adição de simplexes permite construir politopos com formas geométricas arbitrárias para as predições de saída, definidas por vértices que são padrões de treinamento selecionados, como veremos posteriormente.

De modo similar a FMMNN, SAM permite a sobreposição entre as categorias internas, se estas possuem a mesma predição (não permite, portanto, sobreposição entre as predições). Este fato é garantido através do "teste de sobreposição de predições", que é executado após a expansão da categoria, como abordaremos em detalhes na seção 2.1.4. Este teste de sobreposição constitui um dos principais elementos que define SAM, uma vez que para a expansão da categoria, neste modelo, não é necessário o uso de nenhum parâmetro adicional, como o de vigilância, comentado na seção 1.5.

As subseções descritas a seguir descrevem os principais passos das etapas de treinamento e de processamento.

[2]Um simplex ou hiper-tetraedro é o volume fechado em $\mathbb{R}^n$ com fronteiras lineares (hiperplanos), delimitado pelo menor número possível de vértices $(n+1)$. Um simplex em $\mathbb{R}^2$ é um triângulo, e em $\mathbb{R}^3$ é um tetraedro.

2.1 Etapa de Treinamento

Tabela 2.1: Nomenclatura associada à *Simplex* ARTMAP.

n	Dimensão do padrão de entrada $\mathbf{I}$
$\mathbf{I}$	Padrão de entrada, $\mathbf{I} = (I_1, \ldots, I_n) \in [0,1]^n$
	sem codificação em complemento
P_d	Predição desejada para o padrão de entrada $\mathbf{I}$
N_c	Número de categorias
C_i	i-ésima categoria, $i = 1, \ldots, N_c$
$P(C_i)$	Predição associada da categoria C_i
N_i	Número de vetores da categoria C_i
$\mathbf{w}_{ij}$	Vetor de peso (vértice) j-ésimo
	da categoria C_i, $j = 1, \ldots, N_i$
$T_i^t(\mathbf{I})$	Função de Escolha da Categoria (FEC) C_i,
	durante o treinamento
$T_i^p(\mathbf{I})$	Função da Escolha de Categoria (FEC) C_i,
	durante o processamento
$\mathbf{c}_i(\mathbf{p}_i)$	Centro da função Gaussiana dentro da categoria C_i
	dada pelos índices $\mathbf{p} = (p_{i2}, \ldots, p_{in}) \in \mathbb{N}^{N_i - 1}$
α_{ik}	Número de funções Gaussianas na direção do vetor $\mathbf{w}_{ik} - \mathbf{w}_{i1}$
σ	Desvio da função Gaussiana
$A_i(\alpha_i)$	Conjunto de vetores da categoria C_i
Γ	Limiar para as funções de escolha de categoria (FEC)
r_i	Estado de rejeição da categoria C_i
	($r_i = 1$, se a C_i é rejeitada, senão, $r_i = 0$)

A tabela 2.1 resume a nomeclatura que usaremos na formulação de SAM. Cada categoria C_i é definida pela sua predição associada $P(C_i)$ junto com $1 \leq N_i \leq n+1$ vetores $\mathbf{w}_{ij}, j = 1, \ldots, N_i$. No momento da sua criação, uma categoria tem um único vetor, igual ao padrão de entrada ($N_i = 1, \mathbf{w}_{i1} = \mathbf{I}$) que provoca a criação da categoria. À medida que a C_i codifica os padrões de entrada, o N_i cresce, e a categoria se converte em um simplex, quando $N_i = n + 1$. Uma categoria C_i, portanto, pode ter menos de $n + 1$ vetores, mas não mais.

2.1.1 Função de escolha de categoria

Analogamente às redes ART clássicas, a função de escolha $T_i^t(\mathbf{I})$ da categoria C_i durante o treinamento assume o valor $T_i^t(\mathbf{I}) \lesssim 1$, se o padrão de entrada cai dentro da categoria, e $T_i^t(\mathbf{I}) \gtrsim 0$, em caso contrário. Com este objetivo, para cada categoria C_i, SAM determina se o padrão de entrada $\mathbf{I}$ está dentro ou fora de seu simplex, mediante um somatório de funções gaussianas centradas em pontos interiores do simplex. Este somatório fornece valores não-nulos, somente dentro do simplex e valores zero, fora do simplex. Cada função gaussiana é dada pela seguinte expressão:

$$f(\mathbf{I}, \mathbf{c}_i(\mathbf{p}_i)) = exp\left(-\frac{\|\mathbf{I} - \mathbf{c}_i(\mathbf{p}_i)\|^2}{2\sigma^2}\right) \tag{2.1}$$

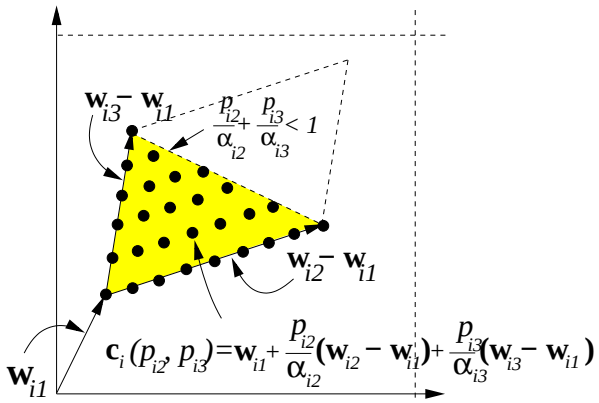


Figura 2.1: A categoria simplex em $\mathbb{R}^2$ e os centros $C_i(p_{i1}, p_{i2})$ das funções gaussianas.

O parâmetro σ é o desvio da função gaussiana. O seu centro é o vetor $\mathbf{c}_i(\mathbf{p}_i)$, dado pelo vetor de índices inteiros $\mathbf{p}_i = (p_{i2}, \ldots, p_{iN_i}) \in \mathbb{N}^{N_i-1}$ ($\mathbb{N}$ é o conjunto dos números naturais). A fig. 2.1 mostra estes centros $\mathbf{c}_i(\mathbf{p}_i)$, num exemplo em $\mathbb{R}^2$, os quais são calculados seguindo a expressão:

$$\mathbf{c}_i(\mathbf{p}_i) = \mathbf{w}_{i1} + \sum_{k=2}^{N_i} \frac{p_{ik}}{\alpha_{ik}}(\mathbf{w}_{ik} - \mathbf{w}_{i1}) \tag{2.2}$$

$$\mathbf{p}_i = (p_{i2}, \ldots, p_{iN_i}) \in \mathbb{N}^{N_i-1} \tag{2.3}$$

onde o $\mathbf{w}_{ik}$ é o vetor de peso (vértice) k-ésimo da categoria C_i, e o α_{ik} é o número de funções gaussianas na direção do vetor $\mathbf{w}_{ik} - \mathbf{w}_{i1}, k = 2, \ldots, N_i$, definido pela equação:

$$\alpha_{ik} = \left\lceil \frac{\|\mathbf{w}_{ik} - \mathbf{w}_{i1}\|}{2\sigma\sqrt{2\log 2}} \right\rceil, \quad k = 2, \ldots, N_i \tag{2.4}$$

Nesta expressão, $\lceil x \rceil$ designa a parte inteira por excesso (o menor inteiro maior ou igual a x). Esta definição de α_{ik} se deve impôr que, cada função gaussiana, tenha o valor $f(\mathbf{I}, \mathbf{c}_i(\mathbf{p}_i)) = 1/2$ no ponto médio entre os dois centros das funções gaussianas vizinhos na direção do vetor $\mathbf{w}_{ik} - \mathbf{w}_{i1}$.

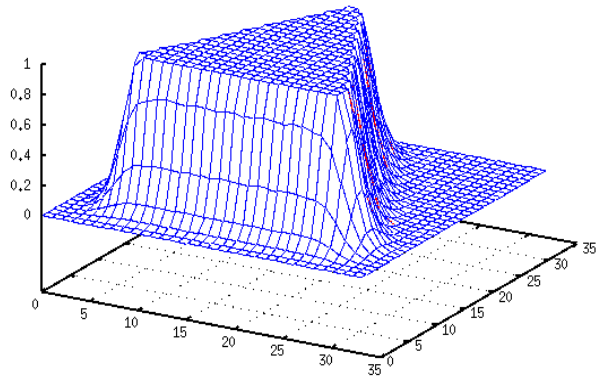
A função de escolha de categoria $T_i^t(\mathbf{I})$ para a categoria C_i, durante o treinamento, é dada pela eq. 2.5, apresentado na fig. 2.2 para $\mathbb{R}^2$. Os vetores $\mathbf{p}_i$ determinam os centros das funções gaussianas $\mathbf{c}_i(\mathbf{p}_i)$ dentro do simplex (fig. 2.1). Os vetores de índices $\mathbf{p}_i$, que geram os centros de funções gaussianas dentro do simplex da C_i de vértices $\mathbf{w}_{i1}, \ldots, \mathbf{w}_{i(n+1)}$, são os pertencentes ao conjunto $A_i(\alpha_i)$, definidos pela eq. 2.6 (ver fig. 2.1).

$$T_i^t(\mathbf{I}) = min\left[1, \sum_{\mathbf{p}_i \in A_i(\alpha_i)} exp\left(-\frac{\|\mathbf{I} - \mathbf{c}_i(\mathbf{p}_i)\|^2}{2\sigma^2}\right)\right]$$
$$\alpha_i = (\alpha_2, \ldots, \alpha_{N_j}) \tag{2.5}$$

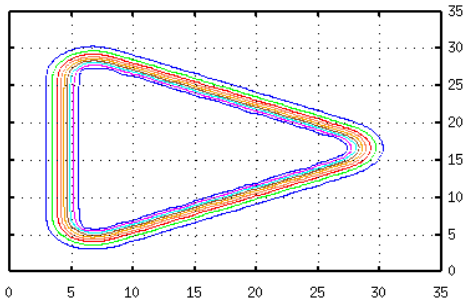
$$A_i(\alpha_i) = \Bigg\{\mathbf{p}_i \in \mathbb{N}^{N_i - 1} : p_{ik} = 0, \ldots, \alpha_{ik}; k = 2, \ldots, N_i;$$
$$\frac{p_{il}}{\alpha_{il}} + \frac{p_{im}}{\alpha_{im}} \leq 1, \forall l, m = 2, \ldots, N_i; l \neq m\Bigg\} \tag{2.6}$$

2.1.2 Competição

Similarmente às redes ART, SAM seleciona a categoria C_I, não rejeitada ($r_i = 0$), com a função de escolha $T_i^t(\mathbf{I})$ máxima. Porém, dado que as funções de escolha de SAM assumem valores quase-nulos fora das categorias, é possível que $T_i^t(\mathbf{I}) = 0, \forall i$. Nesta situação, não existiria nenhuma categoria ganhadora e, portanto, o $I = -1$. Para discriminar este tipo de situação, a categoria ganhadora C_I deve satisfazer a condição $T_I^t(\mathbf{I}) > \Gamma$, sendo o $\Gamma \gtrsim 0$ um valor mínimo necessário para



(a)



(b)

Figura 2.2: Exemplo da função de escolha de categoria $T_i^t(\mathbf{I})$ (a) e o seu contorno (b) para uma categoria simplex em $\mathbb{R}^2$.

que uma categoria seja ganhadora. Pode ocorrer também que todas as categorias se encontrem rejeitadas ($r_i = 1, \forall i = 1, \ldots, N_c$), e assim, também não existiria nenhuma categoria ganhadora (neste caso, igualmente o $I = -1$). Em ambos os casos, SAM aprende o padrão de entrada mediante a expansão de uma categoria existente, ou mediante a criação de uma categoria nova (subseção 2.1.4). Portanto, o índice I da categoria ganhadora é dado por:

$$I = \begin{cases} arg \max\limits_{i=1,\ldots,N_c} \{T_i^t(\mathbf{I})\} & \exists C_i : T_i^t(\mathbf{I}) > \Gamma, r_i = 0 \\ -1 & \text{, em caso contrário} \end{cases} \tag{2.7}$$

2.1.3 Teste de predição

Se a predição $P(C_I)$ associada à categoria ganhadora C_I é distinta da predição desejada P_d, para o padrão de treinamento $\mathbf{I}$, então este padrão cai dentro de uma categoria com uma predição incorreta. Para corrigir este erro na aprendizagem, a categoria C_I divide-se em N_i categorias ($N_i \leq n+1$), cada uma associada a um vértice $\mathbf{w}_{ij}$. Neste caso, a categoria ganhadora C_I é rejeitada ($r_I = 1$), e repete-se o processo de competição (subseção 2.1.2) para selecionar uma nova categoria ganhadora. Se, ao contrário, a $P(C_I) = P_d$, então as predições, a desejada e a associada à categoria, coincidem, e o padrão alcança a ressonância com a categoria ganhadora C_I. Neste caso, como a $T_I^t(\mathbf{I}) > \Gamma$, o padrão de entrada cai dentro da categoria C_I, de modo que esta não tem que ser expandida para aprender este padrão.

2.1.4 Expansão e criação de categorias

Quando nenhuma categoria é ganhadora na competição ($I = -1$), ou porque $T_i^t(\mathbf{I}) < \Gamma, \forall i$, ou porque o $r_i = 1, \forall i$, SAM tenta aprender o padrão de treinamento expandindo as categorias existentes até o dito padrão. A fig. 2.3 apresenta os vários casos que se podem produzir em $\mathbb{R}^2$. No painel (a), o padrão é codificado por uma categoria com um único vetor $\mathbf{w}_{i1}$ ($N_i = 1$), a categoria que passa a ser um segmento de reta (neste caso, $N_i = 2$ e $\mathbf{w}_{i2} = \mathbf{I}$). No painel (b), o padrão é codificado por uma categoria com dois vetores ($N_i = 2$), que passa a ser um simplex (um triângulo em $\mathbb{R}^2$, com $N_i = 3$). Nos painéis (c) e (d), o padrão de entrada é codificado por um conjunto de categorias. Nos dois primeiros casos, a expansão consiste simplesmente em adicionar o padrão de entrada à categoria ganhadora. No entanto, o último caso é o mais complexo, porque é necessário selecionar os vetores mais próximos das várias categorias. SAM gerencia esta situação da seguinte forma: os n vetores mais próximos, pertencentes às categorias com a predição desejada, são selecionados e utilizados para criar uma nova categoria com o padrão de entrada (painel (d)). Isto torna possível alguns "espaços vazios" entre as categorias, como mostra o painel (d).

A categoria nova (ou a categoria expandida, como nos painéis (a) e (b)), poderiam cobrir regiões que pertencem às outras categorias C_l com a $P(C_l) \neq P_d$. Neste caso, produziria-se sobreposição de predições, e a categoria expandida deveria ser contraída. O **teste de sobreposição** de predições efetua-se, simplesmente, cal-

culando a função de escolha T_i^t, da categoria C_i expandida, para todos os centros $\mathbf{c}_l$ das categorias C_l com a $P(C_l) \neq P(C_i)$. Se $T_i^t(\mathbf{c}_l) \neq 0$ para algum centro $\mathbf{c}_l$ nestas condições, então existe sobreposição entre as predições e a categoria C_i não deve ser expandida. Este teste de "sobreposição de predições" substitui, de algum modo, o teste de vigilância nas redes ART, dado que atua como critério limitante na expansão das categorias. Além do mais, não é imposto um limite de tamanho à categoria, e o parâmetro de vigilância ρ, utilizado nas redes ART, e muito influente nos seus resultados, não é mais necessário.

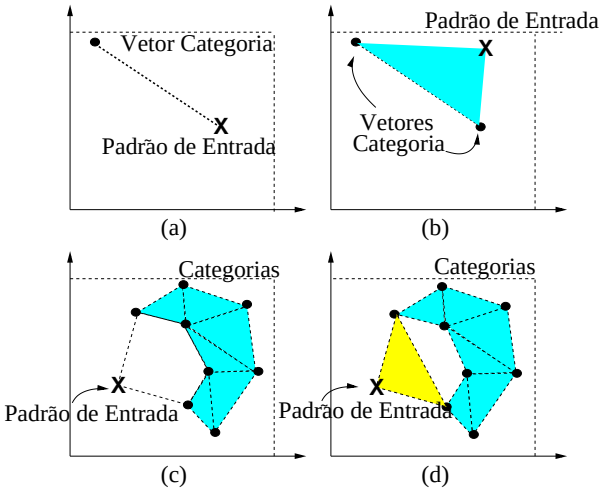


Figura 2.3: Exemplos de aprendizagem em $\mathbb{R}^2$.

Especificamente, se o $I = -1$ (i.e., o $\mathbf{I}$ não cai dentro de nenhuma categoria), SAM procura as categorias C_i mais próximas, e também os vetores $\mathbf{w}$ mais próximos do $\mathbf{I}$ com a predição P_d (é usada a distância euclidea). Existem duas situações:

1. Se não existe nenhuma categoria C_i com a predição $P(C_i) = P_d$, então uma categoria nova C_{N_c+1} é criada com o $N_{N_c+1} = 1$ e o padrão de entrada $\mathbf{I}$ como seu único vetor: $\mathbf{w}_{(N_c+1)1} = \mathbf{I}$.

2. Se existe uma, ou várias categorias C_i com a $P(C_i) = P_d$, os n vetores mais próximos pertencentes a estas categorias, são selecionados (pode ocorrer que existam somente $r < n$ vetores deste tipo). Também aqui existem dois casos possíveis:

   2.1. Se estes $r \leq n$ vetores selecionados pertencem à mesma categoria C_i, e, além do mais, o $N_i < n + 1$, então a categoria C_i não é um simplex

completo (seria um único vetor, ou um segmento de reta, um triângulo em $\mathbb{R}^3$, ou, em $\mathbb{R}^n$, um subconjunto, limitado por n vetores de um sub-espaço vetorial de dimensão $r \leq n$. Neste caso, o $\mathbf{I}$ é adicionado como um vetor novo a C_i, que se expande até alcançar o $\mathbf{I}$, como se indica na figura 2.3 (painéis (a) e (b)). Se esta expansão gera sobreposição entre as predições, a categoria C_i é contraída, retornando ao seu estado anterior, através da supressão do vetor $\mathbf{I}$ adicionado. Neste caso, uma categoria nova C_{N_c+1} é criada somente com um vetor $\mathbf{w}_{(N_c+1)1} = \mathbf{I}$ (categoria mono-vetor).

2.2. Se os $r \leq n$ vetores selecionados não pertencem à mesma categoria, ou se eles pertencem à mesma categoria C_i, mas o $N_i = n + 1$ (a categoria C_i é um simplex completo), então SAM não pode utilizar uma categoria já existente para codificar $\mathbf{I}$. Neste caso, SAM cria uma categoria nova C_{N_c+1} com os r vetores selecionados e o padrão de entrada $\mathbf{I}$ (fig. 2.3 painel (d)), de modo que $N_{N_c+1} = r + 1$. Desta maneira, o conjunto de categorias com predição P_d é aumentado com uma categoria nova, que cubra a região definida pelos r vetores e o $\mathbf{I}$. No caso de ser o $r < n$, a categoria criada não é um simplex, já que possui menos de $n + 1$ vértices (por exemplo, em $\mathbb{R}^2$, seria um segmento de reta). A sobreposição de predições é testada novamente e, em caso de produzir-se, os r vetores são suprimidos de C_{N_c+1} , que passa a ser uma categoria mono-vetor com o $N_{N_c+1} = 1$ e o $\mathbf{w}_{(N_c+1)1} = \mathbf{I}$.

A fig. 2.4 mostra o diagrama de fluxo do treinamento de SAM para um padrão. O algoritmo proposto é simples e, durante a aprendizagem de um padrão de treinamento, a categoria é expandida somente na direção daquele padrão, cobrindo o espaço de entrada entre eles sem sobreposição de predições.

A sobreposição entre as categorias, com diferentes predições, é impedida pelo teste de sobreposição de predições. Esta sobreposição, entre as duas categorias C_i e C_j, se avalia calculando a função de escolha T_j^t da categoria C_j, nos centros $\mathbf{c}_i(\mathbf{p}_i)$ das funções gaussianas, que definem a FEC da C_i. Mais formalmente, definimos a função de sobreposição $O(C_i, C_j)$ (*overlapping*) entre as categorias C_i e C_j, da seguinte forma:

$$O(C_i, C_j) = \begin{cases} 1 & \exists \mathbf{p}_i \in A_i(\alpha_i) : T_j^t(\mathbf{c}_i(\mathbf{p}_i)) > \Gamma \\ 0 & \text{em caso contrário} \end{cases} \tag{2.8}$$

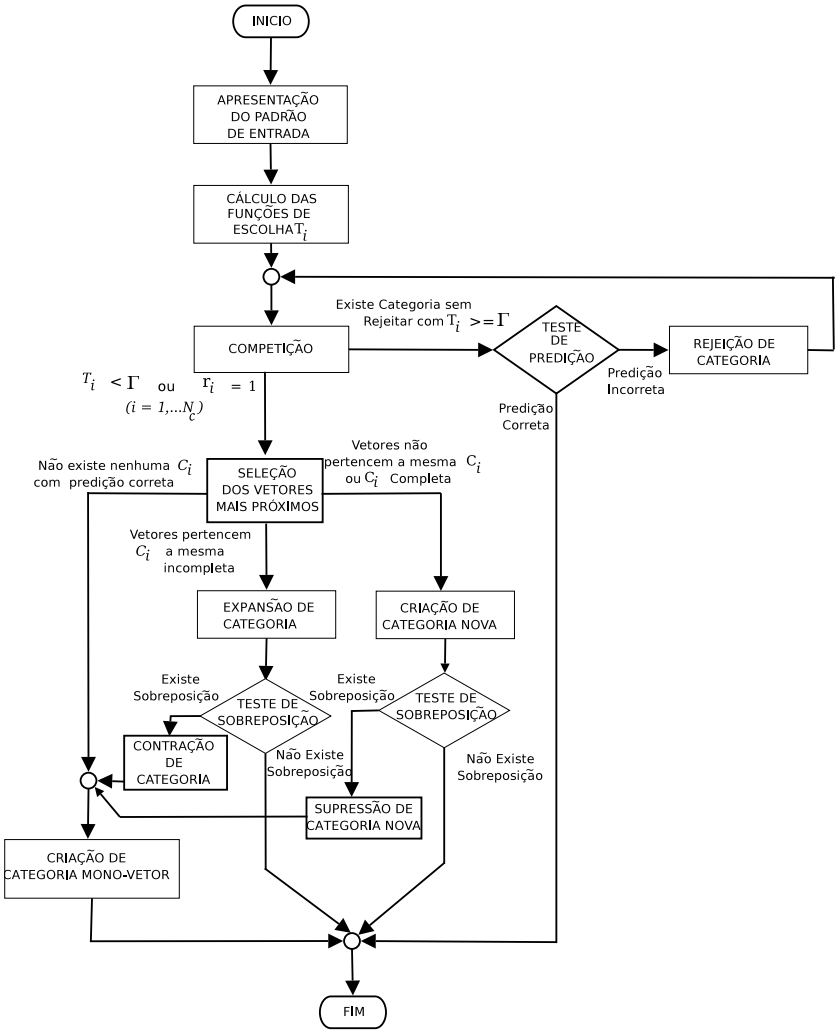


Figura 2.4: Diagrama de fluxo da etapa de treinamento em SAM.

onde o $\mathbf{c}_i(\mathbf{p}_i)$ é o centro da categoria C_i dado pelos índices $\mathbf{p}_i = (p_{i2}, \ldots, p_{iN_i}) \in \mathbb{N}^{N_i-1}$ e o $\Gamma \gtrsim 0$ é o limiar da função de escolha. De acordo com esta definição, $O(C_i, C_j) = 1$, se existe sobreposição entre C_i e C_j, e o $O(C_i, C_j) = 0$, em caso contrário.

O desvio σ das funções gaussianas e o limiar Γ são os únicos parâmetros ajustáveis de SAM. O valor de σ não deve ser muito alto, pois em tal caso, a função de escolha assumiria valores não nulos fora do volume do simplex. Por outro lado, tão pouco pode ser muito baixo, uma vez que então seriam necessárias muitas gaussianas para cobrir o volume do simplex, o qual incrementaria p custo computacional de SAM. Experimentalmente, podemos comprovar que um valor de $\sigma = 0.01$ é o valor máximo

necessário para que a função de escolha não assuma valores não nulos fora do volume do simplex. O parâmetro Γ também pode ser ajustado, e o seu valor deveria ser próximo a zero. Se o valor de σ é baixo, então as funções $T_i^t(\mathbf{I})$ não alcançam valores apreciáveis fora dos simplexes, de modo que o valor de Γ não tem muito impacto nos resultados.

2.1.5 Resumo da etapa de treinamento

Baseado nestes fundamentos, o algoritmo de *Simplex* ARTMAP, na etapa de treinamento, funciona como segue:

1. Apresentação de um novo padrão de entrada $\mathbf{I}$.

2. Cálculo das funções de escolha, $T_i^t(\mathbf{I}), i = 1, \ldots, N_c$ (eq. 2.5).

3. Competição. Se $T_i^t(\mathbf{I}) > \Gamma$ para alguma categoria C_i com $r_i = 0$, então a categoria C_I não rejeitada ($r_I = 0$), com o maior $T_i^t(\mathbf{I})$, é a candidata para codificar $\mathbf{I}$. Ir para o passo 4 (*Teste de predição*).

Se, ao contrário, todas as categorias estão rejeitadas ($r_i = 1, \forall i$), ou $T_i^t(\mathbf{I}) < \Gamma, \forall i$, então o $I = -1$. Neste caso, ir para o passo 5 (*Seleção dos vetores mais próximos*).

4. Teste de predição. Se a $P(C_I) = P_d$, então a categoria C_I classifica corretamente o padrão de entrada $\mathbf{I}$, e ele é codificado pela dita categoria (*Ressonância*). Neste caso, o $\mathbf{I}$ não é aprendido, porque já cai dentro da RRC da C_I. Finalizar.

Se, ao contrário, a $P(C_I) \neq P_d$, dividir C_I em N_I categorias mono-vetor, ($N_I \leq n + 1$) uma associada à cada vértice $\mathbf{w}_{Ij} \in C_I$. Ir para o passo 2 (*Competição*) para pesquisar uma nova categoria candidata.

5. Seleção dos vetores mais próximos. Se não existe nenhuma categoria C_i com predição $P(C_i) = P_d$, ir para o passo 8 (*Criação de categoria mono-vetor*). Caso contrário, seleciona os n vetores mais próximos a $\mathbf{I}$, com a predição P_d (pode ocorrer que somente existam $r < n$ vetores deste tipo).

6. Expansão de categoria. Se os r vetores mais próximos selecionados pertencem à mesma categoria C_i e o $N_i < n + 1$, adicionar o $\mathbf{I}$ como um vetor novo da C_i: o $N_i = N_i + 1$ e o $\mathbf{w}_{iN_i} = \mathbf{I}$ (figura 2.3, painéis (a) e (b)). Se esta expansão gera sobreposição de categorias com predições diferentes, retornar a C_i ao seu estado inicial (contração de categoria), e ir para o passo 8 (*Criação de categoria mono-*

vetor). Caso contrário, finalizar.

7. Criação de categoria nova. Se os r vetores mais próximos selecionados não pertencem à mesma categoria, ou se eles pertencem à mesma categoria C_i mas o $N_i = n + 1$, criar uma categoria C_{N_c+1} nova com os r vetores e o padrão de entrada $\mathbf{I}$ (fig. 2.3 painel (d)). Se o C_{N_c+1} sobrepõe-se com as categorias já existentes de predições diferentes, suprimir a C_{N_c+1}, e ir para o passo 8 (*Criação de categoria mono-vetor*). Em caso contrário, finalizar.

8. Criação de categoria mono-vetor. Criar uma categoria nova C_{N_c+1}, com o $N_{N_c+1} = 1$ e o $\mathbf{w}_{(N_c+1)1} = \mathbf{I}$. Finalizar.

Estes passos estão descritos em pseudocódigo com mais detalhe na seção H.1 do apêndice H.

2.2 Etapa de Processamento

A operação de SAM, durante a etapa de processamento, é como segue. A função de escolha de categoria, durante o processamento, se define de modo similar à etapa de treinamento. No entanto, no processamento, o padrão de entrada pode cair fora de qualquer simplex, de modo que $T_i^t(\mathbf{I}) < \Gamma, \forall i$. Neste caso, o padrão é codificado pela categoria mais próxima (i.e., àquela categoria que possui o vetor mais próximo ao padrão de entrada). Portanto, definimos a função $\psi(\mathbf{I})$ de modo que $\psi(\mathbf{I}) = 1$, se $T_i^t(\mathbf{I}) > \Gamma$ para alguma categoria i, $\psi(\mathbf{I}) = 0$, em caso contrário:

$$\psi(\mathbf{I}) = \Theta\left[\sum_{i=1}^{N_c} \Theta(T_i^t(\mathbf{I}) - \Gamma)\right] \tag{2.9}$$

onde a função $\Theta(x)$ está definida por:

$$\Theta(x) = \begin{cases} 1 & x > 0 \\ 0 & x \leq 0 \end{cases} \tag{2.10}$$

A função de escolha de categoria, durante a etapa de processamento, $T_i^p(\mathbf{I})$ é dada por:

$$T_i^p(\mathbf{I}) = \psi(\mathbf{I})T_i^t(\mathbf{I}) + (1 - \psi(\mathbf{I}))\left(1 - \frac{1}{\sqrt{n}} \min_{k=1,\ldots,N_i} \|\mathbf{I} - \mathbf{w}_{ik}\|\right) \tag{2.11}$$

De modo que, $T_i^p(\mathbf{I}) = T_i^t(\mathbf{I})$, se existe alguma categoria com $T_i^t(\mathbf{I}) > \Gamma$ (caso $\psi(\mathbf{I}) = 1$). Caso contrário (caso $\psi(\mathbf{I}) = 0$), $T_i^p(\mathbf{I})$ é decrescente com a distância entre o $\mathbf{I}$ e o vértice da C_i mais próximo ao $\mathbf{I}$ (o fator de escala $1/\sqrt{n}$, se deve a que $0 \leq \|\mathbf{I} - \mathbf{w}_{ik}\| \leq \sqrt{n}$). Portanto, neste caso, $T_i^p(\mathbf{I})$ é máximo para a categoria C_i com o vértice mais próximo ao $\mathbf{I}$.

O padrão de entrada $\mathbf{I}$ é codificado pela categoria C_I com a função de escolha $T_i^p(\mathbf{I})$ máxima:

$$I = arg \max_{i=1,\ldots,N_c} \{T_i^p(\mathbf{I})\} \tag{2.12}$$

Finalmente, a saída de SAM é a predição $P(C_I)$ associada à categoria ganhadora C_I. Esta etapa de processamento está descrita na seção H.2 do apêndice H.

2.3 Configuração experimental

O desempenho de SAM foi testado com o conjunto de dados *"Circle-in-the-Square"* (CIS) [143], muito usado na literatura em problemas de classificação, e no conjunto que denominaremos *"Concentric Circle In the Square"* (CCIS) (fig. 2.5), que definimos como generalização do conjunto anterior, com três círculos concêntricos. Ambos os conjuntos de dados são bi-dimensionais, de modo que permitem a representação gráfica das categorias, e das fronteiras entre as predições criadas por SAM.

Para cada um dos conjuntos de dados usados no trabalho experimental, tanto neste capítulo como nos seguintes, calcularemos uma série de medidas que permitem caracterizar a sua complexidade como problemas de classificação, desde distintos pontos de vista (separabilidade linear, mistura entre as predições diferentes, eficiência das características de entrada para a discriminação entre as categorias, etc.). Estas medidas de complexidade se definem em [89], e se descrevem de modo resumido no apêndice B. A tabela 2.2 mostra os valores destas medidas para os problemas CIS e CCIS, usados no trabalho experimental de validação de SAM.

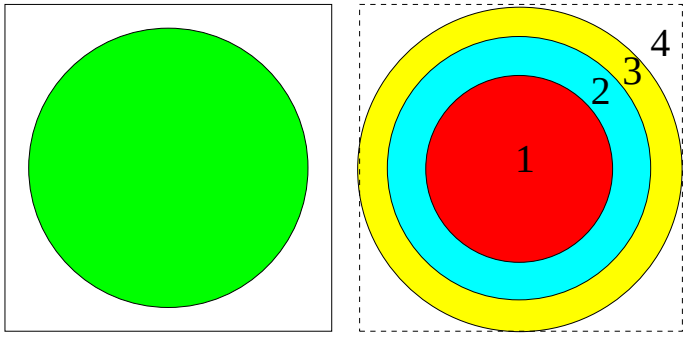


Figura 2.5: Região das predições de saída do conjunto de dados CIS (painel esquerdo) e do conjunto de dados CCIS (painel direito).

2.3.1 Conjunto de dados CIS

Neste conjunto de dados, o problema de classificação consiste em distingüir entre os pontos dentro e fora do círculo (fig. 2.5 painel esquerdo)). Os pontos são gerados randomicamente, dentro e fora do círculo. Testamos vários tamanhos de círculo, dados pelas porcentagens da área do quadrado coberta pelo círculo, abrangendo de 10% a 70% (não utilizamos os valores superiores a 70%, por corresponder aos círculos que saem fora do quadrado-unidade). A razão de utilizar vários tamanhos do círculo é que deste modo varia a população de padrões de cada predição que, lembremos, está uniformemente distribuída, e, portanto, a complexidade do problema. Para cada porcentagem desta área, foram gerados 20 conjuntos de treinamento e 20 conjuntos de teste (10000 padrões para cada um). Na tabela 2.2 se pode observar que as várias medidas de complexidade $f2$ e $f3$ crescem com a porcentagem de área coberta pelo círculo, ainda que existam exceções. A medida $f1$ varia de forma errática, porém sempre com valores muito maiores que 1, indicando a não separabilidade linear deste conjunto de dados. As medidas $n2$, $n3$ e $t2$, por sua vez, não refletem significantes variações em função da área.

2.3.2 Conjunto de dados CCIS

Este conjunto de dados, inspirado no problema T5 de [120], contém padrões pertencentes à 4 categorias, associadas aos 3 círculos concêntricos, e à região exterior a todos eles (fig. 2.5, painel direito). A metodologia de experimentação foi similar à utilizada no caso do problema CIS, usando 20 conjuntos de treinamento e 20 de teste, com 10000 padrões gerados, aleatoriamente, para cada um destes conjuntos.

Tabela 2.2: Medidas de complexidade para os problemas CIS (variando a porcentagem da área coberta pelo círculo) e CCIS.

Medida	CIS							CCIS	Intervalo	Ideal
	10	20	30	40	50	60	70			
$f1$	165.6	131.2	383.1	853.8	230	1054.9	451.3	900.4	$[0, +\infty)$	< 1
$f2$	0.35	0.50	0.62	0.71	0.79	0.87	0.94	0.99	$[0, 1]$	$\ll 1$
$f3$	1.52	1.97	2.62	3.49	4.87	7.68	16.5	212.7	$[1, +\infty)$	~ 1
$n2$	0.77	0.71	0.73	0.67	0.75	0.75	0.81	0.77	$[0, +\infty)$	$\ll 1$
$n3$	2.9E-3	4E-3	6E-3	5E-3	9E-3	7E-3	9E-3	0.02	$[0, 1]$	$\ll 1$
$t1$	0.94	0.97	0.96	0.98	0.95	0.97	0.99	0.99	$(0, 1]$	$\ll 1$
$t2$	0.0002							0.0002	$(0, +\infty)$	$\ll 1$

A maioria das medidas de complexidade de CCIS ($f1$, $f2$, $f3$, $n3$ e $t1$, na tabela 2.2) refletem valores visivelmente maiores que os de CIS, com a exceção de $n2$, com um valor similar. Todas as medidas, exceto $n3$ e $t2$, são elevadas em comparação com os valores ideais, indicados na última coluna.

2.4 Resultados

Para a validação comparativa de SAM, avaliamos os resultados das duas redes ART mais populares, que são *Fuzzy* ARTMAP [28] (FAM) e *Distributed* ARTMAP [36] (DAM).Por outro lado, para ter uma referência dos resultados obtidos por um classificador alheio ao modelo ART, avaliamos também a eficiência alcançada pela Máquina de Vetores de Suporte [136] (SVM), por ser, atualmente, um dos algoritmos de referência em problemas de clasificação. O apêndice A proporciona uma breve descrição dos princípios e funcionamento de uma SVM.

A tabela 2.3 resume os resultados obtidos por SAM, FAM, DAM e pela SVM. Cada linha apresenta a taxa de erro, em percentual, ($\%\epsilon$) e o número de categorias (#C) para estes classificadores, e para cada porcentagem do tamanho do círculo no conjunto de dados CIS. Os valores de cada linha são a média sobre os 20 experimentos (utilizamos, para cada experimento, um par de conjuntos diferentes de treinamento-teste). As últimas linhas mostram a média e o desvio padrão (STD), sobre todos os tamanhos dos círculos. No caso de FAM e DAM, indicam-se os melhores resultados médios (a taxa de erro mínima) entre todos os valores da vigilância, e as funções de escolha, sobre os conjuntos de validação, conforme os parâmetros registrados na tabela 2.4. A *Lei de Weber* e a *Choice-By-Difference*, descritas na secção 1.3.2,

Tabela 2.3: O erro (%ϵ) e o número de categorias (#C), ou vetores de suporte (#sv), obtidos usando SAM, FAM, DAM e SVM para o conjunto de dados CIS (colunas). As linhas são as diferentes porcentagens da área do círculo. Os menores erro e #C das redes ART e não ART estão em negrito.

	SAM		FAM		DAM		SVM	
$\bar{\rho}$	–		0.90		0.90		–	
%p	%ϵ	#C	%ϵ	#C	%ϵ	#C	%ϵ	#sv
10	0.7	460	1.0	213	1.0	183	0.19	558
20	0.9	544	1.4	218	1.3	187	0.20	559
30	1.1	574	1.8	222	1.7	186	0.21	566
40	1.3	584	2.0	226	2.1	188	0.26	581
50	1.4	608	2.2	227	2.5	187	0.24	582
60	1.5	631	2.4	229	3.0	188	0.25	593
70	1.6	654	2.5	235	4.1	190	0.25	608
Média	**1.2**	**579**	**1.9**	**225**	**2.3**	**187**	**0.23**	**578**
STD	0.33	64.0	0.54	7.3	0.94	2.2	0.03	18.5

foram investigados como funções de escolha por FAM. Os resultados foram muito similares, como apresentam-se na tabela 2.5, no entanto, quando utilizamos a *Lei de Weber* observamos ser ligeiramente melhores[3]. A SVM foi desenvolvida usando a biblioteca *Torch3* [40], utilizando o $C = 100$, e os melhores resultados foram obtidos utilizando os núcleos gaussianos, com o desvio igual a $\sigma = 0.1$. A fig. 2.6 apresenta (painel (a), à esquerda) um exemplo dos simplexes (triângulos em $\mathbb{R}^2$) criados por SAM para o conjunto de dados CIS (usando uma porcentagem de 30% da área do quadrado). O painel (b), à direita, mostra os círculos (bases radiais) das funções gaussianas associadas às categorias simplex de ambas as predições.

A tabela 2.6 mostra a taxa de erro e o número de categorias (ou vetores de suporte) obtidos pelos classificadores SAM, FAM, DAM e SVM para o conjunto de datos CCIS. Os resultados também foram conseguidos através da média dos 20 experimentos, cada um deles com um par de conjuntos diferentes de treinamento-teste. A figura 2.7 apresenta os simplexes (painel (a), à esquerda) e os círculos

[3]As tabelas de resultados somente reúnem o melhor valor alcançado por FAM.

Tabela 2.4: Os valores dos parâmetros nos diferentes modelos utilizados para a validação de SAM, nos conjuntos de dados CIS e CCIS.

FAM	$\alpha = 0.01, \beta = 1, \rho_{ab} = 1$, Lei de Weber e Choice-By-Difference
DAM	$\alpha = 0.01, \beta = 1, p = 10$
parâmetros comuns	$\bar{\rho} = 0.00 : 0.05 : 0.95$ 1 época de treinamento
SVM	$C = 100$, Núcleo Gaussiano com desvio $\in \{0.1, 0.5, 1, 2.5, 5, 7.5, 10, 12, 15, 17, 20\}$
SAM	$\sigma = 0.01, \Gamma = 0.01$

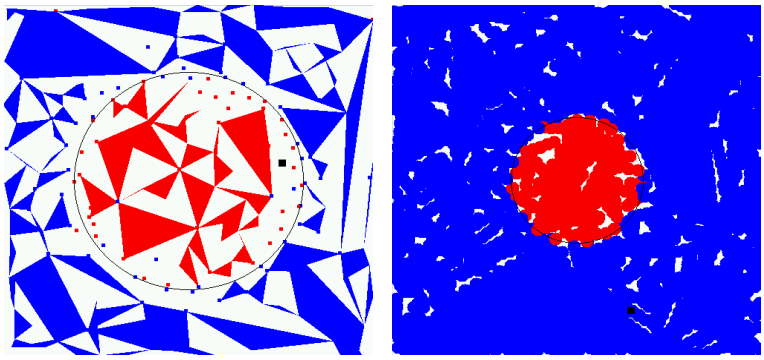


Figura 2.6: Os triângulos (painel (a), à esquerda) e as bases de funções gaussianas (painel (a), à direita) criados por SAM para o conjunto de dados CIS.

(painel (b), à direita) das funções gaussianas criadas por SAM para o conjunto de dados CCIS. Destacamos também a similaridade dos resultados alcançados por FAM, com as duas funções de escolha para o conjunto de dados CCIS, na tabela 2.5.

Os classificadores mencionados foram também testados no conjunto de dados *Iris Plant* da UCI *Machine Learning Repository* [12]. No entanto, a taxa de erro foi de 0% para todos eles, por isto não é aceitável como uma referência para comparação.

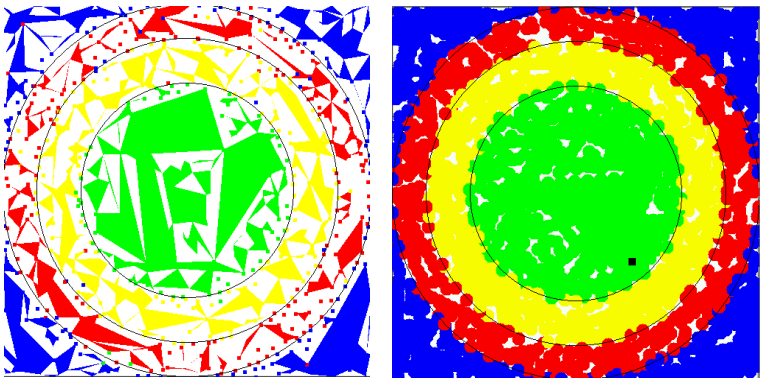
Tabela 2.5: Erro (em percentual) e o número de categorias criadas por FAM usando a *Lei de Weber* (LW) e *Choice-by-difference* (CBD), como funções de escolha, para os conjuntos de dados CIS (nas diferentes porcentagens de área do círculo) e CCIS.

CIS				
–	LW		CBD	
%p	%ϵ	#C	%ϵ	#C
10	0.98	214.6	0.97	213.4
20	1.36	219.4	1.36	218.4
30	1.77	222.2	1.70	222.4
40	1.97	224.6	2.00	226.1
50	2.18	226.2	2.17	227.2
60	2.38	229.2	2.32	229.6
70	2.51	234.8	2.61	235.2
CCIS				
–	LW		CBD	
–	%ϵ	#C	%ϵ	#C
–	6.00	278.6	6.05	277.2

2.5 Discussão

O classificador SVM fornece o melhor dos resultados para todos os experimentos, como se pode observar nas tabelas 2.3 e 2.6. No entanto, ele não permite aprendizagem *on-line* e além do mais, o número de vetores de suporte é maior que o número de categorias obtidos usando FAM (para ambos os conjuntos de dados) e DAM (apenas em CIS). Comparando com as técnicas baseadas em ART, a taxa de erro obtida utilizando SAM é menor que utilizando FAM (para ambos os conjuntos de dados) e DAM (para o conjunto de dados CIS), porém o número de categorias é o mais alto (comparável ao número de vetores de suporte criados pela SVM para o conjunto de dados CIS, e o dobro para CCIS). Estes resultados mostram a capacidade de SAM em aprender categorias, ajustando muito bem as fronteiras entre as predições (figuras 2.6 e 2.7), mas com o número muito elevado de categorias.

Aparentemente, as categorias triangulares na fig. 2.6 (painel (a), à esquerda) não cobrem o espaço de entrada, mas devemos considerar que a escolha da categoria é dada pelas funções gaussianas (eq. 2.5) e, no painel da direita, os seus círculos cobrem todo o espaço de entrada, de modo que não é necessário um crescimento maior



(a) (b)

Figura 2.7: As regiões no conjunto de dados CCIS (painel (a), à esquerda). As bases das funções gaussianas criadas por SAM para este conjunto de dados (painel (b), à direita).

Tabela 2.6: Erro médio e o número de categorias obtidos usando SAM, FAM, DAM e SVM para o conjunto de dados CCIS.

	SAM		FAM ($\bar{\rho} = 0.90$)		DAM ($\bar{\rho} = 0.90$)		SVM	
	%ϵ	#C	%ϵ	#C	%ϵ	#C	%ϵ	#sv
Média	4.1	1141	6.0	**279**	**3.4**	994	0.9	560
STD	0.30	60.0	0.32	9.7	0.79	222	0.12	18.6

dos triângulos. Existe também várias categorias mono-vetores, como mostra o painel (a), à esquerda (pontos isolados). Destacamos que SAM somente aprende/armazena os vértices do simplex $\mathbf{w}_{ij}$, a partir dos quais se calculam os centros das funções gaussianas (eq. 2.3), que não se armazenam. O número de vetores $\mathbf{w}_{ij}$ (não apresentados nas tabelas 2.3 e 2.6) é também menor que o número de categorias, por causa do compartilhamento de vetores entre elas.

Os resultados obtidos por FAM e DAM correspondem ao melhor desempenho para valores diferentes do ρ e a função de escolha (no caso de FAM, conforme a tabela 2.5). SAM não depende do parâmetro de vigilância, como as redes FAM e DAM, mas faz uso dos parâmetros Γ e σ (desvio do núcleo gaussiano), que deve assumir um valor compromisso que cubra o espaço de entrada de forma eficiente. Portanto, a supressão da vigilância, que constitui uma vantagem comparativa em

relação às redes ART, por suprimir um parâmetro a ajustar, fica neutralizada pela introdução da desviação σ das funções gaussianas, que define a precisão com a que se aprendem as fronteiras entre as categorias.

A taxa de erro obtida mostra que a capacidade de aprendizagem de SAM é comparável, ou maior que estes classificadores, exceto em CCIS, no qual a DAM obtém o melhor resultado (3.41% de erro). Porém, este erro, ligeiramente inferior ao das redes ART, obtém-se ao custo de um número de categorias bastante superior, relacionado com o processo de quebras de categorias (seção 2.1.3).

Uma boa parte dos erros de classificação de SAM, é devido ao valor fixo do parâmetro σ, que leva a certa sobreposição entre as categorias (de iguais predições) nas fronteiras entre as predições (figuras 2.6, painel da direita, e 2.7, painel da direita), de modo que seria necessário um valor inferior de σ.

Finalmente, tem-se que considerar que o número de centros de funções gaussianas aumenta muito rapidamente quando aumenta a dimensão n do espaço de entrada (e também quando se reduz o valor de σ), o qual limitaria a sua eficiência em problemas de classificação de dimensão elevada. Por esta razão, o seguinte passo no trabalho de investigação realizado é a redefinição das categorias internas, de modo que a função de escolha se possa calcular mais eficientemente. O objetivo, que abordaremos no seguinte capítulo, é a definição de categorias internas com formas poliédricas irregulares (politopos), ao invés de simplexes, por se tratar de uma forma geométrica mais geral.

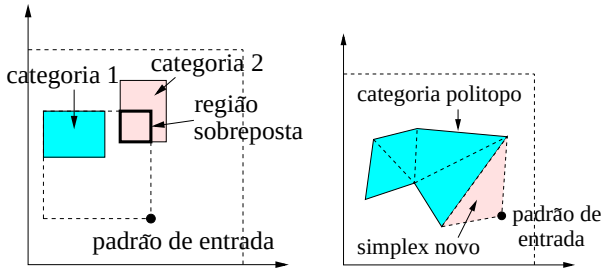
Capítulo 3

PolyTope ARTMAP

Os problemas e limitações associados à *Simplex* ARTMAP nos levaram a formulação de *PolyTope* ARTMAP (PTAM) [62, 64]. Esta segunda proposta usa categorias internas com geometria de politopo irregular, definido por uma série de hiperplanos limitados, que compõem os lados do politopo. Estes hiperplanos se definem, de modo que, compõem aproximações lineares por partes (*piece-wise linear*) das fronteiras entre as distintas predições de saída. Os vértices de cada categoria politopo são vetores de pesos que correspondem a alguns padrões de treinamento previamente codificados pela dita categoria. O número de vértices de um politopo depende da complexidade geométrica do conjunto de dados. Estes vértices são selecionados de maneira que o esboço do politopo se ajuste à fronteira entre as predições de saída. Cada predição de saída pode ser associada a um ou mais politopos. Cada categoria politopo é definida por vários vértices, em vez de um único vetor de pesos como nas outras redes ART. A capacidade das categorias irregulares para aproximar as fronteiras entre as predições de saída pode ser superior que as de categorias com geometria predefinida, e isto pode permitir menor erro e categorias internas.

Contudo, a Função de Escolha de Categoria (FEC) nas redes tradicionais assume o valor 1, se o padrão de entrada cai dentro da categoria e, um valor menor que 1, decrescente com a distância entre o padrão e a RRC, em caso contrário. O cálculo direto de uma FEC equivalente para uma categoria com geometria politopo, é complexo. No caso de politopos convexos é possível, usando as equações dos hiperplanos que os delimitam, determinam se um padrão cai dentro ou fora de um politopo. Isto não é possível no caso de politopos não convexos. No nosso caso, para simplificar a sua descrição matemática, e seguindo as idéias sugeridas por *Simplex*

ARTMAP, representamos cada politopo dividindo-o em vários simplexes adjacentes. A FEC do politopo é calculada usando a FEC de cada simplex que, similarmente às FEC das redes ART tradicionais, assume o valor 1 se o padrão de entrada se encontra dentro de um destes simplexes. De outra forma, a FEC é menor que 1 e diminui com a distância entre o padrão e o politopo.



(a) (b)

Figura 3.1: (a) Aprendizagem rápida de FAM em $\mathbb{R}^2$. Quando a categoria 1 aprende o padrão de entrada, mantendo, simultaneamente, a sua forma retangular, ela se sobrepõe com a categoria 2, que tem uma predição associada diferente. (b) A expansão de uma categoria politopo de PTAM em $\mathbb{R}^2$ através da criação de um simplex novo (triângulo em $\mathbb{R}^2$) somente se realiza na direção que vai desde a categoria ao padrão de entrada.

As considerações realizadas na seção 1.5, mostram a relação existente entre a geometria predefinida das categorias internas nas redes ART clássicas, e a existência do parâmetro e o teste de vigilância, assim como da importância do tamanho da categoria na função de escolha, e na sobreposição entre as categorias. Entretando, as categorias em PTAM não têm a geometria predefinida, se bem que elas, internamente, estão integradas por um conjunto de simplexes adjacentes, não sobrepostos. Por outro lado, como o simplex cobre o menor volume fechado entre os seus vértices, a categoria politopo pode ser expandida somente até o padrão de entrada– e não em outras direções– usando os mesmos princípios que *Simplex* ARTMAP, através da adição de um simplex novo definido por determinados vértices do politopo e pelo padrão de entrada (fig. 3.1 (b)). Este comportamento é diferente das redes ART tradicionais que, como no caso de FAM (fig. 3.1 (b)), necessitam sobreporem as suas categorias internas, para poder ser expandida mantendo a sua geometria predefinida.

As categorias politopo de PTAM podem ser construídas de maneira que cubram somente regiões do espaço de entrada populadas por padrões com a mesma predição de saída (se existem padrões com uma predição diferente entre a categoria

e o padrão de entrada, a categoria é ajustada como descreveremos posteriormente). Por esta razão, é possível expandir as categorias de PTAM sem se sobreporem entre si, e, portanto, PTAM pode impedir a dita sobreposição sem evitar a expansão. Esta limitação tem importantes conseqüências sobre a sua operação. Por um lado, introduz um fator limitante na expansão das categorias, que torna relativamente razoável: uma categoria poder ser expandida somente até que "choque" (se sobreponha) com outra categoria. Deste modo, se suprime a sobreposição massiva entre as categorias que caracteriza as redes ART tradicionais, nas quais, com freqüência, é o tamanho da categoria o aspecto que decide a qual delas codifica o padrão de entrada, e não as posições relativas de padrão e categoria. Consequentemente, já não é necessário usar um critério, que nas redes ART é o tamanho da categoria, para limitar a sua expansão. Na realidade, este último é um critério não definido diretamente pelo conjunto de dados, uma vez que o tamanho máximo das categorias, determinado pelo parâmetro de vigilância, não pode ser aprendido durante a etapa de treinamento. Então deve ser determinado externamente, e portanto, ajustado para cada problema. Então, a ausência de sobreposição de categorias permite a PTAM suprimir o parâmetro de vigilância.

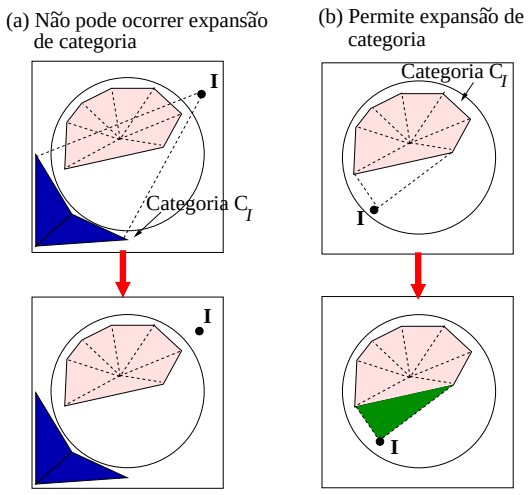


Figura 3.2: Exemplos de expansões de categoria em PTAM com o problema *Circle-In-The-Square* (CIS) em $\mathbb{R}^2$ [143]. Painel (a): A categoria ganhadora C_I não pode ser expandida, porque existe uma categoria com predição distinta entre a C_I e o padrão de entrada **I**. Painel (b): A categoria ganhadora C_I se expande até o padrão de entrada **I** sem sobreposição (painel (b)).

Por outro lado, dado que as categorias não podem se sobreporem, o padrão de entrada não pode cair dentro de várias categorias. Portanto, não é necessário um

critério para decidir qual categoria tem maior FEC, quando o padrão cai dentro de várias categorias[1]. Consequentemente, a FEC de PTAM não leva em consideração o tamanho da categoria.

Cada vez que uma categoria se expande para aprender um padrão de treinamento, PTAM –similarmente a FMMNN – deve avaliar se a categoria expandida se sobrepõe com outras categorias, para impedir, em tal caso, a dita sobreposição. Quer dizer, PTAM tem que determinar se existe outra categoria, entre o padrão de entrada e a categoria que o pretende codificar (a ganhadora da competição). O alcance deste objetivo requer utilizar testes geométricos para determinar a intersecção entre as categorias, técnicas que se descrevem na subseção 3.1.4 e com maior detalhe no apêndice F. Este processo constitui um teste, que determina se a categoria ganhadora pode, ou não, ser expandida até o padrão de entrada (ou seja, codificá-lo). Este teste, denominado "Teste de Sobreposição", vem a substituir o teste de vigilância das redes ART tradicionais[2], e avalia se a categoria ganhadora, logo depois de ser expandida até o padrão de entrada, se sobrepõe com outra categoria que se encontra entre ambos. Se existe sobreposição (painel (a) na fig. 3.2), a categoria ganhadora é rejeitada (*reset*) e contraída ao seu estado anterior à expansão, e uma categoria nova é a ganhadora e tenta codificar o padrão de entrada. Em caso contrário, a expansão é permitida (painel (b) na fig. 3.2)

O teste de sobreposição impede que uma categoria possa cobrir regiões ocupadas por outras categorias. Em particular, se estas têm uma predição associada distinta, este teste impede a uma categoria cobrir regiões com padrões de entrada com uma predição diferente. Entretanto, este teste, por si só, não é suficiente para impedir a sobreposição de categorias com predições associadas distintas. De fato, uma categoria C_i, com predição associada $P(C_i)$, pode cobrir uma região populada por padrões com predições desejadas $P_d \neq P(C_i)$, se nenhum padrão com predição P_d caiu dentro daquela região anteriormente. Logo, uma categoria deve ser capaz de corrigir uma expansão errada que realizou previamente. Consideremos que um padrão de treinamento $\mathbf{I}$, com predição desejada $P_d \neq P(C_i)$, seja apresentado a rede posteriormente, e $\mathbf{I}$ caia dentro da categoria C_i. Neste caso, C_i é corrigida, removendo o simplex que contém $\mathbf{I}$. De modo que, se $\mathbf{I}$ voltasse a se apresentar no futuro, já não cairia dentro de C_i. Os vértices do simplex suprimido, que não pertençam a

[1]Como ocorre nas redes ART, nas quais o tamanho determina a categoria que apresenta maior FEC (concretamente, a categoria de menor tamanho).

[2]Entretanto, diferentemente do teste de vigilância, o teste de sobreposição não requer de nenhum parâmetro sintonizável.

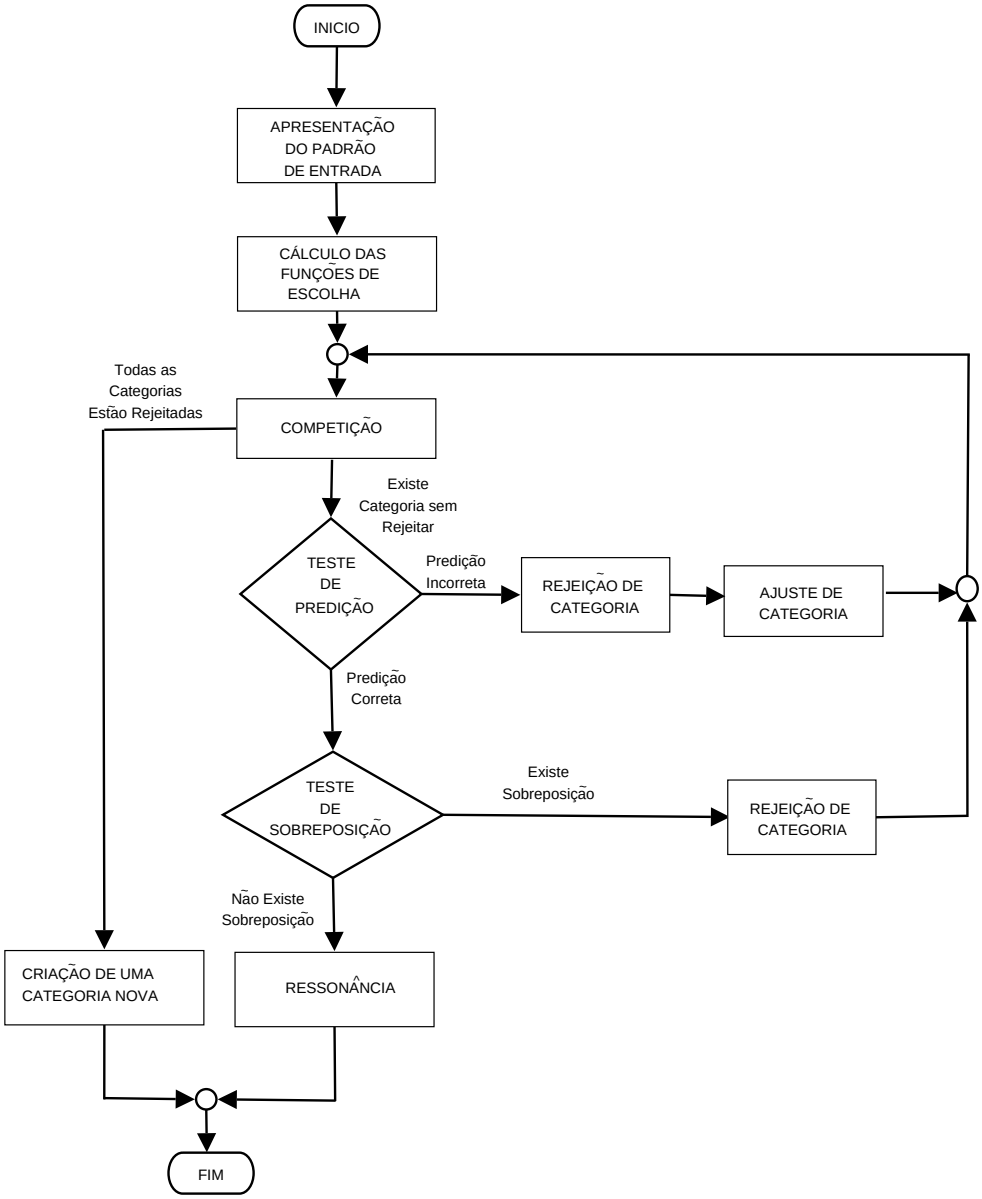


Figura 3.3: Diagrama de fluxo da etapa de treinamento em PTAM.

nenhum outro simplex, serão apresentados à rede como padrões de entrada para ser aprendidos novamente. Este é o passo de Ajuste de Categorias (subseção 3.1.3).

Além da limitação imposta pela não-sobreposição sobre a expansão de uma categoria C_i, a predição desejada P_d para o padrão de entrada $\mathbf{I}$ deve ser comparada com a predição $P(C_i)$ associada à dita categoria, uma vez que o padrão não deveria ser codificado por uma categoria com uma predição incorreta. Com este objetivo, e de modo similar a etapa de *Match Tracking* nas redes ART tradicionais (ainda que sem incrementar a vigilância) o Teste de Predição (subseção 3.1.3) determina se a predição desejada P_d é igual a predição associada $P(C_i)$. Se $P_d \neq P(C_i)$, a categoria C_i é rejeitada (*reset*), e uma nova categoria é selecionada, com base na FEC, para tentar codificar ao padrão de entrada[3]. Em caso contrário, a categoria C_i passa no teste de predição, e o padrão de treinamento $\mathbf{I}$ é codificado pela mesma na etapa de ressonância (subseção 3.1.5).

Finalmente, se nenhuma categoria passa em ambos os testes, PTAM cria uma categoria nova (subseção 3.1.6). Como as categorias politopo têm, ao menos, um simplex definido por $n+1$ vetores de pesos (vértices), sendo n a dimensão do espaço de entrada, a criação de uma categoria nova com um único simplex (categoria mono-simplex) requer n padrões de treinamento com predição desejada P_d. Se a categoria nova se sobrepõe com outras categorias, ou existem menos de n vetores de pesos com predição P_d, então não é possível criar uma categoria nova com um único simplex, e PTAM cria uma categoria nova de vetor único (categoria mono-vetor) com o padrão de treinamento $\mathbf{I}$[4].

Podemos ter uma visão global de todo este processo na fig. 3.3, que resume os passos da etapa de treinamento em PTAM.

[3]Obviamente, o teste de predição implica que nenhuma categoria C_i com predição associada $P(C_i) \neq P_d$ pode codificar ao padrão de entrada. Portanto, este teste deveria ser prévio ao teste de sobreposição por razões de eficiência, como veremos nas seguintes seções.

[4]Nos nossos trabalhos consideramos a possibilidade de criar categorias com $1 < p < n+1$ vetores, de modo similar a SAM, mas por razões de simplicidade, consideramos somente as categorias mono-vetor, categorias simplex ou politopo.

3.1 Etapa de Treinamento

As subseções a seguir descrevem os principais passos da etapa de treinamento. As categorias politopo de PTAM são consideradas como uma série de simplexes adjacentes. Concretamente, cada categoria C_i ($i = 1, \ldots, N_c$, onde N_c é o número de categorias) contém N_i^s simplexes $\{S_{i1}, \ldots, S_{iN_i^s}\}$. Cada simplex S_{ij} está definido por $n+1$ vetores de pesos ou vértices $\{\mathbf{w}_{ij1}, \ldots, \mathbf{w}_{ij(n+1)}\}$, e está delimitado por $n+1$ hiperplanos $\{h_{ij1}, \ldots, h_{ij(n+1)}\}$. Por sua vez, cada hiperplano h_{ijk} está definido por n vetores $\{\mathbf{w}_{ijk1}, \ldots, \mathbf{w}_{ijkn}\}$. Em algumas situações, uma categoria C_i pode estar definida por um único vetor denotado por $\mathbf{w}_i$ (em tal caso, a denominaremos de categoria mono-vetor), e não possui nenhum simplex ($N_i^s = 0$). A tabela 3.1 resume a nomeclatura usada.

Tabela 3.1: Nomenclatura associada à PTAM.

n	Dimensão do padrão de entrada $\mathbf{I}$
$\mathbf{I}$	Padrão de entrada, $\mathbf{I} = (I_1, \ldots, I_n)$, $I_l \in [0, 1]$, $l = 1, \ldots, n$ sem codificação em complemento
P_d	Predição desejada para o padrão de entrada $\mathbf{I}$
N_c	Número de categorias
C_i	i-ésima categoria, $i = 1, \ldots, N_c$
$P(C_i)$	Predição associada à categoria C_i
N_i^s	Número de simplexes na categoria C_i
S_{ij}	j-ésimo Simplex da categoria C_i, $j = 1, \ldots, N_i^s$
$\mathbf{w}_{ijl}$	l-ésimo Vetor de peso (vértice) do simplex S_{ij} na categoria C_i, $l = 1, \ldots, n+1$
h_{ijk}	k-ésimo Hiperplano do simplex S_{ij} na categoria C_i, $k = 1, \ldots, n+1$
$\mathbf{w}_{ijkl}$	l-ésimo Vetor de peso do hiperplano h_{ijk} do simplex S_{ij} na categoria C_i, $l = 1, \ldots, n$
$T_i^t(\mathbf{I})$	Função de Escolha da Categoria (FEC) C_i, durante o treinamento
$T_{ij}(\mathbf{I})$	Função de Escolha do simplex S_{ij} na categoria $C_i, j = 1, \ldots, N_i^s$
$T_i^p(\mathbf{I})$	Função de Escolha da Categoria (FEC) C_i, durante o processamento
r_i	Estado de rejeição da categoria C_i ($r_i = 1$ se C_i é rejeitada, senão $r_i = 0$)
$\mathbf{w}_i$	Vetor de peso da categoria mono-vetor C_i ($N_i^s = 0$)
$\widehat{\mathbf{w}}_{ijk}$	Vetor de peso do simplex S_{ij} que não pertence ao hiperplano h_{ijk}
$\mathbf{w}_{ijk}^*$	Vetor diretor (normal) do hiperplano h_{ijk} (apêndice D)

3.1.1 Função de Escolha de Categoria

A FEC de uma categoria politopo C_i de PTAM, durante o treinamento, para um padrão de entrada $\mathbf{I}$, denotada por $T_i^t(\mathbf{I})$, é definida como o máximo das funções de escolha $T_{ij}(\mathbf{I})$ dos simplexes $S_{ij}, j = 1, \ldots, N_i^s$ que compõem C_i :

$$T_i^t(\mathbf{I}) = \max_{j=1,\ldots,N_i^s} \{T_{ij}(\mathbf{I})\} \qquad i = 1, \ldots, N_c \tag{3.1}$$

A função de escolha $T_{ij}(\mathbf{I})$ do simplex S_{ij} se define de maneira que $T_{iJ}(\mathbf{I}) = 1$, se $\mathbf{I}$ cai dentro do simplex S_{iJ}. Se, ao contrário, $\mathbf{I}$ cai fora do simplex S_{ij}, a função $T_{ij}(\mathbf{I})$ se define de modo que $0 < T_{ij}(\mathbf{I}) < 1$ e, além do mais, é decrescente com a distância $d(\mathbf{I}, S_{ij})$ entre o padrão de entrada e o simplex. Portanto, $T_i^t(\mathbf{I}) = 1$, somente se $T_{iJ}(\mathbf{I}) = 1$ para algum simplex S_{iJ} de C_i, isto é, se $\mathbf{I}$ cai dentro de C_i. Se, pelo contrário, $\mathbf{I}$ cai fora de C_i, então $T_i^t(\mathbf{I}) = T_{iJ}(\mathbf{I})$, onde o S_{iJ} é o simplex mais próximo de $\mathbf{I}$ na categoria C_i. Descreveremos ambos os casos separadamente.

Padrão de entrada situado no interior de um simplex

O simplex S_{ij} está definido por $n+1$ hiperplanos $\{h_{ij1}, \ldots, h_{ij(n+1)}\}$. O hiperplano h_{ijk} é definido por n vetores de pesos $\{\mathbf{w}_{ijk1}, \ldots, \mathbf{w}_{ijkn}\}$, que atuam como vértices do simplex. Consequentemente, o hiperplano h_{ijk} pode ser descrito pela seguinte equação:

$$sgn(\phi_{ijk}(\widehat{\mathbf{w}}_{ijk}))\phi_{ijk}(\mathbf{I}) = 0 \tag{3.2}$$

Onde a função $sgn(x)$ é a tradicional função sinal:

$$sgn(x) = \begin{cases} 1 & x > 0 \\ 0 & x = 0 \\ -1 & x < 0 \end{cases} \tag{3.3}$$

O vetor $\widehat{\mathbf{w}}_{ijk}$ é o vetor de peso do simplex S_{ij} que não pertence ao hiperplano h_{ijk}[5]. A função $\phi_{ijk}(\mathbf{x})$ é dada por:

[5]Dado que um simplex S_{ij} tem $n+1$ vetores, e um hiperplano h_{ijk} está definido por n vetores, somente existe um vetor em S_{ij} que não pertence a h_{ijk}.

$$\phi_{ijk}(\mathbf{x}) = \begin{vmatrix} x_1 - w_{ijk1,1} & \dots & x_n - w_{ijk1,n} \\ w_{ijk2,1} - w_{ijk1,1} & \dots & w_{ijk2,n} - w_{ijk1,n} \\ \dots & & \\ w_{ijkn,1} - w_{ijk1,1} & \dots & w_{ijkn,n} - w_{ijk1,n} \end{vmatrix} \tag{3.4}$$

Onde $w_{ijkl,m}$ é o m-ésimo componente do vetor $\mathbf{w}_{ijkl}$ (ver apêndice D para detalhes). Por conveniência, definiremos:

$$g_{ijk}(\mathbf{I}) \equiv sgn(\phi_{ijk}(\widehat{\mathbf{w}}))\phi_{ijk}(\mathbf{I}) \tag{3.5}$$

A função $g_{ijk}(\mathbf{I})$ permite dividir o espaço de entrada $[0,1]^n$ em duas regiões, uma para a qual $g_{ijk}(\mathbf{I}) > 0$ e, outra definida por $g_{ijk}(\mathbf{I}) < 0$. O fator $sgn(\phi_{ijk}(\widehat{\mathbf{w}}))$ na eq. 3.5 garante que $g_{ijk}(\mathbf{I}) > 0$ no lado do hiperplano h_{ijk} que está no interior do simplex S_{ij}, e $g_{ijk}(\mathbf{I}) < 0$ no exterior do simplex. Um padrão de entrada $\mathbf{I}$ cai no interior do simplex S_{ij}, somente se ele cai no lado interior dos $n+1$ hiperplanos $h_{ij1}, \dots, h_{ij(n+1)}$ do simplex S_{ij}. Finalmente, podemos definir a FEC do simplex S_{ij}, para os casos em que $\mathbf{I}$ cai dentro do simplex S_{ij}, segundo o indicado anteriormente, como:

$$T_{ij}(\mathbf{I}) = 1 \qquad se \quad g_{ijk}(\mathbf{I}) > 0, \forall k = 1, \dots, n+1 \tag{3.6}$$

Portanto, $T_{ij}(\mathbf{I}) = 1$, se o padrão $\mathbf{I}$ cai dentro dos $n+1$ hiperplanos de S_{ij}.

Padrão de entrada situado no exterior de um simplex

A partir do comentado nos parágrafos anteriores, a função de escolha $T_{ij}(\mathbf{I})$ do simplex S_{ij}, deve satisfazer $0 < T_{ij}(\mathbf{I}) < 1$, somente se o $\mathbf{I}$ cai fora do simplex, e ser decrescente com a distância entre o padrão e o simplex. Para cumprir estas condições, definimos:

$$T_{ij}(\mathbf{I}) = 1 - \frac{d(\mathbf{I}, S_{ij})}{\sqrt{n}} \qquad se \quad \exists k \in \{1, \dots, n+1\}, g_{ijk}(\mathbf{I}) < 0 \tag{3.7}$$

Dado que $\mathbf{I} \in [0,1]^n$, a longitude da diagonal do hiper-retângulo $[0,1]^n$ é $\sqrt{n}$, de modo que $0 \leq d(\mathbf{I}, S_{ij}) < \sqrt{n}$. Portanto, $T_{ij}(\mathbf{I}) = 1$, se $d(\mathbf{I}, S_{ij}) = 0$, e $T_{ij}(\mathbf{I}) = 0$, se $d(\mathbf{I}, S_{ij}) = \sqrt{n}$. Usamos um valor aproximado de $d(\mathbf{I}, S_{ij})$, devido a que o seu cálculo

preciso é complexo para $n > 2$. Com este objetivo, consideramos a hiperesfera centrada no centróide $\mathbf{c}_{ij}$ dos vértices do simplex:

$$\mathbf{c}_{ij} = \frac{1}{n+1} \sum_{l=1}^{n+1} \mathbf{w}_{ijl} \tag{3.8}$$

e, cujo raio R_{ij} é a maior distância entre este centróide e os vértices do simplex:

$$R_{ij} = \max_{l=1,\ldots,n+1} \{\|\mathbf{w}_{ijl} - \mathbf{c}_{ij}\|\} \tag{3.9}$$

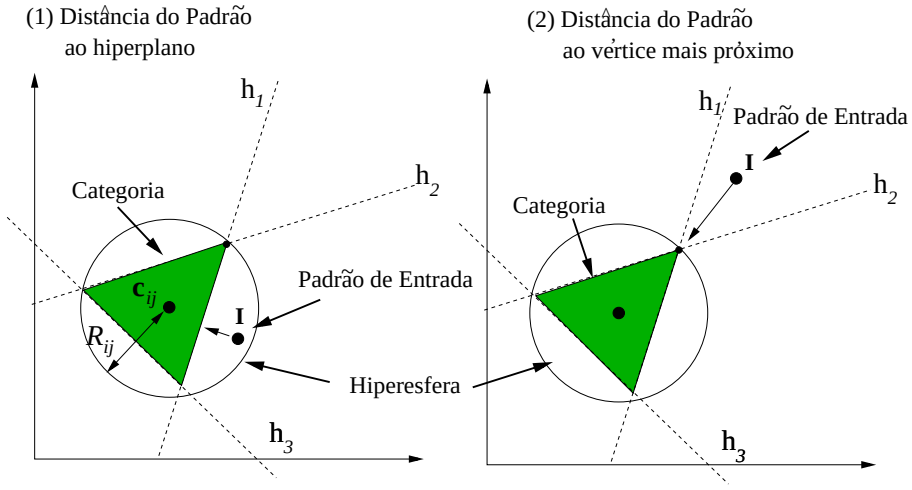


Figura 3.4: (1) Se o padrão se encontra dentro da hiperesfera, a distância do padrão-simplex se aproxima pela distância ao seu hiperplano (reta em $\mathbb{R}^2$) mais próximo. (2) Se o padrão de entrada se encontra fora da hiperesfera (círculo em $\mathbb{R}^2$), a distância do padrão-simplex se aproxima pela distância ao seu vértice mais próximo.

Considerando a posição relativa do padrão de entrada e desta hiperesfera, existem dois casos (fig. 3.4):

1. Se o padrão de entrada cai dentro desta hiperesfera ($\|\mathbf{I} - \mathbf{c}_{ij}\| < R_{ij}$), então $d(\mathbf{I}, S_{ij})$ é aproximada pela distância entre o $\mathbf{I}$ e o seu hiperplano mais próximo h_{ijk} no simplex S_{ij}:

$$d(\mathbf{I}, S_{ij}) \simeq \min_{k \in \mathcal{S}} \frac{|g_{ijk}(\mathbf{I})|}{\|\mathbf{w}^*_{ijk}\|}, \quad \|\mathbf{I} - \mathbf{c}_{ij}\| < R_{ij} \tag{3.10}$$

onde $\mathbf{w}^*_{ijk}$ é o vetor diretor do (perpendicular ao) hiperplano h_{ijk} (ver apêndice E), e $|g_{ijk}(\mathbf{I})|/\|\mathbf{w}^*_{ijk}\|$ é a distância entre o $\mathbf{I}$ e o h_{ijk}. O conjunto $\mathcal{S}$ inclui os índices k dos hiperplanos h_{ijk} do simplex S_{ij}, para que $g_{ijk}(\mathbf{I}) < 0$, ou seja, os hiperplanos do simplex para os quais $\mathbf{I}$ cai no seu lado exterior.

2. Se $\mathbf{I}$ cai fora da hiperesfera, então a distância padrão-simplex é aproximada pela distância mínima entre o $\mathbf{I}$ e um vértice do S_{ij}:

$$d(\mathbf{I}, S_{ij}) \simeq \min_{l=1,\ldots,n+1}\{\|\mathbf{I} - \mathbf{w}_{ijl}\|\}, \quad \|\mathbf{I} - \mathbf{c}_{ij}\| \geq R_{ij} \tag{3.11}$$

Em resumo, se pode escrever a distância $d(\mathbf{I}, S_{ij})$, de maneira compacta, como:

$$d(\mathbf{I}, S_{ij}) \simeq \begin{cases} \min\limits_{k:g_{ijk}(\mathbf{I})<0} \dfrac{|g_{ijk}(\mathbf{I})|}{\|\mathbf{w}^*_{ijk}\|} & \|\mathbf{I} - \mathbf{c}_{ij}\| < R_{ij} \\ \min\limits_{l=1,\ldots,n+1} \{\|\mathbf{I} - \mathbf{w}_{ijl}\|\} & \|\mathbf{I} - \mathbf{c}_{ij}\| \geq R_{ij} \end{cases} \tag{3.12}$$

e a função de escolha $T_{ij}(\mathbf{I})$ do simplex S_{ij} é dada por:

$$T_{ij}(\mathbf{I}) = \begin{cases} 1 & g_{ijk}(\mathbf{I}) > 0, \quad k = 1, \ldots, n+1 \\ 1 - \dfrac{d(\mathbf{I}, S_{ij})}{\sqrt{n}} & \text{senão} \end{cases} \tag{3.13}$$

3.1.2 Competição

Similarmente às redes ART tradicionais, em PTAM existe também uma competição entre as categorias politopo, na qual é escolhida somente a categoria C_I com a maior FEC e que não esteja rejeitada ($r_I = 0$):

$$T^t_I(\mathbf{I}) = \max_{i=1,\ldots,N_c} \left\{T^t_i(\mathbf{I}) : r_i = 0\right\} \tag{3.14}$$

Se todas as categorias estão rejeitadas ($r_i = 1, i = 1, \ldots, N_c$), uma categoria nova é criada (subseção 3.1.6). A fig. 3.5 resume esta etapa.

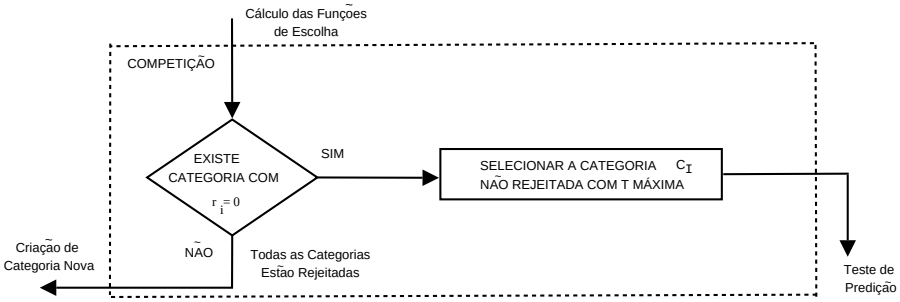


Figura 3.5: Diagrama de fluxo da etapa de competição entre as categorias.

3.1.3 Teste de Predição e Ajuste de Categoria

Se a predição $P(C_I)$ da categoria C_I ganhadora é distinta da predição desejada P_d para o padrão de entrada $\mathbf{I}$, então C_I não pode codificar o padrão de entrada. Este **Teste de Predição** (TP) compara ambas predições. Se $P(C_I) = P_d$, C_I passa no TP e o seguinte passo (Teste de Sobreposição, subseção 3.1.4) é executado. Senão, $P(C_I) \neq P_d$, C_I não passa no TP e ela é rejeitada ($r_I = 1$). Se, além do mais, $T_I^t(\mathbf{I}) = 1$, o padrão de entrada cai dentro da RRC da C_I. Isto significa que a região coberta por C_I está populada também por um padrão $\mathbf{I}$ com $P_d \neq P(C_I)$. Neste caso, o simplex S_{IJ}, que verifica $T_{IJ}(\mathbf{I}) = 1$, é removido, para poder corrigir a categoria C_I, de modo que se evite este erro de classificação. Esta é a etapa de **Ajuste de Categoria**. Estas duas etapas representa-se no diagrama de fluxo da fig. 3.6. O processo competitivo seleciona outra categoria não-rejeitada, com a qual se repete o teste de predição. Expandimos o processo de rejeição de categoria no diagrama de fluxo da fig. 3.7.

Se o simplex S_{IJ} removido possui vetores $\mathbf{w}_{IJl}$ que não pertencem a outros simplexes em C_I, então cada vetor $\mathbf{w}_{IJl}$ é apresentado a PTAM como um novo

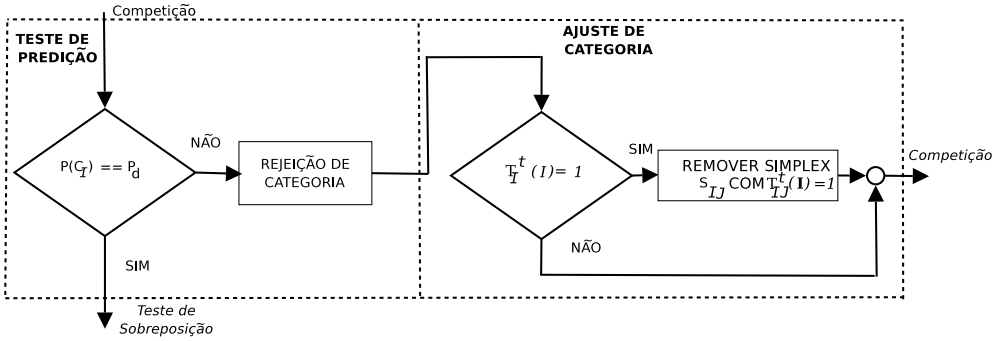


Figura 3.6: Diagrama de fluxo das etapas de Teste de Predição e Ajuste de Categoria.

padrão de treinamento, depois do atual $\mathbf{I}$, para ser codificado pelas outras categorias. Se outro simplex, S_{pq}, cobrir $\mathbf{w}_{IJl}$, durante a classificação de $\mathbf{I}$, S_{pq} poderia ser removido, quando $\mathbf{w}_{IJl}$ fosse apresentado outra vez e, este fato, poderia levar a um laço infinito (um *loop*). Para impedir esta situação, no momento de suprimir o simplex S_{IJ}, PTAM cria uma categoria nova mono-vetor C^*, para cada vetor $\mathbf{w}_{IJl}$ de S_{IJ}, que não pertence a nenhum outro simplex. O Teste de Sobreposição (seção 3.1.4) impede que nenhum simplex cubra a categoria C^*, que é removida, quando $\mathbf{w}_{IJl}$ é apresentado como novo padrão de entrada. Assim, $\mathbf{w}_{IJl}$ não pode remover qualquer simplex na segunda apresentação e, estes *loops* não são possíveis.

Finalmente, no caso de PTAM, não tem sentido a etapa de *Match Tracking*, comum nas redes ART, quando não coincidem as predições desejada e a associada à categoria ganhadora. A razão é que o *Match Tracking* consiste na elevação da vigilância, de modo que se rejeite a dita categoria ganhadora. Por outro lado, esta elevação da vigilância reduz o tamanho máximo permitido para que uma categoria alcance a ressonância, de maneira que as futuras ganhadoras devam ser mais específicas, ou seja, com um tamanho menor. Dado que PTAM, nem usa o parâmetro de vigilância, nem se baseia no tamanho da categoria para selecionar a categoria ganhadora, não se faz necessário incluir a etapa de *Match Tracking*.

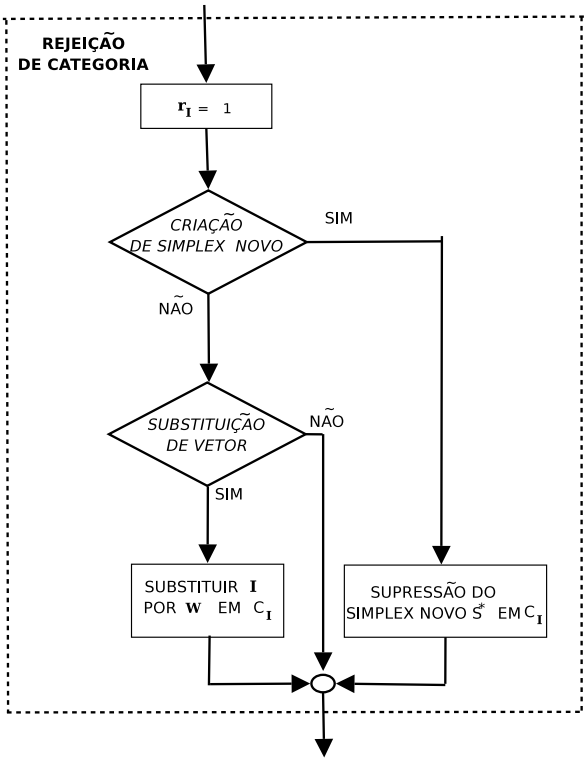


Figura 3.7: Diagrama de fluxo da rejeição de categoria.

3.1.4 Teste de Sobreposição

Se a categoria ganhadora C_I passa no teste de predição, o **Teste de Sobreposição** (TS) determina se C_I se sobrepõe com as outras categorias, quando se expande até o **I**. Se existe sobreposição, C_I não passa no TS, e é rejeitada ($r_I = 1$). Neste caso, uma categoria nova, não-rejeitada, é selecionada pelo processo competitivo. Se C_I passa no TS, então C_I codifica o padrão de entrada **I** (etapa de **Ressonância**, subseção 3.1.5). A fig. 3.8 ilustra o diagrama de fluxo da etapa do TS.

Existem várias situações possíveis no TS, dependendo da posição relativa do **I** e da C_I, que estão ilustradas na fig. 3.9:

1. O **I** cai dentro da RRC da categoria C_I (i.e., $T_I^t(\mathbf{I}) = 1$). Neste caso, a expansão da categoria não é necessária para cobrir o **I**, porque já está cobrindo-o. Como C_I não tem que ser expandida, a sobreposição com outras categorias não é possível, e a C_I passa no TS (fig. 3.9 (1)).

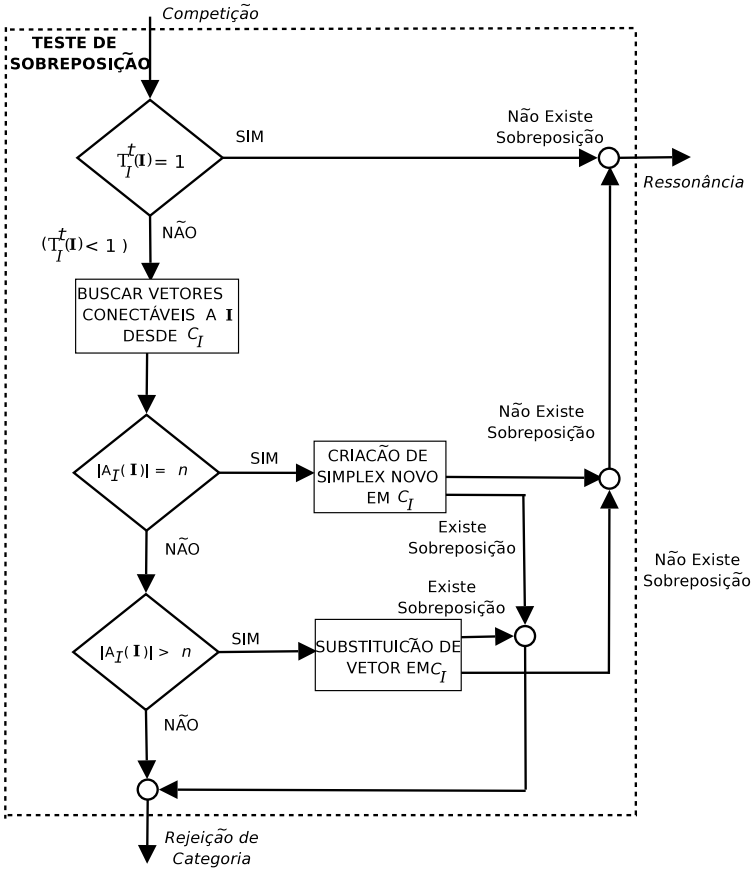


Figura 3.8: Diagrama de fluxo da etapa do Teste de Sobreposição.

2. O $\mathbf{I}$ cai fora da RRC da categoria C_I (i.e., $T_I^t(\mathbf{I}) < 1$). A C_I pode aprender o $\mathbf{I}$ expandindo-se até o $\mathbf{I}$ (incluindo o $\mathbf{I}$ como um vetor de peso, ou substituindo um vetor de peso da C_I pelo $\mathbf{I}$). Em ambos os casos, novos segmentos, entre o $\mathbf{I}$ e os vértices da C_I, devem ser criados, de forma que não se possam sobrepor com as categorias existentes. Além do mais, deve ser calculado o conjunto $\mathcal{A}_I(\mathbf{I})$ dos vetores de pesos na categoria C_I, que são conectáveis ao $\mathbf{I}$, sem sobreposição com as outras categorias. A função $C(\mathbf{w}, \mathbf{I})$ (eq. F.6 no apêndice F.1) permite determinar se o vetor de peso $\mathbf{w}$ e o padrão de entrada $\mathbf{I}$ são conectáveis[6], de modo que $C(\mathbf{w}, \mathbf{I}) = 1$ em tal caso, e em caso contrário, $C(\mathbf{w}, \mathbf{I}) = 0$. Portanto, o conjunto $\mathcal{A}_I(\mathbf{I})$ pode ser definido como:

$$\mathcal{A}_I(\mathbf{I}) = \{\mathbf{w}_{Ijl} : C(\mathbf{w}_{Ijl}, \mathbf{I}) = 1\} \tag{3.15}$$

[6]Para avaliar esta conectividade, é determinado se existe intersecção entre o segmento $\overline{\mathbf{Iw}}$ e algum simplex de qualquer categoria existente, como se descreve no apêndice F.1.

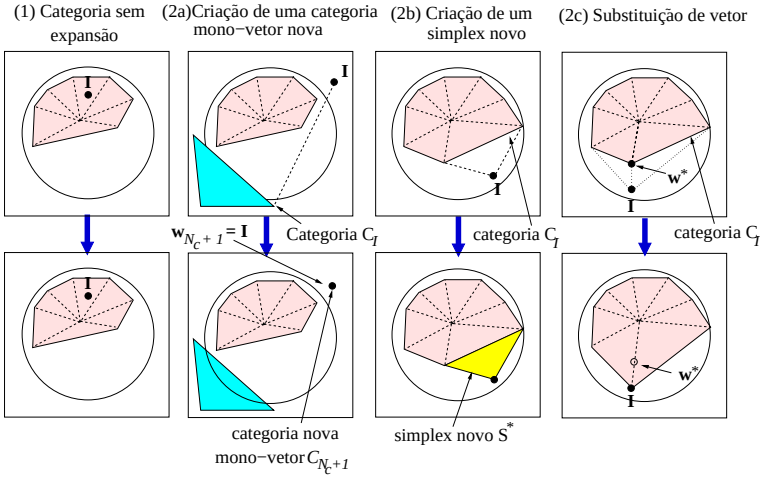


Figura 3.9: Exemplos de aprendizagem da categoria politopo para o problema *Circle-In-the-Square* (CIS) em $\mathbb{R}^2$ [143]. (1) O padrão de entrada $\mathbf{I}$ cai dentro da C_I, que não é expandida. (2a) Menos que $n = 2$ vetores de pesos são conectáveis do $\mathbf{I}$ em C_I: a C_I não é expandida e o $\mathbf{I}$ cria uma categoria mono-vetor nova C_{N_c+1}, com $\mathbf{w}_{N_c+1} = \mathbf{I}$. (2b) Os $n = 2$ vetores de pesos são conectáveis do $\mathbf{I}$, e a C_I se expande até o $\mathbf{I}$ pela criação de um simplex novo S^*. (2c) O número de vetores de pesos conectáveis do $\mathbf{I}$ é 3 ($> n = 2$): o vetor $\mathbf{w}^*$ é substituído pelo $\mathbf{I}$, sem perda de volume para a categoria C_I.

Dependendo do $|\mathcal{A}_I(\mathbf{I})|$ (número de elementos do conjunto $\mathcal{A}_I(\mathbf{I})$, ou número de vetores conectáveis desde o $\mathbf{I}$ em C_I), existem três casos:

2.1. Se $0 \leq |\mathcal{A}_I(\mathbf{I})| < n$ (fig. 3.9 (2a)), não existem vértices conectáveis suficientes do $\mathbf{I}$, na C_I, para criar um simplex novo entre eles, porque os segmentos do $\mathbf{I}$ para os vértices da C_I se sobrepõem com outras categorias. Consequentemente, a categoria C_I não passa no TS, porque não pode ser expandida até o $\mathbf{I}$, sem sobreposição.

2.2. Se $|\mathcal{A}_I(\mathbf{I})| = n$ (fig. 3.9 (2b)), um simplex novo S^*, pertencente a C_I, pode ser criado com vértices no $\mathbf{I}$ e nos n vetores de pesos do $\mathcal{A}_I(\mathbf{I})$. Se o S^* se sobrepõe com as outras categorias, o S^* é removido e a C_I não passa no TS. Senão, a C_I passa no TS.

2.3. Se $|\mathcal{A}_I(\mathbf{I})| > n$ (fig. 3.9 (2c)), algum vetor de peso no $\mathcal{A}_I$ pode ser substituído pelo $\mathbf{I}$, sem perda de volume para a categoria C_I, como veremos a seguir. Se os simplexes modificados na C_I, depois da substituição do vetor, não se sobrepõem com as outras categorias, a C_I passa no TS.

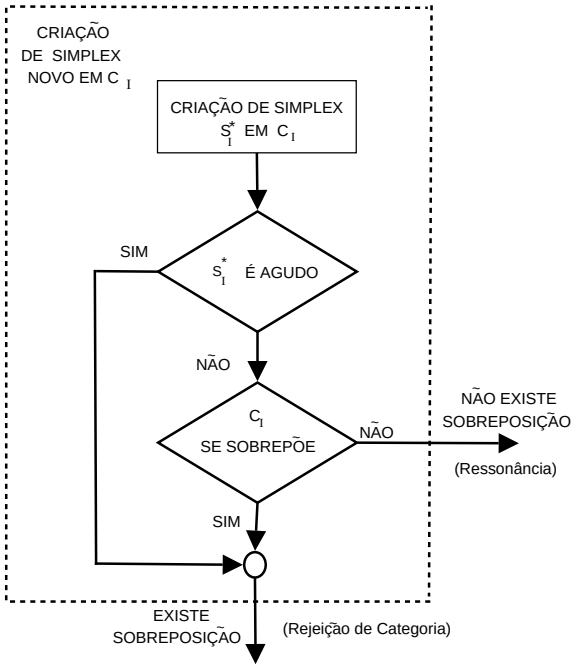


Figura 3.10: Diagrama de fluxo da etapa de criação de simplex novo.

As duas seguintes subseções detalham mais os casos (2b) e (2c).

Expansão da categoria: criação de um simplex novo

Quando $|\mathcal{A}_I(\mathbf{I})| = n$, o TS determina se a categoria C_I pode ser expandida até o $\mathbf{I}$, pela criação de um simplex novo $S^* = \mathcal{A}_I \cup \{\mathbf{I}\}$ entre a C_I e o $\mathbf{I}$ (fig. 3.10). A sobreposição entre o S^* e as outras categorias é testado usando a função O_{sc} (sobreposição entre a categoria e o simplex), definida na equação F.8 (apêndice F.2). Esta função é definida de modo que $O_{sc}(S^*, C_i) = 1$, se o simplex S^* e a categoria C_i se sobrepõem e, senão, $O_{sc}(S^*, C_i) = 0$. No primeiro caso, o simplex novo S^* se sobrepõe com a categoria C_i $(i \neq I)$, de modo que o S^* é removido e a categoria C_I não passa no TS. No segundo caso, a categoria C_I passa no TS. A criação de um simplex novo, aumenta a quantidade de informação armazenada pela rede– o número de vetores de pesos e os simplexes–,uma vez que inclui um novo vetor e um novo simplex, similarmente à criação de novas categorias nas redes ART clássicas.

O simplex novo pode ser muito agudo, se seus hiperplanos estiverem quase que paralelos. Neste caso, o volume do simplex é muito pequeno, e mesmo quando

ele cobre as regiões populadas por padrões com predições corretas, isto não aumenta significativamente o volume da categoria, ainda que contribua a aumentar a informação armazenada. Não obstante, se um simplex agudo cobre uma região com uma predição diferente, por seu pequeno volume, é muito pouco provável que padrões futuros de treinamento, com aquela predição, caiam dentro deste simplex. Portanto, o passo de Ajuste de Categoria (subseção 3.1.3) não pode removê-lo. Os simplexes agudos impedem a expansão de outras categorias, dificultando a aprendizagem e contribuindo à sua proliferação. Então, propomos remover estes simplexes agudos, de maneira que as categorias cubram, mais eficientemente, o espaço de entrada. O volume de um simplex em $\mathbb{R}^n$ pode ser calculado usando o determinante de *Cayley-Menger* [73], mas a sua complexidade computacional é muito elevada. Como alternativa para entender a criação destes simplexes agudos, PTAM assume que algum ângulo entre os hiperplanos de um simplex deste tipo é muito pequeno. Especificamente, o PTAM cria um simplex novo, S_{ij}, com vetores de pesos $\{\mathbf{w}_{ij1}, \ldots, \mathbf{w}_{ij(n+1)}\}$, somente se os ângulos θ_{klm} entre os vetores $\mathbf{w}_{ijl} - \mathbf{w}_{ijk}$ e $\mathbf{w}_{ijm} - \mathbf{w}_{ijk}$, $k, l, m = 1, \ldots, n+1, k \neq l \neq m$, estão acima do valor limiar θ_{min}. O ângulo θ_{klm} é calculado usando o produto interno (escalar) dos vetores $\mathbf{w}_{ijl} - \mathbf{w}_{ijk}$ e $\mathbf{w}_{ijm} - \mathbf{w}_{ijk}$:

$$\theta_{klm} = arcos \left[\frac{(\mathbf{w}_{ijl} - \mathbf{w}_{ijk})(\mathbf{w}_{ijm} - \mathbf{w}_{ijk})}{\|\mathbf{w}_{ijl} - \mathbf{w}_{ijk}\| \|\mathbf{w}_{ijm} - \mathbf{w}_{ijk}\|} \right], \quad k, l, m = 1, \ldots, n+1, k \neq l \neq m \tag{3.16}$$

O valor de θ_{min} deve ser pequeno o suficiente para remover apenas os simplexes agudos. Das considerações gráficas sobre os simplexes agudos em $\mathbb{R}^2$, transladáveis a $\mathbb{R}^n$, assumimos $\theta_{min} = 10$ graus, como sendo um valor razoável para este limiar.

Expansão da categoria: substituição do vetor de pesos

Se o número de vetores de pesos conectáveis com o $\mathbf{I}$ na categoria C_I é maior que n ($|\mathcal{A}_I(\mathbf{I})| > n$), o TS determina se a C_I pode ser expandida até o $\mathbf{I}$, substituindo algum vetor de peso $\mathbf{w} \in \mathcal{A}_I(\mathbf{I})$ da C_I pelo $\mathbf{I}$ (fig. 3.11). Se a C_I se sobrepõe com outras categorias, depois da substituição, a C_I não passa no TS. A substituição dos vetores de pesos não aumenta a quantidade de informação aprendida pela rede, uma vez que não se incrementa o número de vetores de pesos nem o número de simplexes.

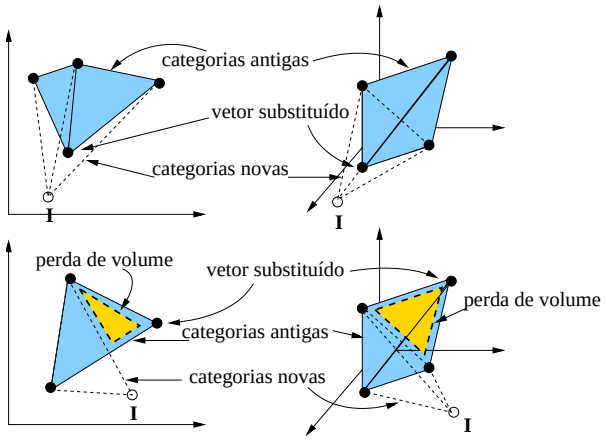


Figura 3.11: Exemplos de substituição de um vetor de peso do simplex por **I**, em $\mathbb{R}^2$ (painel esquerdo) e em $\mathbb{R}^3$ (painel direito), sem perda de volume (painel superior) e com perda de volume (painel inferior) do simplex.

A fig. 3.11 apresenta que, dependendo da posição relativa do padrão e do simplex, a substituição de um vetor de peso pode reduzir (painel inferior) ou não (painel superior) o volume de um ou de vários simplexes. Este fato poderia conduzir à perda de informação aprendida. Por outro lado, a fig. 3.12 (painel esquerdo), em $\mathbb{R}^2$, apresenta que um simplex perde volume pela substituição de um vértice, quando o padrão de entrada **I** cai nas regiões 1, 2 e 3; mas não, quando este cai nas regiões A, B ou C. Se o **I** cai do lado exterior (fora do simplex) dos n hiperplanos que se cruzam com o **w**, o **I** pode substituir o **w** sem perda de volume do simplex. Senão, os simplexes definidos pelo **w**, perdem volume (painel direito da fig. 3.12: a perda de volume é devido a que o **I** cai no lado interior do h_2 e do h_3). Consideremos o conjunto $H_{ij}(\mathbf{w}) = \{k = 1, \ldots, n+1 : \mathbf{w} \in h_{ijk}\}$, integrado pelos hiperplanos do simplex S_{ij} que contêm o vetor **w**. O padrão de entrada **I** pode substituir ao vetor **w** no caso de satisfazer a condição:

$$\prod_{k \in H_{ij}(\mathbf{w})} \Phi(g_{ijk}(\mathbf{I})) > 0 \tag{3.17}$$

A função $g_{ijk}(\mathbf{I})$ é definida na eq. 3.5. Por sua vez, a função $\Phi(x)$ está definida como:

$$\Phi(x) = \begin{cases} 1 & x \leq 0 \\ 0 & x > 0 \end{cases}$$

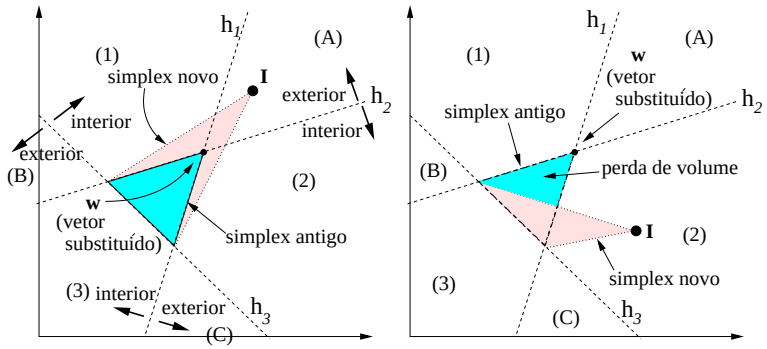


Figura 3.12: Exemplo da condição de substituição do vetor, em $\mathbb{R}^2$. Os padrões nas regiões A, B e C podem substituir os vetores de pesos do antigo simplex (painel esquerdo), mas os padrões nas regiões 1, 2 e 3 não podem fazê-lo, porque a substituição reduz o volume do simplex (painel direito).

Se o **I** cai no lado interior de algum hiperplano $h_{ijk} \in H_{ij}(\mathbf{w})$, então $g_{ijk}(\mathbf{I}) > 0$ e $\Phi(g_{ijk}(\mathbf{I})) = 0$, logo a condição 3.17 não é cumprida. Se $g_{ijk}(\mathbf{I}) < 0, \forall k \in H_{ij}(\mathbf{w})$, então o **I** satisfaz a condição 3.17. Neste caso, o vetor de peso mais próximo $\mathbf{w}^*$ do **I** no $\mathcal{A}_I(\mathbf{I})$ que satisfaz a condição 3.17 é o candidato para a substituição. O vetor $\mathbf{w}^*$ é substituído pelo **I**, somente se os simplexes S_{ij} para os quais $\mathbf{w}^* \in S_{ij}$ não se sobreponham com as outras categorias, depois da substituição do vetor. Esta condição significa que $O_{sc}(S_{ij} \setminus \{\mathbf{w}^*\} \cup \{\mathbf{I}\}, C_i) = 0$ (a função O_{sc} é definida na eq. F.8 no apêndice F.2), $\forall S_{ij} : \mathbf{w}^* \in S_{ij}, \forall C_i, i \neq I$. Se nenhum vetor de peso da C_I satisfaz esta condição, a C_I não passa no TS. O processo de substituição de um vetor de pesos se representa graficamente no diagrama da fig. 3.13.

3.1.5 Ressonância

Se a C_I passa em ambos os testes, de Predição e de Sobreposição, dois casos são possíveis, como se indica no diagrama de fluxo da fig. 3.14:

1. Se o **I** cai dentro da RRC da C_I (em tal caso, a $T_I(\mathbf{I}) = 1$), a C_I já cobre o **I** e não tem que ser expandida para incluí-lo, de modo que a aprendizagem não é necessária . Nesta situação, a C_I se mantém sem modificação (fig. 3.9 (1)).

2. Se o **I** cai fora da C_I ($T_I(\mathbf{I}) < 1$), a C_I foi expandida até o **I**, ou substituindo algum vetor de peso da C_I pelo padrão de entrada **I**, ou criando um simplex novo. Neste último caso, se a C_I já é a categoria mais ativa, isto é, se

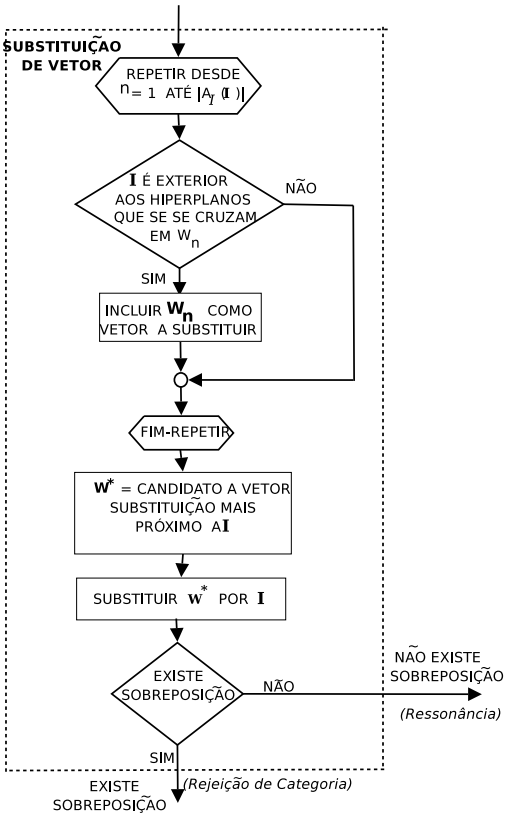


Figura 3.13: Diagrama de fluxo da etapa de substituição de um vetor de pesos.

$T_I(\mathbf{I}) = max\{T_i(\mathbf{I})\}_{i=1}^{N_c}$, então a expansão de C_I não era necessária para que C_I codificasse o padrão de entrada. De fato, C_i seria a ganhadora, e alcançaria a ressonância com o padrão de entrada, se ele for apresentado outra vez. Em conseqüência, e para evitar a criação de simplexes redundantes, o simplex novo S^* é suprimido, de modo que C_I retorna ao seu estado prévio. Este fato poderia provocar que PTAM criasse uma única categoria mono-simplex, somente se apresentassem, sucessivamente, todos os padrões de treinamento com a mesma predição desejada. Esta categoria, ao ser um simplex único, não poderia modelar a geometria do problema adequadamente, o que seria indesejável. No entanto, geralmente os padrões de treinamento estão embaralhados, assim, os padrões com predições diferentes estão misturados, e este erro não pode ocorrer.

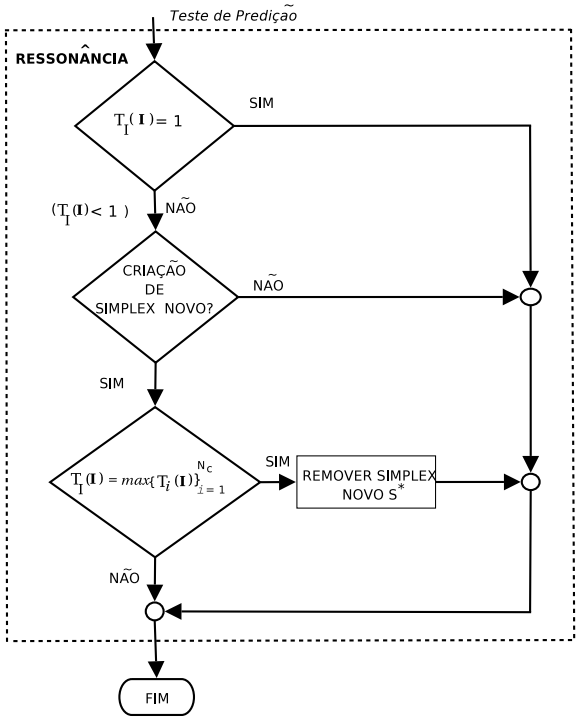


Figura 3.14: Diagrama de fluxo da etapa de Ressonância.

3.1.6 Criação de categorias novas

Se nenhuma categoria passa em ambos os testes, de Predição e de Sobreposição, PTAM seleciona os n vetores de pesos conectáveis mais próximos do $\mathbf{I}$, pertencentes as categorias com a predição desejada P_d. A função $C(\mathbf{w}, \mathbf{I})$ (eq. F.6 no apêndice F.1) determina se um vetor de peso $\mathbf{w}$ é conectável desde o $\mathbf{I}$, quer dizer, se o segmento $\overline{\mathbf{wI}}$ interseccciona com alguma categoria C_i. Se existem n ou mais vetores de pesos conectáveis desde o $\mathbf{I}$, com predição P_d, os n vetores mais próximos e o $\mathbf{I}$ criam um simplex novo S^*. Se o S^* não se sobrepõe com outras categorias, e não é um simplex agudo (subseção 3.1.4), PTAM cria uma categoria nova C_{N_c+1} com o simplex S^* e a predição P_d (neste caso, $N^s_{N_c+1} = 1$, $S_{(N_c+1)1} = S^*$ e $P(C_{N_c+1}) = P_d$). Se existe menos de n vetores de pesos conectáveis com predição P_d, ou bem se o S^* se sobrepõe com outros simplexes, ou o S^* é agudo, então PTAM remove o S^* e cria uma categoria mono-vetor nova C_{N_c+1}, sem nenhum simplex ($N^s_{N_c+1} = 0$), com o vetor de peso $\mathbf{w}_{N_c+1} = \mathbf{I}$ e a predição $P(C_{N_c+1}) = P_d$. Este processo se mostra em detalhe na fig. 3.15.

PTAM necessita, no mínimo, $n+1$ padrões de treinamento com a mesma predição

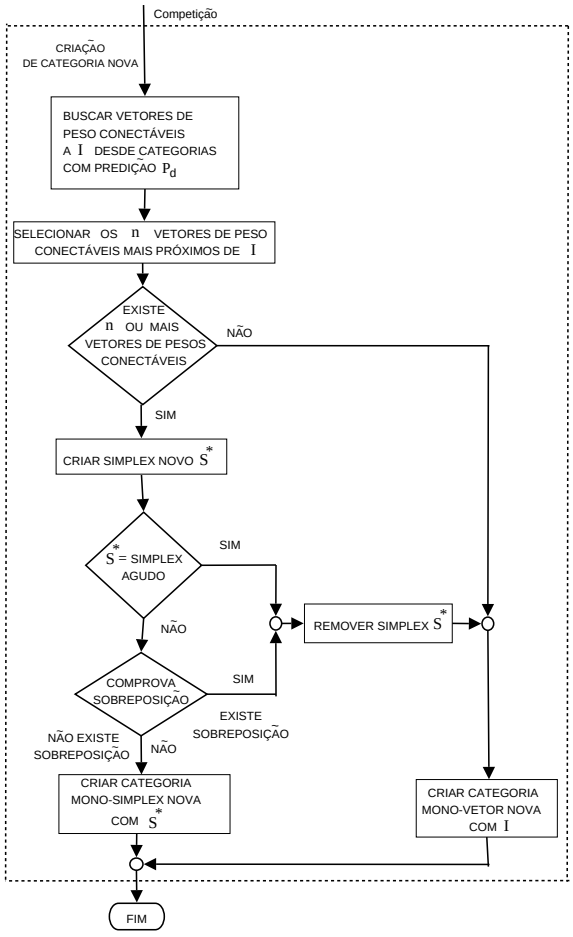


Figura 3.15: Diagrama de fluxo da etapa de criação de categoria nova.

para criar uma categoria politopo. Se não existe nenhuma predição com mais de $n+1$ padrões de treinamento, PTAM cria uma categoria mono-vetor para cada padrão. Durante o processamento, a FEC $T_i^p(\mathbf{I})$ de cada categoria mono-vetor C_i ($N_i^s = 0$) depende da distância euclideana entre o seu vetor de peso $\mathbf{w}_i$ e o $\mathbf{I}$ (eq. 3.18 na subseção 3.2). Então, o $\mathbf{I}$ é codificado pela sua categoria mono-vetor mais próxima, isto é, PTAM se comporta como um classificador do vizinho mais próximo (*Nearest Neighbour Classifier*) [45]. Neste caso, dado que cada predição tem menos de $n+1$ padrões, o tamanho do conjunto de treinamento disponível não é suficiente para aprender corretamente o problema, que é severamente afetado pela "maldição da dimensionalidade" [11].

3.1.7 Resumo da etapa de treinamento

A etapa de treinamento de PTAM, para cada padrão de entrada, pode ser resumida nos seguintes passos:

1. Apresentação de um novo padrão de entrada $\mathbf{I}$.

2. Cálculo das Funções de Escolha de Categorias (FEC): Calcular $T_i^t(\mathbf{I}), \forall i = 1, \ldots, N_c$ (eqs. 3.1 e 3.13) para todas as categorias.

3. Competição: Selecionar a categoria C_I com o $r_I = 0$ e maior FEC: $T_I(\mathbf{I}) = \max\{T_i(\mathbf{I}) : r_i = 0\}_{i=1}^{N_c}$. Se todas as categorias estão rejeitadas ($r_i = 0, i = 1, \ldots, N_c$), então ir para o passo 8 (*Criação de uma categoria nova*).

4. Teste de Predição: Se $P(C_I) \neq P_d$ (a predição associada à C_I é diferente da predição desejada), rejeitar a C_I ($r_I = 1$) e ir para o passo 5 (*Ajuste de Categoria*). Senão, ir para o passo 6 (*Teste de Sobreposição*).

5. Ajuste de Categoria: Se $T_I(\mathbf{I}) = 1$ ($\mathbf{I}$ cai dentro da RRC da categoria C_I), então remover o simplex S_{IJ} para o qual $T_{IJ}(\mathbf{I}) = 1$. Incluir, no conjunto de padrões de entrada, aqueles vetores do simplex S_{IJ} que não pertencem a nenhum outro simplex da C_I. Ir para o passo 3 (*Competição*).

6. Teste de Sobreposição: Expandir a C_I até o $\mathbf{I}$, ou criando um simplex novo S^*, ou substituindo algum vetor de peso $\mathbf{w}^*$ da C_I pelo $\mathbf{I}$. Se a C_I se sobrepõe com outras categorias, depois da expansão, rejeitar a C_I ($r_I = 1$), retornar a C_I para seu estado anterior (suprimir o S^* ou restaurar o $\mathbf{w}^*$, segundo proceda) e voltar ao passo 3 (*Competição*). Senão, ir para o passo 7 (*Ressonância*).

7. Ressonância: Codificar o padrão de entrada pela C_I. Se a expansão provocou a criação de um simplex novo S^* e além do mais $T_I(\mathbf{I}) = \max\{T_i(\mathbf{I})\}_{i=1}^{N_c}$, então suprimir o S^*. Finalizar.

8. Criação de uma categoria nova: Criar um simplex novo S^* com o $\mathbf{I}$ e os seus n vetores de pesos conectáveis mais próximos, com a predição P_d. Se o S^* não é um simplex agudo e não se sobrepõe com outras categorias, criar uma categoria nova mono-simplex C^* com o S^*. Se existe menos de n vetores de pesos conectáveis do $\mathbf{I}$, o S^* é um simplex agudo ou o S^* se sobrepõe com outras categorias, remover o S^* e criar uma categoria nova mono-vetor C^* com o $\mathbf{I}$. Finalizar.

Estes passos se encontram descritos, em pseudocódigo, com mais detalhe no

apêndice I.1.

3.2 Etapa de Processamento

A saída de PTAM para cada padrão de entrada, durante o processamento, é a predição associada da categoria C_i com a maior $T_i^p(\mathbf{I})$ (eq. 3.18). A FEC de uma categoria mono-vetor C_i ($N_i^s = 0$) é calculada usando a distância euclideana entre o seu vetor de peso $\mathbf{w}_i$ e o $\mathbf{I}$, de modo que a $T_i^p(\mathbf{I})$, durante o processamento, define-se da seguinte forma:

$$T_i^p(\mathbf{I}) = \begin{cases} \max\limits_{j=1,\ldots,N_i^s} \{T_{ij}(\mathbf{I})\} & N_i^s > 0 \\ 1 - \dfrac{\|\mathbf{I} - \mathbf{w}_i\|}{\sqrt{n}} & N_i^s = 0 \end{cases} \tag{3.18}$$

onde $T_{ij}(\mathbf{I})$ se define na eq. 3.13. A FEC é decrescente com a distância entre o padrão de entrada $\mathbf{I}$ e o vetor $\mathbf{w}_i$ associado à categoria mono-vetor C_i. O processo competitivo seleciona a categoria C_I com a maior FEC.

$$T_I^p(\mathbf{I}) = \max_{i=1,\ldots,N_c} \{T_i^p(\mathbf{I})\} \tag{3.19}$$

Finalmente, a saída de PTAM é a predição $P(C_I)$ associada à categoria escolhida C_I. A etapa de processamento de PTAM se encontra descrita, em pseudocódigo, no apêndice I.2.

3.3 Complexidade computacional

O custo computacional de cada iteração (processamento de um padrão de treinamento) em PTAM é o seguinte. O custo das funções $\phi_{ijk}(\mathbf{I})$ e $g_{ijk}(\mathbf{I})$ (eqs. 3.4 e 3.5, respectivamente) é $\mathcal{O}(n^3)$, devido a eles usarem determinantes de n-ordem [95]. Para avaliar se o padrão de entrada cai dentro do simplex, devem ser calculados $n+1$ valores de $g_{ijk}(\mathbf{I})$, que é de ordem $\mathcal{O}(n^4)$. Se o padrão cai fora do simplex, mas dentro da hiper-esfera (eq. 3.10), deve ser calculado $\|\mathbf{w}_{ijk}^*\|$ para os $n+1$ hiperplanos em S_{ij}. Assim, devem ser calculados n determinantes D_{1l} (eq. E.2), $\|\mathbf{w}_{ijk}^*\|$ é $\mathcal{O}(n^4)$, logo $d(\mathbf{I}, S_{ij})$ (eq. 3.10) é $\mathcal{O}(n^5)$. Se o padrão cai dentro da hiper-esfera, $d(\mathbf{I}, S_{ij})$

(eq. 3.11) é $\mathcal{O}(n^2)$. Além do mais, $T_{ij}(\mathbf{I})$ e $T_i^t(\mathbf{I})$ (ou $T_i^p(\mathbf{I})$) são $\mathcal{O}(n^5)$. Uma vez que o sistema de n equações lineares em I_{sh} (eq. F.4) é $\mathcal{O}(n^2)$, a função O_{ls} (eq. F.5) é $\mathcal{O}(n^3)$, devido necessitar o cálculo I_{sh} para os $n+1$ hiperplanos de um simplex. O cálculo do conjunto $\mathcal{A}_I(\mathbf{I})$ (secção 3.1.4) é $\mathcal{O}(n^4)$, devido o $(n+N_I^s)\sum_{i=1}^{N_c} N_i^s$ valores de O_{ls} devem ser calculados[7]. A função O_{ss} (eq. F.7) para o teste de sobreposição é $\mathcal{O}(n^5)$, porque devem ser calculados $(n+1)^2$ valores de O_{ls}, então a criação de um novo simplex é $\mathcal{O}(n^5)$. A supressão dos simplexes agudos é de $\mathcal{O}(n^4)$ (eq. 3.16), porque é necessário calcular os n^3 ângulos, os quais são, cada um, $\mathcal{O}(n)$. O passo de substituição de vetor é $\mathcal{O}(n^4)$, devido que devem ser calculados os $n+N_I^s$ valores de $g_{Ijk}(\mathbf{I})$ (que é $\mathcal{O}(n^3)$), mas o teste de sobreposição para o simplex modificado é também de $\mathcal{O}(n^5)$. O custo do teste de predição, os passos de ajuste de categoria e ressonância, não dependem de n. A criação de uma categoria nova necessita $\sum_{i=1}^{N_c}(n+N_i^s)$ cálculos de O_{ls}, que é $\mathcal{O}(n^4)$, para selecionar os vetores conectáveis de $\mathbf{I}$, e o teste de sobreposição para o simplex novo, que é $\mathcal{O}(n^5)$.

No total, o custo de PTAM para cada iteração é $\mathcal{O}(n^5)$. Enquanto este custo alto dificulta o uso de PTAM com os conjunto de dimensões altas, para n baixo sua velocidade é comparável às outras redes ART. Especificamente, a implementação atual de PTAM gasta em torno de 40 seg. para o conjunto de dados CIS (10000 padrões de treinamento e 10000 padrões de teste), em um PC de uso geral (Pentium IV com 512MB RAM), e executa mais rápido que DAM (560 seg.), três vezes mais lento que as redes GAM e EAM (15 seg.), e oito vezes mais lento que FAM e FasArt (5 seg.).

3.4 Configuração experimental

PTAM é testado e comparado com outras redes ART, e não-ART, com uma série de conjunto de dados de referência (*benchmarks*), enumerados na tabela 3.2. Para ilustrar graficamente as categorias politopo criadas por PTAM, a grande parte destes conjuntos de dados são bi-dimensionais. Também empregamos conjuntos de dados com ruído e a sobreposição de categorias, para testar o comportamento de PTAM nestas situações. Por último, PTAM foi testado com conjuntos de dados multi-dimensionais reais. A complexidade de cada um destes conjuntos de dados, segundo se descreve no apêndice B, se mostra nas tabelas 3.3 e 3.4. Os seguintes

[7]Cada categoria politopo C_i com $N_i^s > 0$ é definido por $(n+1)+(N_i^s-1) = n+N_i^s$ vetores de pesos (vértices).

Tabela 3.2: Lista dos conjuntos de dados usados no trabalho experimental. Ver o texto para detalhes.

No.	Nome	Predições	Observações
Conjunto de dados bi-dimensionais			
1	CIS	2	# padrões/predição $\propto$ área
2	Chess	2	50% padrões/predição
3	T3	5	20% padrões/predição
4	T4	5	20% padrões/predição
5	T5	6	# padrões/predição $\propto$ área
6	T6	3	# padrões/predição $\propto$ área
7	T7	3	50%-30%-20% padrões
Conjunto de dados bi-dimensionais com ruído			
8	CIS-ruído	2	nível de ruído 0.01:0.01:0.05
9	T5-ruído	6	nível de ruído 0.01:0.01:0.03
Conjunto de dados bi-dimensionais com sobreposição de categoria			
10	4G1	4	$d = 0.134$ (sobreposição baixa)
11	4G2	4	$d = 0.120$ (sobreposição média)
12	4G3	4	$d = 0.084$ (sobreposição alta)
2D Conjunto de dados com geometria irregular			
13	Form	2	50%-50% padrões
Conjuntos de dados reais multi-dimensionais			
14	PID	2	768 padrões, 8 entradas
15	Abalone	3	4177 padrões, 8 entradas

parágrafos brevemente descrevem cada um destes conjuntos de dados.

3.4.1 Conjuntos de dados bi-dimensionais

Os conjuntos 1-7 (CIS, *Chess* e T3-T7 na tabela 3.2) são conjuntos de dados artificiais bi-dimensionais (fig. 3.16), com geometrias retangular (*Chess* e T4) e circular (CIS, T3 e T5-T7). O conjunto de dados CIS (*Circle-In-The-Square*) [143] é freqüentemente usado na literatura para comparação entre as redes ART, e já o usamos na validação experimental de SAM, no capítulo anterior. Para este conjunto

Tabela 3.3: Medidas de complexidade para os conjuntos de dados bi-dimensionais (*Chess*-T7) e com sobreposição entre as predições (4G1-4G3), usados no trabalho experimental (os valores para o conjunto CIS se referem a uma porcentagem da área de 70%).

	$f1$	$f2$	$f3$	$n2$	$n3$	$t1$	$t2$
CIS	451.3	0.94	16.5	0.81	9E-3	0.9893	2E-4
Chess	2110.7	1.00	909.1	0.76	1E-2	0.9914	2E-4
T3	0.13	0.30	8.62	0.54	8E-3	0.9558	2E-4
T4	0.13	0.25	7.38	0.52	1E-2	0.9362	2E-4
T5	1417.7	1.00	181.8	0.81	3E-2	0.9938	2E-4
T6	0.22	0.50	3.68	0.78	7E-3	0.9740	2E-4
T7	0.21	0.50	5.71	0.77	8E-3	0.9787	2E-4
4G1	4E-4	0.35	2000	0.84	8E-2	0.9986	2E-4
4G2	4E-4	0.36	5000	0.89	0.12	0.9994	2E-4
4G3	9E-4	0.53	2500	1.22	0.31	1.0000	2E-4
Intervalo	$[0, +\infty)$	$[0, 1]$	$[1, +\infty)$	$[0, +\infty)$	$[0, 1]$	$(0, 1]$	$(0, +\infty)$
Ideal	< 1	$\ll 1$	~ 1	$\ll 1$	$\ll 1$	$\ll 1$	$\ll 1$

de dados, os resultados são valores médios dos círculos com tamanho abrangendo de 10% a 70% da área do quadrado. Os conjuntos de dados T3-T7 foram usados em [120] na comparação entre FasArt, FAM e DAM. Em particular, o conjunto T5 é uma generalização a 5 círculos concêntricos, do conjunto de dados CCIS, usado no trabalho experimental de SAM. O conjunto *Chess*, também conhecido como "*Generalized XOR*", é também freqüentemente usado na literatura [66, 128]. O número de padrões, para cada predição, é proporcional à sua área para os conjuntos de dados CIS, T5 e T6 (tabela 3.2), enquanto as predições nos conjuntos de dados *Chess*, T3 e T4 têm o mesmo número de padrões.

A tabela 3.3 mostra as medidas de complexidade para estes conjuntos de dados. A complexidade do conjunto de dados CIS corresponde a um círculo que cobre 70% do quadrado unidade. Destacam as complexidades dos conjuntos *Chess* e T5, especialmente nas medidas $f1$, $f2$, $f3$ e $n3$, comparadas com os conjuntos T3, T4, T6 e T7, que mostram os valores baixos para toda as medidas. CIS também tem uns valores elevados, especialmente de $f1$, em relação aos conjuntos menos

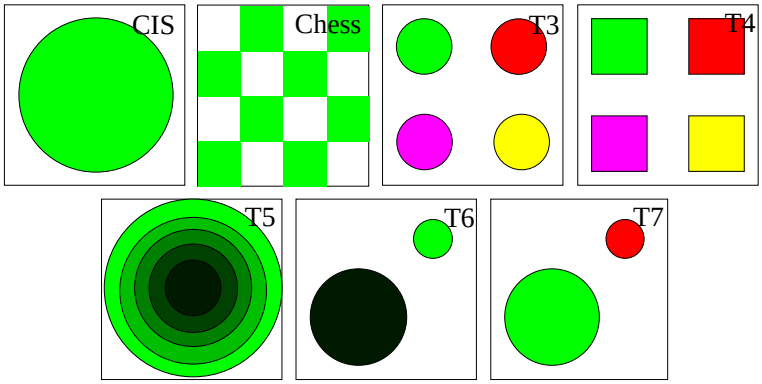


Figura 3.16: Conjuntos de dados bi-dimensionais CIS, *Chess*, T3, T4, T5, T6 e T7 usados no trabalho experimental. Em T6, o número de padrões de cada categoria é proporcional ao seu tamanho (75.8% fora dos círculos, 19.7% no círculo grande e 4.5% no círculo menor). Em T7, as populações são 50%-30%-20% [120].

complexos. Finalmente, T7 parece ser, globalmente, mais complexo que T6, uma vez que proporciona valores superiores de $f3$, $n3$ e $t1$, enquanto que T6, somente supera a T7, em $f1$ e $n2$. Entretanto, as predições com menor superfície, têm mais padrões em T7 que em T6 e, portanto, T7 deveria ser um problema mais fácil de aprender que T6. De todos os modos, a diferença nos seus valores não são muito elevados.

3.4.2 Conjuntos de dados bi-dimensionais com ruído

Os conjuntos de dados 8 e 9 (CIS-ruído e T5-ruído, na tabela 3.2) são gerados adicionando ruído gaussiano aos conjuntos de dados CIS e T5, respectivamente. Vários níveis de ruídos foram utilizados, dados pelo padrão de desvio de uma distribuição gaussiana, com valores nos intervalos de 0.01 a 0.05 e de 0.01 a 0.03, com incremento de 0.01, para CIS-ruído e T5-ruído, respectivamente. No caso de CIS-ruído, as suas medidas de complexidade (tabela 3.4) crescem com o nível de ruído, como é esperado, exceto $f1$, que decresce, e $t1$, que mostra os valores variáveis. A mesma situação se produz em T5-ruído, com $f1$ decrescente e $t1$ variável, mas, neste caso, $f3$ (inversa da máxima eficiência da característica) decresce com o nível de ruído.

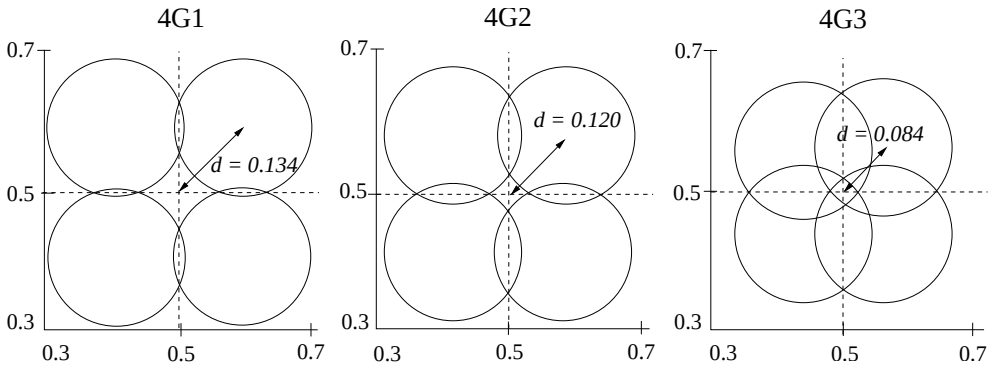


Figura 3.17: Os círculos envolvendo as distribuições gaussianas, com raios 2σ ($\sigma = 0.05$ é o desvio da gaussiana) para os problemas de classificação 4G1, 4G2 e 4G3, consistem de 4 distribuições gaussianas, parcialmente sobrepostas. A figura indica as distâncias d, entre o centro de cada círculo e o centro da unidade quadrado (0.5, 0.5), para o conjunto 4G1 (sobreposição baixa), 4G2 (sobreposição média) e 4G3 (sobreposição alta).

3.4.3 Conjuntos de dados bi-dimensionais com sobreposição de categorias

Os conjuntos de dados 10-12 (4G1, 4G2 e 4G3, na tabela 3.2) são usados na literatura [5] para avaliar os algoritmos de classificação, quando se sobrepõem as predições. Eles têm 4 predições de saída cada um, dado por uma distribuição probabilística gaussiana, com desvio $\sigma = 0.05$ (fig. 3.17). A sobreposição entre eles vai diminuindo com a distância d, entre os seus centros e o centro do quadrado $[0, 1] \times [0, 1]$, que assume os valores $d = 0.134$ (conjunto 4G1), $d = 0.120$ (conjunto 4G2) e $d = 0.084$ (conjunto 4G3). Estes conjuntos de dados mostram os valores muito altos de $n2$, e os valores altos de $f3$ (inversa da máxima eficiência da característica), $n3$ (erro de um classificador do vizinho mais próximo) e $t1$ (inversa da ordem máxima de aderência). Também destaca o reduzido valor de $f1$.

3.4.4 Conjuntos de dados com formas irregulares

Dado que os conjuntos de dados CIS, *Chess* e T5-T7 têm geometrias circulares ou retangulares, completamos nosso experimento através da comparação de PTAM com as melhores redes ART retangulares e circulares, num conjunto de dados bi-dimensionais com geometria irregular. Este conjunto de dados artificial, o qual

Tabela 3.4: Medidas de complexidade para os conjuntos de dados CIS-ruído, T5-ruído, Pima e Abalone.

	$f1$	$f2$	$f3$	$n2$	$n3$	$t1$	$t2$
CIS-r 0.01	3842.8	0.96	15.3	0.94	3E-2	0.9916	2E-4
CIS-r 0.02	3238.9	0.99	26.3	0.97	6E-2	0.9988	2E-4
CIS-r 0.03	235.49	1.00	25.8	1.00	0.10	0.9986	2E-4
CIS-r 0.04	610.33	1.00	20.2	0.97	0.13	0.9978	2E-4
CIS-r 0.05	69.99	1.00	15.1	1.00	0.15	0.9971	2E-4
T5-r 0.01	2916.0	1.00	123.4	0.99	0.11	0.9998	2E-4
T5-r 0.02	3232.9	1.00	62.9	1.00	0.22	1.0000	2E-4
T5-r 0.03	1057.9	1.00	40.2	0.99	0.32	0.9998	2E-4
Pima	1.5	0.61	2.04	0.89	0.29	1.0000	1E-2
Abalone	3.54	0.44	1.45	0.95	0.43	0.9985	1.9E-3
Intervalo	$[0, +\infty)$	$[0, 1]$	$[1, +\infty)$	$[0, +\infty)$	$[0, 1]$	$(0, 1]$	$(0, +\infty)$
Ideal	< 1	$\ll 1$	~ 1	$\ll 1$	$\ll 1$	$\ll 1$	$\ll 1$

chamamos de *Form* (conjunto 13 da tabela 3.2), não é especificamente adequado nem para as categorias de geometria circular nem retangular, como os conjuntos anteriores. O *Form* tem duas predições associadas para os padrões de entrada, que caem dentro e fora da curva na fig. 3.18. Esta curva está baseada num exemplo de máscara usada para a análise das regiões de textura irregular em visão computacional [57], e a sua expressão matemática, em versão paramétrica, é a seguinte:

$$x = 0.5 + r(\theta)cos\theta; \quad y = 0.5 + r(\theta)sin\theta \tag{3.20}$$

$$r(\theta) = max\left\{r_{min}, r_0 + \sum_{n=0}^{N} r_0 f(n)cos[(2n+1)\theta + \psi(n)]\right\}$$

$$0 \leq \theta < 2\pi \tag{3.21}$$

onde $0 \leq r_0 \leq 0.5$ é o raio médio da curva, $0 \leq f(n) \leq 1$ e $0 \leq \psi(n) \leq 2\pi$ são números pseudo-randômicos. O N é a ordem da curva, quando $N = 0$ tem-se um círculo, e a sua irregularidade aumenta com a ordem. Neste experimento, utilizamos o $r_0 = 0.5$ e o $N = 3$. Para tornar mais fácil a rotulação do padrão de entrada, foi requerido um raio mínimo $r_{min} = 0.08$. Finalmente, as coordenadas x, dos pontos da

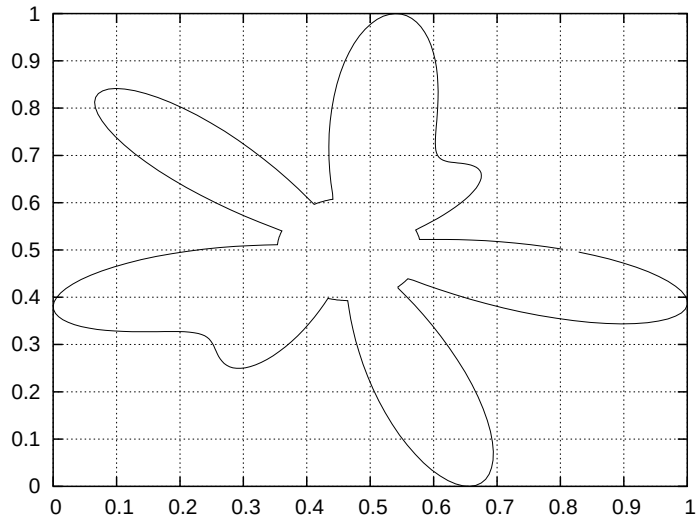


Figura 3.18: Fronteiras entre as duas predições no conjunto de dados *Form*.

curva, foram transladadas e escaladas de modo que o $min\{x\} = 0$ e o $max\{x\} = 1$, e igual para as coordenadas y.

3.4.5 Conjuntos de dados com formas irregulares e irregularidade crescente

A fim de estudar o comportamento das distintas redes ART, quando aumenta a irregularidade das fronteiras entre as predições de saída (ou categorias) de um problema, realizamos experimentos para avaliar estas redes sobre o problema *Form*, variando a ordem N da curva, que determina o seu nível de irregularidade, no intervalo de $N = 2, 3, \ldots, 10$. As curvas, obtidas variando N, se mostram na figura 3.19. Em cada curva, o problema de classificação consiste em discriminar entre os padrões de entrada situados dentro e fora da curva. A tabela 3.5 mostra os valores das medidas de complexidade descritas no apêndice B para o problema *Form*, variando N. Estes valores são variáveis, e em geral, não experimentam um destacado incremento com N. Algumas medidas, no caso de $f2$, $n2$ e $t1$, não variam excessivamente com N. A medida $n3$ (erro de um classificador do vizinho mais próximo) é a que mostra, mais evidente, um comportamento crescente com a ordem da curva, ainda que sempre é inferior ao de T5 (tabela 3.3).

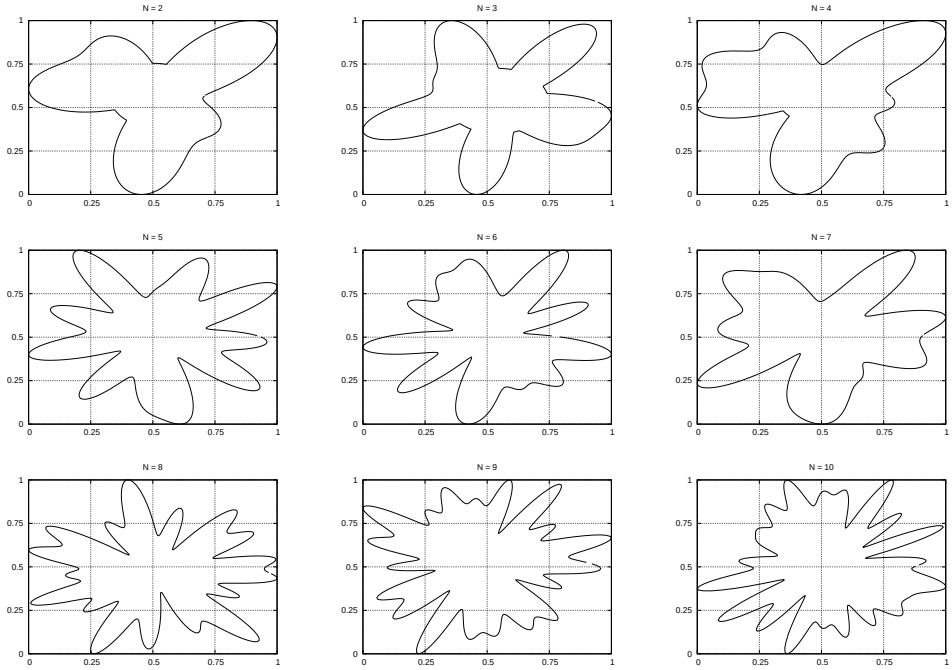


Figura 3.19: Fronteiras entre as duas predições no problema *Form*, variando N.

3.4.6 Conjuntos de dados reais, multi-dimensionais

Os conjuntos de dados 13 e 14 (Pima Indians Diabetes (PID) e Abalone, tabela 3.2), ambos, disponíveis na UCI *Machine Learning Repository* [12], contêm dados reais de 8-dimensões, com 2 e 3 predições de saída, respectivamente, e eles são registrados como de elevada dificuldade na literatura. Para o conjunto de dados PID, FAM obteve 38% de erro [128] e Safe-μARTMAP, 32% de erro [65][8]. Para o conjunto de dados Abalone, MLP obteve um erro de 36% [39] e FAM em torno dos 50% (367 categorias) [5]. Os padrões de entrada no problema Abalone têm 1 componente discreto (3 valores) e 7 componentes contínuos. A saída desejada é um número inteiro, com valores no conjunto $\{1 \ldots 29\}$. Para tarefas de classificação, a documentação do Abalone especifica que se agrupem os padrões de entrada em 3 predições de saída: predição 1, inclui padrões com saída desejada 1-8; saídas desejadas 9 e 10, estão incluídas na predição 2 e, na predição 3, inclui padrões com saídas

[8]Neste artigo, usam 576 padrões de treinamento e 192 padrões de teste. Entretanto, nós usamos 192 padrões de treinamento, 384 de validação e 192 de teste. Ainda que, se poderia pensar, que o tamanho do conjunto de treinamento é reduzido, este fator não importa para a validação comparativa dos distintos classificadores, no nosso trabalho, uma vez que todos usam a mesma configuração experimental.

Tabela 3.5: Medidas de complexidade para os conjuntos de dados *Form*, com $N = 2, \ldots, 10$.

N	$f1$	$f2$	$f3$	$n2$	$n3$	$t1$	$t2$
2	0.97	0.97	35.7	0.79	1.0E-2	0.979	2E-4
3	2.39	0.97	40.0	0.78	1.7E-2	0.987	2E-4
4	0.89	0.97	29.3	0.78	1.2E-2	0.982	2E-4
5	21.46	0.98	50.7	0.80	1.8E-2	0.986	2E-4
6	23.30	0.97	36.7	0.78	1.8E-2	0.988	2E-4
7	8.27	0.95	22.1	0.80	1.7E-2	0.990	2E-4
8	5.43	0.98	62.9	0.78	2.5E-2	0.999	2E-4
9	4.47	0.99	78.7	0.79	2.3E-2	0.992	2E-4
10	11.72	0.98	55.2	0.76	2.3E-2	0.987	2E-4
Intervalo	$[0, +\infty)$	$[0, 1]$	$[1, +\infty)$	$[0, +\infty)$	$[0, 1]$	$(0, 1]$	$(0, +\infty)$
Ideal	< 1	$\ll 1$	~ 1	$\ll 1$	$\ll 1$	$\ll 1$	$\ll 1$

desejadas 11-29. Os valores de complexidade para estes dois conjuntos de dados se mostram na tabela 3.4, e são, em geral, mais baixos que os correspondentes aos conjuntos de dados ruidosos. Apesar de ser problemas próximos à separabilidade linear, como indica o baixo valor de $f1$, e a baixa sobreposição entre as categorias em cada dimensão ($f2$), o valor relativamente elevado de $n2$ (quociente entre as distâncias médias intra- e inter-categoria) e, sobretudo, o valor elevado de $n3$ (erro do vizinho mais próximo), induz a uma idéia da complexidade real destes dois conjuntos.

3.5 Resultados

Desenvolvemos experimentos comparando PTAM com as redes ART mais populares com estes conjuntos de dados, incluindo *Fuzzy* ARTMAP [28], *Gaussian* ARTMAP [144], *Distributed* ARTMAP [36], *Ellipsoid* ARTMAP [3] e *FasArt* [17]. Também registramos os resultados obtidos por SVM [136], por considerarmos este, um algoritmo de referência para problemas de classificação, tal como fizemos com SAM. Os valores dos parâmetros ajustáveis dos diferentes algoritmos são registrados na tabela 3.6. A *Lei de Weber* e a *Choice-By-Difference*, descritas na secção 1.3.2, foram investigados como funções de escolha por FAM. Os resultados foram muito

similares, mas ligeiramente melhores, quando utilizamos a *Lei de Weber* (os resultados alcançados por FAM com as duas funções de escolha se mostram na tabela 3.7; as outras tabelas de resultados somente reúnem o melhor valor alcançado por FAM). Para os classificadores ART, incluindo PTAM, foram executadas 5 épocas de treinamento para os conjuntos de dados 1-7, e 1 época de treinamento para os conjuntos de dados 8-14, para reduzir o custo computacional das simulações. FasArt utilizou $\gamma_a = \gamma_b = 1$ para os conjuntos de dados PID e Abalone, porque, usando $\gamma_a = 10$ (valor empregado em [120]), as categorias eram muito estreitas e elas não cobriam todo o espaço de entrada. A SVM foi executada usando a biblioteca de algoritmos de aprendizagem *Torch* [40]. Foram investigados os núcleos gaussiano, polinomial, sigmóide e escalar. Os melhores resultados foram obtidos com os núcleos gaussianos. O valor de desvio, com o desempenho médio melhor sobre os conjuntos de validação, foi selecionado para cada conjunto de dados (conforme tabela 3.8).

Tabela 3.6: Os valores dos parâmetros para os diferentes classificadores.

FAM	$\alpha = 0.01, \beta = 1$, $\rho_{ab} = 1$, Lei de Weber e Choice-By-Difference
GAM	$\gamma = 0.2$, $\gamma = 0.8$ (PID e Abalone)
DAM	$\alpha = 0.01, \beta = 1$, $p = 10$ (representante de aumento de contraste)
EAM	$\mu = 0.5, D = \sqrt{n}/\mu, \beta = 1, \alpha = 0.1$
FasArt	$\bar{\rho}_b = 1, \beta = 1, \gamma_a = \gamma_b = 10$, $\gamma_a = \gamma_b = 1$ (PID e Abalone)
parâmetros comuns das redes ART	$\bar{\rho} = 0.00 : 0.02 : 0.98$ 1-5 épocas de treinamento
SVM	$C = 0.01, 0.1, 1, 100, 1000$, Núcleo polinomial, sigmóide, escalar e gaussiano com desvio $\in \{0.1 : 0.1 : 1, 2.5, 5, 7.5, 10, 12, 15, 17, 20\}$

Gostaríamos de enfatizar que o uso de várias épocas de treinamento, no caso das redes ART, não significa que estas redes necessitem de várias épocas para aprender o problema. Realmente, estas redes permitem aprendizagem *on-line*, de modo que podem aprender o problema com somente uma apresentação do conjunto de treinamento, propriedade que não exibem os classificadores com treinamento *off-line*, como MLP, RBF ou SVM. No entanto, dado que usualmente os resultados obtidos pelas redes ART com várias épocas são superiores aos obtidos com somente uma época, e com o objetivo de compará-los com os resultados alcançados pela SVM, as

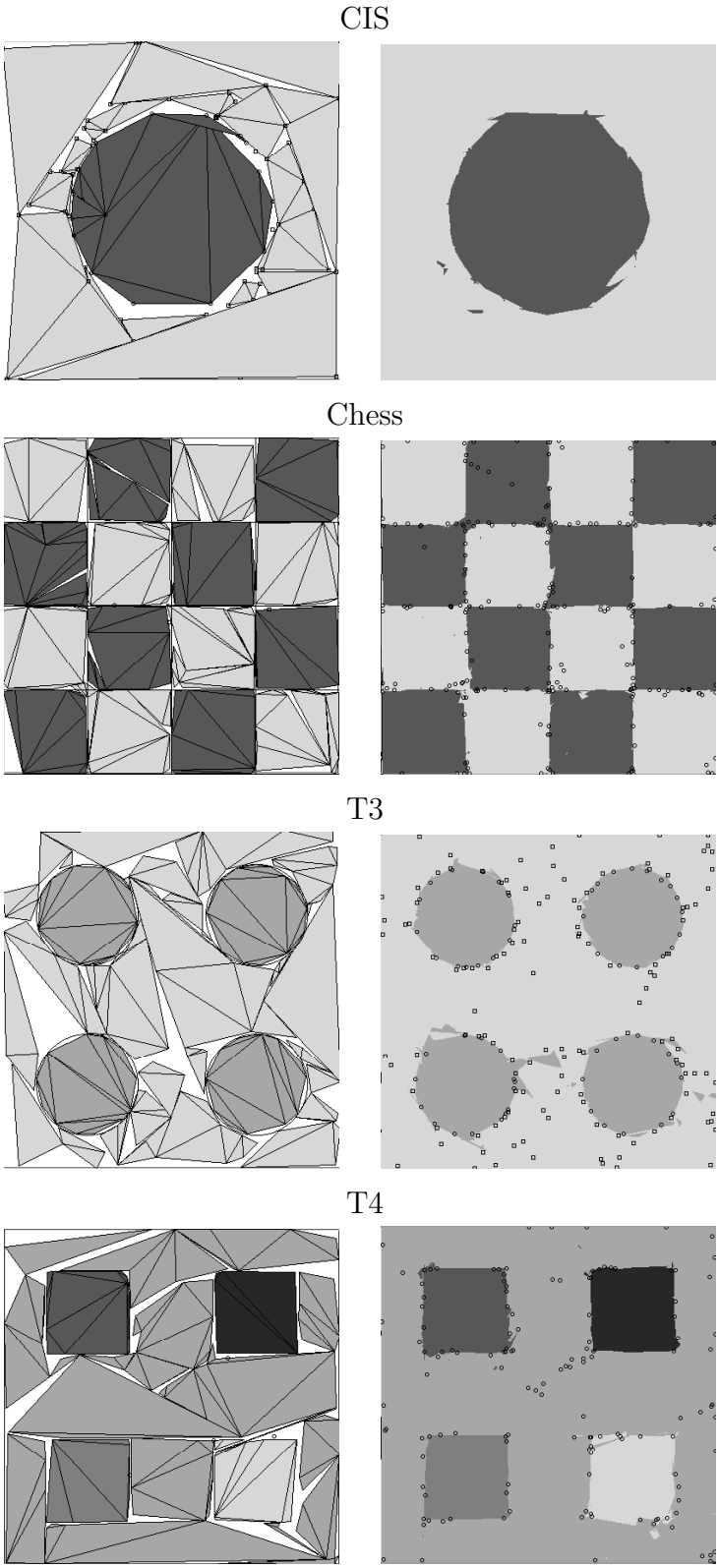


Figura 3.20: Exemplos de categorias criadas por PTAM (painéis da esquerda) e regiões de classificação (painéis da direita) para os conjuntos de dados CIS e *Chess*. Os painéis da direita também apresentam os vetores de pesos criados por PTAM (círculos pequenos).

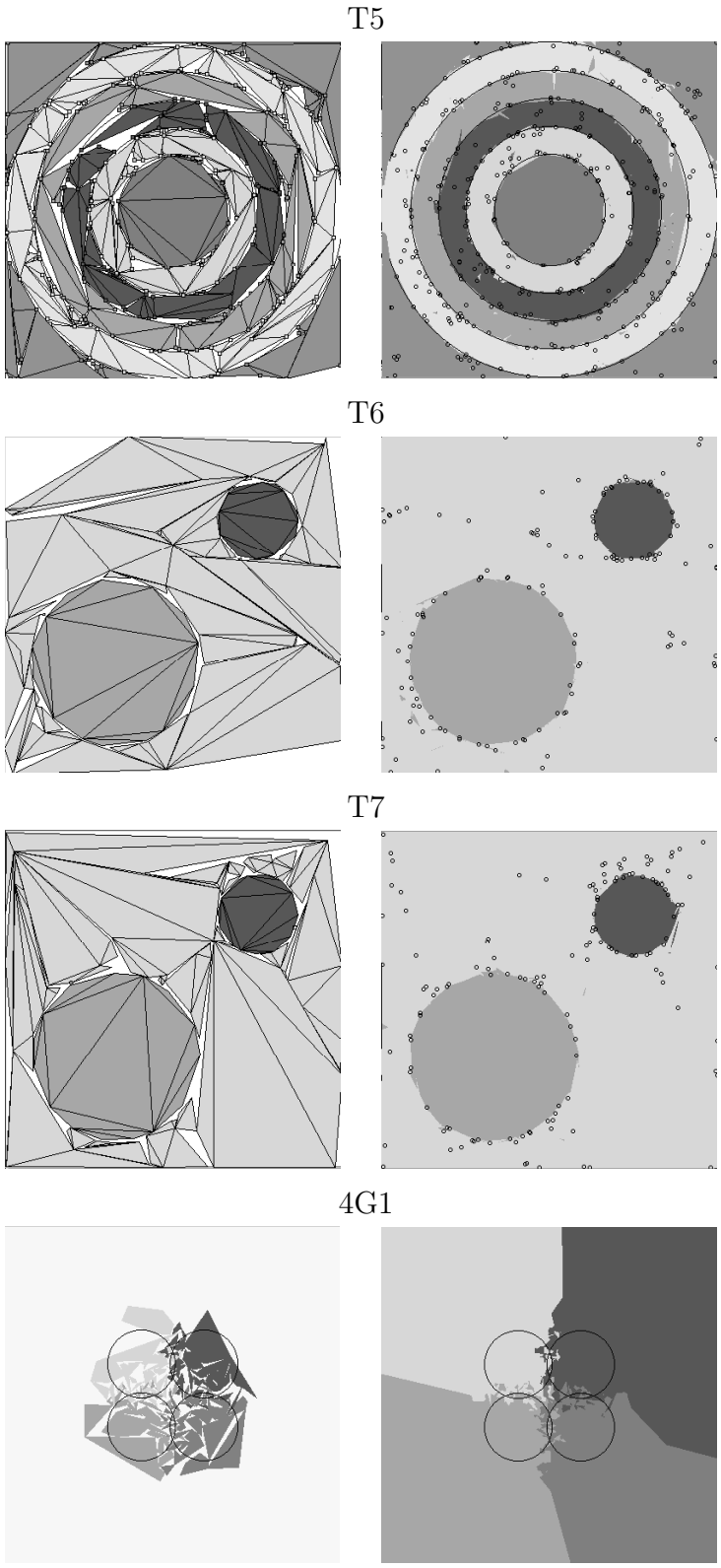


Figura 3.21: Exemplo de categorias criadas por PTAM (painéis esquerdos) e regiões de classificação (painéis direitos) para os conjuntos de dados T5, T6, T7 e 4G1.

Tabela 3.7: O erro (em percentual) e o número de categorias criadas por FAM, usando a *Lei de Weber* (WL) e *Choice-by-difference* (CBD), como funções de escolha nos distintos problemas bi-dimensionais.

FAM	WL		CBD	
	$\%\epsilon$	#C	$\%\epsilon$	#C
CIS	2.15	171	2.21	172
Chess	1.81	96	1.70	88
T3	2.16	100	2.32	114
T4	0.56	35	1.02	41
T5	6.04	438	6.02	443
T6	1.64	81	1.60	86
T7	1.61	86	1.63	86

redes ART, e PTAM entre elas, também empregaram 5 apresentações do conjunto de treinamento.

Alguns autores [65, 128] registram os resultados obtidos pelas redes ART com a vigilância base zero ($\bar{\rho} = 0$). Entretanto, o erro de classificação e o número de categorias obtidos por uma rede ART depende fortemente do valor de vigilância, como mostram as figs. 3.22 a 3.25 mais adiante. Nos nossos experimentos utilizamos o método de validação-cruzada para determinar o melhor $\bar{\rho}$ para cada conjunto de dados. Para cada um dos conjuntos de dados 1-12, criamos 20 trios, cada um composto de um conjunto de treinamento, um de validação e um de teste, contendo cada um deles 10000 padrões gerados randomicamente. Para cada classificador ART (FAM, GAM, DAM, EAM e FasArt), testamos valores de $\bar{\rho}$ no intervalo de 0.00 a 0.98, com um incremento de 0.02. Para cada valor de vigilância, foram treinados 20 versões de classificadores, um para cada conjunto de treinamento, e eles foram testados no seu conjunto de validação correspondente. As figs. 3.22 a 3.25 representam a taxa de erro (em percentual) em função do número de categorias, que denotaremos por #C (número de vetores de pesos #W para PTAM ou vetores de suporte #SV para SVM), para os diferentes valores de vigilância e para cada conjunto de dados, pela média dos 20 conjuntos de validação. O valor de vigilância $\bar{\rho}$ que fornece a melhor relação entre o erro e o número de categorias no conjunto de validação, foi selecionado para cada rede ART, conforme tabela 3.8. Este valor

Tabela 3.8: Valores selecionados para os parâmetros ajustáveis (ρ para as redes ART, C e o desvio σ em SVM) nos diferentes conjuntos de validação. CIS-r e T5-r significam CIS-ruído e T5-ruído, respectivamente.

	FAM	GAM	DAM	EAM	FasArt	SVM	
Data set	$\bar{\rho}$	ρ	$\bar{\rho}$	ρ	ρ	C	σ
CIS	0.88	0.04	0.9	0.84	0.68	100	0.1
Chess	0.46	0.06	0.7	0.94	0.92	1000	0.1
T3	0.2	0.10	0.6	0.94	0.9	1000	0.1
T4	0.3	0.12	0.76	0.94	0.82	1000	0.1
T5	0.92	0.02	0.94	0.96	0.94	100	0.1
T6	0.54	0.04	0.88	0.92	0.84	1000	0.1
T7	0.48	0.1	0.78	0.82	0.86	1000	0.1
CIS-n 0.01	0.9	0.1	0.9	0.94	0.8	1000	0.9
0.02	0.9	0.06	0.9	0.92	0.88	100	0.9
0.03	0.92	0.02	0.9	0.9	0.9	10	0.7
0.04	0.9	0.02	0.9	0.9	0.92	10	0.9
0.05	0.92	0.02	0.9	0.9	0.92	1	0.5
T5-n 0.01	0.9	0.36	0.95	0.96	0.88	1000	0.7
0.02	0.88	0.02	0.95	0.02	0.88	10	0.4
0.03	0.92	0.02	0.95	0.02	0.32	10	0.2
Form	0.36	0.12	0.88	–	–	–	–
	0.92	–	0.94	–	–	–	–
4G1	0.9	0.02	0.75	0.96	0.54	10	0.9
4G2	0.94	0.02	0.65	0.8	0.3	10	0.9
4G3	0.94	0.02	0.1	0.9	0.02	1	0.7
PID	0.8	0.2	0.6	0.6	0.6	1000	10
Abalone	0.95	0.2	0.8	0.4	0.0	1000	2.5

determina o ponto de operação (marcado com um quadrado vazio nas figs. 3.22-3.25), para a etapa de teste de cada classificador ART, em cada conjunto de dados. Por exemplo, o ponto de operação de FAM para o conjunto de dados CIS (fig. 3.22 painel superior) usa $\bar{\rho} = 0.88$ e obtém um erro de 2.15% e 171 categorias sobre o conjunto de validação. GAM é uma exceção perceptível, devido seu erro e #C aumentarem com a vigilância, então seu valor selecionado é, geralmente, próximo a zero.

Os 20 classificadores treinados, com o valor da vigilância $\bar{\rho}$ selecionado, foram testados nos 20 conjuntos de testes. O erro médio e #C sobre os conjuntos de dados 1-7 e o seu desvio padrão, estão registrados nas duas linhas inferiores da tabela 3.9 e representados na fig. 3.26. A mesma metodologia experimental é usada para os conjuntos de dados bi-dimensionais com ruído– CIS-ruído e T5-ruído–, e para os conjuntos de dados com sobreposição 4G1, 4G2 e 4G3.

Os painéis esquerdos das figs. 3.20 e 3.21 apresentam as Regiões de Representação das Categorias (RRCs) criadas por PTAM para os conjuntos de dados CIS, *Chess*, T3, T4, T5, T6, T7 e 4G1. As fronteiras das categorias (segmentos de linha em $\mathbb{R}^2$) compõem uma aproximação linear por partes das fronteiras entre as predições de saída. As categorias politopo podem não cobrir totalmente o espaço de entrada, quando não é necessário para classificar corretamente o conjunto de treinamento, como na esquina inferior esquerda no conjunto de dados *Chess* (fig. 3.20, painel inferior esquerdo). Os painéis da direita apresentam os vetores de pesos e as fronteiras entre as predições criadas por PTAM. Nestas figuras é observada a elevada precisão com a que PTAM aprende estas fronteiras, apesar de algumas imprecisões em T3. Também se aprecia que a mesma representação se poderia obter com um menor número de vetores.

3.5.1 Conjuntos de dados bi-dimensionais

A tabela 3.9 registra os resultados dos testes dos sete classificadores para os conjuntos de dados bi-dimensionais 1-7. A SVM obtém o menor erro para todos eles, exceto para T4, onde DAM obtém o menor erro. Entretanto, é necessário mais vetores de suporte que categorias das redes ART para os conjuntos de dados CIS, *Chess* e T5. A seguir, analisamos, por separado, o comportamento das redes ART em cada conjunto de dados.

PTAM é a rede ART com menor erro (1.0%) no CIS, menor que as redes circulares GAM e EAM, e seu #W (número de vetores de pesos ou vértices $\mathbf{w}$) é o segundo menor deles (102). As outras redes ART necessitam mais categorias para obter uma taxa de erro igual. GAM obtém o menor número de categorias #C (43), com maior erro. FAM e EAM têm o mesmo erro que GAM (1.3%), porém muito mais categorias. DAM obtém taxa de erro igual e mais categorias que FAM e EAM. FasArt tem o maior erro (1.8%) e número de categorias (350).

A tabela 3.8 registra os valores de vigilância para cada classificador e conjunto de dados. A maioria dos valores de vigilância são altos, que confirma que os melhores resultados são obtidos limitando o tamanho da categoria, ao invés de usar $\bar{\rho} = 0$. GAM é uma destacável exceção, devido a sua taxa de erro e o número de categorias (#C), ambos aumentarem com o valor de vigilância (figs. 3.22 a 3.25). Assim, o ponto de operação selecionado é frequentemente $\bar{\rho} = 0$. Ao contrário do comportamento comum das redes ART, que normalmente obtêm o menor erro com

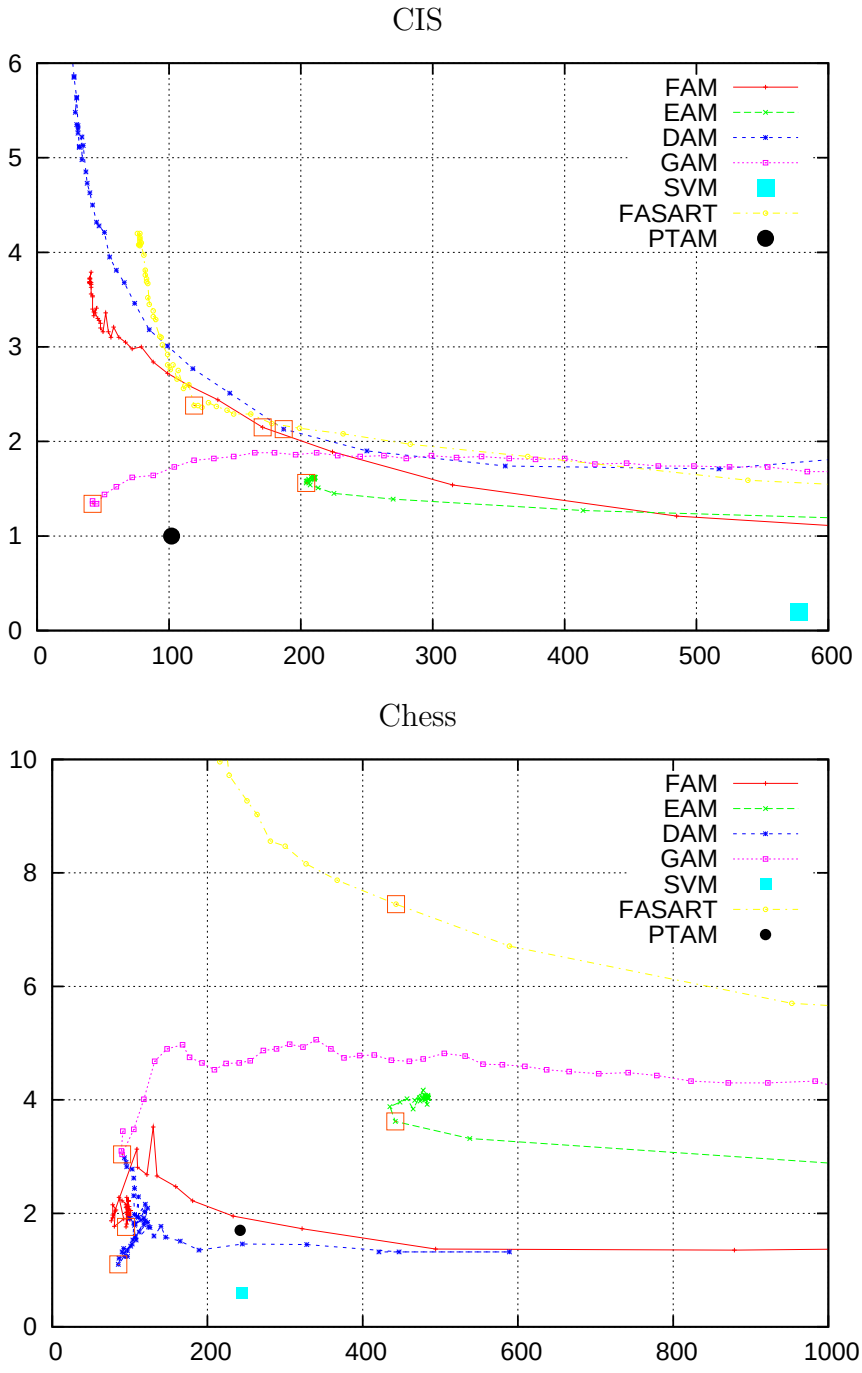


Figura 3.22: O erro (em %) de classificação em função do número de categorias nos conjuntos de validação para cada classificador nos conjuntos de dados CIS e *Chess*. O ponto de operação selecionado está marcado com um quadrado vazio.

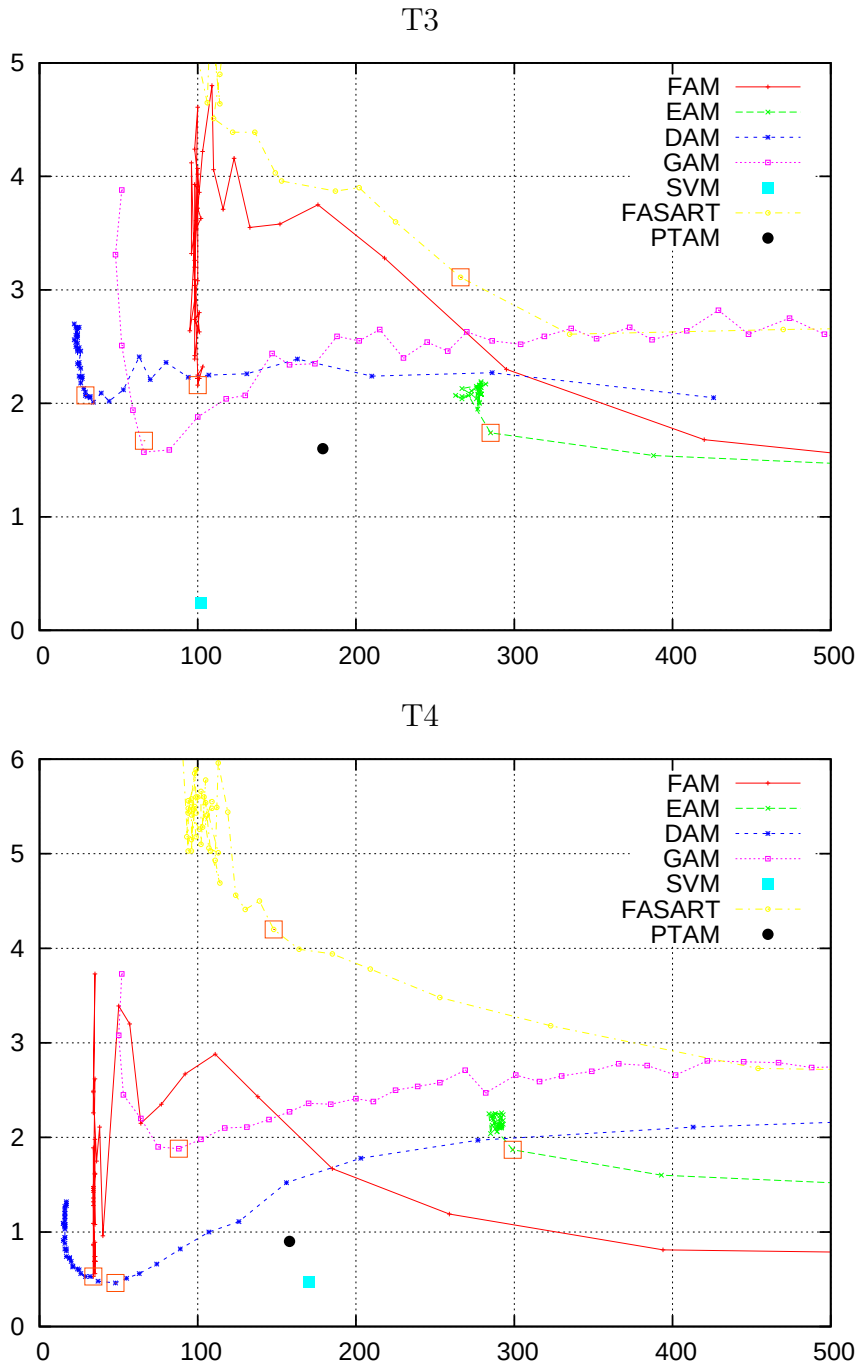


Figura 3.23: O erro (em %) de classificação em função do número de categorias nos conjuntos de validação para cada classificador nos conjuntos de dados T3 e T4.

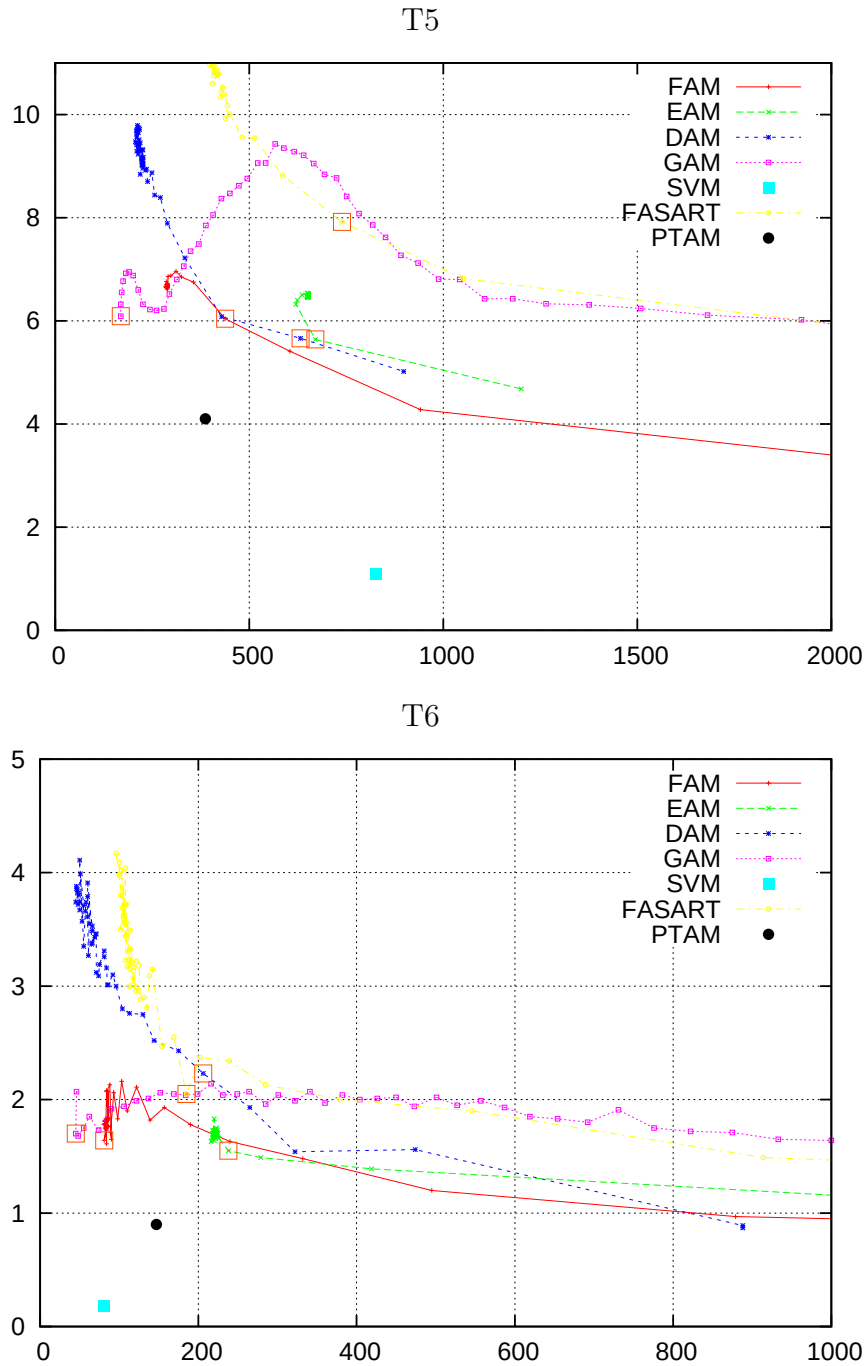


Figura 3.24: O erro (em %) de classificação em função do número de categorias nos conjuntos de validação para cada classificador nos conjuntos de dados T5 e T6.

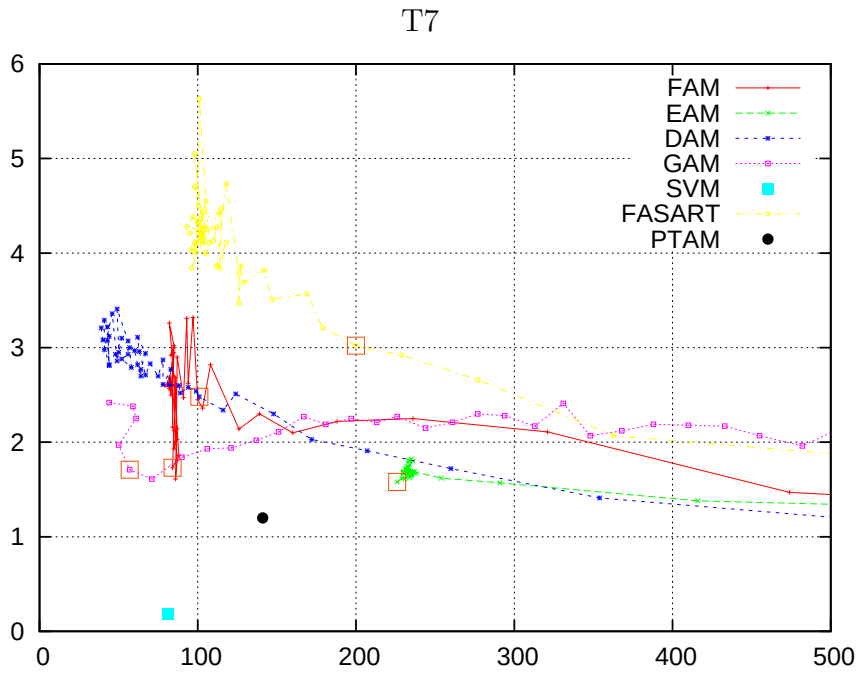


Figura 3.25: O erro (em %) de classificação em função do número de categorias nos conjuntos de validação, para cada classificador, no conjunto de dados T7.

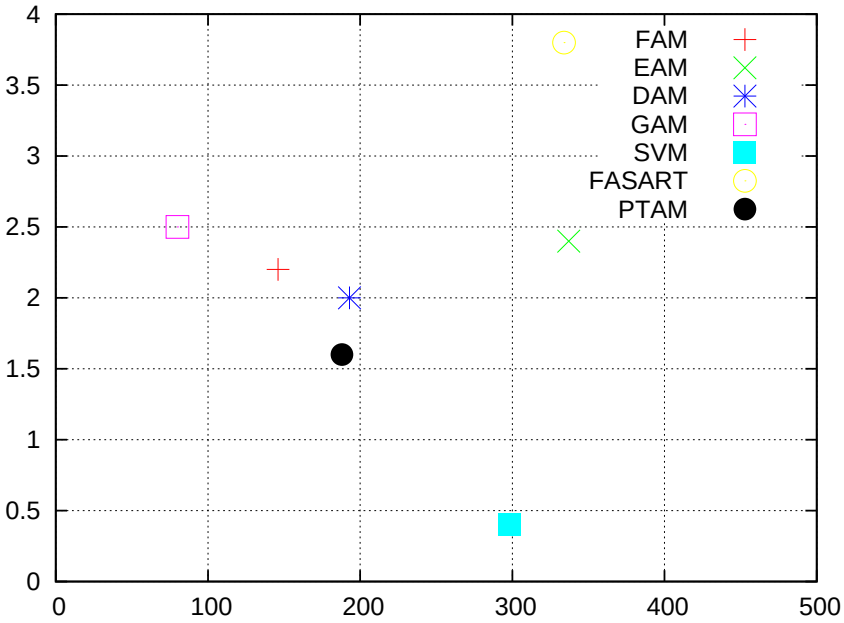


Figura 3.26: Erro de teste (em %) em função do número de categorias médios sobre os conjuntos de dados CIS, *Chess* e T3-T7.

maior número de categorias e vice-versa, tendo seu valor de vigilância, geralmente, $\bar{\rho} \simeq 0$.

Tabela 3.9: Erro de teste (ϵ, em %) e #C (número de vetores de pesos #W para PTAM e número de vetores de suporte #SV para SVM) obtidos por cada classificador com os conjuntos de dados bi-dimensionais(1-7). O melhor ϵ e #C obtidos por um classificador ART e um não-ART estão em negrito.

	PTAM		FAM		GAM		DAM		EAM		FasArt		SVM	
	ϵ	#W	ϵ	#C	ϵ	#C	ϵ	#C	ϵ	#C	ϵ	#C	ϵ	#SV
CIS	**1.0**	102	1.3	188	1.3	**43**	1.3	238	1.3	195	1.8	350	**0.2**	578
Chess	1.7	242	1.8	95	3.0	90	**1.1**	**89**	3.4	443	4.7	443	**0.6**	245
T3	**1.6**	179	2.2	100	**1.6**	67	2.1	**42**	1.7	285	3.1	266	**0.2**	102
T4	1.0	158	0.5	**34**	1.9	89	**0.4**	48	1.9	300	4.1	149	0.5	170
T5	**4.1**	387	5.8	439	6.1	**169**	5.6	630	5.6	670	7.8	739	**1.1**	827
T6	**0.9**	147	1.6	81	1.7	**45**	1.6	193	1.5	239	2.1	185	**0.2**	81
T7	**1.2**	141	1.9	85	1.8	**57**	1.9	113	1.7	226	3.1	207	**0.2**	81
Média	**1.6**	188	2.2	146	2.5	**80**	2.0	193	2.4	337	3.8	334	**0.4**	298
STD	**1.1**	100	1.7	137	1.7	**44**	1.7	206	1.5	167	2.0	205	**0.3**	291

DAM é o melhor classificador ART para o conjunto de dados *Chess* (tabela 3.9), tanto em erro (1.1%) e como em #C (89), e PTAM obtém o segundo menor erro (1.7%). FAM obtém maior erro que PTAM (1.8%), apesar da geometria retangular deste conjunto de dados. FasArt e as redes circulares GAM e EAM, que não são adequadas para a geometria de *Chess*, obtêm 3% de erro ou mais. A rede PTAM também cria menos vetores de pesos que categorias em EAM ou FasArt.

PTAM obtém o menor erro (1.6%) no conjunto de dados T3. As redes GAM e EAM, apesar da geometria circular de T3, não são melhores que PTAM. O #W de PTAM (179) está na média entre a rede GAM (67) e EAM (285). As redes FAM e a DAM obtêm erros acima de 2%, provavelmente devido à sua geometria retangular, e DAM cria menos categorias. FasArt obtém a maior taxa de erro (3.1%) com muitas categorias, entretanto, menos que EAM.

FAM e DAM obtêm o menor erro (0.5%) entre os classificadores ART, ainda menor que SVM, com poucas categorias, para o conjunto de dados retangular T4.

Neste caso, DAM obtém ligeiramente mais categorias que FAM. PTAM obtém mais erro (1.0%) que as redes ART retangulares, e menos erro que as redes circulares GAM e EAM (1.9%). Como nos conjuntos de dados anteriores, EAM cria mais categorias que GAM. A rede FasArt também cria menos categorias que PTAM, no entanto com o maior erro.

PTAM obtém o menor erro entre as redes ART (4.1%), e o segundo menor #W (387), no conjunto de dados T5, que é o mais complexo. Apenas GAM cria menos categorias que PTAM, entretanto com maior erro (6.1%). As redes FAM, DAM e EAM obtêm mais erro e categorias que PTAM. FasArt obtém o maior erro (7.8%) e #C.

Os conjuntos de dados T6 e T7 têm a mesma geometria, no entanto eles diferem nos números de padrões de cada predição. Todos os classificadores têm menor erro em T6, coincidindo com as medidas de complexidade, que alcançam um valor superior para T7 (tabela 3.3). PTAM obtém de maneira mais evidente o menor erro (0,9% e 1,2%, respectivamente) entre os classificadores ART para ambos os conjuntos de dados. O seu #W está na média entre as redes FAM-GAM (baixo #C) e EAM-FasArt (alto #C). As redes circulares (GAM e EAM) não obtêm menor erro que as retangulares (FAM e DAM) nestes conjuntos de dados. FasArt obtém o maior erro para ambos os conjuntos de dados.

Tabela 3.10: Erro (ϵ, em %) e #C obtidos em [120] por FAM, DAM e FasArt nos conjuntos CIS e T3-T7.

	FAM		DAM		FasArt	
	Erro	#C	Erro	#C	Erro	#C
CIS	6.3	23.2	11.1	12.3	3.7	63.4
T3	11.5	53.1	12.1	33.4	7.3	122.3
T4	3.8	31.9	4.3	30.7	8.2	139.2
T5	15.5	124.8	19.1	125.4	4.0	278.7
T6	5.2	21.2	21.1	8.6	3.6	62.2
T7	5.4	20.7	4.7	12.9	3.7	57.7

Os resultados para os conjuntos de dados CIS e T3-T7 são diferentes aos obtidos em [120] (tabela 3.10), onde os erros registrados são maiores e o número de categorias é menor, que em nossos experimentos. Algumas possíveis razões são: i) as

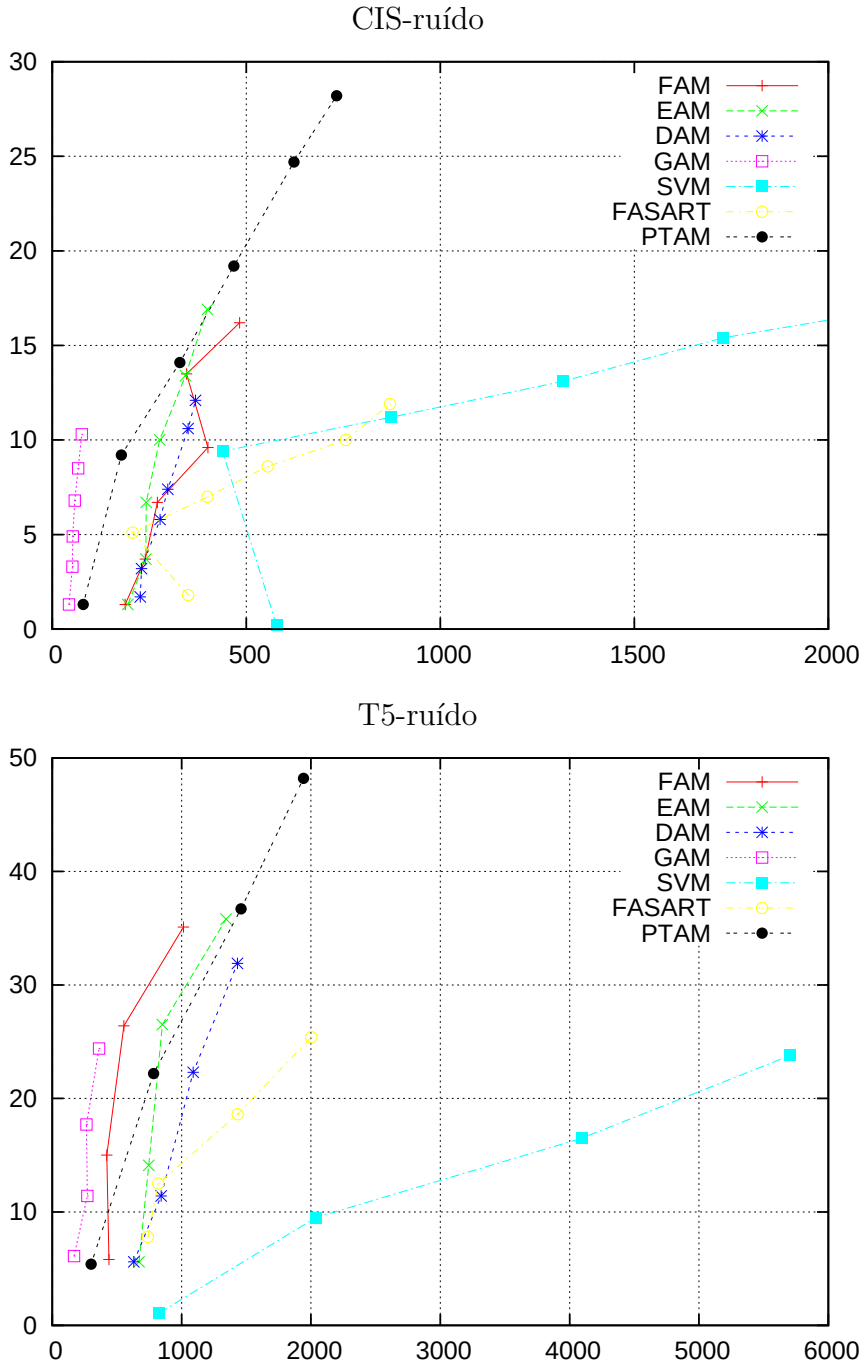


Figura 3.27: A taxa média de erro (em %) em função do número de categorias sobre os conjuntos de teste obtidos pelos diferentes classificadores para os conjuntos de dados CIS-ruído e T5-ruído. Cada ponto corresponde a um nível de ruído diferente (0.00:0.01:0.05 para CIS-ruído e 0.00:0.01:0.03 para T5-ruído).

diferenças nos números de padrões de treinamento e de teste (2000 padrões em [120], 10000 padrões nos nossos experimentos) e nas populações das categorias para alguns conjuntos de dados; ii) otimizamos o valor de vigilância para as redes ART clássicas ($\bar{\rho} \neq 0$ em nossas simulações, exceto para GAM), enquanto que em [120] usam $\bar{\rho} = 0$ para FAM, DAM e FasArt. Por isso, obtivemos menor erro, com maior #C em FasArt, FAM e DAM. Isto pode ter sido a causa dos bons resultados alcançados por FAM e DAM, comparado ao FasArt, em relação ao artigo citado.

	PTAM			FAM		DAM	
Conjunto de Dados	N_s	N_w	N_p/n	N_c	N_p/n	N_c	N_p/n
CIS	78	102	**258**	188	**376**	238	**1071**
Chess	183	242	**608**	95	**190**	89	**401**
T3	134	179	**447**	100	**200**	42	**189**
T4	122	158	**402**	34	**68**	48	**216**
T5	282	387	**951**	439	**878**	630	**2835**
T6	114	147	**375**	81	**162**	193	**869**
T7	101	141	**343**	85	**170**	113	**509**
Média	145	194	**483**	146	**292**	193	**870**
STD	69	95	233	137	274	206	926

Tabela 3.11: Número de categorias (N_c), simplexes (N_s) e parâmetros N_p divididos por n dimensão do espaço de entrada ($n = 2$ para todos os conjuntos de dados na tabela) armazenados por PTAM, FAM e DAM.

É interessante comparar o número de parâmetros (N_p) armazenados pelas diferentes redes. No caso de PTAM, $N_p = (n+2)N_s + nN_w$, sendo N_s e N_w os números de vetores de pesos e simplexes, respectivamente (ver apêndice G). Dado que cada categoria em PTAM requer vários vetores, usualmente armazena mais parâmetros que as clássicas redes ART, o qual é esperável, uma vez que as categorias politopo são mais flexíveis que as categorias hiper-retângulares. A tabela 3.11 N_p, dividido por n (dimensão do espaço de entrada), para PTAM, FAM e DAM, nos sete conjuntos de dados bi-dimensionais. Em média, PTAM armazena 65% mais parâmetros que FAM, para o qual $N_p = 2nN_c$ (N_c é o número de categorias). Por outro lado, o número de parâmetros de DAM é dado por $N_p = (4n + 1)N_c$, e este armazena 80% mais parâmetros que PTAM (DAM armazena menos parâmetros que PTAM somente em *Chess*, T3 e T4, devido a DAM criar menos categorias que vetores

PTAM).

3.5.2 Conjuntos de dados bi-dimensionais com ruído

Todos os classificadores aumentam os seus erros e o número de categorias com o nível de ruído, como está registrado na tabela 3.12 e na fig. 3.27 para os conjuntos de dados CIS-ruído e T5-ruído, mas os seus resultados são diferentes. A tabela também apresenta o erro de *Bayes* [142], calculado segundo se indica no apêndice C, para ambos conjuntos de dados. A vigilância selecionada para estes resultados encontram-se na tabela 3.8. GAM obtém o menor erro (10% com o nível de ruído 0.05 máximo) em CIS-ruído para todos os níveis. Em T5-ruído, GAM alcança um erro ligeiramente maior que SVM, que alcança o limite de *Bayes*. Entretanto, a SVM utiliza um enorme número de vetores de suporte, enquanto o #C de GAM aumenta muito lentamente com o nível de ruído. Notavelmente, GAM cria menos categorias em CIS-ruído com o nível 0.05 que os outros classificadores sem ruído. O melhor comportamento de GAM pode estar relacionado com a natureza gaussiana do ruído adicionado. FAM, DAM e EAM registram números de categorias similares, contudo DAM alcança o erro ligeiramente menor que FAM e EAM. FasArt alcança o erro similar a DAM com o nível máximo de ruído, mas com mais categorias. PTAM alcança o erro e o número de vetores mais altos que as outras redes ART. De fato, PTAM cria um número de categorias bastante elevado, entretanto menor que da rede FasArt e da SVM.

3.5.3 Conjuntos de dados com sobreposição de predições

Nos conjuntos de dados de categorias sobrepostas 4G1, 4G2 e 4G3, PTAM também registra um baixo desempenho, como mostra a tabela 3.13. A primeira linha desta tabela registra o erro de *Bayes* (apêndice C) para cada conjunto de dados. A rede GAM exibe o melhor comportamento global, devido a ele alcançar o erro *Bayes* em cada conjunto de dados com um número muito baixo de categorias. FasArt e SVM também alcança o erro de *Bayes*, mas sua complexidade (determinada pelos seus números de categorias e vetores de suporte, respectivamente) é muito maior, especialmente a da SVM. DAM obtém mais erro que GAM com uma complexidade similar. EAM obtém mais erro que DAM, com muito mais categorias. FAM claramente alcança mais erro que EAM e DAM, com um número de catego-

Tabela 3.12: Os resultados dos testes para os conjuntos de dados 2-D com ruído, juntamente com o erro de *Bayes* (apêndice C). O melhor erro (ϵ, em %) e #C obtidos por um classificador ART e não-ART estão em negrito.

		CIS-ruído					T5-ruído		
		0.01	0.02	0.03	0.04	0.05	0.01	0.02	0.03
Bayes	ϵ	1.7	3.3	5.0	6.5	8.0	8.2	16.1	23.8
PTAM	ϵ	9.2	14.1	19.2	24.7	28.3	21.9	36.6	46.8
	#W	178	329	468	623	733	759	1421	1890
FAM	ϵ	3.7	6.7	9.6	13.5	16.2	15.0	26.4	35.1
	#C	240	271	401	346	483	421	554	1014
GAM	ϵ	**3.3**	**4.9**	**6.8**	**8.5**	**10.3**	**11.4**	**17.7**	**24.4**
	#C	**52**	**53**	**58**	**67**	**76**	**272**	**265**	**362**
DAM	ϵ	3.2	5.8	7.4	10.6	12.1	11.5	22.3	31.9
	#C	230	278	298	350	369	842	1090	1432
EAM	ϵ	3.7	6.7	10.0	13.4	16.9	14.1	26.5	35.8
	#C	241	242	277	343	400	746	851	1344
FasArt	ϵ	5.1	7.0	8.6	10.0	11.9	12.5	18.6	25.4
	#C	207	400	555	757	870	819	1433	2002
SVM	ϵ	9.4	11.2	13.1	15.4	16.7	**9.5**	**16.5**	**23.8**
	#SV	439	873	1315	1728	2105	2039	4094	5707

rias intermediário. PTAM alcança o maior erro entre todos os classificadores, no entanto, cria menos vetores que categorias de FasArt e suporte de vetores de SVM. As vigilâncias selecionadas para as redes ART estão registradas na tabela 3.8.

3.5.4 Conjuntos de dados com formas irregulares

Nos experimentos realizados com conjuntos de dados *Form*, usamos FAM e DAM como as melhores redes ART com geometria retangular[9]. Assim mesmo, dado que os resultados obtidos por EAM são similares quanto ao erro, aos obtidos por GAM, entretanto com um número muito superior de categorias, usaremos GAM como a melhor rede ART circular. A fig. 3.29 apresenta um exemplo de categorias de politopo (painel (a)) e as bordas entre as predições (painel (b)) criadas por PTAM para

[9]Os resultados obtidos por FAM e DAM (tabela 3.9) são equiparáveis, e muito superiores aos obtidos por FasArt, a outra rede ART com geometria retangular.

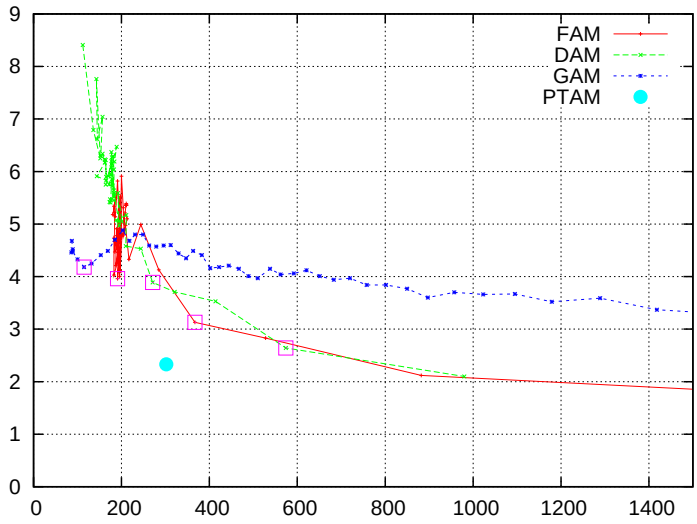


Figura 3.28: O erro médio da validação (em %) e o #C obtidos por FAM, DAM e GAM (variando a vigilância) e por PTAM (os pontos de operação selecionados são os quadrados vazios).

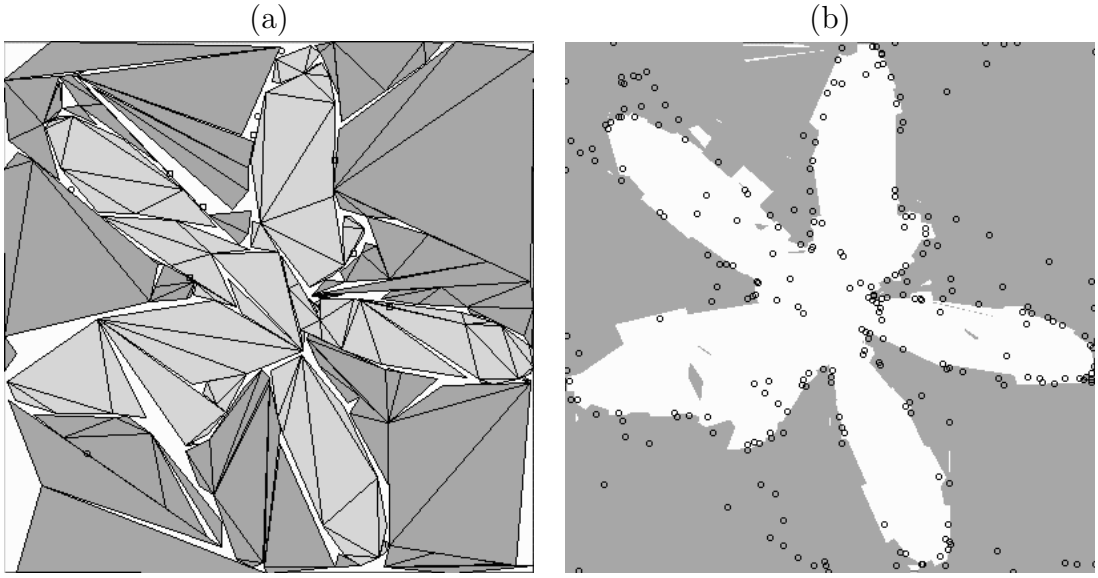


Figura 3.29: (a) Exemplos de categorias criadas por PTAM para o conjunto de dados *Form*. (b) As regiões de classificação e os vetores de pesos criados por PTAM para este conjunto de dados.

Tabela 3.13: O erro de teste (ϵ, em %) e o #C (#W para PTAM e #SV para SVM) para os conjuntos de dados 2-D com sobreposição de predições (4G1, 4G2 e 4G3). O melhor ϵ e o #C obtidos por um classificador ART e não-ART estão em negrito.

	4G1		4G2		4G3	
	ϵ	#C	ϵ	#C	ϵ	#C
Bayes	5.7	–	8.7	–	21.8	–
PTAM	25.6	632	34.0	905	52.5	1869
FAM	15.0	212	19.3	372	39.2	801
GAM	**5.8**	**41**	**8.8**	**51**	22.3	**123**
DAM	10.3	52	14.5	70	32.5	161
EAM	11.9	425	17.2	582	35.2	1937
FasArt	**5.8**	703	**8.8**	1000	**21.9**	2250
SVM	**5.8**	1551	**8.7**	2352	**21.9**	5173

Tabela 3.14: O erro e o #C obtidos na etapa de teste por PTAM e as melhores redes ART, retangulares e circulares (para as quais o valor de $\bar{\rho}$ é registrado) para o conjunto de dados *Form*. O melhor erro e o #C estão em negrito.

	PTAM	GAM	DAM		FAM	
$\bar{\rho}$	–	0.12	0.88	0.94	0.36	0.92
Erro (%)	**2.3**	4.3	3.9	2.7	4.1	3.3
#C	302	**115**	270	572	190	368

este conjunto de dados. As redes FAM, DAM e GAM foram treinadas e testadas com a mesma metodologia de validação-cruzada nos conjuntos de dados bi-dimensionais, ainda que no caso de *Form*, com 10 conjuntos de treinamento / validação / teste, ao invés de 20, como nos demais conjuntos bi-dimensionais.

A fig. 3.28 mostra os resultados da validação obtidos pelas redes FAM, DAM e GAM (variando $\bar{\rho}$ no intervalo de $0.00 : 0.02 : 0.98$), e PTAM, enquanto que na tabela 3.14 registra-se os resultados dos testes. PTAM obtém menos erro de teste (2.3%) que as melhores redes ART retangulares e circulares. GAM obtém mais erro que PTAM, para todos os valores de vigilância, e o erro de teste é duas vezes o erro de PTAM. As redes FAM e DAM necessitam em torno de 1000 categorias para

obter o mesmo erro que PTAM (fig. 3.28). Selecionamos dois pontos de operação para DAM ($\bar{\rho} = 0.88$ e $\bar{\rho} = 0.94$) e outros dois para FAM ($\bar{\rho} = 0.36$ e $\bar{\rho} = 0.92$), ambos marcados com quadrados vazios na fig. 3.28. DAM obtém maior erro (2.7%) que PTAM, usando $\bar{\rho} = 0.94$, e mais categorias (572). Usando $\bar{\rho} = 0.88$, DAM obtém um erro ainda maior (3.9%) que PTAM, com um número ligeiramente inferior de categorias (270). FAM obtém maior erro que PTAM, para os dois valores de vigilância selecionados.

3.5.5 Resultados com irregularidade crescente

A metodologia utilizada nos experimentos com os conjuntos de dados irregulares, variando a sua ordem N, é similar à empregada na seção anterior ($N = 3$). Os resultados obtidos pelas redes FAM, GAM, DAM e EAM e ssEAM [6] sobre os conjuntos de validação se mostram nos gráficos 3.30- 3.32. Cada linha une os pontos, cada um dado pelo erro e pelo número de categorias, obtidos variando a vigilância no intervalo de $\rho = 0.00 : 0.02 : 0.98$, para um valor de N (as distintas linhas correspondem-se com os valores de distintos N, indicados na figura). Pode-se observar que, ainda que existam exceções, a tendência geral, é a de obter os erros e os números de categorias elevados, à medida que cresce N (complexidade da fronteira entre as predições).

É interessante comparar os intervalos de erro e o número de categorias, no que se situam os gráficos de cada rede. Por exemplo, EAM obtém menos erro (2-5%) que GAM e FAM (2-7%), e que DAM (2-10%). Em relação ao número de categorias, os intervalos são variáveis: 300-600 para EAM, 100-600 para DAM, 150-900 para FAM e 50-2200 para GAM. De fato, para as vigilâncias baixas (extremo esquerdo dos gráficos) o erro de FAM e GAM varia entre os 2-7%, e com as vigilâncias altas (extremo direito), variam no intervalo de $1.5 - 3.5\%$. Enquanto que DAM obtém erros de 5-10%, para as vigilâncias baixas, e de 2-5%, para as vigilâncias altas, intervalos superiores aos das redes FAM e GAM. Por sua vez, EAM é a que alcança resultados menos dependentes da vigilância (intervalos mais estreitos), tanto em erro como em número de categorias, situando-se, em níveis de erro, similar a GAM e FAM.

O gráfico 3.32 (painel inferior) mostra os resultados obtidos por PTAM sobre os conjuntos de validação, variando N (neste caso, trata-se de pontos e não de linhas, uma vez que PTAM não usa a vigilância). Estes resultados mostram um intervalo

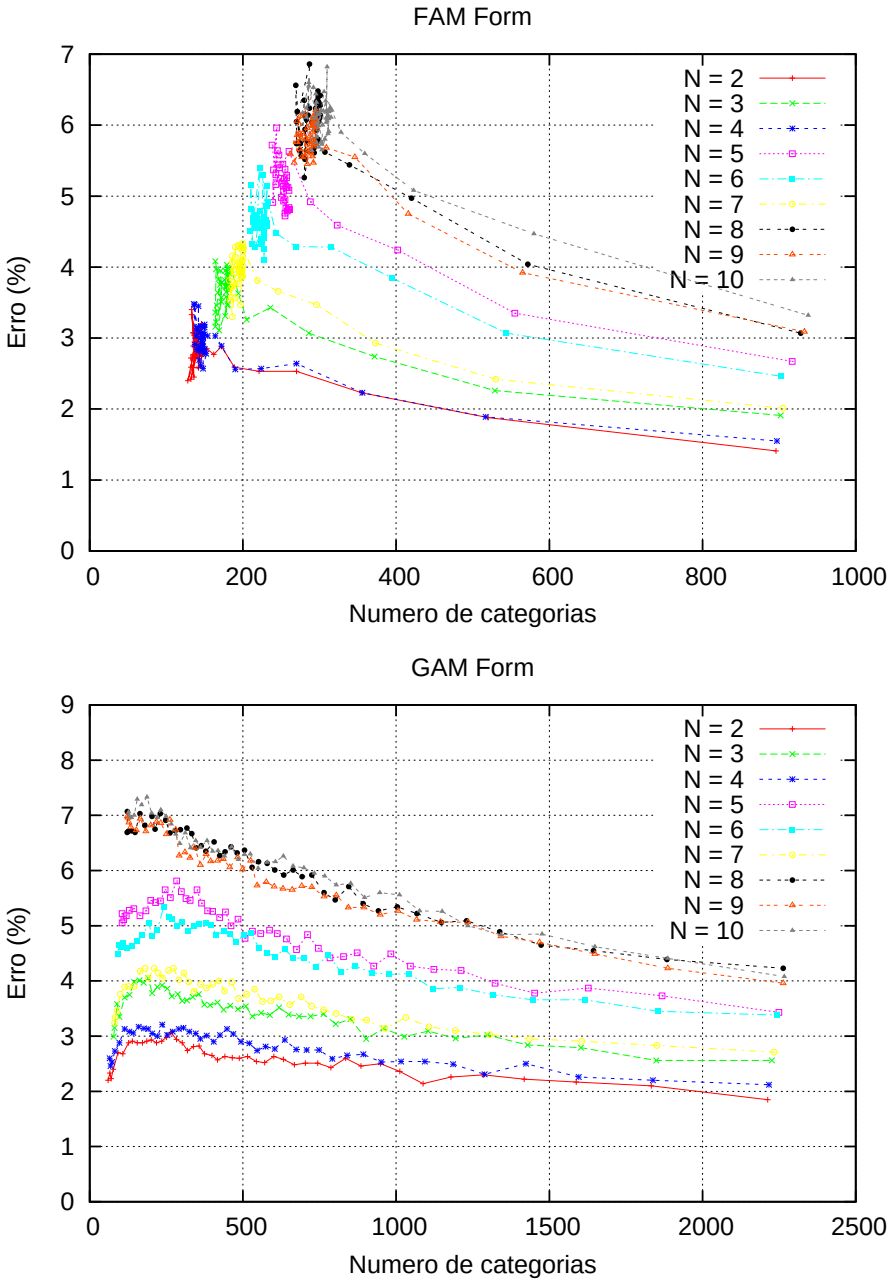


Figura 3.30: O erro (em %) de classificação em função do número de categorias obtido por FAM e GAM sobre os conjuntos de validação no problema *Form* variando N.

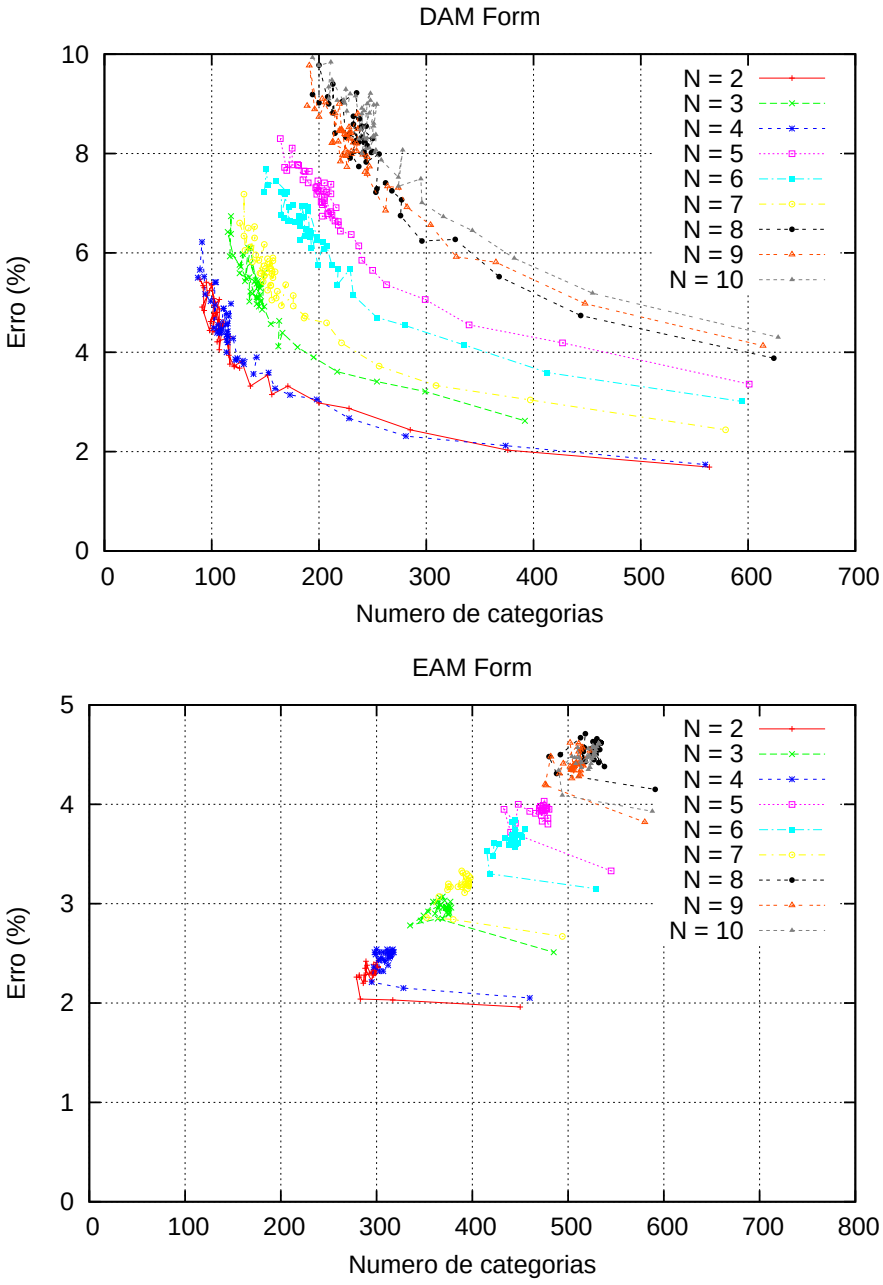


Figura 3.31: O erro (em %) de classificação em função do número de categorias obtido por DAM e EAM sobre os conjuntos de validação no problema *Form*, variando N.

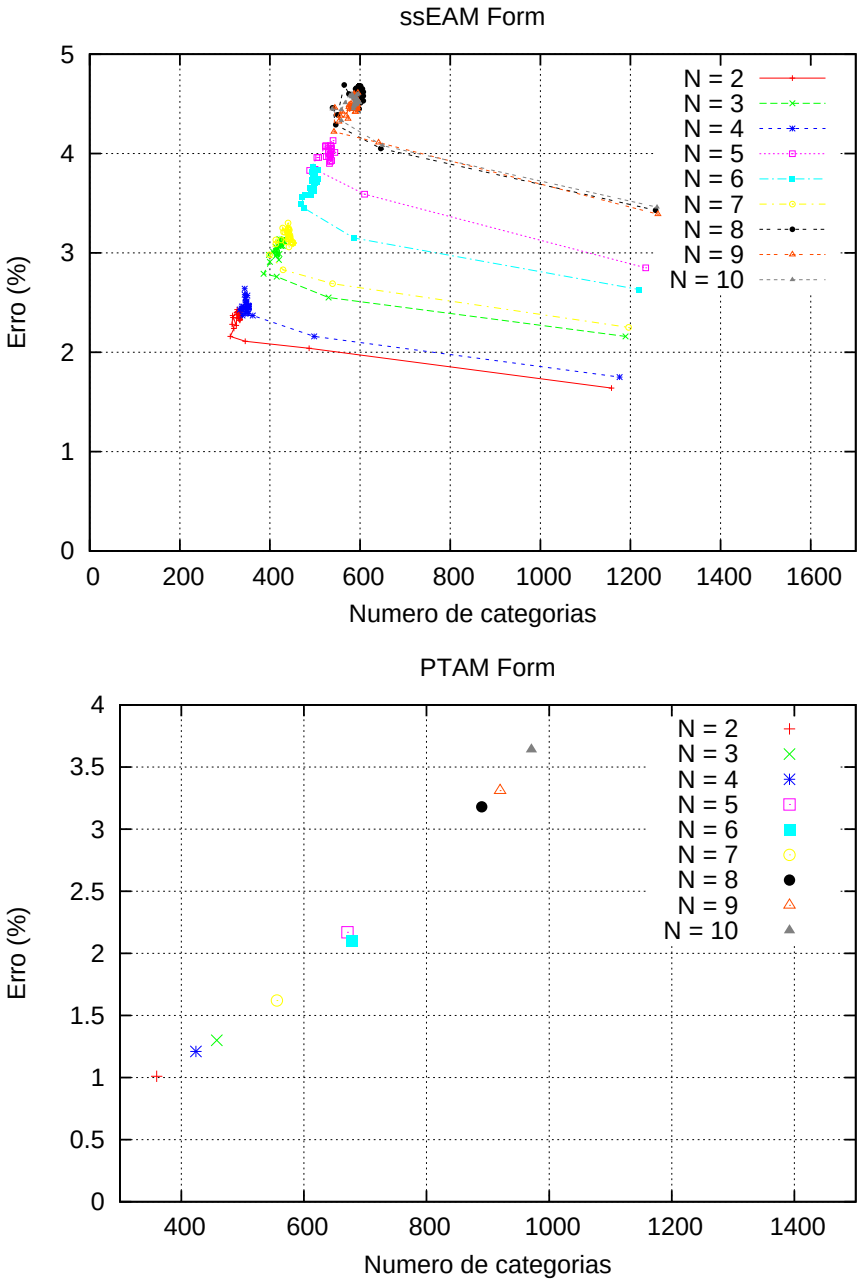


Figura 3.32: O erro (em %) de classificação em função do número de categorias obtido por ssEAM e PTAM sobre os conjuntos de validação no problema *Form*, variando N.

Tabela 3.15: Valor de vigilância base $\bar{\rho}$ que obtém um melhor compromisso entre o erro o número de categorias mediado sobre os conjuntos de validação de *Form*, variando N.

N	FAM	GAM	DAM	EAM	ssEAM
2	0.68	0.00	0.82	0.92	0.92
3	0.52	0.00	0.78	0.00	0.92
4	0.26	0.06	0.80	0.92	0.92
5	0.28	0.00	0.86	0.94	0.92
6	0.38	0.00	0.86	0.94	0.92
7	0.68	0.00	0.86	0.92	0.92
8	0.56	0.06	0.86	0.94	0.94
9	0.74	0.00	0.86	0.94	0.94
10	0.70	0.00	0.84	0.94	0.94

Tabela 3.16: O erro (ϵ, em %) e o #C dos testes obtidos por PTAM e as melhores redes ART retangulares e circulares para o conjunto de dados *Form*, variando N. O melhor erro e o #C estão em negrito.

	PTAM		FAM		GAM		DAM		EAM		ssEAM	
N	ϵ	#C	ϵ	#C	ϵ	#C	ϵ	#C	ϵ	#C	ϵ	#C
2	**1.0**	360	2.4	128	3.5	**97**	3.2	155	2.1	283	2.1	312
3	**1.3**	458	3.1	169	4.3	**93**	4.1	161	3.0	373	2.9	387
4	**1.2**	424	2.6	148	2.4	**68**	3.6	153	2.3	295	2.4	339
5	**2.2**	671	4.7	255	5.6	**112**	5.4	263	3.5	440	3.8	488
6	**2.1**	679	4.1	228	5.1	**104**	4.7	254	3.3	418	3.5	469
7	**1.6**	556	3.4	186	3.8	**88**	4.2	221	2.8	353	2.9	399
8	**3.2**	890	5.1	280	6.6	**122**	6.2	296	4.3	488	4.3	547
9	**3.3**	920	5.7	267	7.2	**125**	6.6	304	4.1	477	4.2	543
10	**3.6**	961	5.8	290	7.4	**133**	7.0	296	4.3	494	4.3	560
Média	**2.2**	659	4.1	217	5.1	**105**	5.0	234	3.3	402	3.4	449
STD	1.0	228	1.3	61	1.7	21	1.4	63	0.8	81	0.8	93

de erro de 1-4%, para todos os valores de N, inferior aos erros obtidos pelas demais redes ART para todos os valores de N. No entanto, o número de categorias se situa no intervalo de 200-1000, em geral, superior ao alcançado pelas outras redes ART.

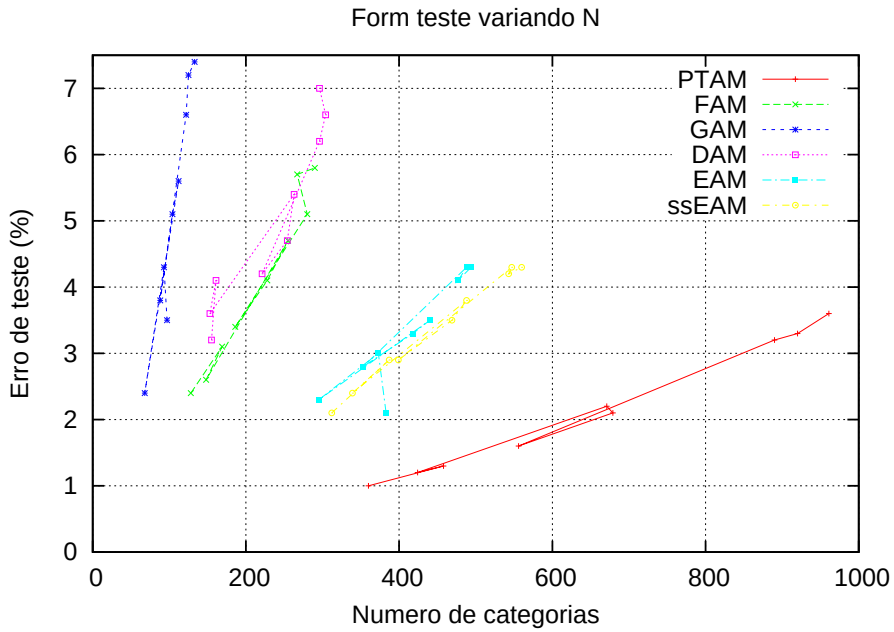


Figura 3.33: Comparação dos resultados dos testes obtidos com as redes PTAM, FAM, GAM, DAM, EAM e ssEAM no problema *Form*, variando N.

Para cada rede ART e cada valor de N, selecionamos o valor de vigilância base $\bar{\rho}$, que obtém um melhor compromisso entre o erro o número de categorias (tabela 3.15), mediado sobre os conjuntos de validação. Este valor de vigilância é o usado para a etapa de teste. Os resultados de teste obtidos pelas distintas redes (FAM, GAM, DAM, EAM, ssEAM) encontram-se na tabela 3.16, e estão representados graficamente na figura 3.33. Nelas pode se observar que a rede PTAM obtém um erro de teste inferior às outras redes ART, para todo os valores de N, com uma baixa desviação típica. Em contrapartida, o número criado de categorias é superior, especialmente, em comparação com FAM, DAM e GAM. Portanto, estes resultados mostram uma elevada capacidade de aprendizagem de fronteiras irregulares entre as predições, ainda que, PTAM não é capaz de alcançar esta melhor aprendizagem de modo eficiente, quer dizer, com um número de categorias equiparável ao obtido pelas demais redes ART.

3.5.6 Conjuntos de dados reais multi-dimensionais

Tabela 3.17: O erro de teste (ϵ, em %) e o #C (#W para PTAM e #SV para SVM) para os conjuntos de dados multi-dimensionais PID e Abalone. O melhor ϵ e o #C obtidos, pelos classificadores ART e não-ART, estão em negrito.

		PTAM	FAM	GAM	DAM	EAM	FasArt	SVM
PID	ϵ	43.6	**31.1**	37.4	**30.9**	34.3	**30.8**	**24.5**
	#C	142	63	**19**	25	22	29	100
Abalone	ϵ	50.3	**42.9**	57.4	**43.3**	47.6	44.1	**34.2**
	#C	**27**	369	60	131	174	341	764

Os conjuntos de alta-dimensão, PID e Abalone, têm um número reduzido de padrões e, portanto, o procedimento experimental usual [65] é o método de validação cruzada Kfold (nos exemplos citados na literatura, normalmente $K = 4$). No entanto, quando existem parâmetros a otimizar, como a vigilância nas redes ART, é recomendável, para não polarizar os resultados, utilizar conjuntos de treinamento, validação (para selecionar os valores ótimos dos parâmetros) e teste, como fizemos nos experimentos anteriores. Portanto, geramos 20 conjuntos de treinamento, 20 de validação e 20 de teste, contendo 25%, 50% e 25% de padrões de entrada, respectivamente; randomicamente selecionados do conjunto original de dados. Especificamente, utilizamos 192 padrões de treinamento, 384 de validação e 192 de teste para o conjunto de dados PID, e 1044 padrões de treinamento, 2088 de validação e 1044 de teste para o conjunto de dados Abalone. Os valores da vigilância base $\bar{\rho}$ no conjunto $\{0.0, 0.2, 0.4, 0.6, 0.8, 0.95\}$ foram testados, e selecionamos o valor com a melhor relação entre a taxa de erro e o #C, sobre os 20 conjuntos de validação, conforme tabela 3.8.

A tabela 3.17 registra os resultados médios de cada classificador com o seu melhor valor da vigilância, sobre os 20 conjuntos de teste. A SVM é visivelmente o melhor classificador para PID e Abalone (24% e 34% de erro, respectivamente), fazendo uso de 52% e de 73% dos padrões de treinamento, como vetores de suporte, respectivamente. Para o conjunto de dados PID, este erro foi também obtido pela rede AFC [128], com vigilância zero, porém usando 576 padrões de treinamento e 192 de teste (portanto, existe a possibilidade de que o treinamento fosse mais fácil, ao usar mais padrões). As redes FasArt, DAM e FAM obtêm o menor erro en-

tre as redes ART (30%), entretanto FAM cria duas vezes o número de categorias (60) que as redes FasArt (29) e DAM (25). As redes EAM e GAM obtêm maior erro (34-37%), com poucas categorias. PTAM obtém um erro (43%) superior ao alcançado pelas outras redes ART, e o maior número de categorias (74% dos padrões de treinamento).

No conjunto de dados Abalone, FAM e DAM obtêm o menor erro (43%), mas FAM cria o maior #C (370), e DAM cria somente 131 categorias. As redes FasArt e EAM obtêm maior erro que FAM e DAM, com menos categorias. PTAM obtém um erro de 50%, somente abaixo de GAM (57%), mas é igual ao erro registrado por FAM (59%) em [5]. Além disso, PTAM cria o menor número de categorias (27) entre as redes ART.

3.6 Discussão

A média e o desvio padrão dos resultados dos testes médios (ϵ e #C), alcançados pelos diferentes classificadores nos conjuntos de dados bi-dimensionais 1-7, são registrados nas últimas linhas da tabela 3.9 e, as médias estão representadas na fig. 3.26. A SVM alcança o menor erro médio (0.4%), com um elevado número de vetores de suporte (298). PTAM alcança o menor erro médio entre as redes ART (1.6%), e o menor erro absoluto para os cinco, dos sete, conjuntos de dados (CIS, T3, T5, T6 e T7). Nos outros dois conjuntos (*Chess* e T4, ambos com geometria retangular), PTAM não obtém muito mais erro que a melhor rede ART (DAM). Somente as redes GAM e FAM criam menos categorias (80 e 146, respectivamente) que PTAM (188). DAM obtém o segundo menor erro médio (2%) entre as redes ART, e similar em #C (193) ao PTAM. A rede FAM obtém mais erro médio que DAM, e menos categorias, o qual não foi esperado a partir dos trabalhos anteriores na literatura [120] (este fato pode ter ocorrido, devido o valor de vigilância ter sido selecionado para ambas as redes). A rede EAM obtém um erro similar ao GAM, no entanto, EAM cria o maior #C, e GAM o mais baixo. O erro (3.8%) e o #C (334) de FasArt são os maiores de entre todas as redes ART examinadas. PTAM também obtém menos erro que as melhores redes retângular (DAM) e circular (GAM), com um número de categorias baixo, nos conjuntos de dados irregulares *Form* (tabela 3.14), e variando a ordem N da forma irregular (tabela 3.16), ainda que a sua complexidade (número de vetores) mais elevada que das outras redes ART.

PTAM cria um simplex novo durante a ressonância, somente se a categoria que alcançou a ressonância não é a mais ativa delas (a ganhadora) (subseção 3.1.5). Também desenvolvemos experimentos, nos quais um simplex novo é criado toda vez que um padrão de entrada cai fora da categoria que atingiu a ressonância. Nestes experimentos, a média do erro de testes e o #W obtidos pela rede PTAM, para os conjuntos de dados 1-7, são 1.4% e 262 vetores de pesos, respectivamente. O erro reduz 0.2, porém o número de vetores de pesos #W aumenta 39%. Então, a criação de simplexes novos, em todos os casos, não reduz significativamente a taxa de erro, mas contribui à proliferação de categorias. Isto justifica nossa hipótese de criar um simplex novo, somente se a categoria que alcançou a ressonância não é a mais ativa delas.

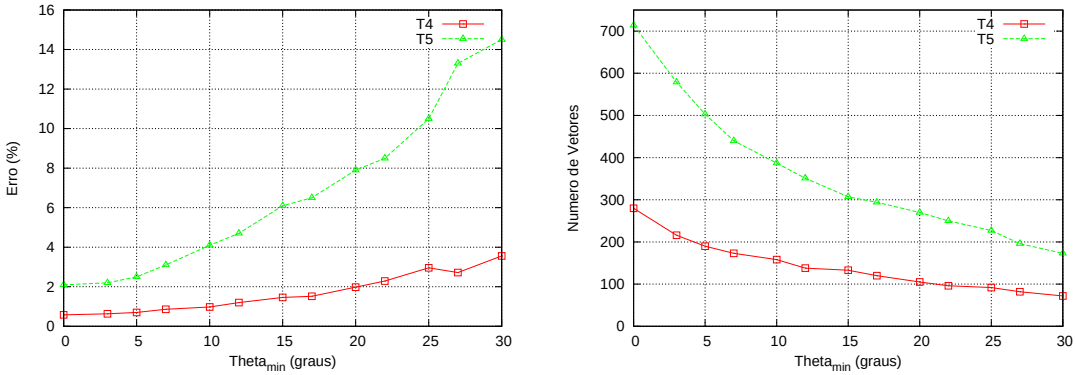


Figura 3.34: Erro (painel esquerdo) e número de vetores (painel direito) obtidos por PTAM variando θ_{min} nos conjuntos de dados T4 e T5.

Também fizemos experimentos relacionados com o parâmetro θ_{min}, utilizado para descartar os simplexes agudos (subseção 3.1.4). A fig. 3.34 mostra o comportamento típico do erro e do número dos vetores obtidos por PTAM, variando θ_{min}, nos conjuntos de dados T4 e T5 (eles foram selecionados como representantes das geometrias retangulares e circulares, mas encontramos o mesmo comportamento nos outros conjuntos de dados). A fig. 3.34 registra que altos valores de θ_{min} incrementa o erro e reduz o número de vetores. Neste caso, PTAM é incapaz de ajustar as fronteiras entre as predições de saída, porque descartam-se muitos simplexes, assim o número de simplexes e de vetores é baixo, mas o erro é alto. Por outro lado, quando θ_{min} se reduz, PTAM rejeita um número baixo de simplexes, assim sua habilidade de aprendizagem é mais elevada, o que leva a baixar o erro e gerar mais simplexes e vetores. No limite $\theta_{min} \to 0$, nenhum simplex é descartado, assim o número de sim-

plexes e de vetores são elevados (final à esquerda do painel da direta na fig. 3.34). Portanto, um valor baixo de θ_{min} deve ser utilizado para descartar somente os simplexes agudos. Baseado nos experimentos desenvolvidos com os vários conjuntos de dados, selecionamos como o valor aceitável de $\theta_{min} = 10$ graus. Dado que o ângulo mímimo para considerar um simplex como agudo não parece depender do conjunto de dados, este valor deve ser válido para qualquer conjunto de dados, assim não necessita ser otimizado através da validação-cruzada.

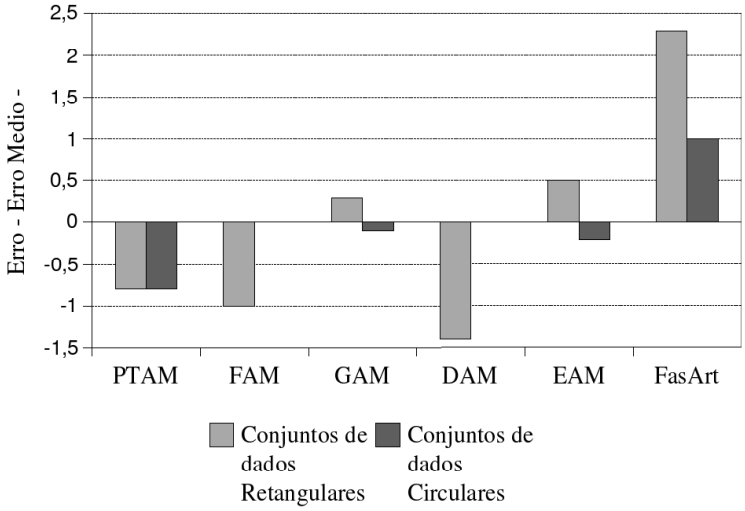


Figura 3.35: Diferença entre o erro obtido por cada classificador ART e o erro médio para cada conjunto de dados. Esta diferença é a média sobre os conjuntos de dados retangulares (barra esquerda) e circulares (barra direita). A diferença é negativa, quando o classificador trabalha melhor que a média dos conjuntos de dados com uma dada geometria e, senão, é positiva.

É interessante avaliar o comportamento de cada classificador dependendo da geometria do conjunto de dados. Em primeiro lugar, PTAM é a rede ART com o menor desvio de erro (*standard deviation*) (1.1) entre os conjunto de dados 1-7 (FAM, GAM e DAM têm o desvio de 1.7, EAM de 1.5 e FasArt de 2.0), como se observa na tabela 3.9. Portanto, os resultados obtidos por PTAM são menos dependentes do conjunto de dados que as outras redes ART. Com o propósito ilustrativo, agrupamos os resultados de cada classificador para os conjuntos de dados, retangulares e circulares. Para cada conjunto de dados, calculamos o erro médio sobre todas as redes ART. Para cada rede, foi calculada a diferença entre o seu erro e este erro médio. Se um classificador trabalha melhor que a média, esta diferença é negativa e, senão, é positiva. Finalmente, para cada classificador, calculamos suas diferenças

médias sobre os conjuntos de dados circulares (CIS, T3 e T5-T7), e sobre os conjuntos de dados retangulares (*Chess* e T4). A fig. 3.35 mostra estas duas médias, para cada classificador. As redes FAM e DAM têm uma diferença negativa alta para os conjuntos de dados retangulares (indicando um erro inferior à média) e, quase com diferença zero, para os circulares (indicando um erro similar à média). Os classificadores circulares (GAM e EAM) exibem um comportamento oposto: diferenças negativas, para conjunto de dados circulares (menor erro que a média), e positiva, para os retangulares (com geometria distinta à das redes EAM e GAM). A FasArt tem uma diferença positiva alta para os conjunto de dados retangulares e circulares (seu erro é pior que a média, em ambas geometrias). Então, o comportamento dos classificadores, ambos, circulares e retangulares, dependem da geometria do conjunto de dados, e seus resultados médios são os melhores (diferenças negativas) para os conjuntos de dados com a sua mesma geometria. Entretanto, PTAM é a única rede com diferença negativa alta (erro inferior à média) para ambos os conjuntos de dados, retangulares e circulares, logo ela trabalha igualmente bem para ambas as geometrias. Portanto, os resultados de PTAM não dependem da geometria do conjunto de dados, devido às suas categorias politopo irregulares, as quais não são especialmente adequadas para nenhuma geometria particular.

Os bons resultados da rede PTAM são possíveis sem vigilância e sem nenhum parâmetro ajustável, o qual é uma característica muito pouco freqüente no âmbito das redes neurais nem dos algoritmos de classificação. Esta característica torna PTAM mais fácil para ser usada que as outras redes ART, uma vez que o usuário não tem que conhecer a operaąão interna da rede nem o significado dos seus parâmetros internos para selecionar um valor ótimo dos mesmos. Por outro lado, esta característica reduz o tempo de computação necessário para aplicar a rede a um problema dado, já que os ajustes de parâmetros, quando existem, geralmente requerem testes de validação cruzada, os quais podem gerar um custo significativo no tempo de processamento. De outra maneira, a validação cruzada requer incluir padrões de entrada nos conjuntos de validação cruzada, o que pode reduzir o número de padrões de treinamento disponível. Por exemplo, nos conjuntos PID e Abalone, a necessidade de dedicar padrões aos conjuntos de validação, diminui o número de padrões disponíveis para o treinamento. Este fato pode ser importante, quando um baixo número de padrões de entrada estão disponíveis, que é normal nos conjuntos de dados reais.

Os resultados com os conjuntos de dados CIS-ruído e T5-ruído (tabela 3.12),

e 4G1, 4G2 e 4G3 (tabela 3.13) mostram que a PTAM obtém maior erro que as outras redes ART, apesar dos seus bons resultados com os conjuntos de dados livres de ruídos. PTAM também cria um elevado número de categorias, ainda que é menor que a FasArt, e inferior em alguns casos ao número de vetores de suporte criados pela SVM. Isto se deve a que PTAM aproxima as fronteiras entre as predições de saída, seguindo estritamente a informação contida no conjunto de treinamento, tanto no passo de expansão como no de ajuste de categorias. Na presença de ruído ou sobreposição, os padrões de entrada pertencentes às categorias diferentes são misturados e, eles quebram os simplexes no passo de ajuste de categorias, que levam a criação de categorias mono-vetores ruidosas. Por esta razão, a investigação futura contempla modificar as etapas de expansão e ajuste de categoria, possivelmente incluindo a informação estatística sobre os padrões de entrada, para aumentar a robustez de PTAM, em relação ao ruído e a sobreposição de categorias. Nas situações onde as predições se sobrepõem, um ajuste de categorias baseado na contração das suas fronteiras, sem quebra de simplexes, possivelmente, seria o mais recomendável. O comportamento de PTAM nos conjuntos de dados reais de alta-dimensão, PID e Abalone (tabela 3.17), é pior que as outras redes ART, com o maior erro no conjunto de dados PID, ainda que neste caso um número de categorias é variável, o maior para PID e o menor para Abalone.

Capítulo 4

Overlapping Polytope ARTMAP

No capítulo anterior propomos e avaliamos a rede *PolyTope* ARTMAP (PTAM), baseada em categorias com forma geométrica de politopo irregular. Esta rede possui como característica, também relevante, o fato de não permitir a sobreposição entre as categorias. Este aspecto é de grande importância no seu funcionamento porque, suprime a necessidade de usar o tamanho da categoria como critério no processo de competição entre as categorias. Como conseqüência imediata, desaparece também a necessidade de usar um tamanho máximo de categoria e se suprime o parâmetro de vigilância, de modo que PTAM não possui nenhum parâmetro de ajuste.

Como indicamos na seção 1.5, o uso de categorias irregulares sugere, de certo modo, que estas categorias não deveriam se sobreporem. De fato, se as categorias pudessem se sobreporem entre elas, seria necessário, como nas redes ART, levar em consideração o tamanho da categoria para decidir a categoria ganhadora, quando o padrão de entrada cai em uma região de sobreposição entre as categorias. Portanto, os benefícios obtidos por uma geometria de categoria irregular (maior capacidade para aproximar as fronteiras entre as predições) se perdem, em certa medida, se as categorias se sobrepõem, uma vez que então será o seu tamanho, e não a sua forma geométrica (disposição das suas fronteiras), o que determina, em muitos casos, a categoria candidata a codificar o padrão de entrada.

Por outro lado, o uso de geometrias irregulares tem como principal objetivo que a categoria ocupe apenas as regiões do espaço de entrada populadas por padrões com a sua mesma predição de saída, seja qual for a sua forma geométrica[1]. Entretanto,

[1]Isto não é possível nas redes ART, porque as suas categorias, quando cobrem as regiões popu-

ocupar a região populada com padrões com a mesma predição que a categoria, não tem tanta utilidade se também outras categorias, com outra predição associada, ocupam esta região.

Apesar desta contradição entre geometria irregular das categorias e sobreposição entre elas, seria interessante avaliar a contribução realizada, exclusivamente, pelo uso de categorias irregulares, sem levar em consideração a ausência de sobreposição. O objetivo seria avaliar em que medida as categorias irregulares contribuem a melhorar, ou a piorar, a operação das redes ART, sem levar em consideração a não sobreposição. Para isto, e como complemento do trabalho anterior, neste capítulo propomos um modelo alternativo ao PTAM, que denominaremos *Overlapping Poly-Tope* ARTMAP (OPTAM) [63], no qual as categorias têm forma irregular, como em PTAM, mas se podem sobreporem entre si.

A possibilidade de sobreposição entre as categorias permite OPTAM simplificar algumas fases da etapa de treinamento em relação ao PTAM. Dado que as categorias são politopos irregulares, como em PTAM, a mesma função de escolha de categoria $T_i^t(\mathbf{I})$ (eq. 3.1) é válida também para OPTAM. Por outro lado, a etapa de Teste de Sobreposição já não tem sentido, ao se permitir a sobreposição entre as categorias. É necessário também introduzir um Teste de Vigilância, similar ao usado pelas redes ART que impõe uma limitação na expansão das categorias, que em OPTAM podem se sobrepor. Neste teste, é necessário definir o tamanho da categoria, que agora tem uma forma de politopo irregular. Finalmente, a função de escolha deve levar em consideração este tamanho, quando o padrão cai em regiões de sobreposição entre as categorias. Poderia-se pensar em distintas alternativas para avaliar o tamanho de uma categoria politopo. No entanto, o cálculo direto ou aproximado do seu volume em $\mathbb{R}^n$ levaria a um alto custo computacional, e fortemente crescente com a dimensão n do espaço de entrada. Por esta razão, no caso de OPTAM, optamos por definir o tamanho da categoria de modo similar a *Fuzzy* ART/ARTMAP, de modo que a complexidade computacional do seu cálculo é inferior e aumenta mais lentamente com a dita dimensionalidade. As seguintes seções descrevem as etapas de treinamento e processamento em OPTAM.

ladas por padrões com a sua mesma predição, têm que ocupar também outras regiões (populadas por padrões com predições distintas) para manter a sua forma geométrica predefinida (seção 1.5).

4.1 Etapa de Treinamento

Dado que OPTAM está estreitamente relacionado com PTAM, devido a ambos usarem categorias com forma de politopo irregular, usaremos aqui a nomenclatura empregada no capítulo anterior, que se encontra na tabela 3.1. A fig. 4.1 resume os passos principais da etapa de treinamento em OPTAM, descreveremos a seguir.

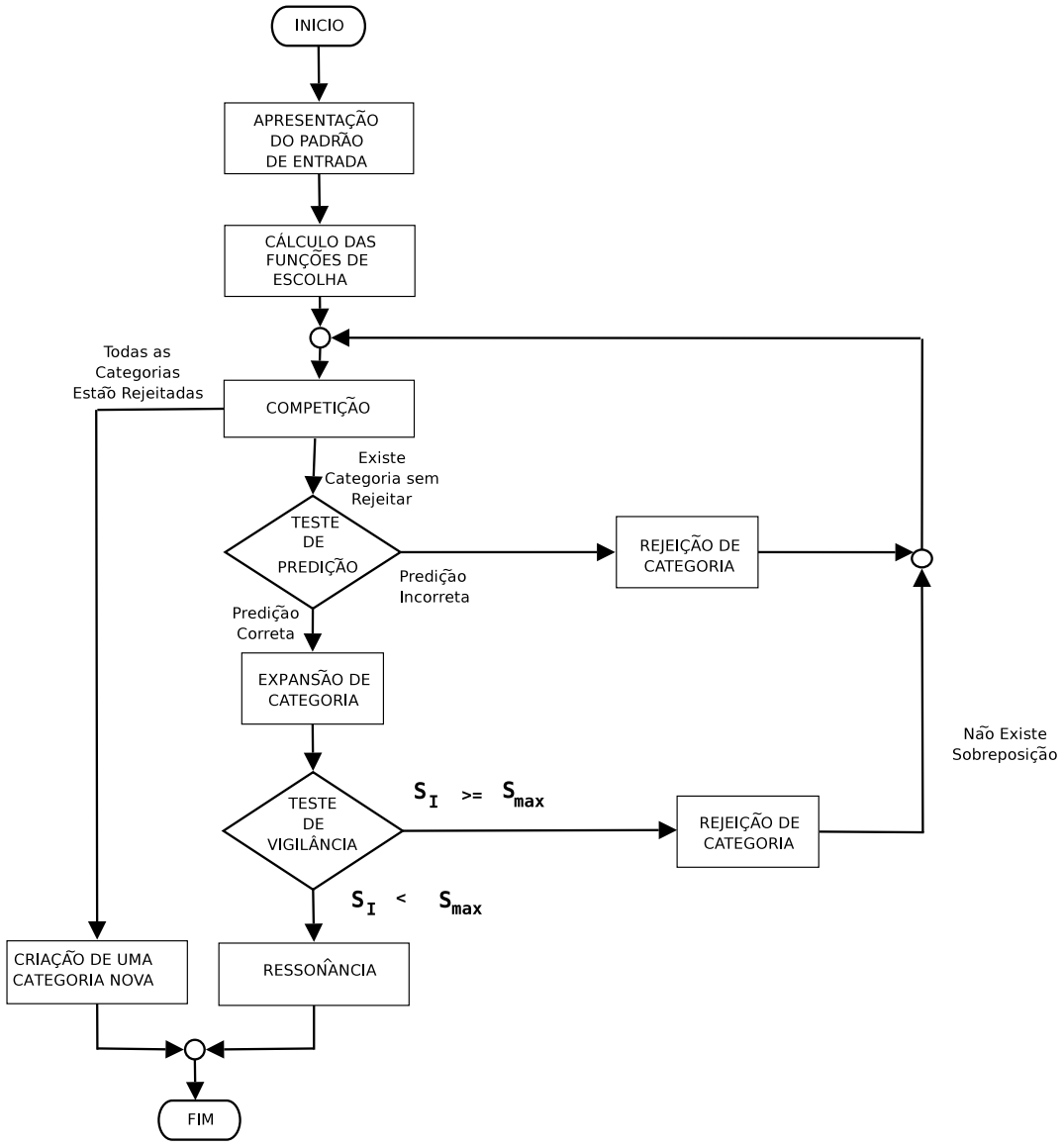


Figura 4.1: Diagrama de fluxo da etapa de treinamento em OPTAM.

4.1.1 Função de escolha de categoria

Uma vez que OPTAM permite a sobreposição entre as categorias, a função de escolha da categoria deve levar em consideração a possibilidade de que o padrão de entrada $\mathbf{I}$ caia dentro de várias categorias sobrepostas. Como se comentou na introdução deste capítulo, em tal caso o padrão é codificado, como nas redes ART tradicionais, pela categoria mais específica (a de menor tamanho). Esta questão, entretanto, não se considera durante a etapa de treinamento, uma vez que quando um padrão cai dentro de uma ou várias categorias, nenhuma delas aprende (se expande até o padrão de entrada), porque já o contém. Ao contrário, o tamanho da categoria somente se levará em consideração durante a etapa de processamento. Por esta razão, e similarmente a PTAM, a função de escolha de categorias, na rede OPTAM, é distinta no treinamento e no processamento. A função de escolha $T_i^t(\mathbf{I})$ de uma categoria politopo C_i, durante o treinamento é definida como:

$$T_i^t(\mathbf{I}) = \max_{j=1,\ldots,N_i^s} \{T_{ij}(\mathbf{I})\} \qquad i = 1, \ldots, N_c \tag{4.1}$$

A função $T_i^t(\mathbf{I})$ é o máximo das funções $T_{ij}(\mathbf{I})$ dos simplexes $S_{i1}, \ldots, S_{iN_i^s}$ da categoria C_i. Por sua vez, a função $T_{ij}(\mathbf{I})$ associada ao simplex S_{ij} se define da mesma forma que a função de escolha do simplex em PTAM (eq. 3.13):

$$T_{ij}(\mathbf{I}) = \begin{cases} 1 & g_{ijk}(\mathbf{I}) > 0, \quad k = 1, \ldots, n+1 \\ 1 - \dfrac{d(\mathbf{I}, S_{ij})}{\sqrt{n}} & \text{senão} \end{cases} \tag{4.2}$$

Esta definição estabelece que $T_{ij}(\mathbf{I}) = 1$, dentro do simplex S_{ij}, e $T_{ij}(\mathbf{I}) < 1$, fora, sendo neste caso, descendente com a distância $d(\mathbf{I}, S_{ij})$ entre o padrão de entrada $\mathbf{I}$ e o simplex S_{ij}. A função $g_{ijk}(\mathbf{I})$, já usada pela rede PTAM (eq. 3.5), verifica que $g_{ijk}(\mathbf{I}) = 0$, se o $\mathbf{I} \in h_{ijk}$, a $g_{ijk}(\mathbf{I}) < 0$, se o $\mathbf{I}$ se encontra no lado do hiperplano h_{ijk} exterior ao simplex S_{ij} e a função $g_{ijk}(\mathbf{I}) > 0$ no lado interior ao simplex. Deste modo, a eq. 4.2 estabelece que $T_{ij}(\mathbf{I}) = 1$, se o $\mathbf{I}$ cai no lado interior de todos os hiperplanos de S_{ij}, ou seja, dentro do simplex S_{ij}, e $T_{ij}(\mathbf{I}) < 1$ e decresce linearmente com a $d(\mathbf{I}, S_{ij})$, se o $\mathbf{I}$ cai fora do simplex. Finalmente, esta distância $d(\mathbf{I}, S_{ij})$ é definida em OPTAM da mesma forma que em PTAM (equação 3.12).

4.1.2 Competição

Igual que nas redes ART clássicas, em OPTAM o processo competitivo entre as categorias internas seleciona como categoria ganhadora C_I aquela não rejeitada ($r_i = 0$) com a função de escolha máxima:

$$T_i^t(\mathbf{I}) = \max_{i=1,\ldots,N_c} \{T_i^t(\mathbf{I}) : r_i = 0\} \tag{4.3}$$

A categoria ganhadora C_I é a única que permanece ativa, e se converte na candidata a codificar o padrão de entrada. Se todas as categorias se encontram rejeitadas ($r_i = 0, i = 1, \ldots, N_c$), então OPTAM cria uma categoria nova (seção 4.1.6).

4.1.3 Teste de predição

A predição $P(C_I)$ associada à categoria ganhadora C_I se compara com a predição desejada P_d para o padrão de entrada (teste de predição). Se $P_d = P(C_I)$, a categoria ganhadora C_I supera o teste de predição e tenta a expansão até o padrão de entrada. Caso contrário, a C_I é rejeitada ($r_I = 1$), e retorna-se à competição entre as categorias para selecionar uma nova categoria ganhadora.

OPTAM não utiliza *Match Tracking* [28], que é comum nas redes ART, porque o *Match Tracking* eleva a vigilância até que a categoria ganhadora seja rejeitada, possibilitando a escolha de uma categoria alternativa. No entanto, o aumento da vigilância, na rede OPTAM, dificulta a escolha das outras categorias, uma vez que reduz o tamanho máximo que estas podem alcançar, o qual diminui a sua capacidade de expansão. Finalmente, dado que OPTAM permite a sobreposição entre as categorias, não tem sentido a etapa de ajuste de categorias que é realizada em PTAM, quando um padrão cai dentro de uma categoria com uma predição distinta.

4.1.4 Expansão da categoria ganhadora

A aprendizagem consiste em expandir a C_I até o $\mathbf{I}$. Como no caso de PTAM, existem várias situações possíveis, descritas na fig. 4.2:

1. O $\mathbf{I}$ cai no interior da categoria C_I ($T_I^t(\mathbf{I}) = 1$). Neste caso, a C_I não necessita ser expandida para cobrir o $\mathbf{I}$ (fig. 4.2, painel 1).

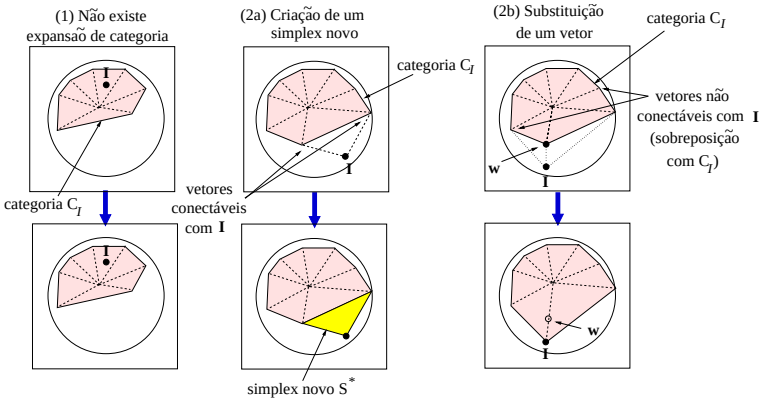


Figura 4.2: Exemplos da aprendizagem de categorias politopo com OPTAM para o problema "*Circle-In-the-Square*" (CIS), em $\mathbb{R}^2$.

2. O **I** cai fora da categoria ganhadora C_I ($T_I^t(\mathbf{I}) < 1$), e portanto a C_I tenta ser expandida até o **I**. Em PTAM, para determinar se a expansão se efetua mediante a criação de um simplex novo, ou mediante a substituição de um vértice da C_I, avalia-se o número de vetores na C_I conectáveis com o **I**, sem sobrepor-se com a C_I, nem com as outras categorias (conjunto $\mathcal{A}_I(\mathbf{I})$ definido na eq. 3.15). Entretanto, dado que a sobreposição entre as categorias está permitida, OPTAM avalia o número de vértices na C_I conectáveis com o **I**, sem sobrepor-se com a C_I, ainda que se possam se sobreporem com as outras categorias. A razão de que estes vetores devam ser conectáveis com o **I**, sem sobreporem-se com a própria C_I, é que se devem encontrar no lado da C_I, em frente ao **I**, e não no lado oposto, para serem substituídos pelo **I**, ou para formar parte de um simplex novo, junto com o **I**, como se indica nos painéis (2a) e (2b) da fig. 4.2. Portanto, o conjunto de vetores conectáveis no $\mathcal{A}_I(\mathbf{I})$ passa a estar definido em OPTAM da seguinte forma:

$$\mathcal{A}_I(\mathbf{I}) = \{\mathbf{w}_{Ijl} : O_{ls}(\mathbf{w}_{Ijl}, \mathbf{I}, S_{Ij}) = 0, \quad \forall j = 1, \ldots, N_I^s\} \tag{4.4}$$

onde a função $O_{ls}(\mathbf{w}_{Ijl}, \mathbf{I}, S_{Ij})$, que determina se o segmento $\overline{\mathbf{w}_{Ijl}, \mathbf{I}}$ e o simplex S_{Ij} se sobrepõem, é definida na eq. F.5 do apêndice F.1. Dependendo do número de vetores do $\mathcal{A}_I(\mathbf{I})$, denotado pelo $|\mathcal{A}_I(\mathbf{I})|$, podem se produzir duas situações ilustradas na fig. 4.2 (painéis 2a e 2b).

2.1. Criação de um novo simplex entre o **I** e a C_I, que se adiciona à categoria C_I. Se existem n vértices na C_I conectáveis desde o **I**, sem sobreposição

com a C_I ($|\mathcal{A}_I(\mathbf{I})| = n$), a expansão se realiza criando um novo simplex (fig. 4.2, painel 2a) com os n vértices conectáveis e o $\mathbf{I}$. Neste caso, e dado que as categorias podem se sobreporem livremente, não é necessária a supressão de simplexes agudos, como ocorre em PTAM.

2.2. Substituição de um vértice da C_I pelo $\mathbf{I}$. Se existe mais de n vértices da C_I conectáveis com o $\mathbf{I}$ ($|\mathcal{A}_I(\mathbf{I})| > n$), então a C_I pode ser expandida substituindo algum vértice da C_I pelo $\mathbf{I}$ (fig. 4.2, painel 2b). A seleção do vértice $\mathbf{w} \in \mathcal{A}_I(\mathbf{I})$ que deve ser substituído pelo $\mathbf{I}$ realiza-se da mesma forma que em PTAM. Concretamente, para evitar que esta substituição reduza o volume da C_I, substitui-se o vetor $\mathbf{w} \in \mathcal{A}_I(\mathbf{I})$ mais próximo a $\mathbf{I}$ que satisfaça a condição 3.17 (fig. 3.12). No caso de OPTAM, dado que se permite a sobreposição entre as categorias, não tem que testar se a categoria expandida, depois da substituição do vetor, se sobrepõe com as outras categorias.

Observamos que o caso $|\mathcal{A}_I(\mathbf{I})| < n$, não se pode produzir em OPTAM, uma vez que, se C_I não é uma categoria mono-vetor, sempre vai ter, ao menos, n vetores conectáveis com o $\mathbf{I}$, sem se sobreporem com a C_I. O caso $|\mathcal{A}_I(\mathbf{I})| < n$ produz-se em PTAM, somente quando existe vetores $\mathbf{w}$ na C_I que poderiam conectar-se com o $\mathbf{I}$, sem se sobreporem com a C_I, porém o segmento $\overline{\mathbf{wI}}$ sobrepõe-se com as outras categorias. Dado que OPTAM permite a dita sobreposição, este caso não se pode produzir.

4.1.5 Teste de vigilância

Nas redes ART, o teste de vigilância determina se a categoria ganhadora C_I pode ser expandida até o $\mathbf{I}$. O parâmetro de vigilância ρ determina o tamanho máximo permitido para a categoria. Na rede OPTAM, o tamanho S_i (*size*) de uma categoria politopo C_i se define de um modo similar ao de *Fuzzy* ARTMAP (seção 1.3.2), tentando representar o intervalo, em cada dimensão, sobre o qual se estendem os padrões que alcançaram a ressonância com a categoria:

$$S_i = \sum_{l=1}^{n} M_{il} - m_{il} \tag{4.5}$$

Nesta expressão, M_{il} e m_{il} designam os valores máximo e mínimo dos vértices da categoria C_i na dimensão l:

$$M_{il} = \max_{j=1,\ldots,N_i^s} \left\{ \max_{k=1,\ldots,n+1} \{w_{ijk,l}\} \right\}$$
$$m_{il} = \min_{j=1,\ldots,N_i^s} \left\{ \min_{k=1,\ldots,n+1} \{w_{ijk,l}\} \right\}$$

Por outro lado, o $w_{ijk,l}$ denota a l-ésima componente do vetor de pesos $\mathbf{w}_{ijk}$. Deste modo, o tamanho S_i de uma categoria C_i é a soma dos intervalos dos componentes de seus vértices $\mathbf{w}_{ijk}, j = 1, \ldots, N_i^s; k = 1, \ldots, n+1$. Pode se observar que $0 \leq S_i \leq n$, dado que $m_{il} \geq 0$,$M_{il} \leq 1$ e $m_{il} \leq M_{il}$. O tamanho máximo S_{max} permitido para uma categoria em OPTAM, determinado pela vigilância ρ, se define como:

$$S_{max} \equiv (1 - \rho)n \tag{4.6}$$

O valor $\rho = 0$ implica no $S_{max} = n$, de modo que qualquer categoria pode cobrir completamente o espaço de entrada $[0, 1]^n$. Ao contrário, se $\rho = 1$, então o $S_{max} = 0$, de modo que cada categoria pode ser um vetor único.

O teste de vigilância comprova se o tamanho S_I da categoria ganhadora C_I, depois de ser expandida até o $\mathbf{I}$, é superior a S_{max}. Se o $S_I \geq S_{max}$, a C_I é rejeitada ($r_I = 1$) e se seleciona outra categoria ganhadora (seção 4.1.2), com a qual se repete o mesmo processo. Se, ao contrário, o $S_I < S_{max}$, depois da expansão, então a categoria ganhadora C_I codifica o padrão de entrada (ressonância).

No caso de PTAM, o padrão de entrada somente se aprende, se a categoria que alcançou a ressonância não foi a primeira ganhadora. No entanto, em OPTAM o padrão de entrada se aprende sempre, tanto se a categoria que atingiu a ressonância foi a primeira ganhadora, como se não[2]. A razão é que, em OPTAM, as categorias podem se sobreporem. Se a categoria que atingiu a ressonância C_I, que tem a predição correta $P(C_I) = P_d$, não se expande até o padrão de entrada $\mathbf{I}$, outras categorias C_i ($i \neq I$) com a predição associada $P(C_i) \neq P_d$, podem ocupar essa região (ainda que não existam nelas padrões de treinamento com predição desejada $P(C_i)$). Conseqüentemente, o padrão de entrada $\mathbf{I}$ ficaria coberto por categorias

[2]Se o padrão de entrada cai dentro da categoria, então não é aprendido, como se indicou no caso 1 da seção 4.1.4.

C_i, com predição incorreta, e não pela categoria C_I, com a predição correta, o qual provocaria erros de classificação. Pelo contrário, expandindo sempre a categoria que atingiu a ressonância C_I (tanto se for a primeira ganhadora como senão), se garante que esta cubra o padrão de entrada, ainda que as outras categorias C_i ($i \neq I$), com predições distintas, também possam cobri-lo, uma vez que se permite a sobreposição de categorias (neste caso, o tamanho da categoria determina a categoria que codificou o padrão de entrada no processamento, como se indica na seção 4.2).

4.1.6 Criação de categorias novas

Nas redes ART clássicas, quando nenhuma categoria supera os testes de predição e vigilância, se cria uma categoria nova, definida por um único vetor de pesos igual ao padrão de entrada. Em OPTAM, igual que em PTAM, a criação de um simplex novo S^* requer ao menos n vetores de pesos (vértices) pertencentes às categorias com a mesma predição que o padrão de entrada. OPTAM requer que estes n vetores pertençam às categorias mono-vetores, a fim de que, em caso de criação de uma categoria nova, nenhum vetor pertença a mais de uma categoria. Se existem estes n vetores, OPTAM cria uma categoria nova C_{N_c+1} que somente contém o simplex S^* (neste caso, $N^s_{N_c+1} = 1$ e $S_{(N_c+1)1} = S^*$). Diferente da rede PTAM, esta nova categoria C_{N_c+1} pode sobrepor-se com outras categorias. Por outro lado, a categoria nova C_{N_c+1} deve passar no teste de vigilância e, portanto, deve satisfazer $S_{N_c+1} < S_{max}$. Se não existem n vetores deste tipo, ou bem se $S_{N_c+1} \geq S_{max}$ (a categoria nova não passa no teste de vigilância), OPTAM cria uma categoria nova C_{N_c+1}, sem nenhum simplex ($N^s_{N_c+1} = 0$) e com um único vetor igual ao padrão de entrada $\mathbf{w}_{N_c+1} = \mathbf{I}$.

4.1.7 Resumo da etapa de treinamento

O processamento de um padrão de entrada, durante a etapa de treinamento de OPTAM, pode ser resumido nos seguintes passos (conforme fig. 4.1):

1. Apresentação do padrão de entrada $\mathbf{I}$.

2. Cálculo das funções de escolha de categorias: Calcular a $T^t_i(\mathbf{I}), i = 1, \ldots, N_c$ (eqs. 4.1, 4.2) para todas as categorias.

3. Competição: Selecionar a categoria C_I com $r_I = 0$ e a maior FEC: $T_I^t(\mathbf{I}) = \max\{T_i^t(\mathbf{I}) : r_i = 0\}_{i=1}^{N_c}$. Se todas as categorias estão rejeitadas ($r_i = 0, i = 1, \ldots, N_c$), então ir para o passo 8 (*Criação de uma categoria nova*).

4. Teste de Predição: Se a $P(C_I) \neq P_d$ (a predição associada da C_I é diferente da predição desejada), rejeitar a C_I ($r_I = 1$) e ir para o passo 3 (*Competição*). Senão, seguir para o passo 5 (*Expansão de categoria*).

5. Expansão de categoria: Expandir a C_I até o $\mathbf{I}$, ou bem criando um simplex novo S^*, com o $\mathbf{I}$ e os vetores de C_I conectáveis com o $\mathbf{I}$, ou substituindo algum vetor de peso $\mathbf{w}$ na C_I pelo $\mathbf{I}$. Ir para o passo 6 (*Teste de Vigilância*).

6. Teste de Vigilância: Se o $S_I \geq S_{max}$, depois da expansão, rejeitar a C_I ($r_I = 1$), retornar a C_I para seu estado anterior (suprimir o S^* ou restaurar o $\mathbf{w}$, segundo proceda) e voltar ao passo 3 (*Competição*). Senão, ir para o passo 7 (*Ressonância*).

7. Ressonância: Codificar o padrão de entrada pela C_I. Finalizar.

8. Criação de uma categoria nova: Selecionar os n vetores de pesos mais próximos conectáveis a $\mathbf{I}$ pertencentes às categorias mono-vetor com a predição P_d. Criar uma categoria nova C_{N_c+1} com um único simplex definido pelo $\mathbf{I}$ e os n vetores selecionados. Se existem menos de n vetores de pesos nestas condições, ou se a categoria nova não passa no teste de vigilância ($S_{N_c+1} \geq S_{max}$), remover a C_{N_c+1} e o simplex e criar uma nova categoria mono-vetor C_{N_c+1} com o $\mathbf{w}_{N_c+1} = \mathbf{I}$. Finalizar.

Estes passos se encontram descritos, em pseudocódigo, com mais detalhe na seção J.1 do apêndice J.

4.2 Etapa de Processamento

Durante a etapa de processamento, a saída de OPTAM, diante de um padrão de entrada, é a predição associada à categoria com a maior função de escolha de categoria. A FEC da categoria C_i, durante o processamento, que denotaremos por $T_i^p(\mathbf{I})$, leva em consideração, como comentado anteriormente, o tamanho da categoria, de modo que quando o padrão de entrada cai dentro de várias categorias, $T_i^p(\mathbf{I})$ é o maior para aquela com o tamanho S_i menor. Em primeiro lugar, definimos a função de $\chi_i(\mathbf{I})$ entre a categoria C_i e o padrão $\mathbf{I}$, durante o processamento, da

seguinte forma:

$$\chi_i(\mathbf{I}) = \begin{cases} \max\limits_{j=1,\ldots,N_i^s} \{T_{ij}(\mathbf{I})\} & N_i^s \geq 1 \\ 1 - \dfrac{\|\mathbf{I} - \mathbf{w}_i\|}{\sqrt{n}} & N_i^s = 0 \end{cases} \tag{4.7}$$

onde $T_{ij}(\mathbf{I})$ se define na eq. 3.13. Se a C_i é uma categoria politopo, a similaridade $\chi_i(\mathbf{I})$ é o máximo dos $T_{ij}(\mathbf{I})$ dos seus simplexes $S_{ij}, j = 1, \ldots, N_i^s$. Se, ao contrário, a C_i é uma categoria mono-vetor, então $\chi_i(\mathbf{I})$ é linearmente decrescente com a distância entre o $\mathbf{I}$ e o seu vetor de pesos $\mathbf{w}_i$.

Analogamente ao *Fuzzy* ARTMAP, quando o padrão de entrada se encontra numa região onde se sobrepõem várias categorias, para as quais a $\chi_i(\mathbf{I}) = 1$, àquela com o menor tamanho deve apresentar a máxima atividade $T_i^p(\mathbf{I})$. Denotando por:

$$\Theta(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{4.8}$$

definimos a função $\psi(\mathbf{I})$ como:

$$\psi(\mathbf{I}) = \Theta\left[\sum_{l=1}^{N_c} \Theta[\chi_l(\mathbf{I}) - 1] - 2\right] \tag{4.9}$$

Esta função verifica $\psi(\mathbf{I}) = 1$, se existem várias categorias com $\chi_i(\mathbf{I}) = 1$, ou seja, se o $\mathbf{I}$ cai dentro de várias categorias. Se, ao contrário, somente existe uma categoria C_i que satisfaça $\chi_i(\mathbf{I}) = 1$, então $\psi(\mathbf{I}) = 0$. A partir da $\psi(\mathbf{I})$, a função de escolha $T_i^p(\mathbf{I})$ para o processamento da categoria C_i se define da seguinte forma:

$$T_i^p(\mathbf{I}) = [1 - \psi(\mathbf{I})]\chi_i(\mathbf{I}) + \psi(\mathbf{I})\frac{\Theta[\chi_i(\mathbf{I}) - 1]}{\alpha + S_i} \tag{4.10}$$

onde $\alpha \gtrsim 0$ e o S_i foi definido na eq. 4.5. Quando $\psi(\mathbf{I}) = 1$, $T_i^p(\mathbf{I}) = 0$ para as categorias com $\chi_i(\mathbf{I}) < 1$, e $T_i^p(\mathbf{I}) = 1/(\alpha + S_i)$ para as demais categorias, de modo que $T_i^p(\mathbf{I})$ é decrescente com o tamanho S_i da categoria. Se, ao contrário, somente existe uma única categoria C_i com a $\chi_i(\mathbf{I}) = 1$, então $\psi(\mathbf{I}) = 0$ e $T_i^p(\mathbf{I}) = \chi_i^p(\mathbf{I})$. OPTAM codifica o padrão de entrada pela C_I com $T_i^p(\mathbf{I})$ máxima:

$$T_I^p(\mathbf{I}) = \max_{i=1,\ldots,N_c} \{T_i^p(\mathbf{I})\} \tag{4.11}$$

Finalmente, OPTAM proporciona como saída a predição $P(C_I)$ associada à categoria ganhadora C_I. O pseudocódigo, desta etapa de processamento, se inclui na seção J.2 do apêndice J.

4.3 Resultados

Realizamos experimentos comparando a rede OPTAM com PTAM e com algumas das redes ART mais populares: *Distributed* ARTMAP (DAM), com categorias retangulares, e *Gaussian* ARTMAP (GAM), que utiliza categorias com geometria hiper-elipsóide. Selecionamos estas redes por ser as que obtêm os melhores resultados em nossas simulações com problemas de ambas geometrias, como comprobamos no capítulo anterior. Também, neste caso, comparamos os resultados com os obtidos pela SVM [136]. Utilizamos dois problemas com geometria circular ("*Circle-in-the-Square*" (CIS) [143] e T5 [120]) e dois problemas com geometria retangular (*Chess* ou "*Generalized XOR*" [66] e T4 [120]), para avaliar cada rede em problemas com sua mesma geometria e com geometrias distintas (fig. 4.3). As complexidades destes conjuntos de dados se mostram na tabela 3.3.

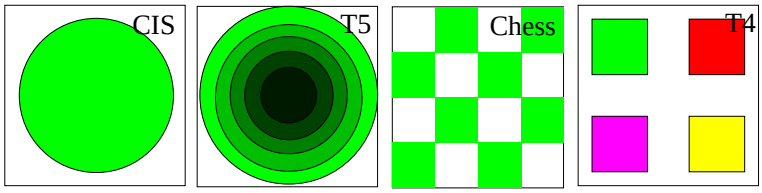


Figura 4.3: Conjuntos de dados CIS, T5 (circulares) e *Chess*, T4 (retangulares), com 2, 6, 2 e 5 predições de saída, respectivamente, usados nos experimentos de validação de OPTAM.

A vigilância ρ foi otimizada para cada rede ART e para cada problema, mediante um processo de validação cruzada igual ao empregado no capítulo anterior (20 conjuntos de treinamento, validação, teste com 10000 padrões cada um). As figs. 4.4 e 4.5 mostram a evolução do erro em função do número de categorias, pela média dos conjuntos de validação, para OPTAM, DAM, GAM, SVM e PTAM, variando a vigilância nos problemas CIS, *Chess*, T4 e T5. Os resultados de teste, obtidos com

os melhores valores de ρ sobre os conjuntos de testes, são mostrados na tabela 4.1. A rede OPTAM utiliza os valores $\rho = 0$ para os problemas CIS, T4 e T5, e $\rho = 0.75$ para *Chess*.

Tabela 4.1: O erro de teste ϵ e o número de categorias internas (os vetores para OPTAM-PTAM e os vetores de suporte para SVM) #C para cada classificador e problema, valores médios e desviação típica. O menor ϵ e o #C absoluto obtidos por uma rede ART e não ART indicam-se em negrito.

	CIS		Chess		T4		T5		Média		SDV	
	ϵ	#C	ϵ	#C	ϵ	#C	ϵ	#C	ϵ	#C	ϵ	#C
OPTAM	**0.6**	119	4.8	478	0.9	170	**2.6**	356	2.2	281	1.93	166.3
PTAM	1.0	102	1.7	242	1.0	158	4.1	387	**2.0**	222	1.47	124.0
GAM	1.3	**43**	3.0	90	1.9	89	6.1	**169**	3.1	**98**	2.14	52.3
DAM	1.3	238	**1.1**	**89**	**0.4**	**48**	5.6	630	2.1	251	2.37	265.4
SVM	**0.2**	578	**0.8**	475	0.6	271	**1.1**	827	**0.7**	538	0.38	231.2

Também aplicamos OPTAM ao conjunto de dados *Form*, variando N. A figura 4.6 mostra as curvas erro-categorias de OPTAM nos conjuntos de validação, obtidas variando $\bar{\rho}$ e parametrizadas pela ordem N de irregularidade, que se pode comparar com os gráficos correspondentes para FAM, GAM, DAM e PTAM (figs. 3.30-3.32). Finalmente, a fig. 4.7 mostra o erro de teste em função do número de categorias, com os valores de $\bar{\rho}$ selecionados, para os distintos valores de N e para as redes OPTAM, PTAM, FAM, GAM, DAM, EAM e ssEAM. Os resultados dos testes obtidos por estas redes mostram-se na tabela 4.2.

4.4 Discussão

A tabela 4.1 mostra que OPTAM não obtém menos erro que a SVM, que continua sendo o melhor classificador. O comportamento de OPTAM é variável: obtém um erro ligeiramente inferior (0.9%) que PTAM (1.0%) em T4 e o maior erro (4.8% contra 1.7% usando PTAM) em *chess*, que são os dois conjuntos de dados retangulares. Nos conjuntos de dados circulares OPTAM supera claramente a PTAM (0.6% contra 1.0% em CIS e 2.6% contra 4.1% em T5), DAM e GAM. O comportamento de OPTAM é também variável em número de categorias criadas, o qual é maior

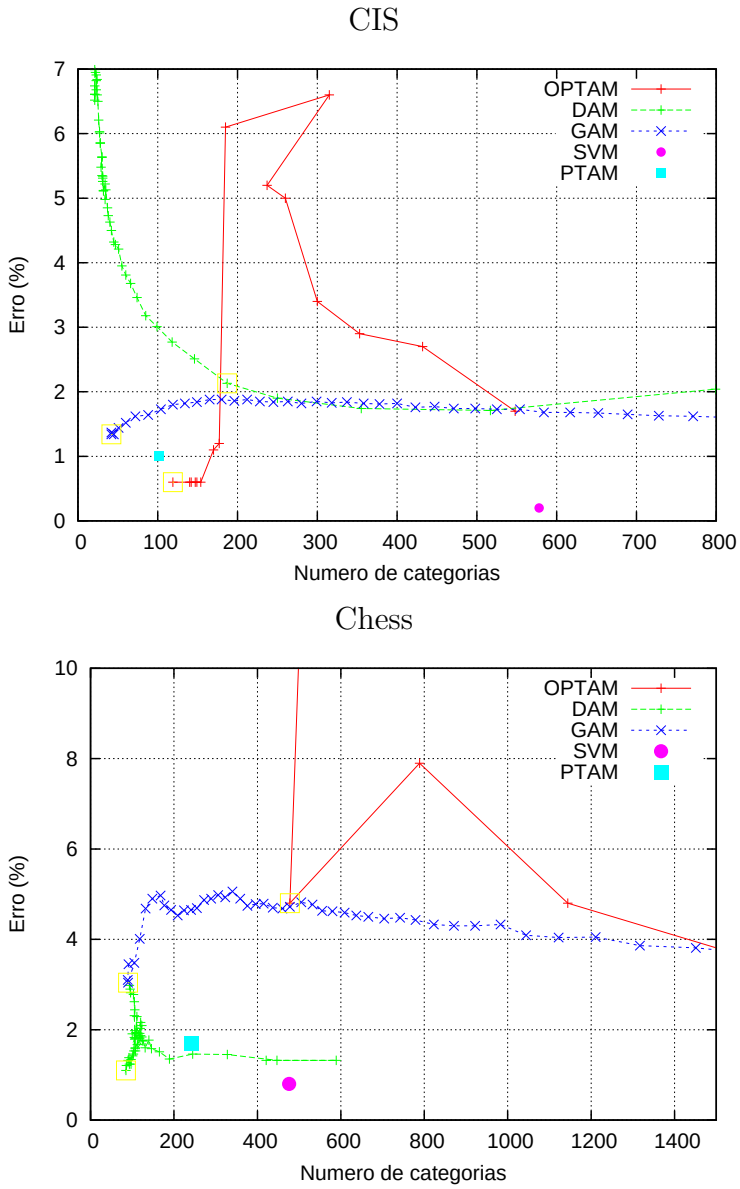


Figura 4.4: Erro de classificação em função do número de categorias sobre os conjuntos de validação para os problemas CIS e *Chess*, variando a vigilância. O ponto com o melhor compromisso entre ambas magnitudes indica-se com um quadrado vazio.

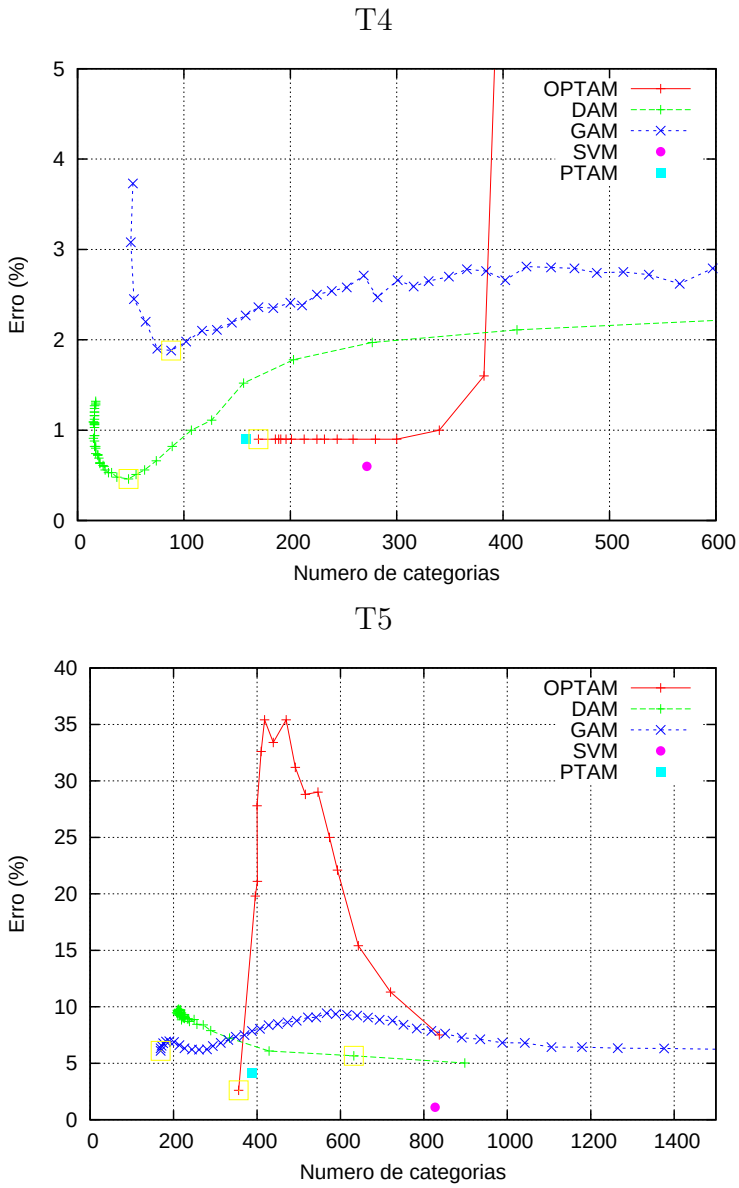


Figura 4.5: Erro de classificação em função do número de categorias sobre os conjuntos de validação para os problemas T4 e T5, variando a vigilância. O ponto com o melhor compromisso entre ambas magnitudes indica-se com um quadrado vazio.

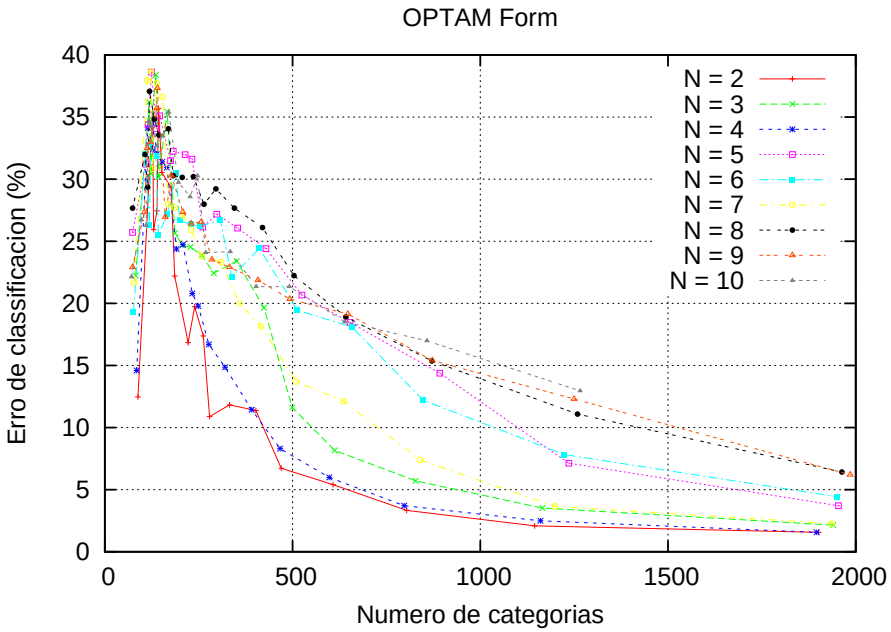


Figura 4.6: A taxa de erro em função do número de categorias sobre o conjunto de validação para OPTAM no problema *Form*, variando N.

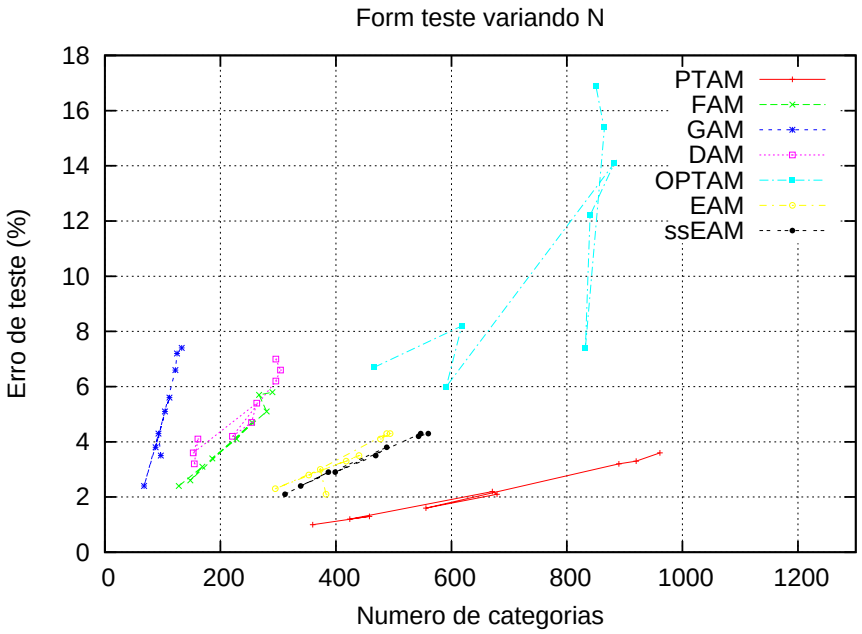


Figura 4.7: Comparação dos resultados de teste obtidos com OPTAM, PTAM, FAM, GAM, DAM, EAM e ssEAM no problema *Form*, variando N.

que PTAM nos conjuntos de dados CIS, *Chess* e T4, e o menor em T5. As figuras 3.3 e 3.4 também apresentam grandes variações nos erros de validação e número de vetores obtidos por OPTAM, variando a vigilância, com relação as outras redes ART.

Na média, OPTAM e PTAM obtêm erros (2.2% e 2.0%) similares a DAM (2.1%), e visivelmente inferiores a GAM (3.1%)[3]. Por outro lado, PTAM cria menos categorias (222) que as redes DAM (251) e OPTAM (281). Portanto, a sobreposição de categorias politopo não reduz o erro em relação ao PTAM ou DAM e, além do mais, aumenta o número de categorias criadas em comparação com estas duas redes. Finalmente, o erro alcançado por PTAM varia menos entre os problemas, uma vez que seu desvio típico é o menor entre as redes ART (1.5 frente à 1.9 para OPTAM, 2.1 para DAM e 2.4 para GAM).

É ilustrativo analisar o comportamento de OPTAM no problema *Chess*. Usando o $\rho = 0$, leva a $S_{max} = n$ pela eq. 4.6 e OPTAM somente cria duas categorias sobrepostas, que cobrem todo o espaço de entrada. Dado que no problema *Chess* as duas predições estão distribuídas entre as várias regiões separadas, estas duas categorias não são capazes de discriminar entre estas regiões, e o erro obtido é muito elevado (em torno aos 50%, como se indica na parte esquerda da fig. 4.4, painel inferior). Por conseguinte, OPTAM necessita vigilâncias elevadas ($\bar{\rho} = 0.75$) para alcançar um erro comparável às outras redes ART, criando um elevado número de categorias (478). No entanto, quando um padrão cai numa região de sobreposição entre as categorias, durante o processamento, a categoria mais pequena necessariamente não é a correta e, por isso, OPTAM obtém um erro de teste comparativamente alto (4.8%). Entretanto, nos problemas CIS, T4 e T5, o valor de $\rho = 0$ proporciona um erro e número de categorias menores, porque neles cada predição tem uma única região associada, e usando o $\rho = 0$ obtemos que o $S_{max} = (1 - \rho)n = n$, de modo que esta região pode ser corretamente coberta por uma única categoria. Por exemplo, em T5 as predições são concêntricas, de modo que as categorias criadas por OPTAM, para as distintas predições, se sobrepõem. Se um padrão se encontra numa região de sobreposição, durante o processamento, é codificado pela categoria mais pequena que, no caso de T5, é a correta. O mesmo ocorre com CIS e T4, mas não em *Chess*, daí o elevado erro de OPTAM neste conjunto de dados.

[3]Realizamos também experimentos adicionais com OPTAM, usando *Match Tracking*, obtendo o maior erro médio (4.2%) e o número de categorias (384), resultados que justificam sua ausência na tabela 4.1.

Tabela 4.2: O erro e o #C de testes obtidos por OPTAM, PTAM e as melhores redes ART retangulares e circulares, para o conjunto de dados *Form*. O melhor erro e #C estão em negrito.

	OPTAM		PTAM		FAM		GAM		DAM		EAM		ssEAM	
N	ϵ	#C	ϵ	#C	ϵ	#C	ϵ	#C	ϵ	#C	ϵ	#C	ϵ	#C
2	6.7	466	**1.0**	360	2.4	128	3.5	**97**	3.2	155	2.1	283	2.1	312
3	8.2	618	**1.3**	458	3.1	169	4.3	**93**	4.1	161	3.0	373	2.9	387
4	6.0	591	**1.2**	424	2.6	148	2.4	**68**	3.6	153	2.3	295	2.4	339
5	14.1	882	**2.2**	671	4.7	255	5.6	**112**	5.4	263	3.5	440	3.8	488
6	12.2	840	**2.1**	679	4.1	228	5.1	**104**	4.7	254	3.3	418	3.5	469
7	7.4	832	**1.6**	556	3.4	186	3.8	**88**	4.2	221	2.8	353	2.9	399
8	15.4	865	**3.2**	890	5.1	280	6.6	**122**	6.2	296	4.3	488	4.3	547
9	15.4	864	**3.3**	920	5.7	267	7.2	**125**	6.6	304	4.1	477	4.2	543
10	16.9	850	**3.6**	961	5.8	290	7.4	**133**	7.0	296	4.3	494	4.3	560
Media	11.4	756	**2.2**	659	4.1	217	5.1	**105**	5.0	234	3.3	402	3.4	449
STD	4.3	155	1.0	228	1.3	61	1.7	21	1.4	63	0.8	81	0.8	93

Os resultados de OPTAM sobre o conjunto de dados *Form*, variando N, mostram variando o $\bar{\rho}$ nos conjuntos de validação (fig. 4.6), uns intervalos, de erro e de categorias, superiores aos alcançados pelas outras redes ART, incluindo PTAM. Concretamente, o erro de OPTAM varia entre o intervalo de 10-40%, para vigilâncias baixas, e o intervalo de 3-7%, para vigilâncias elevadas. Em relação ao número de categorias, estas variam no intervalo de 100-2000. Ambos os intervalos são muito superiores aos das outras redes ART, como se pode comprovar comparando esta figura com as figs. 3.30-3.32, e indicam que OPTAM tem um comportamento mais variável. A partir destes resultados de validação, selecionamos os valores de vigilância $\bar{\rho}$ que proporcionam um compromisso entre o erro e o número de categorias. Usando estes valores selecionados, os resultados de teste, variando N (fig. 4.7), mostram um erro elevado, superior aos 6% para todos os valores de N, e um elevado número de categorias, juntamente com um comportamento mais errático que o das outras redes ART. Os valores médios e as suas desviações (tabela 4.2) mostram que OPTAM obtém os maiores erros e números de categorias que todas as redes ART. Também se pode observar uma variabilidade, comparativamente, elevada nos resultados de

OPTAM (alta STD), especialmente nos erros, se bem que a desviação dos números de categorias é inferior ao da rede PTAM.

Estes resultados inferiores, junto com a sua alta variabilidade, condicionados pelo número de regiões associadas à cada predição, como no caso de *Chess*, mostram um comportamento pouco estável, e sugerem que o uso de categorias politopo é mais eficaz– tanto em erro como em número de categorias– não se permitindo a sobreposição entre elas. Além do mais, torna-se contraditório utilizar categorias politopo irregulares, para uma melhor aprendizagem de problemas com geometrias complexas, e por outro lado se basear mais– como faz OPTAM– no seu tamanho que com a sua geometria, no momento de codificar um padrão por uma, dentre as várias, categorias sobrepostas.

Conclusões e Trabalhos Futuros

Nesta tese de doutoramento apresentamos uma série de redes neurais, inspiradas na *Teoria da Ressonância Adaptativa* que tentam ampliar a sua capacidade de aprendizagem à própria geometria das categorias internas. Estas propostas tratam de determinar em que medida o uso de geometrias genéricas, construídas ao longo do processo de treinamento, melhora o desempenho das redes ART.

Uma conseqüência imediata do uso das categorias com geometria genérica, é que cada categoria não pode definir-se, como na redes ART clássicas, utilizando exclusivamente um vetor de pesos, que atua como representante ou valor esperado da categoria, e uma expressão matemática (função de escolha da categoria), dependente do padrão de entrada e do representante, que define uma forma geométrica constante ao longo do treinamento. Ao contrário, nos modelos que propomos, as categorias genéricas estão definidas por uma série de padrões de treinamento selecionados, que atuam como vértices da categoria[4], uma vez que suponhamos fronteiras de categorias lineares a partes. Adicionalmente, a função de escolha de categoria depende do padrão de entrada, mas também de todos os vértices da categoria e é uma função complexa, envolvendo determinantes de ordem n, no caso de PTAM e OPTAM. É precisamente esta função uma das causas da elevada complexidade computacional dos modelos propostos.

A primeira aproximação, SAM, tem como principal característica o uso de funções gaussianas para sintetizar as funções de escolha de categoria. Estas funções gaussianas assumem valores não nulos somente dentro do volume de um simplex, de modo que as categorias internas têm esta forma geométrica. As funções gaussianas estão centradas em pontos interiores ao simplex, e todas têm a mesma extensão, de-

[4]No caso de SAM, o número de vértices encontram-se limitados no âmbito superior por $n+1$, por tratar-se das categorias simplex. Enquanto que em PTAM, usa as categorias com um número de vetores variável no tempo, que muda à medida que avança o treinamento.

terminada pelo parâmetro σ, do qual depende o número dos centros de gaussianas necessárias para cobrir o volume do simplex. SAM não permite a sobreposição entre as predições, de modo que a expansão de uma categoria está limitada pelas outras predições, e portanto, não necessita o parâmetro de vigilância. A aprendizagem implica criar uma nova categoria simplex entre as categorias existentes e o padrão de treinamento, ou adicionar o padrão à categoria, se esta tem menos de $n+1$ vértices.

Ainda que SAM obtém erros ligeiramente inferiores a FAM e DAM, isto se alcança com o custo de um número de categorias bastante superior, como indicam os resultados no problema CIS e, sobretudo, no CCIS. Além do mais, a supressão de um parâmetro a ajustar como a vigilância, que constitui uma vantagem comparativa em relação às redes ART, é compensada pela introdução do σ, que define a precisão como se aprendem as fronteiras entre as categorias[5]. Finalmente, o número de centros de funções gaussianas aumenta muito rapidamente com a dimensão do espaço de entrada, o qual compromete a aplicabilidade deste método em problemas de dimensionalidade elevada.

Os problemas derivados do uso de funções gaussianas nos sugeriu a possibilidade de calcular a função de escolha das categorias irregulares, com base na fronteira das mesmas, ao invés de no seu volume interior. Este enfoque conduz a usar hiperplanos para determinar se um padrão cai dentro ou fora de um simplex, e neste último caso para calcular a distância entre o padrão e o simplex. Este fato, unido à possibilidade de construir politopos irregulares, mediante a adição de simplexes adjacentes, permite a definição das categorias com geometria de politopo irregular em PTAM. O contorno de uma categoria politopo constitui uma aproximação linear por partes às fronteiras entre as predições de saída num problema de classificação. Esta representação geométrica é mais flexível e, potencialmente mais rica, que a das redes ART existentes, em relação à sua capacidade expressiva para modelar conjuntos de dados. As categorias politopo podem se expandir sem se sobreporem, ao contrário das outras redes ART. Deste modo, a expansão da categoria está limitada, de um modo natural, pelas outras categorias, e não pelo tamanho da categoria, um conceito alheio ao problema de classificação, que requer parâmetros adicionais, como a vigilância. Assim, PTAM não tem nenhum parâmetro ajustável, de modo que a sua operação é completamente automática, sem necessidade de nenhum processo de otimização.

PTAM usa um teste de sobreposição entre as categorias, que determina se a

[5]O parâmetro Γ tem menos relevância nos resultados, sempre que o seu valor seja baixo.

categoria expandida se sobrepõe com as demais. Uma categoria é ajustada quando um padrão de treinamento com uma predição diferente cai dentro do seu volume, e em tal caso o simplex que contém ao padrão é suprimido. PTAM cria uma categoria mono-vetor quando nenhuma categoria com a predição desejada pode expandir-se até o padrão de entrada, sem sobreposição.

PTAM obtém um erro baixo– inferior às redes ART– nos conjuntos bi-dimensionais sem ruído nem sobreposição de predições. Além do mais, a sua operação é menos dependente da geometria do conjunto de dados que as outras redes ART. PTAM também obtém melhores resultados nos conjuntos de dados artificiais com geometria irregular (*Form*), ainda que criando mais categorias que as redes ART quando aumenta a irregularidade da fronteira entre as predições. Estes resultados mostram que as categorias politopo, não sobrepostas, têm mais capacidade para aprender as fronteiras entre as predições que as categorias com geometria predefinida, próprias das redes ART clássicas. O número de vetores criados por PTAM é equiparável ao número de categorias das redes ART. De fato, ainda que cada categoria politopo C_i de PTAM requer $n + N_i^s$ vetores, a sua flexibilidade geométrica permite selecionar os vetores de modo que definam a fronteira da categoria. Ao contrário das redes ART, que necessitam criar várias categorias com formas geométricas predefinidas para aproximar estas fronteiras. Por esta razão, consideramos que é possível melhorar a expansão das categorias em PTAM, de modo que selecione melhor os vértices das categorias e obter, assim, um melhor erro e menos vetores.

Enfatizamos que PTAM obtém estes resultados sem o parâmetro de vigilância. A ausência deste parâmetro ajustável reporta as seguintes vantagens: i) O usuário final não tem que se preocupar pela operação interna de PTAM para determinar o seu valor, e portanto é mais fácil de usar para usuários não familiarizados com o seu funcionamento interno. É certo que as redes ART podem usar vigilância zero, entretanto este valor, geralmente, não proporciona os melhores resultados, como exibe o trabalho experimental. ii) Os testes de validação cruzada necessárias para determinar o valor ótimo da vigilância, podem requerer um custo computacional elevado. iii) Uma parte dos padrões de entrada deve se dedicar aos conjuntos de validação, nos testes de validação-cruzada e não podem ser usados para o treinamento. Este fato pode ser importante para os conjuntos de dados reais, que, frequentemente, têm um reduzido número de padrões.

Ainda que ao longo da tese os resultados obtidos pelos distintos classificado-

res foram apresentados normalmente empregando várias épocas[6], isto se fez com o objetivo de avaliar a máxima eficiência de cada classificador, que normalmente é superior com várias épocas de treinamento que com somente uma época. Especialmente pensando na sua comparação com a SVM que, como classificador com treinamento *off-line*, emprega também várias épocas. Porém, isto não significa que as redes ART necessitem de várias épocas, já que estas redes, e PTAM entre elas, permitem a aprendizagem *on-line* (capítulo 1), e portanto são capazes de aprender razoavelmente bem o conjunto em somente uma época– ainda que possivelmente com um maior erro–. Ao contrário das redes com treinamento *off-line* (SVM ou o *Perceptrón* Multi-camada) que necessitam de várias épocas de treinamento para alcançar resultados aceitáveis.

Outro aspecto relevante é a dependência da informação aprendida em relação da ordem de apresentação temporal dos padrões de entrada, usual nas redes ART. Um fator que contribui à esta dependência em PTAM é o fato de não aprender o padrão de treinamento quando a categoria ressonante já é a primeira ganhadora do processo de competição entre as categorias (subseção 3.1.5). Este critério tem como objetivo reduzir o número de padrões de treinamento aprendidos por uma categoria. A causa do problema é que a expansão de categorias só substitui os vetores quando estes cumprem a condição 3.17 (subseção 3.1.4), de modo que muitos padrões de treinamento provocam a criação de simplexes novos. Este fato reduz a eficiência na seleção dos padrões aprendidos, e portanto, uma categoria de PTAM em geral cria mais padrões e simplexes que os necessários para cobrir a sua região associada. As linhas de trabalho futuras incluem melhorar o processo de expansão de categorias para que a seleção de vetores seja mais eficiente, e não seja necessário usar critérios limitadores como o anterior, que contribuem à dependência com a ordem de apresentação dos padrões.

A eficácia do algoritmo de expansão-contração das categorias de PTAM não é tão elevada em presença de níveis de ruído altos e sobreposição entre as predições. Nestes casos, a etapa de ajuste de categorias provoca a ruptura de simplexes e a criação de categorias mono-vetor ruidosas. Por esta razão, uma linha de trabalho futuro contempla modificar esta etapa, para corrigir a categoria implicada num erro preditivo de uma forma mais eficiente, sem aumentar a informação aprendida (número de vetores e simplexes) e sem processar novamente alguns padrões de treinamento (os pertencentes, exclusivamente, ao simplex suprimido). Relacionado com este ajuste

[6]Cada época é uma apresentação completa do conjunto de treinamento.

de categorias, outro aspecto a modificar é a supressão de simplexes agudos, etapa que propõe um elevado custo computacional. Neste sentido, o método de seleção dos vetores na expansão de uma categoria deveria evitar a ocorrência de simplexes deste tipo.

A complexidade computacional é uma questão ainda aberta sobre PTAM. Esta complexidade é gerada pelos determinantes de ordem n necessários para calcular as funções de escolha, e pelos sistemas de equações lineares, também de ordem n, necessários para avaliar a sobreposição entre as categorias. Por esta razão, decidimos explorar a possibilidade de usar categorias politopo permitindo a sobreposição entre as categorias, com o objetivo de determinar a utilidade real das categorias irregulares por si mesmas nas redes ART existentes, as quais sim permitem a sobreposição. O modelo resultante, denominado OPTAM, é uma versão simplificada de PTAM, mas que segue tendo a vigilância como parâmetro a otimizar. Os resultados obtidos por OPTAM sobre os conjuntos de dados bi-dimensionais são inferiores aos de PTAM e das restantes redes ART, tanto em erro como em número de categorias, exibindo, além do mais, uma grande variabilidade e dependência com as propriedades geométricas das predições. Estes experimentos sugerem que as geometrias irregulares, por si só, não melhoram a capacidade de aprendizagem das redes ART, nem reduzem o número de categorias criadas. Ainda que em problemas simples OPTAM obtém resultados aceitáveis usando $\rho = 0$, o ajuste da vigilância torna-se imprescindível em problemas mais complexos– como no caso de *Chess*, cujas predições estão misturadas–, sem que isso proporcione menos erros ou categorias que PTAM.

A computação local é uma das propriedades básicas que todo modelo neural deve possuir. O processamento num modelo deste tipo deve ser distribuído sobre uma série de unidades (neurônios artificiais) inter-conectadas operando em paralelo, sendo a saída de cada unidade função exclusivamente das suas entradas. A diferença das redes com aprendizagem *off-line*, nas quais é possível o cálculo dos pesos na etapa de treinamento utilizando os algoritmos não locais (como ocorre, por exemplo, se usamos o algoritmo SVM para determinar os pesos de uma rede *feed-forward*), a aprendizagem *on-line*, típica das redes ART, implica que também o algoritmo de treinamento deve admitir uma implementação local, uma vez que a rede ART deve ser capaz de comutar entre os modos de treinamento e processamento. Portanto, o cálculo dos pesos deve ser realizada na própria rede (aprendizagem local), e não mediante um algoritmo externo, em geral não local. Esta questão, logo, é fundamental para a possível implementação em *hardware* de qualquer modelo neural ART.

Os modelos formulados nesta tese de doutorado não admitem uma implementação com computação local, e de fato não incluimos neles nenhum diagrama com a sua arquitetura, como é normal se fazer com os modelos ART. A razão é que a complexidade dos modelos propostos (por exemplo, associada ao uso de vetores de pesos como vértices e não como os protótipos de categorias, ou à implementação do teste de sobreposição, entre outros aspectos) impede a plasmação sobre uma arquitetura relativamente simples, como a correspondente às redes ART clássicas. Por esta razão, nesta tese, formulamos estes modelos na sua vertente algorítmica, sem fazer ênfase na sua possível implementação em *hardware*, ainda que ressaltando a sua inspiração no modelo ART, em cujos conceitos básicos (funções de seleção, valores esperados, teste de predição, ressonância, rejeição) se baseiam. Com esta formulação algorítmica pretendemos avaliar a contribuição das categorias politopo às redes ART convencionais, ainda que prorrogando para médio prazo a sua implementação local. Com este objetivo, estamos trabalhando na implementação mais eficiente de certas partes do algoritmo, de modo que seja possível a sua formulação local.

A principal contribuição desta tese é a introdução das categorias com geometrias irregulares não predefinidas na aprendizagem das redes ART. Esta característica, como se observou ao longo da memória, introduz importantes mudanças na operação destas redes, entre elas a supressão do parâmetro de vigilância. Por estas razões, consideramos que o mais indicado para provar as contribuições dos modelos propostos, foi compará-los com os classificadores *standard*, especialmente os baseados em ART, sobre os problemas de classificação de referência, sobre os quais fosse possível avaliar o desempenho comparativo das distintas aproximações. Nesta perspectiva, adiamos para o trabalho futuro a aplicação dos modelos propostos em problemas de classificação reais de interesse prático.

A partir das considerações anteriores podemos concluir que, ainda que o uso de categorias politopo irregulares não sobrepostas melhora a capacidade das redes ART para aproximar as fronteiras entre as predições, o trabalho realizado e os resultados obtidos mostram que o método de seleção dos vetores, e o procedimento de expansão e ajuste das categorias, são de fundamental importância para uma aprendizagem eficiente (baixo erro e número de vértices). Neste sentido, estamos trabalhando na formulação de aproximações alternativas para as etapas de expansão e contração das categorias, com o objetivo de obter menor erro de classificação com menos vetores, quer dizer, de um modo mais eficiente. Assim, uma redefinição das categorias politopo em termos de hiperplanos, sem simplexes, poderia facilitar a

formulação destas alternativas para a expansão de categorias.

Finalmente, outro objetivo prioritário da investigação futura é reduzir a complexidade computacional de PTAM. Somos conscientes que o uso de categorias politopo, necessariamente, implica numa complexidade superior às das redes ART existentes, que usam geometrias mais simples e menos flexíveis. Porém, a elevada complexidade de PTAM, provocada pelo uso de determinantes e sistemas de equações lineares de ordem n, compromete, como no caso de SAM, a sua aplicabilidade em problemas de dimensionalidade elevada. Por esta razão, estamos trabalhando para simplificar a função de escolha de categoria e uma implementação mais eficiente dos testes de sobreposição entre o segmento-simplex e o simplex-categoria (funções O_{ls} e O_{sc} definidas nos apêndices F.1 e F.2).

Part II

Summary of the Thesis

Preface

The present thesis is the continuation of the work developed in the last 90's by our research group– the Intelligent Systems Group (GSI)– in the field of Adaptive Resonance Theory (ART) neural networks. In this work, we proposed Multi-channel ART (MART), an ART network for the unsupervised learning of multi-channel signals [53, 51, 52]. Among other features, this network uses a different adaptive vigilance for each internal category, in order to adapt the discrimination ability of the network to the signal properties (noise level, variability, etc.). However, this approach is limited by the pre-defined geometry of the internal categories, which is given by the similarity measure used to cluster the input patterns. Our present work tries to overcome this limitation by proposing ART models which use internal categories without any pre-defined geometry. During the training phase, the network learns the geometry of each category using the training patterns codified by this category. The category geometry is not given by some similarity measure, as in the ART networks, but it is defined on-line as the training proceeds. This approach is not possible in unsupervised learning, where a similarity or distance measure must be used to cluster the input patterns. In this case, the similarity measure completely determines the geometry of the internal categories, which try to model the clusters in the input data. On the contrary, categories with general geometry can only be defined if the desired output for a training pattern is available, in order to determine if it belongs to a given internal category. The desired outputs can be used to "learn" the geometry of these categories. Therefore, categories with general geometry are only possible using supervised training, where the desired outputs can be used to learn the geometry of categories. In the present work, the proposed models are oriented to classification problems, and we delayed for future research its extension to other kinds of problems, as regression.

The categories with irregular geometries change in several ways the usual ope-

ration of ART networks. We developed several models with irregular internal categories, based on different principles. Our first proposal, called Simplex ARTMAP [61], was based on simplex-shaped categories, and Gaussian functions were used to define the category choice function. However, the categories of Simplex ARTMAP are limited by the simplex geometry, and the calculation of Gaussian functions in high-dimensional spaces is computationally too expensive. Consequently, we proposed another model, called PolyTope ARTMAP (PTAM) [62, 64], with irregular polytope[7] categories, delimited by selected training patterns. Another feature of PTAM is that categories do not overlap, so there is a test which avoids the category overlap and limits its expansion. Since this limitation makes unnecessary a maximum category size, the vigilance parameter can be removed and PTAM remains without any tuning parameter. We have also proposed a third model, called Overlapping PolyTope ARTMAP (OPTAM) [63] based on overlapped polytope categories in order to evaluate the interest of category overlap among irregular categories.

[7] A polytope is the generalization of the concept of polygon to $\mathbb{R}^n$.

Introduction

The Adaptive Resonance Theory (ART) constitutes a model for the development of artificial neural networks which is inspired by the studies of Grossberg and Carpenter [25]. These networks feature on-line, incremental learning, with self-organized (ART1, ART2, ART3, Fuzzy ART) and supervised learning (Fuzzy ARTMAP (FAM) [28], Gaussian ARTMAP (GAM) [144], Distributed ARTMAP (DAM) [36], Ellipsoid ARTMAP (EAM) [3] and FasArt [17], among others). These models have been widely used in many application fields, including robotics [55], data mining [122], information fusion [34], data clustering [147], multi-channel pattern recognition [51], etc. Default ARTMAP [21] compiles the basic structure of an ARTMAP network.

Some drawbacks of the ART networks, either in supervised and unsupervised applications, have been identified in the literature. The proliferation of internal categories is perhaps the most important one [60, 65]. It has been associated with the presence of noise [107], and with the inefficiency of the category geometry [144], which requires an excessive number of categories in order to cover the input space. Category proliferation has been also associated [138] with overtraining, and cross-validation [102] has been suggested as a solution. Distributed ARTMAP [36] proposes distributed learning as another solution to category proliferation. Parrado-Hernández *et al* [120] quantitatively analyzed the influence of distributed learning in category proliferation for supervised classification problems. One conclusion of this work is that the reduction in the number of categories created by DAM with respect to FAM depends on the geometry of the output predictions in the data set. Specifically, DAM creates less categories than FAM in data sets with non-rectangular category geometries, while the difference between them is low in data sets with rectangular geometry.

The above analysis suggests that category geometry is an important factor in

the performance and the number of categories created by an ART network. While the first ART networks used hyperbox-shaped categories (FAM), some other models have been proposed with non-rectangular category geometries, given by the Category Choice Function (CCF), which acts as a measure of similarity between the input pattern and the category. These functions can be also understood as "support" functions, which take the value 1 inside the category support (region of the input space covered by the category) and < 1 outside, decreasing with the distance between pattern and category. For example, GAM [144] uses CCF with hyperellipsoidal symmetry. Hypersphere ARTMAP (HAM) [2] and Ellipsoid ARTMAP (EAM) [3] use hypersherical and hyperellipsoidal internal Category Representation Regions (CRRs) [4] respectively. Despite of using non-hyperbox CRRs, the category geometry in HAM and EAM[8] is also pre-established (spherical or ellipsoidal) by the CCF. The simmetry of the GAM categories is also pre-established and independent of the geometric structure of the data. However, if the geometric structure of the input data (e.g, the borders among the output predictions) is not similar to the CRR geometry, the network may need more categories in order to learn the data. Specifically, in supervised classification problems the networks with pre-defined CRR geometry are more suited to data sets with similar geometric properties– e.g., output predictions with rectangular geometries for FAM-DAM, or ellipsoidal geometries for GAM, HAM and EAM–, and their performance may decrease otherwise.

The present work goes one step beyond, and proposes to use categories whose geometry is not pre-established. This geometry should be learnt by the network during the training stage, so that its border should be defined by the training patterns. On the other hand, since the category geometry is not pre-defined, this geometry should be defined by some vectors (e.g., selected training patterns), where number may be different for each category (on the contrary, categories in classical ART networks have the same number of vectors). Simplex ARTMAP (SAM, chapter 1) is the first ART model which we proposed without pre-defined internal categories. SAM approximates the border among output predictions using non-overlapping simplex categories, because the simplex is the simplest closest volume in $\mathbb{R}^n$ with the least number of vertices $(n + 1)$. The category choice function is defined as a sum of Gaussian functions. Since categories can not overlap, the vigilance parameter is not necessary. However, the Gaussian spread replaces vigilance as tuning parameter. On the other hand, the number of Gaussian functions required to cover

[8]In EAM, the hyperellipsoid ratio of lengths is fixed, and defined by the non-adaptive parameter μ, which acts as a tuning parameter. It also limits the geometric flexibility of the hyperellipsoids.

the simplex volume quickly increases with the dimension of the input space, which difficults its application to high-dimensional data sets. These limitations suggested us to replace simplex categories defined by Gaussian choice functions by polytope categories, defined by hyperplanes whose vertices are selected training patterns.

Based on these ideas, we proposed Polytope ARTMAP (PTAM, chapter 2), whose internal categories are irregular polytopes. This geometry is more flexible than the pre-defined category geometries in order to approximate the borders among the desired output predictions in the data set. When an internal category in PTAM learns a training pattern, it can expand only towards it, and not in other directions, as happens in the ART networks in order to keep its pre-defined geometry. Thus, the internal categories of PTAM can cover only the regions in the input space populated by patterns with its own output prediction. Consequently, these categories do not need to overlap, as it is usual in classical ART networks. PTAM uses an overlap test which avoids the expansion of a category if it overlaps with other categories. Therefore, the expansion of a category is limited by the other categories, so it does not need to be limited by a maximum category size, as in the ART networks. The vigilance parameter can be removed, so PTAM does not need any tuning parameter.

The relevance of category overlap was evaluated comparing PTAM with an alternative model, called Overlapping PolyTope ARTMAP (OPTAM, chapter 3), which also uses polytope internal categories, but overlap among them is allowed. Since categories overlap in OPTAM, a maximum category size is used to limit its expansion, given by the vigilance parameter, so OPTAM is not a parameter-free network.

Chapter 1

Simplex ARTMAP

The mathematical definition of internal categories requires to define its choice function, which has a high value inside its support and a low value outside it. We used a set of linear borders (straight lines in $\mathbb{R}^2$, planes in $\mathbb{R}^3$, and hyperplanes in $\mathbb{R}^n$) in order to compose a piece-wise approximation to the borders among the output predictions in the data set. We selected the simplest irregular closed volume with linear borders, the simplex, as the elementary block in order to compose the volume of the region associated to each output prediction. The simplex is the closed volume with linear borders defined by the minimum number of vertices ($n+1$ vertices in $\mathbb{R}^n$). The region associated to an output prediction is described as a set of adjacent non-overlapped simplexes. This is the origin of our first proposal of ART network with generic geometry is called Simplex ARTMAP (SAM). Each internal category C_i in SAM is defined by its associated prediction $P(C_i)$ and N_i vectors ($1 \leq N_i \leq n+1$) which act as vertices of the simplex. The table 1.1 reports the nomenclature associated with SAM.

1.1 Training phase

For each category C_i, the category choice function $T_i^t(\mathbf{I})$ during the training phase must be $T_i^t(\mathbf{I}) \simeq 1$ if the input pattern $\mathbf{I}$ falls inside the simplex, and $T_i^t(\mathbf{I}) < 1$ otherwise. In order to determine if $\mathbf{I}$ falls inside or outside the simplex, we used a sum of Gaussian functions (eq. 1.1) whose values are non-zero just inside the simplex volume and zero outside the simplex.

Table 1.1: Nomenclature.

n	Dimension of the input pattern $\mathbf{I}$
$\mathbf{I}$	Input pattern $\mathbf{I} = (I_1, \ldots, I_n) \in [0,1]^n$ non complement-coded
P_d	Desired prediction for the input pattern $\mathbf{I}$
N_c	Number of internal categories
C_i	i-th internal category $i = 1, \ldots, N_c$
$P(C_i)$	Associated prediction of internal category C_i
N_i	Number of vectors of category C_i, $1 \leq N_i \leq n+1$
$\mathbf{w}_{ij}$	j-th weight vector (vertex) of category C_i, $j = 1, \ldots, N_i$
$T_i^t(\mathbf{I})$	Choice function of category C_i during training
$T_i^p(\mathbf{I})$	Choice function of category C_i during processing
$\mathbf{c}_i(\mathbf{p}_i)$	Center of the Gaussian function inside category C_i
	given by indexes $\mathbf{p}_i = (p_{i2}, \ldots, p_{iN_i}) \in \mathbb{N}^{N_i-1}$
α_{ik}	Number of Gaussian functions in the direction of vector $\mathbf{w}_{ik} - \mathbf{w}_{i1}$
σ	Spread of the Gaussian functions
$A_i(\alpha_i)$	Set of index vectors $\mathbf{p}_i$ in category C_i
Γ	Threshold for the category choice functions
r_i	Reset state of category C_i
	($r_i = 1$ if C_i is reset, and $r_i = 0$ otherwise)

$$f(\mathbf{I}, \mathbf{c}_i(\mathbf{p}_i)) = exp\left(-\frac{\|\mathbf{I} - \mathbf{c}_i(\mathbf{p}_i)\|^2}{2\sigma^2}\right) \tag{1.1}$$

Each Gaussian function has a spread σ and a center $\mathbf{c}_i(\mathbf{p}_i)$ given by:

$$\mathbf{c}_i(\mathbf{p}_i) = \mathbf{w}_{i1} + \sum_{k=2}^{N_i} \frac{p_{ik}}{\alpha_{ik}}(\mathbf{w}_{ik} - \mathbf{w}_{i1}) \tag{1.2}$$

$$\tag{1.3}$$

In this equation, $\mathbf{w}_{i1}, \ldots, \mathbf{w}_{iN_i}$ are the vertices of the category C_i. The positive integer indexes $p_{ik}, k = 2, \ldots, N_i$ compose the vector $\mathbf{p}_i = (p_{i2}, \ldots, p_{iN_i}) \in \mathbb{N}^{N_i-1}$, and they define the Gaussian centers inside the shadowed triangle in fig. 1.1, which shows these centers $\mathbf{c}_i(\mathbf{p}_i)$ in a 2D example. The indexes p_{ik} belong to the set $A_i(\alpha_i)$, defined as:

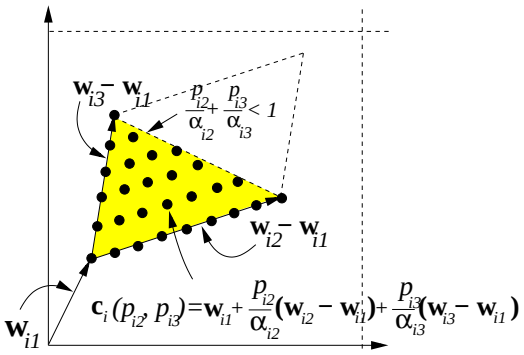


Figure 1.1: Simplex category C_i ($N_i = 3$) in $\mathbb{R}^2$, and centers $\mathbf{c}_i(p_{i2}, p_{i3})$ of the Gaussian functions.

$$A_i(\alpha_i) = \left\{ \mathbf{p}_i \in \mathbb{N}^{N_i - 1} : p_{ik} = 0, \ldots, \alpha_{ik}; k = 2, \ldots, N_i; \right. \\ \left. \frac{p_{il}}{\alpha_{il}} + \frac{p_{im}}{\alpha_{im}} \leq 1, \forall l, m = 2, \ldots, N_i; l \neq m \right\} \tag{1.4}$$

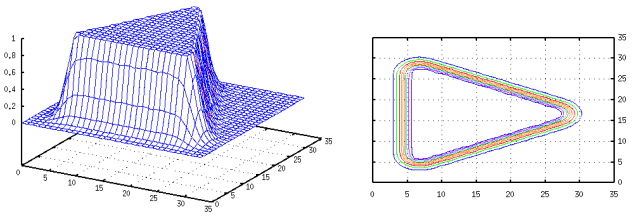
Here, $\mathbb{N}$ is the set of non-negative integers. Also, α_{ik} is the number of Gaussian functions in the direction of the vector $\mathbf{w}_{ik} - \mathbf{w}_{i1}, k = 2, \ldots, N_i$ (eq. 1.5):

$$\alpha_{ik} = \left\lceil \frac{\|\mathbf{w}_{ik} - \mathbf{w}_{i1}\|}{2\sigma\sqrt{2 \log 2}} \right\rceil, \quad k = 2, \ldots, N_j \tag{1.5}$$

The value of α_{ik} is determined imposing that every Gaussian function has the value $1/2$ in the midpoint between two consecutive centers of functions in that segment ($\lceil x \rceil$ is the ceiling function, or the smaller integer not less than x). The choice function $T_i^t(\mathbf{I})$ of category C_i is given by eq. 1.6, and it is shown in fig. 1.2 in $\mathbb{R}^2$.

$$T_i^t(\mathbf{I}) = min \left[1, \sum_{\mathbf{p}_i \in A_i(\alpha_i)} exp\left(-\frac{\|\mathbf{I} - \mathbf{c}_i(\mathbf{p}_i)\|^2}{2\sigma^2} \right) \right] \\ \alpha_i = (\alpha_{i2}, \ldots, \alpha_{N_i}) \tag{1.6}$$

SAM selects the category C_I with the highest $T_i^t(\mathbf{I})$ which is not reset ($r_i = 0$). Since $T_i^t(\mathbf{I}) \simeq 0$ outside the category C_i, a threshold $\Gamma \gtrsim 0$ is imposed in such a way



(a) (b)

Figure 1.2: Example of activation function (a) and its contour (b) of a simplex in $\mathbb{R}^2$.

that there is no winner ($I = -1$) if $T_i^t(\mathbf{I}) < \Gamma, \forall i$. Therefore, the winner category C_I is defined by:

$$I = \begin{cases} arg \max\limits_{i=1,\ldots,N_c} \{T_i^t(\mathbf{I})\} & \exists C_i : T_i^t(\mathbf{I}) > \Gamma, r_i = 0 \\ -1 & \text{otherwise} \end{cases} \tag{1.7}$$

When the winner category C_I learns an input pattern with desired prediction P_d, C_I expands until $\mathbf{I}$. Fig. 1.3 shows several cases: category with one vector $\mathbf{w}$ in the panel (a), class with 2 vectors in the panel (b) and input pattern added to a set of categories (panel (c)). The last case is more complex because it is necessary to select the nearest vectors from several categories. SAM selects the n nearest vectors belonging to categories with the desired prediction in order to create a new category with them and the input pattern (panel (d)). Thus, some "holes" among categories are possible, as panel (d) shows.

SAM avoids the overlap between categories if they have different associated predictions. The overlap between predictions is tested calculating the choice function $T_i^t(\mathbf{c}_l)$ of the expanded category C_i for all the centers $\mathbf{c}_l$ of the categories C_l with $P_l \neq P_i$. If $T_i^t(\mathbf{c}_l) \neq 0$ for some center $\mathbf{c}_l$, then there is overlap and category C_i must not be expanded. This "Prediction Overlap Test" somehow substitutes the vigilance test in the ART networks. However, no bound on the category size is imposed, and the vigilance parameter ρ, with high influence in the results in the classical ART networks, is no longer necessary. The deviation σ of the Gaussian functions is the only parameter to set. We have used a low σ value ($\sigma = 0.01$) in all the experiments.

Based on these foundations, the Simplex ARTMAP algorithm, in the training phase, operates as follows. For each input pattern $\mathbf{I} \in [0, 1]^n$ with desired prediction P_d:

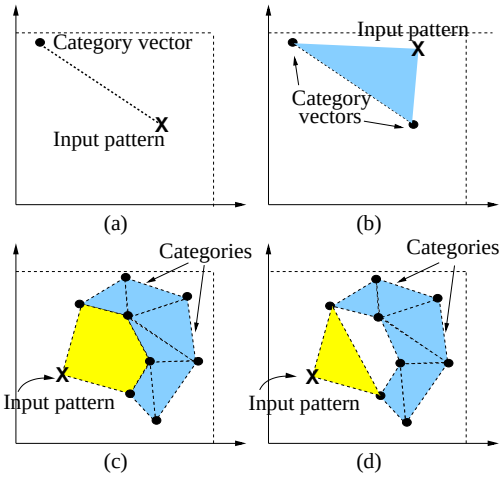


Figure 1.3: Learning examples in $\mathbb{R}^2$.

1. **Step 1**. Calculate the choice functions $T_i^t(\mathbf{I})$ of the N_c categories (eq. 1.6).

2. **Step 2**. If $T_i^t(\mathbf{I}) > \Gamma$ for some category C_i, then the category C_I with the highest $T_i^t(\mathbf{I})$ is winner category, which is the tentative candidate to code the input pattern ($I = -1$ if $T_i^t(\mathbf{I}) < \Gamma, \forall i$). The associated prediction $P(C_I)$ of the winner category C_I is tested.

   2.1. **Step 2.1**. If $P(C_I) = P_d$, then C_I is already the right category for $\mathbf{I}$, so it is assigned to C_I. However, it is not stored as a new vector of category C_I, because C_I already covers $\mathbf{I}$. End.

   2.2. **Step 2.2**. Otherwise, C_I has been incorrectly activated, because it covers regions with prediction $P_d \neq P(C_I)$. In order to correct this error, the category C_I is broken into N_I categories, one for each vertex of C_I. Go to step 1 to search a new winner category.

3. **Step 3**. If there is no winner category ($I = -1$), it is necessary to extend some category with the right prediction P_d, or to create a new category, in order to learn the current pattern. It seems reasonable to seek the nearest categories from the input pattern, so SAM searches the nearest vectors $\mathbf{w}_i$ from $\mathbf{I}$ which belong to categories with prediction P_d (the Euclidean distance is used).

   3.1. **Step 3.1**. If there is no category with prediction P_d, then a new category C_{N_c+1} is created with the input pattern as its only one vector: $N_{N_c+1} = 1, \mathbf{w}_{(N_c+1)1} = \mathbf{I}$. End.

3.2. **Step 3.2**. Otherwise, the n nearest vectors $\mathbf{w}_i$ which belong to categories with prediction P_d are selected (there might be $r < n$ vectors).

3.2.1. If the selected r vectors belong to the same category C_i, and $N_i < n+1$, then C_i is not a whole simplex. In this case, $\mathbf{I}$ is added as a new vector to C_i, which expands towards $\mathbf{I}$ (see fig. 1.3 (a) and (b)). If this expansion generates prediction overlap, C_i is contracted and a new category C_{N_c+1} with just one vector $\mathbf{w}_{(N_c+1)1} = \mathbf{I}$ is created. End.

3.2.2. If the selected r vectors do not belong to the same category, or if they belong to the same category C_i but $N_i = n+1$ (C_i is a simplex), then $\mathbf{I}$ can not be added to C_i. In this case, a new category C_{N_c+1} is created with the selected vectors and $\mathbf{I}$ (see fig. 1.3(d)). The set of categories with prediction P_d is expanded with this new category that covers regions among the r vectors and $\mathbf{I}$. If prediction overlap is present, C_{N_c+1} is removed, and a new single-vector category C_{N_c+1} is created with $N_{N_c+1} = 1$ and $\mathbf{w}_{(N_c+1)1} = \mathbf{I}$. End.

1.2 Processing phase

During the processing phase, the category choice function $T_i^p(\mathbf{I})$ is given by $T_i^t(\mathbf{I})$. If $T_i^t(\mathbf{I}) < \Gamma, \forall i$, $T_i^p(\mathbf{I})$ is defined as the minimum distance between $\mathbf{I}$ and a vector of category C_i. Specifically, $T_i^p(\mathbf{I})$ is given by:

$$T_i^p(\mathbf{I}) = \psi(\mathbf{I}) T_i^t(\mathbf{I}) + (1 - \psi(\mathbf{I})) \left(1 - \frac{1}{\sqrt{n}} \min_{k=1,\ldots,N_i} \|\mathbf{I} - \mathbf{w}_{ik}\| \right) \tag{1.8}$$

where $\psi(\mathbf{I})$ is given by:

$$\psi(\mathbf{I}) = \Theta \left[\sum_{i=1}^{N_c} \Theta(T_i^t(\mathbf{I}) - \Gamma) \right] \tag{1.9}$$

and $\Theta(x)$ is defined by:

$$\Theta(x) = \begin{cases} 1 & x > 0 \\ 0 & x \leq 0 \end{cases} \tag{1.10}$$

If $T_i(\mathbf{I}) > \Gamma$ for some category C_i, then $\psi(\mathbf{I}) = 1$ and $T_i^p(\mathbf{I}) = T_i^t(\mathbf{I}), i = 1, \ldots, N_c$. Otherwise, $T_i(\mathbf{I}) < \Gamma, \forall i = 1, \ldots, N_c$, so that $\psi(\mathbf{I}) = 0$ and $T_i^p(\mathbf{I})$ is decreasing with the minimum distance between $\mathbf{I}$ and a vertex of C_i, scaled by $\sqrt{n}$ because $0 \leq \|\mathbf{I} - \mathbf{w}_{ik}\| \leq \sqrt{n}$. Finally, the output of SAM is the prediction $P(C_I)$ associated to the category C_I with the highest choice function $T_I^p(\mathbf{I}) = \max\{T_i^p(\mathbf{I})\}_{i=1}^{N_c}$.

1.3 Results

The performance of SAM was tested on the well-known "Circle-in-the-Square" (CIS) [143] and "Concentric Circle In the Square" (CCIS) data sets. Both are 2-dimensional data sets, and therefore they allow graphical representation of the categories and the borders among predictions. In the CIS data set, the points inside and outside the circle must be discriminated (see fig. 1.4 (left)). In the CCIS data set, the points must be classified in three concentric circles and outside them (see fig. 1.5 (left)).

	SAM		FAM		DAM		SVM	
$\%p$	$\%\epsilon$	#C	$\%\epsilon$	#C	$\%\epsilon$	#C	$\%\epsilon$	#sv
10	0.7	460	1.0	213	1.0	183	0.19	558
20	0.9	544	1.4	218	1.3	187	0.20	559
30	1.1	574	1.7	222	1.7	186	0.21	566
40	1.3	584	2.0	226	2.1	188	0.26	581
50	1.4	608	2.2	227	2.5	187	0.24	582
60	1.5	631	2.3	229	3.0	188	0.25	593
70	1.6	654	2.6	235	4.1	190	0.25	608
Avg.	**1.2**	**579**	**1.9**	**225**	**2.3**	**187**	**0.23**	**578**

Table 1.2: Error ($\%\epsilon$) and the number of categories (#C), or support vectors (#sv), obtained using SAM, FAM, DAM and SVM in the CIS data set (columns). The rows are different percentages of circle area.

The CIS data set was created by randomly generating points inside and outside the circle. Several circle sizes, given by percentages of the square area covered by the circle ranging from 10% to 70% (values higher than 70% correspond to circles outside the square) were tried. For each percentage, 20 training and 20 test sets (10,000

patterns each one) were generated. The table 1.2 summarizes the results obtained by SAM, FAM, Distributed ARTMAP (DAM) and Support Vector Machine (SVM), which has been used as the reference classifier. Each row shows the error rate (%ϵ) and the number of categories (#C) for these classifiers and for each percentage of circle size. The values in each row are the averages over the 20 trials (in each trial, we used a different training-test sets couple). The last row shows the average over all the circle sizes. In the case of FAM and DAM, only the best results (minimum error rate) for all the vigilance values ($\rho = 0, 0.1, \ldots, 0.9$), and choice functions (Weber Law and Choice-By-Difference) for FAM, are reported. SVM was implemented using the Torch3 library [40], and the best results were obtained by using Gaussian kernels with spread $\sigma = 0.1$. Fig. 1.4 shows an example of the simplexes (triangles in $\mathbb{R}^2$, left panel) and the supports of the Gaussian functions (right panel) created by SAM in the CIS data set (percentage = 30%).

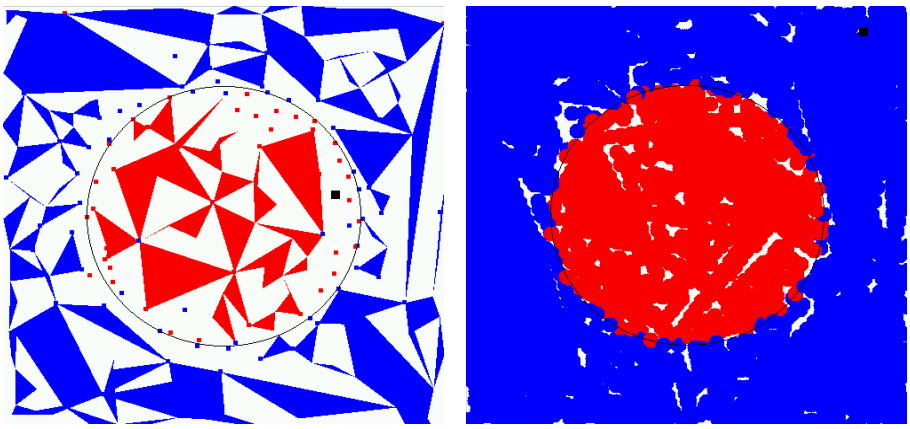


Figure 1.4: Triangles (left) and supports of the Gaussian functions (right) created by SAM in the CIS data set (percentage = 30%).

In the data set CCIS, each classifier must discriminate among points in regions 1, 2, 3 and 4 (see fig. 1.5 (left)). The table 1.3 shows the error rate and the number of categories (or support vectors) obtained by SAM, FAM, DAM and SVM. Results were also averaged over 20 trials, each one with a different training-test sets couple. Fig. 1.5 (right) shows the supports of the Gaussian functions created by SAM in this data set (each support is a circle of radius 2σ).

The mentioned classifiers were also tested on the well-known Iris Plant data set, from the UCI Machine Learning Repository [12]. However, all the classifiers achieved the perfect classification on this data set, so it is not an acceptable benchmark for

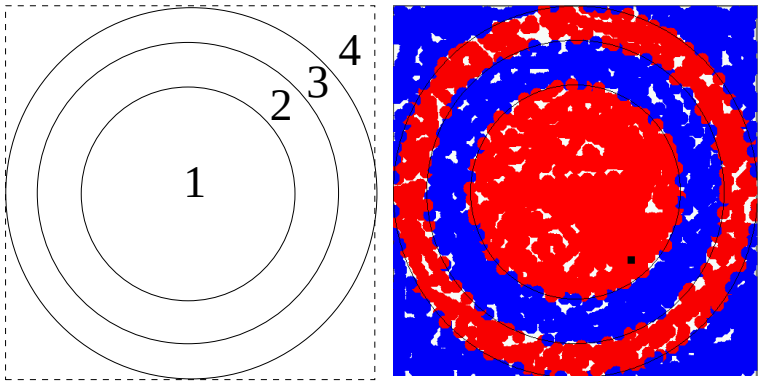


Figure 1.5: Regions of the different output predictions in the data set CCIS (left). Supports of the Gaussian functions created by SAM in this data set (right).

comparison.

SAM		FAM		DAM		SVM	
%ϵ	#C	%ϵ	#C	%ϵ	#C	%ϵ	#sv
4.1	1141	6.0	279	6.7	206	0.9	560

Table 1.3: Avg. error and the number of classes obtained using SAM, FAM, DAM and SVM for the CCIS data set.

1.4 Discussion

SVM provides the best results for all the experiments, although the number of support vectors is higher than the number of categories achieved by FAM or DAM in both data sets. Comparing the ART-based approaches (FAM, DAM and SAM), the error rate achieved by SAM is lower than using DAM or FAM, although SAM creates more categories. This number of categories is comparable to, and higher than, the number of support vectors created by SVM in data sets CIS and CCIS respectively. It is important to emphasize the ability of SAM to learn classes, fitting well the borders among predictions (see figs. 1.4 and 1.5).

Apparently, triangles of classes in fig. 1.4 (left) do not cover the input space, but we must take into account that category activacion is given by Gaussian functions (eq. 1.6) and in the right panel their bases cover the whole input space, so the

triangles do not need to expand more. There are also several one-vector categories, as shown in the left panel. SAM just learns and stores the simplex vertices $\mathbf{w}_{ij}$, from which the centers of the Gaussian functions are calculated (eq. 1.3). The number of vectors $\mathbf{w}_{ij}$ (not shown in the tables 1.2 and 1.3) is lower than the number of categories due to vector sharing among them.

The results achieved by FAM and DAM correspond to the best performance for the different values of ρ and the choice function. In contrast, SAM does not need to tune the vigilance parameter, which is very important to the performance of the ART networks. In particular, FAM with $\rho > 0.95$ obtain less error than SAM, but it creates more categories. The error rate achieved shows that learning ability of SAM is comparable or higher than these classifiers without parameter tuning. Most of the classification errors in SAM are due to the fixed value of the parameter σ, which leads to overlap in the borders among categories (figs. 1.4 (right) and 1.5 (right)).

However, the high number of one-vector categories created by SAM compared to FAM and DAM suggests that the process of category break (section 1.1, step 2.2) creates too many categories and it should be refined. Other drawback of SAM is the need to tune the spread σ of the Gaussian functions, so that vigilance is replaced by σ as a tuning parameter. On the other hand, the number of centers of the Gaussian functions raises exponentially as the dimension of the input space grows and as the spread σ reduces. This fact increases very much computational cost of SAM, and it reduces its interest in high-dimensional data sets. These reason leaded us to redefine the category choice functions in order to overcome these drawbacks, as we shall see in the following chapter.

Chapter 2

PolyTope ARTMAP

An internal category in a typical ART network is usually defined by a weight vector which represents the input patterns codified by the category. The response of each internal category to an input pattern is given by its category choice function (CCF), which depends both on the input pattern and on the category weight vector, and defines the geometry of the category representation region (CRR). The CCF is constant inside the CRR, although some ART variants (FasArt [17], AFC [128]) feature categories with non-constant CCF. The network creates categories that approximate the borders among the output predictions in the data set. This approximation is more accurate, and requires less categories, if the geometries of the categories and the output predictions are similar. For example, FAM and DAM probably need more categories than EAM or HAM in order to achieve equal performance on a data set with circular geometries. However, the categories in PolyTope ARTMAP (PTAM) are irregular polytopes delimited by hyperplanes which compose a piece-wise linear approximation to the border among the output predictions. The category vertices are weight vectors selected among the previous training patterns in such a way that the category borders fit the borders among the output predictions. Each output prediction can be associated to one or several polytopes. Irregular polytope categories are more flexible than categories with pre-defined geometry to fit complex borders among the output predictions, so they might achieve less error. If these categories were more efficient to discriminate among predictions, a less number of categories could even be created. On the other hand, the number of weight vectors is different for each category in PTAM, and it is adaptive during training, as opposite to the other ART networks, whose categories are defined by a fixed number of weight

vectors: category center and deviations in GAM, category center in EAM, vector of ranges in each dimension in FAM, etc.

In order to define mathematically the CCF of irregular polytope categories, we describe each polytope as composed by several adjacent simplexes. The polytope CCF is calculated using a choice function for each simplex. The CCF of the polytope is 1 if the input pattern falls inside one of its simplexes. Otherwise, the CCF is lower than 1 and decreasing with the distance between the pattern and the polytope.

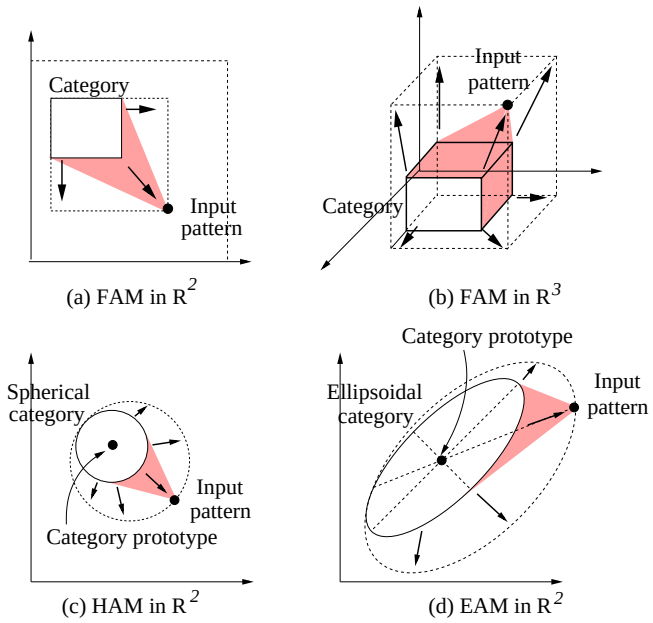


Figure 2.1: Category expansion in fast learning with FAM in $\mathbb{R}^2$ (a) and $\mathbb{R}^3$ (b), HAM in $\mathbb{R}^2$ (c) and EAM in $\mathbb{R}^2$ (d). The dashed line is the new category after learning. The expanded category includes regions not in the direction (shadowed in the figure) from the old category to the input pattern.

In the "fast learning" case (the most usual) of the training in the ART networks, the resonant category C (with prediction $P(C)$) expands towards the training pattern $\mathbf{I}$ (with desired prediction $P(C)$). The pre-defined geometry of the category forces C not only to expand towards $\mathbf{I}$, but also in other directions (fig. 2.1). Thus, C may cover not only the region between it and $\mathbf{I}$, which is supossed to be associated to prediction $P(C)$, but also other regions associated to different predictions. When a training pattern is presented to the network with desired prediction $P \neq P(C)$ and it falls in these regions, C is selected as the winner, but Match Tracking is invoked and another category C' with $P(C') = P$ covers that region (or a new category is

created). The category C is not updated, so C and C' overlap. Therefore, the pre-defined category geometry somehow leads to category overlap (fig. 2.2 (a)), because categories invade regions with different predictions. The Fuzzy Min-Max Neural Network (FMMNN) [130] is a remarkable exception among the ART-based networks, because it uses categories with pre-defined geometry (hyperbox) which do not overlap if they have different associated predictions (similarly to Fuzzy ARTMAP). However, FMMNN uses geometrical techniques in order to test the category overlap[1] and, in this case, the expanded category is contracted. Therefore, when a category in FMMNN tries to expand towards an input pattern, the expansion may be not possible if the hyperbox shape leads to cover regions populated by other predictions.

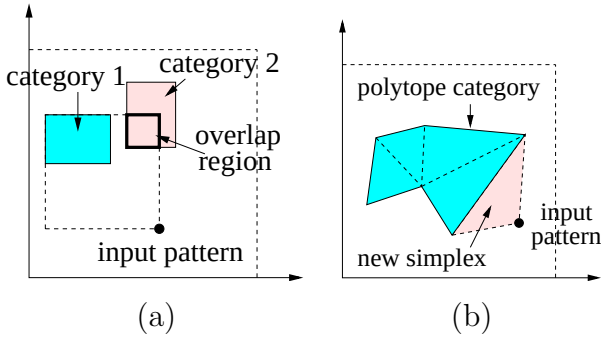


Figure 2.2: (a) FAM learning in $\mathbb{R}^2$. When category 1 learns the input pattern, while keeping its hyperbox shape, it overlaps with category 2, which has a different associated prediction. (b) Expansion of a polytope category in $\mathbb{R}^2$ creating a new simplex (triangle in $\mathbb{R}^2$).

In the ART networks, when a test pattern falls inside an overlap region, the small categories (the most specific ones) are candidates for the resonance in the first place, because its CCF is higher than the CCF of the bigger categories. Thus, the category size is used to select one among several overlapped categories. Also, there is a maximum category size, which limits the category expansion. Since categories can overlap, and they can cover regions with a different desired prediction, if category expansion were not limited, e.g. by this maximum size, the network might create only one category for each output prediction, covering the smallest volume which includes all the training patterns with this prediction. This fact might reduce the discrimination ability of the network, e.g. if each prediction has several non-adjacent regions, so the regions of the different predictions are mixed. Therefore, the ART networks avoid the expansion if the size of the expanded category

[1]Since FMMNN uses hyperbox categories, the test for category overlap is very simple.

surpasses the maximum size, defined by the vigilance parameter[2]. Unfortunately, there is no automatic way to compute the upper limit on the category size for a given data set, so the vigilance must be determined by trial and error, often using cross-validation. In the case of FMMNN, the overlap test limits the category expansion, but a maximum category size, given by the parameter θ, similar to the vigilance, is also used. Both overlap test and maximum category size seem to have the same objective, i.e., to limit the category expansion, so they are somehow redundant. Therefore, FMMNN could work without vigilance, using only the overlap test to limit the category expansion.

We can conclude that pre-defined category geometries are closely related with other typical features of ART networks, such as category overlap, category size and vigilance. PTAM categories have no pre-defined geometry, although they are internally managed as a connected set of adjacent simplexes. Since the simplex covers the smallest volume defined by its vertices, the polytope category can expand only towards the input pattern– and not in other directions– by adding a new simplex with vertices in the polytope and in the input pattern (fig. 2.2 (b)). The polytope categories can be built in such a way that they cover only regions in the input space populated by patterns with its equal output prediction (if there are patterns with a different prediction between the category and the input pattern, the category is adjusted as described below). If category overlap is forbidden, the polytope categories can still expand, as opposed to the other ART networks, because they cover only the regions of the input space populated by patterns with its own output prediction. Therefore, the internal categories of PTAM are not allowed to overlap. This fact has two consequences: (1) since categories do not overlap, the category size is not required to select one among several overlapped categories; (2) since the other categories limit the expansion of a given category, a maximum category size is not required to limit this expansion, so the vigilance parameter is not necessary.

PTAM replaces the vigilance test in the ART networks by the overlap test (subsection 2.1.4), similarly to FMMNN: if a category overlaps with other categories after the expansion towards the training pattern, the category is reset and contracted, and a new category is selected to codify the training pattern. This test avoids that

[2]Although the value $\bar{\rho} = 0$ is often used for classification tasks, the vigilance is raised to positive values when Math-Tracking is invoked, so there is a maximum category size until the presentation of the following training pattern.

a category covers regions populated by input patterns with a different prediction. It limits the category expansion without tuning parameters. Overlap among categories is tested by geometric techniques, which are described in the subsection 2.1.4 and in the appendix M.

The overlap test itself is not enough to avoid category overlap. A category C with prediction $P(C)$ can cover a region populated by patterns with desired prediction $P_d \neq P(C)$ if no training pattern with prediction P_d fell inside that region in the past. So, the categories must be able to correct previous wrong expansions. If a training pattern $\mathbf{I}$ with desired prediction P_d were presented to the network in the future, $\mathbf{I}$ would fall inside the CRR of C. In this case, C could be corrected removing the simplex which contains $\mathbf{I}$. The simplex vertices which do not belong to any other simplex would be used as new input patterns. This is done in the category adjustment step (subsection 2.1.3).

If the active category C passes the overlap test after expansion towards $\mathbf{I}$, the prediction test (subsection 2.1.3) determines if its desired prediction P_d is equal to $P(C)$. The training pattern $\mathbf{I}$ is assigned to the category which passes both overlap and prediction tests (resonance, subsection 2.1.5). If no category passes both tests, PTAM creates a new one (subsection 2.1.6). Since a polytope category must have at least one simplex, defined by $n+1$ vertices, n training patterns with desired prediction P_d are required to create a new single-simplex category. If the new category overlaps with other categories, or there are less than n weight vectors with prediction P_d, a new single-vector category is created with the training pattern $\mathbf{I}$.

2.1 Training phase

The following subsections describe the main steps of the training phase of PTAM. The table 2.1 summarizes the nomenclature used in the text.

2.1.1 Category Choice Function

Since each polytope category C_i is composed by a set of simplexes, its CCF $T_i^t(\mathbf{I})$ during the training phase is defined as the maximum of the choice functions $T_{ij}(\mathbf{I})$ of its simplexes $S_{ij}, j = 1, \ldots, N_i^s$:

n	Dimension of the input space
$\mathbf{I}$	Input pattern, $\mathbf{I} \in [0,1]^n$, not complement-coded
P_d	Desired prediction for the input pattern
N_c	Number of categories
C_i	i-th Category, $i = 1, \ldots, N_c$
$P(C_i)$	Associated prediction of category C_i
N_i^s	Number of simplexes in category C_i
S_{ij}	j-th Simplex of category C_i, $j = 1, \ldots, N_i^s$
$\mathbf{w}_{ijl}$	l-th Weight vector (vertex) of simplex S_{ij} in category C_i, $l = 1, \ldots, n+1$
h_{ijk}	k-th Hyperplane of simplex S_{ij} in category C_i, $k = 1, \ldots, n+1$
$\mathbf{w}_{ijkl}$	l-th Weight vector of hyperplane h_{ijk} of simplex S_{ij} in category C_i, $l = 1, \ldots, n$
$T_i^t(\mathbf{I})$	Category Choice Function (CCF) of category C_i during training
$T_i^p(\mathbf{I})$	CCF of category C_i during processing
$T_{ij}(\mathbf{I})$	Choice Function of simplex S_{ij} in category $C_i, j = 1, \ldots, N_i^s$
r_i	Reset state of category C_i ($r_i = 1$ if C_i is reset, $r_i = 0$ otherwise)
$\mathbf{w}_i$	Weight vector of single-vector category C_i ($N_i^s = 0$)
$\widehat{\mathbf{w}}_{ijk}$	Weight vector of simplex S_{ij} which does not belong to hyperplane h_{ijk}
$\mathbf{w}_{ijk}^*$	Direction vector of hyperplane h_{ijk} (appendix L)

Table 2.1: Nomenclature used in the text.

$$T_i^t(\mathbf{I}) = \max_{j=1,\ldots,N_i^s} \{T_{ij}(\mathbf{I})\} \qquad i = 1, \ldots, N_c \tag{2.1}$$

$T_{ij}(\mathbf{I})$ is defined in such a way that $T_{iJ}(\mathbf{I}) = 1$ if $\mathbf{I}$ falls inside the simplex S_{ij}: in this case, $T_i^t(\mathbf{I}) = T_{iJ}(\mathbf{I}) = 1$ because $\mathbf{I}$ falls inside the CRR of C_i. Otherwise, $0 < T_{ij}(\mathbf{I}) < 1$, and it decreases with the distance $d(\mathbf{I}, S_{ij})$ between the input pattern and the simplex. Thus, $T_i^t(\mathbf{I}) = T_{iJ}(\mathbf{I})$, where S_{iJ} is the nearest simplex from $\mathbf{I}$ in C_i. We will separately describe both cases.

Input pattern inside a simplex

The border of simplex S_{ij} is given by $n+1$ hyperplanes $\{h_{ij1}, \ldots, h_{ij(n+1)}\}$. The hyperplane h_{ijk} is defined by n weight vectors $\{\mathbf{w}_{ijk1}, \ldots, \mathbf{w}_{ijkn}\}$ (simplex vertices), and it can be described by the following equation:

$$g_{ijk}(\mathbf{I}) \equiv sgn(\phi_{ijk}(\widehat{\mathbf{w}}_{ijk}))\phi_{ijk}(\mathbf{I}) = 0 \tag{2.2}$$

where $sgn(x) = 1$ if $x > 0$, $sgn(0) = 0$ and $sgn(x) = -1$ if $x < 0$. The vector $\widehat{\mathbf{w}}_{ijk}$ is the weight vector of simplex S_{ij} which does not belong to hyperplane h_{ijk}[3], and $\phi_{ijk}(\mathbf{x})$ is given by:

$$\phi_{ijk}(\mathbf{x}) = \begin{vmatrix} x_1 - w_{ijk1,1} & \dots & x_n - w_{ijk1,n} \\ w_{ijk2,1} - w_{ijk1,1} & \dots & w_{ijk2,n} - w_{ijk1,n} \\ \dots & & \\ w_{ijkn,1} - w_{ijk1,1} & \dots & w_{ijkn,n} - w_{ijk1,n} \end{vmatrix} \tag{2.3}$$

In this equation, $w_{ijkl,m}$ is the m-th component of $\mathbf{w}_{ijkl}$ (see appendix K). Eq. 2.2 splits $[0,1]^n$ in two regions, one for which $g_{ijk}(\mathbf{I}) > 0$ and another defined by $g_{ijk}(\mathbf{I}) < 0$. The factor $sgn(\phi_{ijk}(\widehat{\mathbf{w}}_{ijk}))$ in eq. 2.2 guarantees that $g_{ijk}(\mathbf{I}) > 0$ if $\mathbf{I}$ falls in the hyperplane side that is inside the simplex, and $g_{ijk}(\mathbf{I}) < 0$ outside the simplex. An input pattern $\mathbf{I}$ falls inside the simplex S_{ij} only if it falls in the inner side of the $n+1$ hyperplanes $h_{ij1}, \dots, h_{ij(n+1)}$ of S_{ij}. So $T_{ij}(\mathbf{I}) = 1$ if and only if $g_{ijk}(\mathbf{I}) > 0$, $\forall k = 1, \dots, n+1$.

Input pattern outside a simplex

The choice function $T_{ij}(\mathbf{I})$ of the simplex S_{ij} must satisfy $0 < T_{ij}(\mathbf{I}) < 1$ outside the simplex and it must be decreasing with the distance between the pattern and the simplex. We define $T_{ij}(\mathbf{I}) = 1 - d(\mathbf{I}, S_{ij})/\sqrt{n}$, where $d(\mathbf{I}, S_{ij})$ is the distance pattern-simplex (note that $0 \leq d(\mathbf{I}, S_{ij}) < \sqrt{n}$, being $\sqrt{n}$ the diagonal length of hyperbox $[0,1]^n$). We use an approximated value of $d(\mathbf{I}, S_{ij})$ since its accurate calculation is computationally demanding for $n > 2$. Let us consider the hypersphere centered in the centroid $\mathbf{c}_{ij}$ of the simplex vertices $\mathbf{c}_{ij} = \sum_{l=1}^{n+1} \mathbf{w}_{ijl}/(n+1)$, whose radius R_{ij} is the maximum distance between this centroid and the simplex vertices $R_{ij} = \max\{\|\mathbf{w}_{ijl} - \mathbf{c}_{ij}\|\}_{l=1}^{n+1}$. We consider two cases:

1. If the input pattern falls inside this hypersphere ($\|\mathbf{I} - \mathbf{c}_{ij}\| < R_{ij}$), $d(\mathbf{I}, S_{ij})$ is approximated by the distance between $\mathbf{I}$ and its nearest hyperplane h_{ijk} in the simplex ($\mathbf{I}$ falls outside S_{ij}, so $g_{ijk}(\mathbf{I}) < 0$):

[3]Since a simplex S_{ij} has $n+1$ vectors and a hyperplane h_{ijk} is defined by n vectors, there is only one vector in S_{ij} which does not belong to h_{ijk}.

$$d(\mathbf{I}, S_{ij}) \simeq \min_{k \in \mathcal{S}} \frac{|g_{ijk}(\mathbf{I})|}{\|\mathbf{w}^*_{ijk}\|}, \quad \|\mathbf{I} - \mathbf{c}_{ij}\| < R_{ij} \tag{2.4}$$

where $\mathbf{w}^*_{ijk}$ is the direction vector of hyperplane h_{ijk} (see appendix L), and $|g_{ijk}(\mathbf{I})| \ / \|\mathbf{w}^*_{ijk}\|$ is the distance between $\mathbf{I}$ and h_{ijk}. The set $\mathcal{S}$ includes the indices k of hyperplanes h_{ijk} in S_{ij} for which $g_{ijk}(\mathbf{I}) < 0$, so that $\mathbf{I}$ falls in the side of h_{ijk} outside S_{ij}.

2. If $\mathbf{I}$ falls outside the hypersphere, then the distance between the pattern and the simplex is approximated by the minimum distance between $\mathbf{I}$ and a vertex of S_{ij}:

$$d(\mathbf{I}, S_{ij}) \simeq \min_{l=1,\ldots,n+1} \{\|\mathbf{I} - \mathbf{w}_{ijl}\|\}, \quad \|\mathbf{I} - \mathbf{c}_{ij}\| \geq R_{ij} \tag{2.5}$$

Finally, the activation function $T_{ij}(\mathbf{I})$ of simplex S_{ij} is given by:

$$T_{ij}(\mathbf{I}) = \begin{cases} 1 & g_{ijk}(\mathbf{I}) > 0, \quad k = 1, \ldots, n+1 \\ 1 - \dfrac{d(\mathbf{I}, S_{ij})}{\sqrt{n}} & \text{otherwise} \end{cases} \tag{2.6}$$

2.1.2 Competition

The competition among polytope categories in PTAM selects the category C_I with the highest CCF which is not reset ($r_I = 0$):

$$T^t_I(\mathbf{I}) = \max_{i=1,\ldots,N_c} \{T^t_i(\mathbf{I}) : r_i = 0\} \tag{2.7}$$

If all the categories are reset ($r_i = 1, i = 1, \ldots, N_c$), $I = -1$ and a new category is created (subsection 2.1.6).

2.1.3 Prediction Test and Category Adjustment

PTAM compares the prediction $P(C_I)$ associated to the winner category C_I with the desired prediction P_d for the training pattern $\mathbf{I}$ in the Prediction Test (PT). If

$P(C_I) = P_d$, C_I passes the PT and the following step (Resonance, subsection 2.1.5) is executed. Otherwise, $P(C_I) \neq P_d$, C_I does not pass the PT and it is reset ($r_I = 1$). If $T_I^t(\mathbf{I}) = 1$, the input pattern falls inside the CRR of C_I, so the region covered by C_I is populated by patterns with $P_d \neq P(C_I)$. In this case, the simplex S_{IJ} which verifies $T_{IJ}(\mathbf{I}) = 1$ is removed in order to adjust category C_I (Category Adjustment). The competitive process selects another non-reset category and the Overlap Test is performed again (subsection 2.1.4).

If $\mathbf{w}_{IJl}$ $(1 \leq l \leq n+1)$ is a weight vector of simplex S_{IJ} which does not belong to any other simplex in C_I, it is presented as a new training pattern, after the current one $\mathbf{I}$, in order to be assigned to other categories. If another simplex S_{pq} covers $\mathbf{w}_{IJl}$ during the classification of $\mathbf{I}$, S_{pq} might be removed when $\mathbf{w}_{IJl}$ is presented again (in a new category adjustment step), and this fact could lead to an infinite loop. In order to avoid these loops, $\mathbf{w}_{IJl}$ creates a new single-vector category C^* when simplex S_{IJ} is removed. Thus, the Overlap Test ensures that no simplex S_{pq} covers C^*. When $\mathbf{w}_{IJl}$ is presented again as a new input pattern, C^* is removed. Therefore, $\mathbf{w}_{IJl}$ can not remove any simplex in the second presentation and infinite loops are not possible.

2.1.4 Overlap Test

The Overlap Test (OT) determines if C_I overlaps with the other categories when it expands towards $\mathbf{I}$. If there is overlap, C_I does not pass the OT and it is reset ($r_I = 1$). In this case, a new non-reset category is selected by the competitive process. If C_I passes the OT, the category C_I codifies $\mathbf{I}$ (Resonance, subsection 2.1.5).

There are several possible situations in the Overlap Test, depending on the relative position of $\mathbf{I}$ and C_I, which are illustrated in fig. 2.3:

1. $\mathbf{I}$ falls inside the CRR of C_I ($T_I^t(\mathbf{I}) = 1$). In this case, C_I already covers $\mathbf{I}$, so it does not expand and overlap with other categories is not possible. Thus, C_I passes the OT (fig. 2.3 (1)).

2. $\mathbf{I}$ falls outside the CRR of C_I ($T_I^t(\mathbf{I}) < 1$). Then, the category C_I can learn $\mathbf{I}$ expanding towards $\mathbf{I}$, either including $\mathbf{I}$ as a weight vector or replacing a weight vector of C_I by $\mathbf{I}$. In both cases, new segments between $\mathbf{I}$ and vertices of C_I must be created, which can not overlap with the existing categories. Therefore,

the set $\mathcal{A}_I(\mathbf{I})$ of weight vectors in category C_I which are connectable from $\mathbf{I}$ without overlap must be created. The function $O_{ls}(\mathbf{I}, \mathbf{w}, S_{ij})$ (line-simplex overlap), defined in eq. M.5 (appendix M), determines if the line segment $\overrightarrow{\mathbf{Iw}}$ overlaps with the simplex S_{ij} ($O_{ls}(\mathbf{I}, \mathbf{w}, S_{ij}) = 1$) or it does not overlap ($O_{ls}(\mathbf{I}, \mathbf{w}, S_{ij}) = 0$). The set $\mathcal{A}_I(\mathbf{I})$ can be defined as $\mathcal{A}_I(\mathbf{I}) = \{\mathbf{w} \in C_I : O_{ls}(\mathbf{I}, \mathbf{w}, S_{ij}) = 0, \forall S_{ij}\}$. Depending on $|\mathcal{A}_I(\mathbf{I})|$ (cardinality of set $\mathcal{A}_I(\mathbf{I})$), there are three cases:

2.1. If $0 \leq |\mathcal{A}_I(\mathbf{I})| < n$ (fig. 2.3 (2a)), there are not enough connectable vertices from $\mathbf{I}$ in C_I to create a new simplex between them, because the segments from $\mathbf{I}$ to C_I overlap with the other categories. C_I does not pass the OT because it can not expand towards $\mathbf{I}$ without overlap.

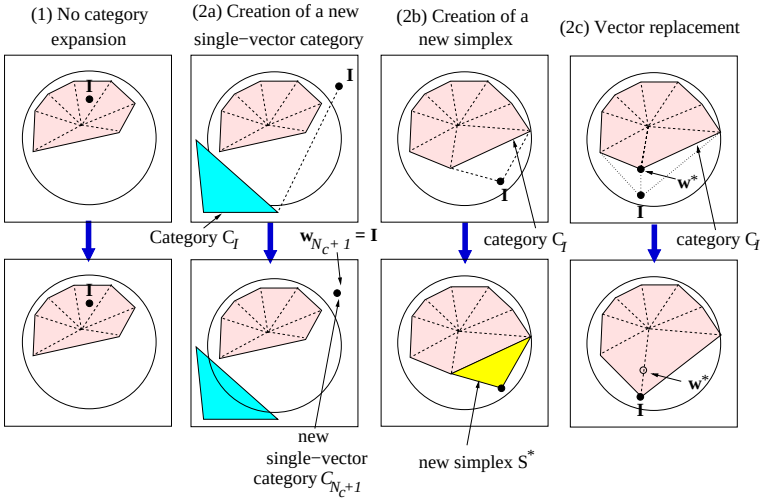


Figure 2.3: Polytope category learning examples for the "Circle-In-the-Square" (CIS) problem in $\mathbb{R}^2$. (1) The input pattern falls inside C_I, which is not expanded. (2a) Less than $n = 2$ weight vectors are connectable from $\mathbf{I}$ in C_I: C_I is not expanded and $\mathbf{I}$ creates a new single-vector category C_{N_c+1} with $\mathbf{w}_{N_c+1} = \mathbf{I}$. (2b) Exactly n weight vectors are connectable from $\mathbf{I}$, and C_I expands towards $\mathbf{I}$ creating a new simplex S^*. (2c) The number of connectable weight vectors from $\mathbf{I}$ is 3 ($> n = 2$): vector $\mathbf{w}^*$ is replaced by $\mathbf{I}$ without volume loss for category C_I.

2.2. If $|\mathcal{A}_I(\mathbf{I})| = n$ (fig. 2.3 (2b)), a new simplex S^* belonging to C_I can be created using $\mathbf{I}$ and the n weight vectors of $\mathcal{A}_I(\mathbf{I})$ (subsection 2.1.4). If S^* overlaps with other categories, C_I is removed and it does not pass the OT. Otherwise, C_I passes the OT.

2.3. If $|\mathcal{A}_I(\mathbf{I})| > n$ (fig. 2.3 (2c)), some weight vector in $\mathcal{A}_I$ may be replaced by $\mathbf{I}$ without volume loss for the category (subsection 2.1.4). If the modified simplexes after the vector replacement do not overlap with the other categories, C_I passes the OT.

The following two subsections give more details about cases (2b) and (2c).

Category expansion: creation of a new simplex

When $|\mathcal{A}_I(\mathbf{I})| = n$, the OT determines if category C_I can expand towards $\mathbf{I}$ creating a new simplex $S^* = \mathcal{A}_I \cup \{\mathbf{I}\}$ among them. The overlap among S^* and the other categories is tested by using the function O_{sc}– overlap between simplex and category– defined in equation M.7 (appendix M) as $O_{sc}(S^*, C_i) = 1$ if simplex S^* and category C_i overlap and $O_{sc}(S^*, C_i) = 0$ otherwise. If the former case holds for some C_i, S^* is removed and C_I does not pass the OT. Otherwise, C_I passes the OT. The category expansion increases the amount of information stored by the network– number of weight vectors and simplexes–, similarly to the creation of new categories in classical ART networks.

The new simplex could be very acute if its hyperplanes are nearly parallel. In this case, the simplex volume is rather small, and even when it covers regions populated by patterns with the right prediction, it does not significantly increase the category volume. Thus, categories with acute simplexes are not efficient to cover the input space. On the other hand, if the acute simplex covers a region with a different prediction, its small volume makes very improbable that future training patterns with that prediction fall inside this simplex, so the Category Adjustment step (subsection 2.1.3) can not remove it. Therefore, acute simplexes difficult the expansion of other categories and contribute to category proliferation. In these cases, PTAM does not create acute simplexes, and the winner category is not expanded.

The volume of a simplex in $\mathbb{R}^n$ can be computed by using the Cayley-Menger determinant [73], but its computational cost is very high. PTAM avoids the creation of acute simplexes assuming that they have some small angle between their hyperplanes. Specifically, PTAM creates a new simplex S_{ij} with weight vectors $\{\mathbf{w}_{ij1}, \ldots, \mathbf{w}_{ij(n+1)}\}$ only if the angle θ_{klm} between vectors $\mathbf{w}_{ijl} - \mathbf{w}_{ijk}$ and $\mathbf{w}_{ijm} - \mathbf{w}_{ijk}$ is above a threshold value θ_{min}. The angle θ_{klm} is calculated using the inner product of vectors $\mathbf{w}_{ijl} - \mathbf{w}_{ijk}$ and $\mathbf{w}_{ijm} - \mathbf{w}_{ijk}$:

$$\theta_{klm} = arcos\left[\frac{(\mathbf{w}_{ijl} - \mathbf{w}_{ijk})(\mathbf{w}_{ijm} - \mathbf{w}_{ijk})}{\|\mathbf{w}_{ijl} - \mathbf{w}_{ijk}\|\|\mathbf{w}_{ijm} - \mathbf{w}_{ijk}\|}\right], \quad k, l, m = 1, \ldots, n+1, k \neq l \neq m \tag{2.8}$$

The value of θ_{min} must be low in order to remove only the acute simplexes. From graphical considerations about acute simplexes in $\mathbb{R}^2$, which are valid for $\mathbb{R}^n$ we assumed that $\theta_{min} = 10$ degree is a reasonable threshold value.

Category expansion: weight vector replacement

If the number of connectable weight vectors in category C_I from $\mathbf{I}$ is higher than n ($|\mathcal{A}_I(\mathbf{I})| > n$), the OT determines if some weight vector in C_I can be replaced by $\mathbf{I}$ (fig. 2.4). The category C_I does not pass OT if it overlaps with other categories after the vector replacement. The replacement of weight vectors does not increase the amount of information learnt by the network– number of weight vectors and simplexes–.

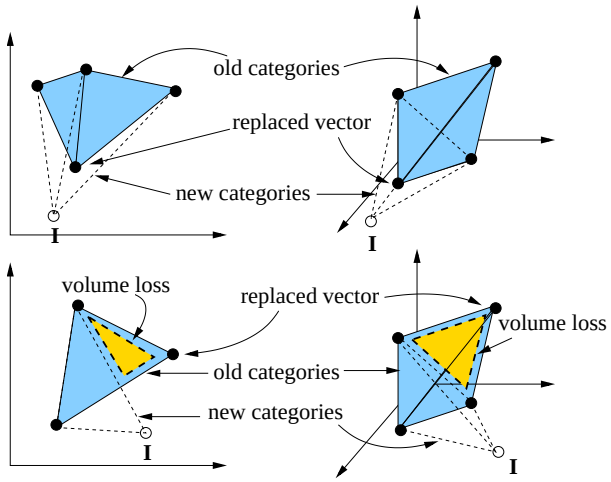


Figure 2.4: Examples of weight vector replacement for simplexes in $\mathbb{R}^2$ (left panels) and $\mathbb{R}^3$ (right panels), without volume loss (upper panels) and with volume loss (lower panels) of the simplex.

The fig. 2.4 shows that, depending on the relative position of a pattern and a simplex, the replacement of a weight vector may reduce (lower panels) or not reduce (upper panels) the volume of one or several simplexes. Fig. 2.5 (left panel) shows in

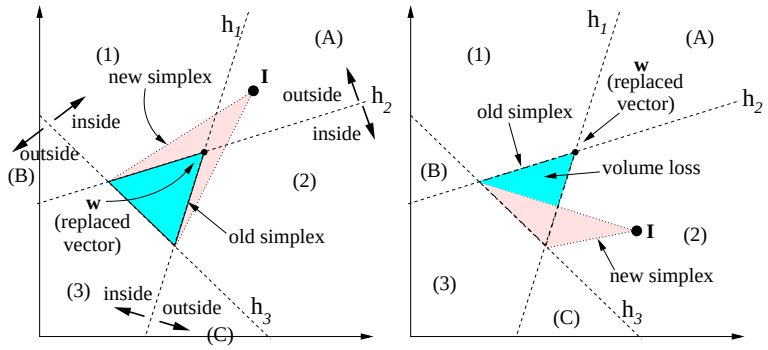


Figure 2.5: Example of vector replacement condition in $\mathbb{R}^2$. Patterns in regions A, B and C can replace weight vectors in the old simplex (left panel), but patterns in regions 1, 2 and 3 can not do it, because the replacement reduces the simplex volume (right panel).

$\mathbb{R}^2$ that a simplex loses volume when the input pattern falls in regions 1, 2 and 3, but not when it falls in regions A, B or C. If $\mathbf{I}$ falls in the outer (out of the simplex) side of the n hyperplanes which join at $\mathbf{w}$, $\mathbf{I}$ can replace $\mathbf{w}$ without simplex volume loss. Otherwise, the simplexes defined by $\mathbf{w}$ lose volume (right panel of fig. 2.5: $\mathbf{I}$ falls in the inner side of h_2 and h_3). Let $H_{Ij}(\mathbf{w}) = \{k = 1, \ldots, n+1 : \mathbf{w} \in h_{Ijk}\}$ be the set of hyperplanes in S_{Ij} which contain the vector $\mathbf{w}$. $\mathbf{I}$ can replace $\mathbf{w}$ in S_{Ij} if:

$$\prod_{k \in H_{Ij}(\mathbf{w})} \Phi[g_{Ijk}(\mathbf{I})] > 0 \tag{2.9}$$

The function $g_{ijk}(\mathbf{I})$ is defined in eq. 2.2, and $\Phi(x)$ is defined by $\Phi(x) = 1$ if $x \leq 0$ and $\Phi(x) = 0$ otherwise. If $\mathbf{I}$ falls in the inner side of some hyperplane $h_{Ijk} \in H_{Ij}(\mathbf{w})$, then $g_{Ijk}(\mathbf{I}) > 0$ and $\Phi[g_{Ijk}(\mathbf{I})] = 0$, so the condition 2.9 is not fulfilled. If $g_{Ijk}(\mathbf{I}) < 0, \forall k \in H_{Ij}(\mathbf{w})$, then $\mathbf{I}$ verifies condition 2.9. The nearest weight vector $\mathbf{w}^*$ from $\mathbf{I}$ in $\mathcal{A}_I(\mathbf{I})$ that satisfies condition 2.9 for each simplex $S_{Ij} \in C_I$ for which $\mathbf{w}^* \in S_{Ij}$ is the candidate to replacement. This vector $\mathbf{w}^*$ is replaced by $\mathbf{I}$ only if these simplexes do not overlap with the other categories after the vector replacement. This condition means that $O_{sc}(S_{Ij} \setminus \{\mathbf{w}^*\} \cup \{\mathbf{I}\}, C_i) = 0$ (function O_{sc} is defined in eq. M.7 in appendix M), $\forall S_{Ij} : \mathbf{w}^* \in S_{Ij}, \forall C_i, i \neq I$. If no weight vector in $\mathcal{A}_I(\mathbf{I})$ satisfies this condition, C_I does not pass the OT.

2.1.5 Resonance

If C_I passes both the Prediction and Overlap tests, then it has the right prediction and it expanded towards $\mathbf{I}$ without overlap, either replacing a new vector or creating a new simplex. However, in order to avoid simplex proliferation, the new simplex is only created if C_I is not the first winner in the competition, i.e., if $T_I^t(\mathbf{I}) \neq max\{T_i^t(\mathbf{I})\}_{i=1}^{N_c}$. Otherwise, the expansion of C_I is not necessary, because if the training pattern were presented again, it would resonate with C_I (which is the first winner). If all the training patterns with the same desired prediction were sucessively presented, PTAM would only create one single-simplex category with that prediction, which is undesirable. However, the training patterns are usually shuffled, so patterns with different predictions are mixed and this problem can not happen.

2.1.6 Creation of new categories

If no category passes both the Overlap and Prediction Tests, PTAM selects the n nearest connectable weight vectors from $\mathbf{I}$ belonging to categories with prediction P_d. The function O_{ls} (eq. M.5 in appendix M) determines if a weight vector is connectable from $\mathbf{I}$. If there are n or more connectable weight vectors from $\mathbf{I}$ with prediction P_d, PTAM creates a new simplex S^* with $\mathbf{I}$ and its n nearest connectable vectors. If S^* is not acute (subsection 2.1.4) and it does not overlap with other categories, PTAM creates a new category C_{N_c+1} with only one simplex (S^*) and associated prediction P_d ($N^s_{N_c+1} = 1$, $S_{(N_c+1)1} = S^*$ and $P(C_{N_c+1}) = P_d$). If S^* is acute, or if it overlaps with other simplexes, then PTAM removes S^* and creates a new single-vector category C_{N_c+1} with zero simplexes ($N^s_{N_c+1} = 0$), weight vector $\mathbf{w}_{N_c+1} = \mathbf{I}$ and prediction $P(C_{N_c+1}) = P_d$. If there are less than n connectable weight vectors with prediction P_d, $\mathbf{I}$ is also added as a new single-vector category.

PTAM needs at least $n+1$ training patterns with the same prediction to create a polytope category. Otherwise, PTAM creates one single-vector category for each training pattern. During the testing phase, the CCF of each single-vector category C_i ($N_i^s = 0$) depends on the Euclidean distance between its weight vector $\mathbf{w}_i$ and $\mathbf{I}$ (eq. 2.6 in subsection 2.1.1). Thus, $\mathbf{I}$ is assigned to its nearest single-vector category, i.e., PTAM behaves as a nearest neighbour classifier. In this case, the number of available training patterns is not enough to correctly learn the data set, which is

severely affected by the "curse of dimensionality" problem [11].

2.1.7 Polytope ARTMAP training phase

The training phase of PTAM can be summarized in the following steps:

1. Presentation of a new input pattern $\mathbf{I}$.

2. Calculation of Category Choice Functions: Calculate $T_i^t(\mathbf{I}), i = 1, \ldots, N_c$ (eqs. 2.1 and 2.6) for all the categories.

3. Competition: Select the category C_I with $r_I = 0$ and the highest CCF: $T_I^t(\mathbf{I}) = \max\{T_i^t(\mathbf{I})\}_{i=1}^{N_c}$. If all the categories are reset, go to step 8 (*Creation of a new category*).

4. Overlap Test: Expand C_I towards $\mathbf{I}$ either creating a new simplex S^*, only if S^* is not an acute simplex (subsection 2.1.4), or replacing some weight vector $\mathbf{w}^*$ in C_I by $\mathbf{I}$. If C_I overlaps with other categories after expansion, reset C_I ($r_I = 1$), return it to its previous state and go to step 3 (*Competition*).

5. Prediction Test: If $P(C_I) \neq P_d$ (the associated prediction of C_I is different from the desired prediction), reset C_I ($r_I = 1$). Otherwise, go to step 7 (*Resonance*).

6. Category Adjustment: If $T_I^t(\mathbf{I}) = 1$ ($\mathbf{I}$ falls inside the CRR of category C_I), then remove simplex S_{IJ} for which $T_{IJ}(\mathbf{I}) = 1$. Include in the set of input patterns the vectors $\mathbf{w}_{IJ1}, \ldots, \mathbf{w}_{IJ(n+1)}$ of simplex S_{IJ} which do not belong to any other simplex in C_I. Go to step 3 (*Competition*).

7. Resonance: Assign the input pattern to C_I. If C_I expanded towards $\mathbf{I}$ creating a new simplex S^*, and $T_I^t(\mathbf{I}) = \max\{T_i^t(\mathbf{I})\}_{i=1}^{N_c}$ (C_I is the first winner), remove S^*. Go to step 1.

8. Creation of a new category: Create a new simplex S^* with $\mathbf{I}$ and its n nearest connectable weight vectors with prediction P_d. If S^* is not acute and it does not overlap with other categories, create a new single-simplex category C_{N_c+1} only with the simplex S^*. If there are less than n connectable weight vectors from $\mathbf{I}$, or if S^* is an acute simplex, or if S^* overlaps with other categories, remove S^* and create a new single-vector category C_{N_c+1} with $\mathbf{w}_{N_c+1} = \mathbf{I}$. Go to step 1.

2.2 Processing phase

The output of PTAM for each input pattern during the processing phase is the associated prediction of the category C_I with the highest $T_i^p(\mathbf{I})$ (eq. 2.10). The CCF of a single-vector category C_i ($N_i^s = 0$) is calculated using the Euclidean distance between its weight vector $\mathbf{w}_i$ and $\mathbf{I}$:

$$T_i^p(\mathbf{I}) = \begin{cases} \max\limits_{j=1,\ldots,N_i^s} \{T_{ij}(\mathbf{I})\} & N_i^s > 0 \\ 1 - \dfrac{\|\mathbf{I} - \mathbf{w}_i\|}{\sqrt{n}} & N_i^s = 0 \end{cases} \tag{2.10}$$

Where $T_i^p(\mathbf{I})$ was defined in eq. 2.6. The competition selects the category C_I with the highest CCF.

$$T_I^p(\mathbf{I}) = \max_{i=1,\ldots,N_c} \{T_i^p(\mathbf{I})\} \tag{2.11}$$

The output of PTAM is the prediction $P(C_I)$ associated to the winner category C_I.

2.3 Computational complexity

The computational cost of each iteration (processing of a training pattern) in PTAM is the following. The cost of functions $\phi_{ijk}(\mathbf{I})$ and $g_{ijk}(\mathbf{I})$ (eqs. 2.3 and 2.2 respectively) is $\mathcal{O}(n^3)$, because they use n-order determinants [95]. In order to evaluate if the input pattern falls inside the simplex, $n+1$ values of $g_{ijk}(\mathbf{I})$ must be calculated, which is $\mathcal{O}(n^4)$. If the pattern falls outside the simplex, but inside the hypersphere (eq. 2.4), $\|\mathbf{w}_{ijk}^*\|$ must be calculated for the $n+1$ hyperplanes in S_{ij}. Since n determinants D_{ijkl} (eq. L.2 in appendix L) must be calculated, $\|\mathbf{w}_{ijk}^*\|$ is $\mathcal{O}(n^4)$, so $d(\mathbf{I}, S_{ij})$ (eq. 2.4) is $\mathcal{O}(n^5)$. If the pattern falls inside the hypersphere, $d(\mathbf{I}, S_{ij})$ (eq. 2.5) is $\mathcal{O}(n^2)$. Therefore, $T_{ij}(\mathbf{I})$ and $T_i^t(\mathbf{I})$, or $T_i^p(\mathbf{I})$, are $\mathcal{O}(n^5)$. Since the system of n linear equations in I_{sh} (eq. M.4) is $\mathcal{O}(n^2)$, the function O_{ls} (eq. M.5) is $\mathcal{O}(n^3)$, because it requires to calculate I_{sh} for the $n+1$ hyperplanes of a simplex. The calculation of set $\mathcal{A}_I(\mathbf{I})$ (section 2.1.4) is $\mathcal{O}(n^4)$, because $(n+N_I^s)\sum_{i=1}^{N_c} N_i^s$ values

of O_{ls} must be calculated[4]. The function O_{ss} (eq. M.6) for the overlap test is $\mathcal{O}(n^5)$, because $(n+1)^2$ values of O_{ls} must be calculated, so the creation of a new simplex is $\mathcal{O}(n^5)$. The removal of acute simplexes is $\mathcal{O}(n^4)$ (eq. 2.8), because it requires to calculate n^3 angles, which are $\mathcal{O}(n)$ each one. The vector replacement step is $\mathcal{O}(n^4)$, because $n + N_I^s$ values of $g_{Ijk}(\mathbf{I})$ (which is $\mathcal{O}(n^3)$) must be calculated, but the overlap test for the modified simplex is also $\mathcal{O}(n^5)$. The costs of the prediction test, category adjustment and resonance steps do not depend on n. The creation of a new category requires $\sum_{i=1}^{N_c}(n + N_i^s)$ calculations of O_{ls}, which is $\mathcal{O}(n^4)$, in order to select the connectable vectors from $\mathbf{I}$, and the overlap test for the new simplex, which is $\mathcal{O}(n^5)$.

Overall, the cost of PTAM for each iteration is $\mathcal{O}(n^5)$. While this high cost difficults the use of PTAM with high-dimensional data sets, for n low its speed is comparable to the other ART networks. Specifically, the current implementation of PTAM takes about 40 s. for the CIS data set (10,000 training and 10,000 test patterns) on a general purpose PC (Pentium IV with 512MB RAM), and it runs much faster than DAM (560 s.), three times slower than GAM and EAM (15 s.) and eight times slower than FAM and FasArt (5 s.).

2.4 Experimental setting

PTAM is tested and compared with other ART and non-ART classifiers on a series of benchmark data sets (table 2.2). Several data sets are two-dimensional, in order to show graphically the polytope categories created by PTAM. Data sets with noise and category overlap have been also used in order to test the behavior of PTAM in these situations. As well, PTAM was tested with real, high-dimensional data sets.

2.4.1 Two-Dimensional data sets

CIS, Chess (also known as "Generalized XOR") and T3-T7 (sets 1-7 in table 2.2) are two-dimensional artificial data sets (fig. 2.6), with rectangular (Chess and T4) and circular (CIS, T3 and T5-T7) geometries. The CIS [143] data set was often

[4]Each polytope category C_i with $N_i^s > 0$ is defined by $(n+1) + (N_i^s - 1) = n + N_i^s$ weight vectors.

No.	Name	Predictions	Observations
Two-dimensional data sets			
1	CIS	2	#patterns/prediction $\propto$ area
2	Chess	2	50% patterns/prediction
3	T3	5	20% patterns/prediction
4	T4	5	20% patterns/prediction
5	T5	6	#patterns/prediction $\propto$ area
6	T6	3	#patterns/prediction $\propto$ area
7	T7	3	50%-30%-20% patterns
Two-dimensional data sets with noise			
8	CIS-noise	2	noise level 0.01:0.01:0.05
9	T5-noise	6	noise level 0.01:0.01:0.03
Two-dimensional data sets with category overlap			
10	4G1	4	$d = 0.134$ (low overlap)
11	4G2	4	$d = 0.120$ (middle overlap)
12	4G3	4	$d = 0.084$ (high overlap)
2D data set with irregular geometry			
13	Form	2	50%-50% patterns
Real multi-dimensional data sets			
14	PID	2	768 patterns, 8 inputs
15	Abalone	3	4177 patterns, 8 inputs

Table 2.2: List of data sets used in the experimental work. See text for details.

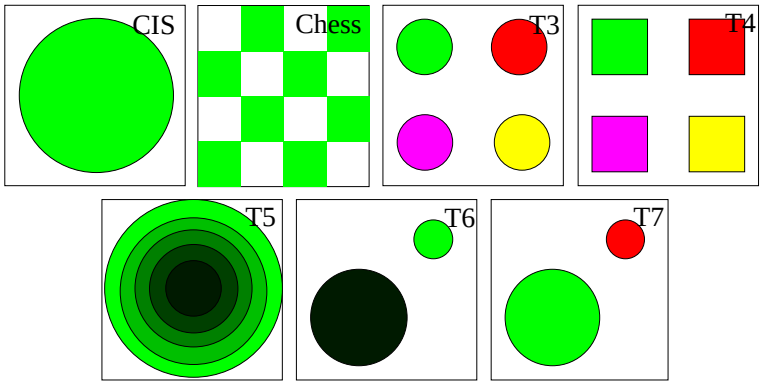


Figure 2.6: Two-dimensional data sets CIS, Chess, T3, T4, T5, T6 and T7 used in the experimental work. In T6 the number of patterns of each category is proportional to its size (75.8% outside the circles, 19.7% in the big circle and 4.5% in the small circle). In T7 the populations are 50%-30%-20% [120].

used in the ART literature. The results achieved with CIS are averaged over seven circle sizes ranging from 10% to 70% of the unit square area, in the same way as the previous chapter. Data sets T3-T7 were used in [120] as benchmarks for FasArt, FAM and DAM. Chess is another benchmark data set frequently used in the literature [128, 66]. The number of patterns for each prediction is proportional

to its area for data sets CIS, T5 and T6 (table 2.2), while predictions in the data sets Chess, T3 and T4 have the same number of patterns.

2.4.2 Noisy two-dimensional data sets

Data sets CIS-noise and T5-noise (sets 8-9 in table 2.2) are generated by adding Gaussian noise to the CIS and T5 data sets respectively. Several noise levels were used, given by the standard deviation of the Gaussian distribution, with values in the range 0.01:0.01:0.05 and 0.01:0.01:0.03 for CIS-noise and T5-noise respectively.

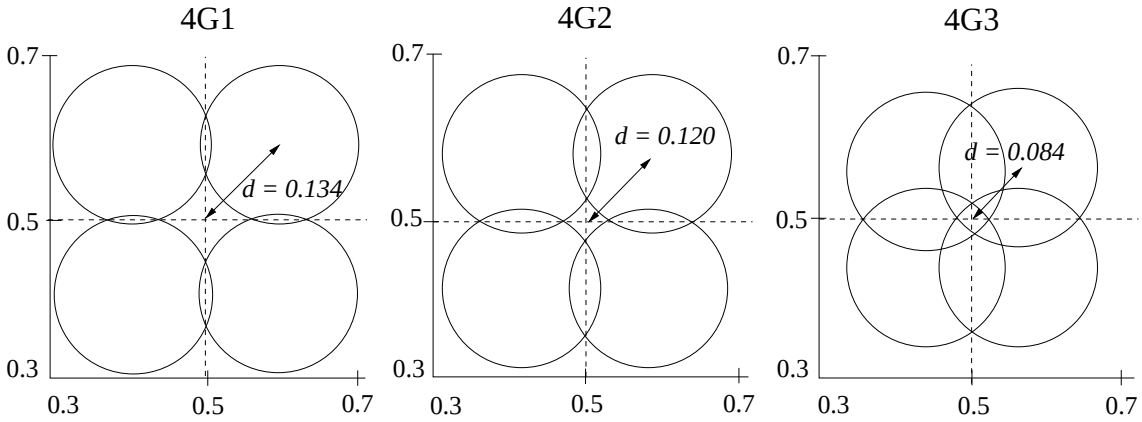


Figure 2.7: Circles surrounding the Gaussian distributions, with radius 2σ ($\sigma = 0.05$ is the Gaussian spread) for the three overlap levels. The distance d between the center of each circle and the unit square center (0.5, 0.5) are $d = 0.134$ in 4G1, $d = 0.120$ in 4G2 and $d = 0.084$ in 4G3.

2.4.3 Two-dimensional data sets with prediction overlap

The data sets 4G1, 4G2 and 4G3 (labeled 10-12 in table 2.2) are used in the literature [5] to evaluate classification algorithms when predictions overlap. Each data set has 4 output predictions, given by Gaussian probability distributions with standard deviation $\sigma = 0.05$. The overlap between them is decreasing with the distance d between its centers and the center of the unit square (fig. 2.7).

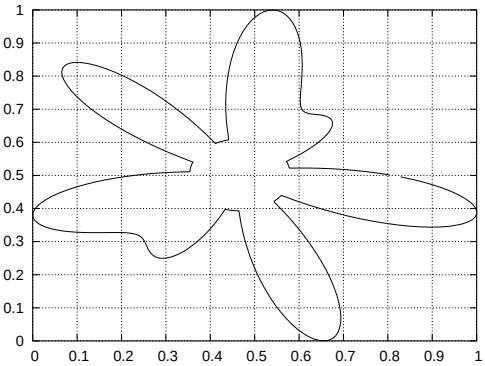


Figure 2.8: Border among the two predictions in data set Form.

2.4.4 Data set with irregular geometry

Since the previous data sets have circular or rectangular geometries, we generated another two-dimensional data set, called Form (set 13 in table 2.2) with irregular geometry and, therefore, not specially suited to circular or rectangular category geometries. Form has two predictions associated to the input patterns falling inside and outside of the curve in fig. 2.8, which is an example of a mask used for the analysis of irregular textured regions in computer vision [57]. Specifically, this curve is described by the following parametric equations:

$$x = 0.5 + r(\theta)cos\theta; \quad y = 0.5 + r(\theta)sin\theta; \quad 0 \leq \theta < 2\pi \tag{2.12}$$

$$r(\theta) = max\left\{r_{min}, r_0 + \sum_{n=0}^{N} r_0 f(n) cos[(2n+1)\theta + \psi(n)]\right\} \tag{2.13}$$

Here, $0 \leq r_0 \leq 0.5$ is the mean radius of the curve, $0 \leq f(n) \leq 1$ and $0 \leq \psi(n) \leq 2\pi$ are pseudo-random numbers. The curve is a circle for $N = 0$ and its irregularity increases with N. We used $r_0 = 0.5$ and $N = 3$. In order to make easier the input pattern labeling, a minimum radius $r_{min} = 0.08$ was required. The coordinates (x, y) where scaled to have minimum 0 and maximum 1.

2.4.5 Real, high-dimensional data sets

Pima Indians Diabetes (PID) and Abalone (both available from the UCI Machine Learning Repository [12]), are real 8-dimensional data sets (labeled 14-15

Table 2.3: Parameter values of the different algorithms.

FAM	$\alpha = 0.01, \beta = 1$ (learning rate), $\rho_{ab} = 1$, Weber Law and Choice-By-Difference
GAM	$\gamma = 0.2$, $\gamma = 0.8$ (PID and Abalone)
DAM	$\alpha = 0.01, \beta = 1$ (learning rate), $p = 10$ (contrast enhancement exponent)
EAM	$\mu = 0.5, D = \sqrt{n}/\mu, \beta = 1, \alpha = 0.1$
FasArt	$\bar{\rho}_b = 1, \beta = 1, \gamma_a = \gamma_b = 10$, $\gamma_a = \gamma_b = 1$ (PID and Abalone)
Common	$\bar{\rho} = 0.00 : 0.02 : 0.98$; 1-5 training epochs
SVM	$C = 0.01, 0.1, 1, 100, 1000$, Gaussian kernel with spread $\in \{0.1 : 0.1 : 1,$ $2.5, 5, 7.5, 10, 12, 15, 17, 20\}$

in table 2.2) with 2 and 3 output predictions respectively, and they are reported as relatively difficult in the literature. FAM achieved 38% error [128] and Safe-μARTMAP [65] 32% error[5] in PID. In Abalone, 36% error was reported with MLP [39] and 50% with FAM (367 categories) [5]. This data set has 8-dimensional input patterns: 1 input is discrete (3 values) and 7 inputs are continuous. The output is an integer number in the range $\{1 \ldots 29\}$. For classification tasks, the input patterns must be grouped in 3 output predictions, associated to outputs 1-8, 9-10 and 11-29. The input patterns in both data sets are pre-processed by scaling each component in the range $[0, 1]$ before feeding to the ART networks. The pre-processing phase associated to the SVM is the usual mean removal and covariance equalization.

2.5 Results

We developed experiments comparing PTAM with the most popular ART networks on these data sets, including Fuzzy ARTMAP [28], Gaussian ARTMAP [144], Distributed ARTMAP [36], Ellipsoid ARTMAP [3] and FasArt [17]. We also report the results achieved by SVM [136], which is a reference for classification tasks. The parameter values of the different algorithms are reported in table 2.3. FAM used the Weber Law and Choice-By-Difference choice functions: the results were similar,

[5]In this paper, the authors use 576 training and 192 testing patterns. We use 192 training, 384 validation and 192 testing patterns for all the neural networks. We are only interested in the comparison between them, so the small size of the training set does not reduce the significance of our experiments.

	FAM	GAM	DAM	EAM	FasArt	SVM	
Data set	$\bar{\rho}$	ρ	$\bar{\rho}$	ρ	ρ	C	σ
CIS	0.88	0.04	0.9	0.84	0.68	100	0.1
Chess	0.46	0.06	0.7	0.94	0.92	1000	0.1
T3	0.2	0.10	0.6	0.94	0.9	1000	0.1
T4	0.3	0.12	0.76	0.94	0.82	1000	0.1
T5	0.92	0.02	0.94	0.96	0.94	100	0.1
T6	0.54	0.04	0.88	0.92	0.84	1000	0.1
T7	0.48	0.1	0.78	0.82	0.86	1000	0.1
CIS-n 0.01	0.9	0.1	0.9	0.94	0.8	1000	0.9
0.02	0.9	0.06	0.9	0.92	0.88	100	0.9
0.03	0.92	0.02	0.9	0.9	0.9	10	0.7
0.04	0.9	0.02	0.9	0.9	0.92	10	0.9
0.05	0.92	0.02	0.9	0.9	0.92	1	0.5
T5-n 0.01	0.9	0.36	0.95	0.96	0.88	1000	0.7
0.02	0.88	0.02	0.95	0.02	0.88	10	0.4
0.03	0.92	0.02	0.95	0.02	0.32	10	0.2
Form	0.36	0.12	0.88	–	–	–	–
	0.92	–	0.94	–	–	–	–
4G1	0.9	0.02	0.75	0.96	0.54	10	0.9
4G2	0.94	0.02	0.65	0.8	0.3	10	0.9
4G3	0.94	0.02	0.1	0.9	0.02	1	0.7
PID	0.8	0.2	0.6	0.6	0.6	1000	10
Abalone	0.95	0.2	0.8	0.4	0.0	1000	2.5

Table 2.4: Selected values of the tuning parameters (ρ in the ART networks, C and spread (σ) in the SVM) on the different validation sets (see text). CIS-n and T5-n stand for CIS-noise and T5-noise respectively.

but slightly better using the Weber Law. For the ART classifiers, including PTAM, 5 training epochs were run on data sets 1-7, and 1 training epoch was run on data sets 8-14, in order to reduce the computational cost of the simulations. FasArt used $\gamma_a = \gamma_b = 1$ in data sets PID and Abalone, because using $\gamma_a = 10$ the categories are too narrow and they do not cover the whole input space. We used the SVM implementation provided by the Torch Machine Learning library [40] with polynomial, sigmoid, dot-product and Gaussian kernels, which usually provided the best results. The value with the best average performance over the validation sets was selected for each data set (table 2.4).

Some authors [128, 65] report the results achieved by the ART networks with zero baseline vigilance ($\bar{\rho} = 0$). However, the error and number of categories achieved by an ART network strongly depend on the vigilance value, as figs. 2.9 and 2.10 below show. We used cross-validation in order to determine the best $\bar{\rho}$ for each data set.

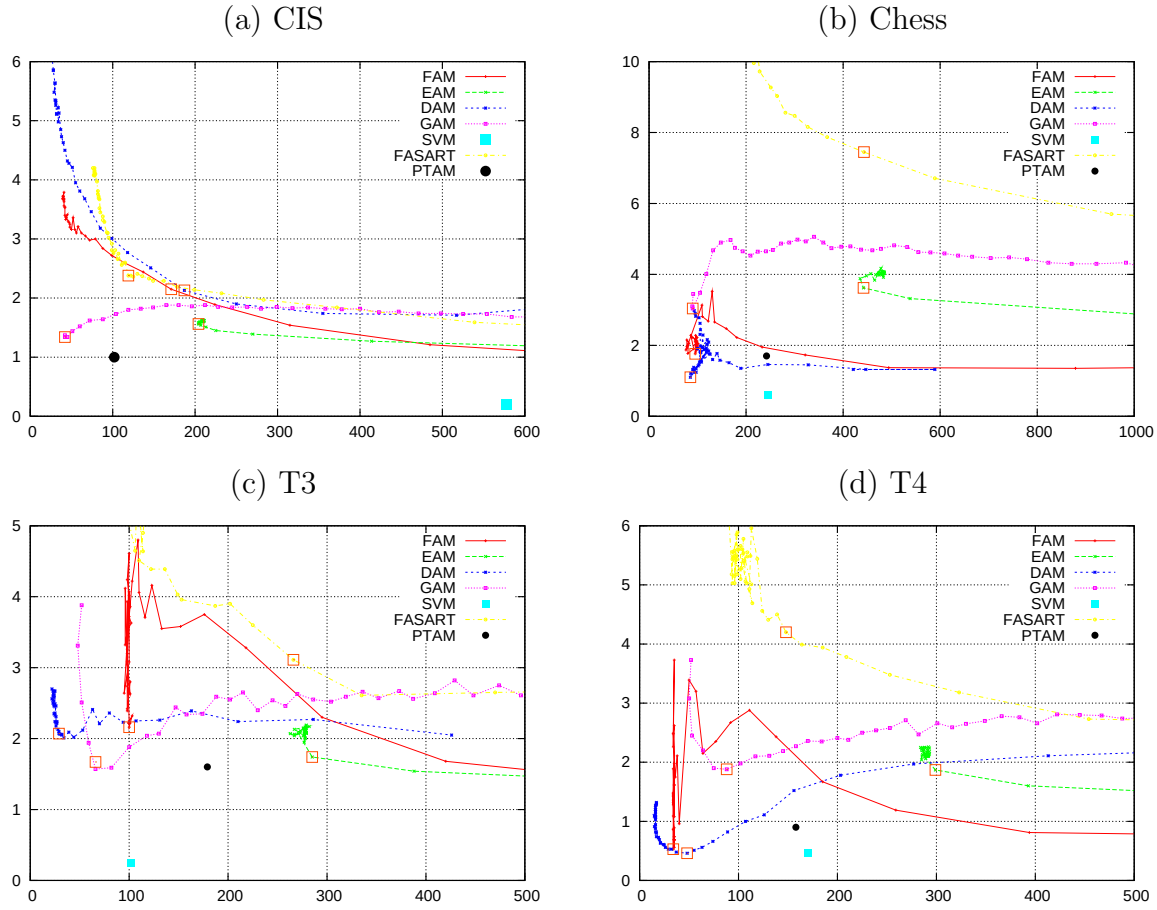


Figure 2.9: Error (in %) against the number of categories on the validation sets in CIS (a), Chess (b), T3 (c) and T4 (d) varying vigilance. The selected operating point, with the best trade-off between the error and the number of categories, is marked with an empty square.

For data sets 1-12, 20 triplets, each one composed of one training, one validation and one test set, were created, containing 10,000 randomly generated patterns each set. For each ART classifier (FAM, GAM, DAM, EAM and FasArt), we tried $\bar{\rho}$ values in the range 0.00:0.02:0.98. For each vigilance value, 20 versions of the classifiers were trained, one for each training set, and they were tested on its corresponding validation set. Figs. 2.9 and 2.10 below represent the percentage of error rate against the number of categories #C (number of weight vectors #W for PTAM or support vectors #SV for SVM), with the different vigilance values and for each data set, averaged over the 20 validation sets. The vigilance value ρ which provides the best trade-off between the error and #C on the validation set (operating point, marked

with an empty square in figs. 2.9 and 2.10) was selected for the test phase. For example, the FAM operating point for data set CIS (panel (a)) has 2.15% error, 171 categories and $\bar{\rho} = 0.88$. The best results were usually achieved with high vigilance, as reported in table 2.4. GAM is a noticeable exception, because its error and #C increase with the vigilance, so its selected value is often $\rho \simeq 0$.

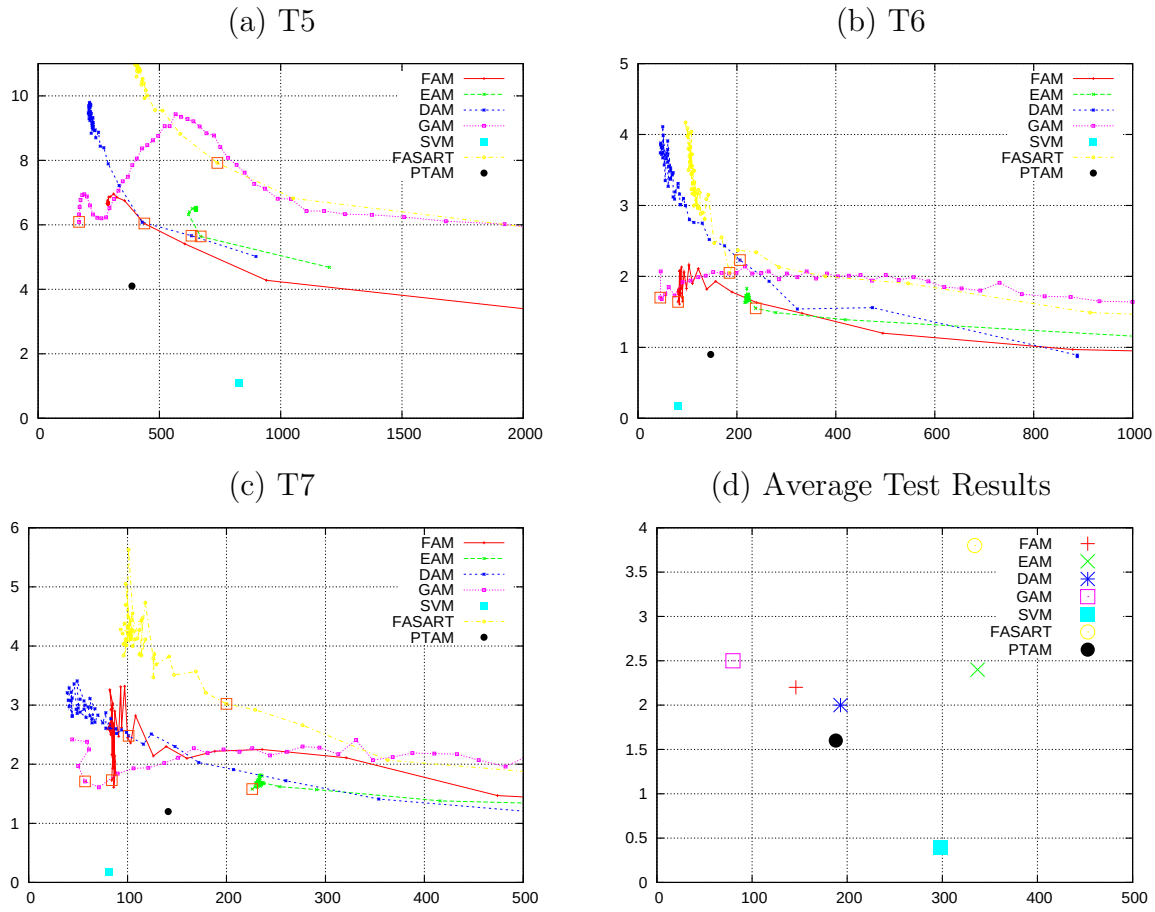


Figure 2.10: Error (in %) against the number of categories in T5 (a), T6 (b) and T7 (c) varying vigilance. The selected operating point is marked with an empty square. Panel (d) shows the average test results of each classifier.

The 20 classifiers trained with the selected vigilance value were tested on the 20 test sets. The average error rate and #C over the test sets are reported in table 2.5, and represented in the panel (d) of fig. 2.10. The same experimental methodology is used with the noisy two-dimensional data sets CIS-noise and T5-noise, and with the overlap data sets 4G1, 4G2 and 4G3.

Panels (a) and (c) in fig. 2.11 show the Category Representation Regions (CRRs)

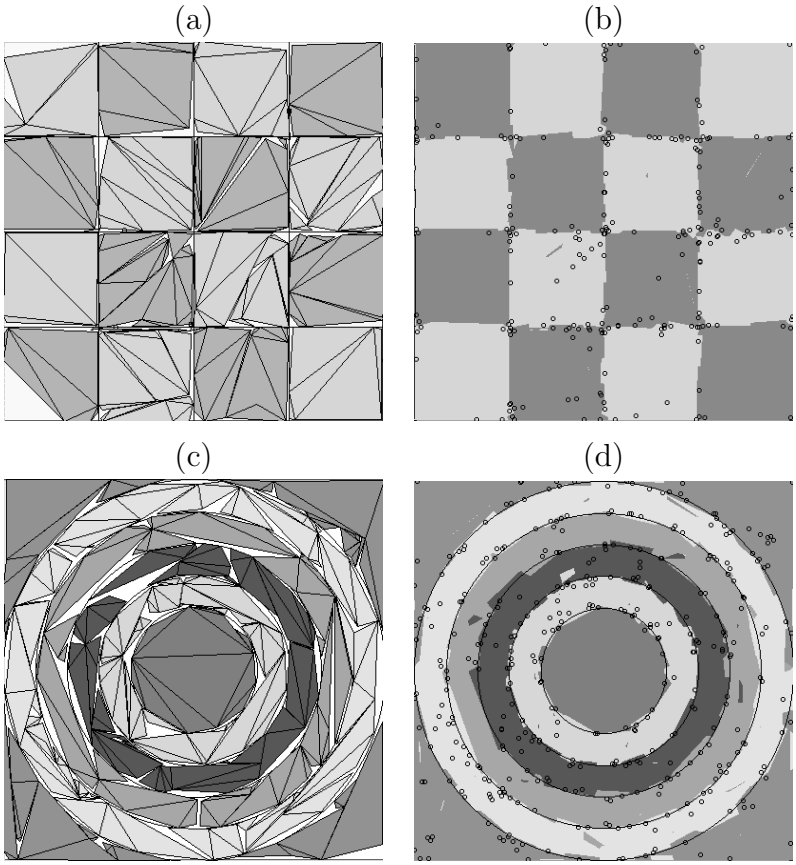


Figure 2.11: Examples of categories and classification regions learnt by PTAM for data sets Chess (panels (a) and (b) respectively) and T5 (panels (c) and (d)).

created by PTAM for data sets Chess and T5 respectively, which are examples of rectangular and circular geometries respectively. Panels (b) and (d) show the weight vectors and the borders among predictions created by PTAM, which achieve quite accurately the desired borders. The category borders (line segments in $\mathbb{R}^2$) in panels (a) and (c) compose a piece-wise linear approximation to the borders among the output predictions. The lower left corner of the panel (a) in fig. 2.11 shows that sometimes polytope categories may not cover the whole input space. However, the regions not covered are assigned to the right output prediction, as panel (b) shows.

2.5.1 Two-dimensional data sets

Table 2.5: Test error (ϵ, in %) and #C (number of weight vectors #W for PTAM and number of support vectors #SV for SVM) achieved by each classifier in two-dimensional data sets. The best ϵ and #C achieved by an ART and non-ART classifier are in bold.

	PTAM		FAM		GAM		DAM		EAM		FasArt		SVM	
	ϵ	#W	ϵ	#C	ϵ	#C	ϵ	#C	ϵ	#C	ϵ	#C	ϵ	#SV
CIS	**1.0**	102	1.3	188	1.3	**43**	1.3	238	1.3	195	1.8	350	**0.2**	578
Chess	1.7	242	1.8	95	3.0	90	**1.1**	**89**	3.4	443	4.7	443	**0.6**	245
T3	**1.6**	179	2.2	100	**1.6**	67	2.1	**42**	1.7	285	3.1	266	**0.2**	102
T4	1.0	158	0.5	**34**	1.9	89	**0.4**	48	1.9	300	4.1	149	0.5	170
T5	**4.1**	387	5.8	439	6.1	**169**	5.6	630	5.6	670	7.8	739	**1.1**	827
T6	**0.9**	147	1.6	81	1.7	**45**	1.6	193	1.5	239	2.1	185	**0.2**	81
T7	**1.2**	141	1.9	85	1.8	**57**	1.9	113	1.7	226	3.1	207	**0.2**	81
Avg.	**1.6**	188	2.2	146	2.5	**80**	2.0	193	2.4	337	3.8	334	**0.4**	298
Std.	**1.1**	100	1.7	137	1.7	**44**	1.7	206	1.5	167	2.0	205	**0.3**	291

The table 2.5 reports the test results of the seven classifiers for two-dimensional data sets 1-7. SVM achieves the lowest error for all the data sets except in T4. However, SVM generates in average more support vectors than categories in PTAM, FAM, GAM and DAM. PTAM achieves the lowest error among the ART networks in five of seven data sets (CIS, T3, T5, T6 and T7). Its error (4.1%) is clearly lower than the other networks in data set T5 (above 5.5%), which is the most complex one. PTAM outperforms the circular ART networks (GAM and EAM) both in rectangular and circular data sets (GAM achieves the equal error in T3). Also, PTAM achieves the second lowest error in the rectangular data set Chess, outperforming FAM. PTAM creates an intermediate number of vectors, less than DAM, EAM and FasArt, but more than GAM and FAM.

DAM achieves the lowest error in the rectangular data sets Chess and T4. FAM achieves an error very similar to, although slightly higher than, DAM in all the data sets. Both networks achieve errors above PTAM, GAM and EAM in circular data set T3. FAM creates less categories than DAM, except in Chess and T3. GAM and EAM achieve similar errors, which are high compared to PTAM, DAM and FAM, specially in the rectangular data sets. However, EAM creates the highest number of categories, and GAM the lowest. Finally, FasArt achieves the highest error both in circular and rectangular data sets, with a high number of categories.

	FAM		DAM		FasArt	
	ϵ	#C	ϵ	#C	ϵ	#C
CIS	6.3	23.2	11.1	12.3	3.7	63.4
T3	11.5	53.1	12.1	33.4	7.3	122.3
T4	3.8	31.9	4.3	30.7	8.2	139.2
T5	15.5	124.8	19.1	125.4	4.0	278.7
T6	5.2	21.2	21.1	8.6	3.6	62.2
T7	5.4	20.7	4.7	12.9	3.7	57.7

Table 2.6: Error (ϵ, in %) and #C achieved in [120] by FAM, DAM and FasArt in CIS and T3-T7.

Results in data sets CIS and T3-T7 are different from [120] (table 2.6), where the reported errors are higher, and the number of categories is lower, than in our experiments. Some possible reasons are the following: i) the differences in the number of training and test patterns (2,000 patterns in [120]) and in the category populations for some data sets; ii) we optimize the vigilance value in the ART networks ($\bar{\rho} \neq 0$ in our simulations except for GAM) while FAM, DAM and FasArt use $\bar{\rho} = 0$ in [120]. Hence, we achieve less error and more categories with FasArt, FAM and DAM. This may be the cause of the good results achieved by FAM and DAM in our simulations, compared to FasArt, with respect to the cited paper.

	PTAM			FAM		DAM	
Data set	N_s	N_w	N_p/n	N_c	N_p/n	N_c	N_p/n
CIS	78	102	**258**	188	**376**	238	**1071**
Chess	183	242	**608**	95	**190**	89	**401**
T3	134	179	**447**	100	**200**	42	**189**
T4	122	158	**402**	34	**68**	48	**216**
T5	282	387	**951**	439	**878**	630	**2835**
T6	114	147	**375**	81	**162**	193	**869**
T7	101	141	**343**	85	**170**	113	**509**
Average	145	194	**483**	146	**292**	193	**870**
Std.	69	95	233	137	274	206	926

Table 2.7: Number of categories (N_c), simplexes (N_s) and parameters N_p divided by the dimension n of the input space ($n = 2$ for all the data sets in the table) stored by PTAM, FAM and DAM.

It is interesting to compare the number of parameters (N_p) stored by the different networks, because it is related with its complexity. In the case of PTAM, this number depends on the number of weight vectors and simplexes (see appendix N). Since each category in PTAM requires several vectors, it usually stores more parameters

than the ART networks. This is expectable, since the polytope categories of PTAM are more flexible than hyperbox categories. The table 2.7 reports N_p, divided by n (dimension of the input space), for PTAM, FAM and DAM, in the seven two-dimensional data sets. In average, PTAM stores 65% more parameters than FAM, for which $N_p = 2nN_c$ (N_c is the number of categories). On the other hand, the number of parameters of DAM is given by $N_p = (4n+1)N_c$, and it stores 80% more parameters than PTAM (DAM stores less parameters than PTAM only in Chess, T3 and T4).

Table 2.8: Test results for 2-D data sets with noise. The best error (ϵ, in %) and #C achieved by an ART and non-ART classifier are in bold.

		CIS-noise					T5-noise		
Method		0.01	0.02	0.03	0.04	0.05	0.01	0.02	0.03
Bayes	ϵ	1.7	3.3	5.0	6.5	8.0	8.2	16.1	23.8
PTAM	ϵ	9.2	14.1	19.2	24.7	28.3	21.9	36.6	46.8
	#W	178	329	468	623	733	759	1421	1890
FAM	ϵ	3.7	6.7	9.6	13.5	16.2	15.0	26.4	35.1
	#C	240	271	401	346	483	421	554	1014
GAM	ϵ	**3.3**	**4.9**	**6.8**	**8.5**	**10.3**	**11.4**	**17.7**	**24.4**
	#C	**52**	**53**	**58**	**67**	**76**	**272**	**265**	**362**
DAM	ϵ	3.2	5.8	7.4	10.6	12.1	11.5	22.3	31.9
	#C	230	278	298	350	369	842	1090	1432
EAM	ϵ	3.7	6.7	10.0	13.4	16.9	14.1	26.5	35.8
	#C	241	242	277	343	400	746	851	1344
FasArt	ϵ	5.1	7.0	8.6	10.0	11.9	12.5	18.6	25.4
	#C	207	400	555	757	870	819	1433	2002
SVM	ϵ	9.4	11.2	13.1	15.4	16.7	**9.5**	**16.5**	**23.8**
	#SV	439	873	1315	1728	2105	2039	4094	5707

2.5.2 Data sets with noise

All the classifiers increase their error and #C with the noise level, as reported in table 2.8 and fig. 2.12 in data sets CIS-noise and T5-noise. The table shows also the Bayes error [142] in both data sets. GAM achieves the lowest error (10% with the highest noise level) in CIS-noise for all the noise levels. In T5-noise, GAM achieves slightly more error than SVM (24%, equal to the Bayes error). However, SVM uses

a huge number of support vectors, while the #C of GAM increases very slowly with the noise level. Remarkably, GAM creates less categories in CIS-noise with noise level 0.05 than the other classifiers without noise. FAM, DAM and EAM report similar number of categories for each noise level, although DAM achieves slightly less error than FAM and EAM. FasArt achieves similar error to DAM with high noise level, but with more categories. PTAM achieves higher error than the other ART networks for the different noise levels, and it creates rather high number of categories, but lower than FasArt and SVM.

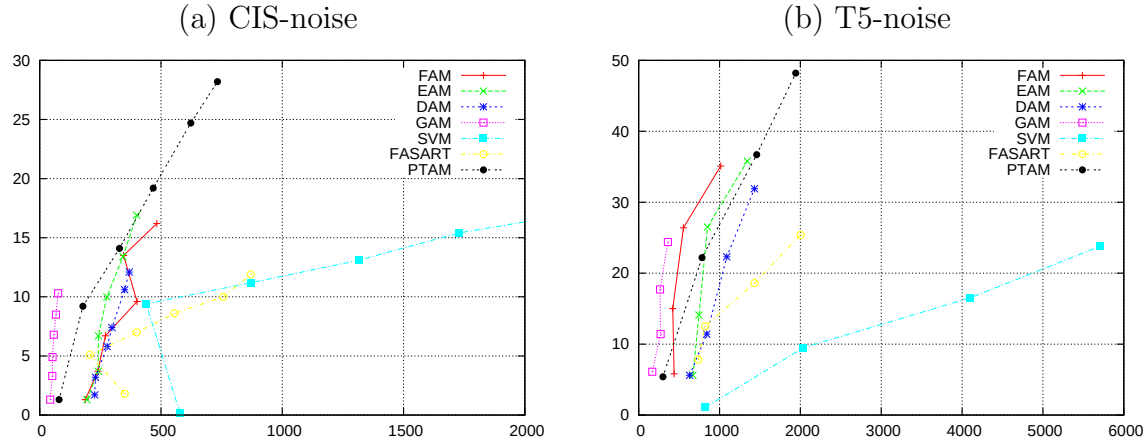


Figure 2.12: Average error (in %) against the number of categories in CIS-noise (a) and T5-noise (b). Each point corresponds to a different noise level (0.00:0.01:0.05 in CIS-noise and 0.00:0.01:0.03 in T5-noise).

2.5.3 Data sets with prediction overlap

In the overlap category data sets 4G1, 4G2 and 4G3, PTAM also reports poor performance (table 2.9). The first row in this table reports the Bayes errors for each data set. GAM shows the best global behavior, because it achieves the Bayes error on each data set with a very low number of categories. FasArt and SVM also achieve the Bayes error, but they are more complex[6]– specially SVM–. DAM achieves more error than GAM with a similar complexity. EAM achieves more error than DAM, with much more categories. FAM achieves clearly more error than EAM and DAM,

[6]The complexity of the classifier is related with its number of parameters, which depends on the number of categories in the ART networks or support vectors in SVM.

Table 2.9: Test error (ϵ, in %) and #C in data sets with prediction overlap. The best ϵ and #C achieved by an ART and non-ART classifier are in bold.

	4G1		4G2		4G3	
	ϵ	#C	ϵ	#C	ϵ	#C
Bayes	5.7	–	8.7	–	21.8	–
PTAM	25.6	632	34.0	905	52.5	1869
FAM	15.0	212	19.3	372	39.2	801
GAM	**5.8**	**41**	**8.8**	**51**	22.3	**123**
DAM	10.3	52	14.5	70	32.5	161
EAM	11.9	425	17.2	582	35.2	1937
FasArt	**5.8**	703	**8.8**	1000	**21.9**	2250
SVM	**5.8**	1551	**8.7**	2352	**21.9**	5173

with an intermediate number of categories. PTAM achieves higher error than the other classifiers, and it only creates less vectors than FasArt and SVM.

2.5.4 Irregular geometry data set

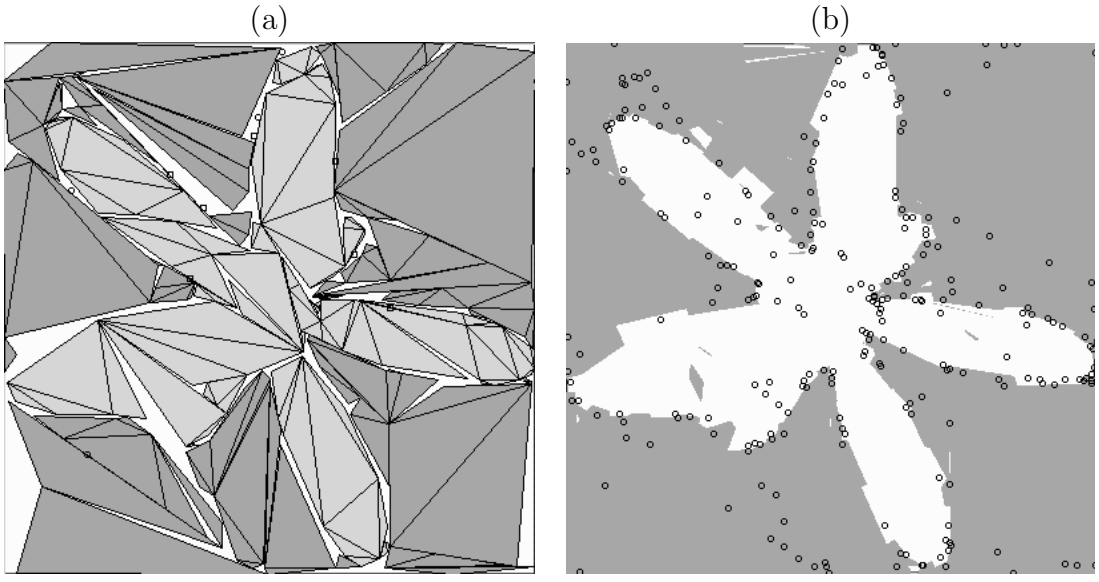


Figure 2.13: (a) Example of categories created (a) and classification regions and weight vectors (b) created by PTAM in the data set Form.

We also compared PTAM with the best rectangular ART networks (DAM and FAM), and with the best circular ART network (GAM), in the irregular data set Form. The fig. 2.13 shows an example of the polytope categories (panel (a)) and the borders among predictions (panel (b)) created by PTAM for this data set. FAM, DAM and GAM were trained and tested using the same cross validation methodology as in the other two-dimensional data sets. The fig. 2.14 shows the validation results achieved by FAM, DAM and GAM (varying $\bar{\rho}$ in the range 0.00:0.02:0.98), and PTAM, and table 2.10 reports the test results. PTAM achieves lower test error (2.3%) than the best rectangular and circular ART networks. GAM achieves more error than PTAM for all the vigilance values, and the test error is twice the error of PTAM. FAM and DAM need about 1000 categories to achieve the same error as PTAM (fig. 2.14). We selected two operating points for DAM and other two points for FAM (marked with empty squares in the figure). DAM achieves higher error (2.7%) than PTAM using $\bar{\rho} = 0.94$, and more categories (572). Using $\bar{\rho} = 0.88$, DAM also achieves more error (3.9%) than PTAM, with slightly less categories (270). FAM achieves higher error than PTAM for the two selected vigilance values.

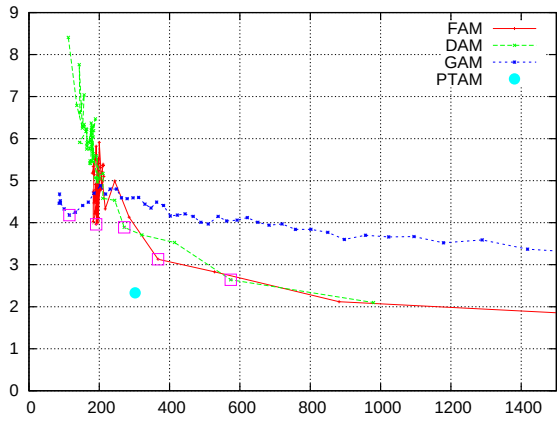


Figure 2.14: Average validation error (in %) and #C achieved by FAM, DAM and GAM (varying vigilance) and by PTAM.

2.5.5 Real, high-dimensional data sets

High-dimensional data sets PID and Abalone have a reduced number of patterns, so the usual experimental procedure [65] is Kfold cross-validation with $k = 4$. In this paper, in order to optimize the vigilance parameter, we created 20 training,

	PTAM	GAM	DAM		FAM	
$\bar{\rho}$	–	0.12	0.88	0.94	0.36	0.92
Error (%)	**2.3**	4.3	3.9	2.7	4.1	3.3
#C	302	**115**	270	572	190	368

Table 2.10: Test error (in %) and #C achieved by PTAM and the best rectangular and circular ART networks (for which the value of $\bar{\rho}$ is reported) in the Form data set. The best error and #C are in bold.

20 validation and 20 test sets, containing 25%, 50% and 25% of the input patterns respectively, randomly selected from the original data set. Specifically, we used 192 training, 384 validation and 192 test patterns in PID, and 1044 training, 2088 validation and 1044 test patterns in Abalone. Baseline vigilance values $\bar{\rho}$ on the set $\{0.0, 0.2, 0.4, 0.6, 0.8, 0.95\}$ were tried, and the value with the best average trade-off between the error and #C over the 20 validation sets was selected (table 2.4).

Table 2.11: Test error (ϵ, in %) and #C in high-dimensional data sets PID and Abalone. The best ϵ and #C achieved by an ART and non-ART classifier are in bold.

		PTAM	FAM	GAM	DAM	EAM	FasArt	SVM
PID	ϵ	43.6	**31.1**	37.4	**30.9**	34.3	**30.8**	**24.5**
	#C	142	63	**19**	25	22	29	100
Abalone	ϵ	50.3	**42.9**	57.4	**43.3**	47.6	44.1	**34.2**
	#C	**27**	369	60	131	174	341	764

The table 2.11 reports the average results of each classifier with its best vigilance value over the 20 test sets. SVM is clearly the best classifier in PID (24%) and Abalone (34%), with 52% and 73% of the training patterns as support vectors respectively. In the PID data set, this error was also achieved by AFC [128] using 576 training and 192 test patterns, and with zero vigilance. FasArt, DAM and FAM achieve the lowest error among the ART networks (about 30%), but FAM creates twice the number of categories (60) than FasArt (29) and DAM (25). GAM and EAM achieve more error (34-37%), with few categories. PTAM achieves the highest error (43%) and number of categories (74% of the training patterns). In the Abalone data set, FAM and DAM achieve the lowest error (43%), but FAM creates the highest #C (370), and DAM creates only 131 categories. FasArt and EAM

achieve higher error than FAM and DAM, with less categories. PTAM achieves 50% error, only below GAM (57%), but it is similar to the one reported by FAM (50%) in [5]. Also, PTAM creates the lowest number of categories (27) among the ART networks.

2.6 Discussion

The average test results (error and #C) obtained by the different classifiers in the two-dimensional data sets 1-7 are reported in the lowest two rows of table 2.5 and represented in fig. 2.10 (d). SVM achieves the lowest global error (0.4%), with a high number of support vectors (298). PTAM achieves the lowest average error among the ART networks (1.6%), and the lowest error in five of seven data sets. In the other two data sets, PTAM does not achieve much more error than the best ART network. Only GAM and FAM create less categories (44 and 137 respectively) than PTAM (188). DAM achieves the second lowest error (2%), with #C (193) similar to PTAM. However, DAM only achieves less error than PTAM in the rectangular data sets Chess and T4. FAM achieves more error than DAM, and less categories. EAM achieves equal error as GAM, and it creates the highest #C. FasArt achieves the highest error (3.8%) and #C. PTAM also achieves less error than the best rectangular and circular ART networks (DAM and GAM), with a low number of categories, on the irregular geometry data set Form (table 2.10).

PTAM creates a new simplex during resonance only if the resonant category is not the most active one (subsection 2.1.5). We developed experiments in which a new simplex is created everytime an input pattern falls outside the resonant category. In these experiments, the average test error and #W achieved by PTAM in data sets 1-7 are 1.4% and 262 weight vectors. The error reduces 0.2, but the number of weight vectors #W increases by 39%. Thus, the creation of new simplexes in all the cases does not significantly reduce the error rate, but it contributes to category proliferation. This justifies our hypothesis of creating a new simplex only if the resonant category is not the most active one.

We also did experiments related with the parameter θ_{min} used to discard acute simplexes (subsection 2.1.4). The fig. 2.15 shows the typical behavior of error and number of vectors achieved by PTAM varying θ_{min} in T4 and T5 (they were selected as representants of rectangular and circular geometries, but we found the same

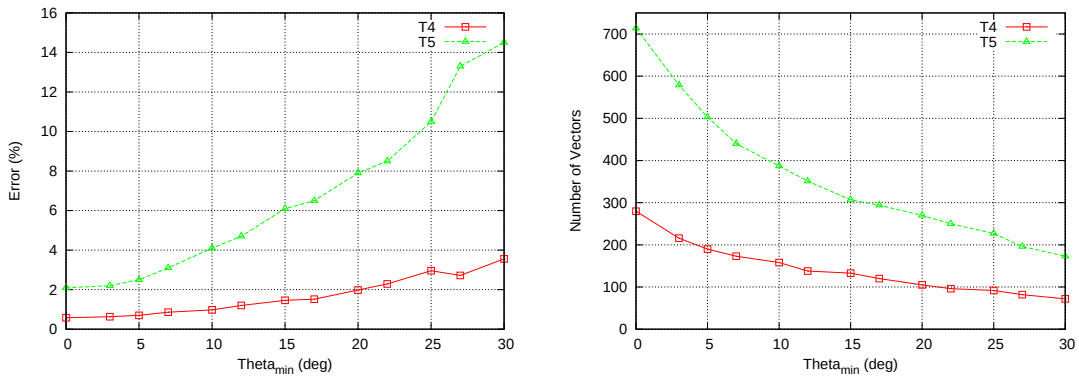


Figure 2.15: Error (left panel) and number of vectors (right panel) achieved by PTAM against θ_{min} in data sets T4 and T5.

behavior in the other data sets). High θ_{min} values increase the error and reduce the number of vectors, because too many simplexes are discarded, so PTAM is unable to fit the borders among the output predictions. With low θ_{min} values, PTAM discards few simplexes, so its learning ability is higher, the error is low and the number of vectors is high. If $\theta_{min} = 0$, no simplex is discarded, so the number of simplexes and vectors boosts (left end of the right panel in fig. 2.15). Therefore, a low θ_{min} value must be used to discard only the acute simplexes. Based on the experiments developed with several data sets, we selected the value $\theta_{min} = 10$ degree as acceptable. Given that the mimimum angle to consider a simplex as acute does not seem to depend on the data set, this value should be valid for any data set, so it does not need to be optimized via cross-validation.

It is interesting to evaluate the behavior of each classifier depending on the data set geometry. The table 2.5 reports that PTAM is the ART network with the lowest standard deviation in the error (1.1) compared to EAM (1.5), FAM, GAM and DAM (1.7), and FasArt (2.0). Thus, the results of the ART networks are more dependent on the data set than the results of PTAM. We grouped the results of each classifier for rectangular and circular data sets. For each data set, we calculated the average error over all the ART networks. Then, we calculated the difference between the error achieved by each classifier and the average error (negative if the error is lower than the average, and positive otherwise). The fig. 2.16 shows the average differences for each classifier over the circular (CIS, T3 and T5-T7) and over the rectangular data sets (Chess and T4). FAM and DAM have high negative difference for the rectangular data sets and zero difference for the circular ones. The

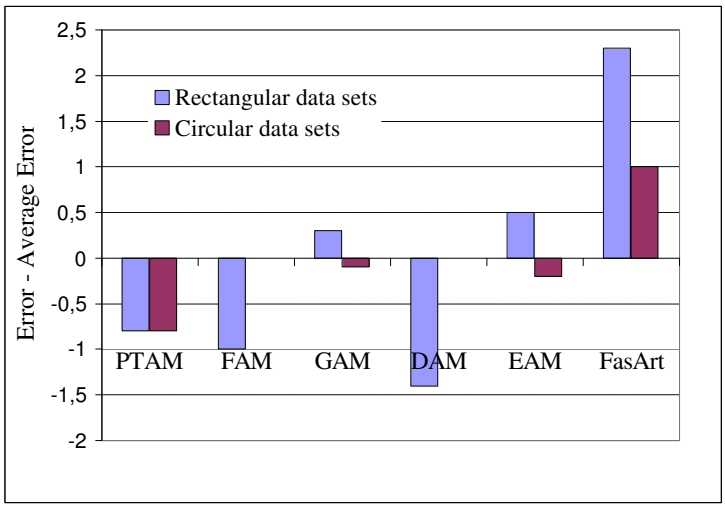


Figure 2.16: Difference between the error (in %) achieved by each ART classifier and the average error for each data set, grouped in rectangular and circular data sets.

circular classifiers (GAM and EAM) have negative (positive) differences for circular (rectangular) data sets. FasArt has high positive difference for rectangular and circular data sets. Thus, the behavior of both circular and rectangular classifiers depend on the data set geometry, and their average results are better (negative differences) for data sets with their equal geometry. PTAM is the only network with high negative difference for both geometries of data sets, because its irregular polytope categories are not specially suited to any particular geometry.

The good results of PTAM are possible without vigilance and, consequently, without any parameter tuning. This feature makes PTAM easier to use than the other ART networks, because this tuning often requires cross validation trials, which may have an important cost in processing time. On the other hand, these trials require validation sets, so that the number of available training patterns is reduced, and learning is difficult. This may be a drawback in real data sets, where the number of available input patterns may be limited by the cost of each data acquisition.

The efficiency of PTAM is not so high in the data sets CIS-noise, T5-noise (table 2.8), and 4G1, 4G2 and 4G3 (table 2.9) and also in PID and Abalone. PTAM also creates a high number of categories on these data sets, although it is lower than FasArt and, specially, SVM. PTAM approximates the borders among the output predictions by strictly following the information contained in the training set, both in

the category expansion and adjustment steps. In the presence of noise or prediction overlap, the input patterns belonging to different categories are mixed and they break simplexes in the category adjustment step, which may lead to the creation of noisy single-vector categories. The future work will use statistical information in the category expansion and adjustment steps in order to make PTAM robust to noise and category overlap.

Chapter 3

Overlapping PolyTope ARTMAP

The most significant features of PTAM are the irregular polytope geometry of its internal categories, and its non-overlapping nature, which requires an overlap test and avoids the vigilance parameter. Although both features are somehow related, it is reasonable to wonder which happens if we use irregular polytope categories but category overlap is allowed. We tried to answer this question proposing a third model, called Overlapping PolyTope ARTMAP (OPTAM), whose internal categories are overlapped irregular polytopes. In this case, a maximum category size is imposed, given by the vigilance parameter, as in the classical ART networks, so OPTAM requires to tune the vigilance parameter. OPTAM also uses the category size to select one among several overlapped categories when the input pattern falls inside an overlap region. On the other hand, OPTAM does not use the overlap test, so it is computationally simpler than PTAM. The following sections describe the training and processing phases of OPTAM, and its application to the some of the benchmark data sets used with PTAM.

3.1 Training phase

The nomenclature used with OPTAM is the same as in the previous chapter (table 2.1). The choice function $T_i^t(\mathbf{I})$ of category C_i is defined as the maximum of its simplexes choice functions, similarly to PTAM:

$$T_i^t(\mathbf{I}) = \max_{j=1,\ldots,N_i^s} \{T_{ij}(\mathbf{I})\} \qquad i = 1, \ldots, N_c \tag{3.1}$$

The choice function $T_{ij}(\mathbf{I})$ of simplex S_{ij} is given by:

$$T_{ij}(\mathbf{I}) = \begin{cases} 1 & g_{ijk}(\mathbf{I}) > 0, \quad k = 1, \ldots, n+1 \\ 1 - \dfrac{d(\mathbf{I}, S_{ij})}{\sqrt{n}} & \text{otherwise} \end{cases} \tag{3.2}$$

The function $g_{ijk}(\mathbf{I})$ is defined in eq. 2.2 in such a way that $g_{ijk}(\mathbf{I}) = 0$ if $\mathbf{I}$ falls in the side of hyperplane h_{ijk} inside simplex S_{ij}, and $g_{ijk}(\mathbf{I}) = 0$ if $\mathbf{I}$ falls outside the simplex S_{ij}. Thus, $T_{ij}(\mathbf{I}) = 1$ if $\mathbf{I}$ falls inside S_{ij}. Otherwise, the distance $d(\mathbf{I}, S_{ij})$ between $\mathbf{I}$ and S_{ij} is computed using the following equation:

$$d(\mathbf{I}, S_{ij}) \simeq \begin{cases} \min\limits_{k:g_{ijk}(\mathbf{I})<0} \dfrac{|g_{ijk}(\mathbf{I})|}{\|\mathbf{w}_{ijk}^*\|} & \|\mathbf{I} - \mathbf{c}_{ij}\| < R_{ij} \\ \min\limits_{l=1,\ldots,n+1} \{\|\mathbf{I} - \mathbf{w}_{ijl}\|\} & \|\mathbf{I} - \mathbf{c}_{ij}\| \geq R_{ij} \end{cases} \tag{3.3}$$

If $\mathbf{I}$ falls outside the simplex S_{ij}, $T_{ij}(\mathbf{I}) < 1$ and it decreases with $d(\mathbf{I}, S_{ij})$, being $T_{ij}(\mathbf{I}) = 1$ if $d(\mathbf{I}, S_{ij}) = 0$ and $T_{ij}(\mathbf{I}) = 0$ if $d(\mathbf{I}, S_{ij}) = \sqrt{n}$ (maximum value of the distance).

In OPTAM, similarly to the classical "winner-takes-all" ART networks, the winner of the competition among internal categories is the category C_I with the highest activation which is non reset ($r_I = 0$):

$$T_I^t(\mathbf{I}) = \max_{i=1,\ldots,N_c} \{T_i^t(\mathbf{I}) : r_i = 0\} \tag{3.4}$$

3.1.1 Prediction Test

Similarly to PTAM, OPTAM compares the prediction $P(C_I)$ associated to the winner category C_I with the desired prediction P_d for the input pattern (Prediction Test). If $P_d = P(C_I)$, the winner category C_I passes the Prediction Test and it codifies the input pattern (Resonance). Otherwise, C_I is reset and a new winner category is selected in the competition. OPTAM does not use *Match-Tracking* [28],

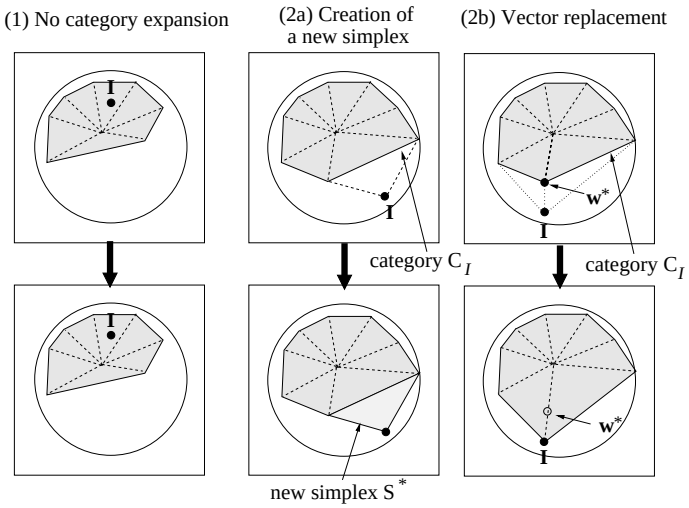


Figure 3.1: Polytope category learning examples for the data set "Circle-In-the-Square" (CIS) in $\mathbb{R}^2$. See text for details.

usual in ART networks, because it reduces the maximum category size and, consequently, its ability of expansion, and it increases the classification error and the number of categories created.

3.1.2 Category expansion

If the winner category C_I passes the Prediction Test, it tries to expand towards $\mathbf{I}$, either creating a new simplex between them, or replacing a vertex in C_I by $\mathbf{I}$. The creation of a new simplex requires n vectors $\{\mathbf{w}_p\}_{p=1}^n \in C_I$ for which the segments $\{\overrightarrow{\mathbf{Iw}_p}\}_{p=1}^n$ do not overlap with C_I (i.e., they are connectables with $\mathbf{I}$). In order to select these vectors, the intersection between $\overrightarrow{\mathbf{Iw}_p}$ and the simplexes of C_I is tested using the function O_{ls} defined in subsection M.1. Depending on the number of connectable vectors there are several cases (fig. 3.1). If $T_I(\mathbf{I}) = 1$, C_I already covers $\mathbf{I}$, so C_I does not need to expand towards $\mathbf{I}$ (panel 1). If $T_I(\mathbf{I}) < 1$, C_I tries to expand towards $\mathbf{I}$. If there are n vertices in C_I connectables with $\mathbf{I}$, C_I expands towards $\mathbf{I}$ creating a new simplex (panel 2a) with the n connectable vertices and $\mathbf{I}$. If there are more than n vertices in C_I connectables with $\mathbf{I}$, then C_I expands towards $\mathbf{I}$ replacing some vertex in C_I by $\mathbf{I}$ (panel 2b). Note that the number of connectable vectors from $\mathbf{I}$ in C_I can not be lower than n, as opposite to PTAM (fig. 2.3, panel 2a), because category overlap is allowed in OPTAM.

The replacement of a vector $\mathbf{w}$ in C_I by $\mathbf{I}$ may reduce the volume of the simplexes to which $\mathbf{w}$ belongs (fig. 2.5, right panel in subsection 2.1.4). In particular, the patterns that can replace $\mathbf{w}$ in fig. 2.5 are those located in region A, which is outside the hyperplanes (lines in $\mathbb{R}^2$) that contain $\mathbf{w}$. Therefore, the vectors $\mathbf{w}$ that can be replaced by $\mathbf{I}$ must verify that $g_{Ijk}(\mathbf{I}) < 0, \forall k : \mathbf{w} \in h_{(I,j,k)}$. If there are several vectors that fulfill this condition, $\mathbf{I}$ replaces the closest one.

3.1.3 Vigilance Test

In the ART networks, the Vigilance Test determines if the winner category C_I can expand towards $\mathbf{I}$. The vigilance parameter ρ determines the maximum category size. The size S_i of a polytope category C_i in OPTAM is defined as the sum of the ranges of its vertices in the n dimensions of the input space, similarly to Fuzzy ARTMAP:

$$S_i = \sum_{l=1}^{n} M_{il} - m_{il}; \qquad M_{il} = \max_{j=1,\ldots,N_i^s} \left\{ \max_{k=1,\ldots,n+1} \{\mathbf{w}_{ijkl}\} \right\} \qquad (3.5)$$

$$m_{il} = \min_{j=1,\ldots,N_i^s} \left\{ \min_{k=1,\ldots,n+1} \{\mathbf{w}_{ijkl}\} \right\}$$

Note that $0 \leq S_i \leq n$. The maximum category size S_{max} allowed in OPTAM is defined as $S_{max} \equiv (1-\rho)n$. If the winner category C_I satisfies $S_I \geq S_{max}$ after the expansion towards $\mathbf{I}$, C_I is reset (Vigilance Test). Then, a new winner category is selected in the competition, and the prediction test and category expansion are repeated again. If $S_I < S_{max}$ after the category expansion, then the winner category C_I achieves the ressonance with the input pattern.

3.1.4 Creation of new categories

In the classical ART networks, if no category passes the Vigilance Test, a new category is created, defined by the input pattern. The categories of OPTAM are defined by $(n+1) + (N_i^s - 1) = n + N_i^s$ weight vectors (vertices). Therefore, the creation of a new category C_{N_c+1} defined by just one simplex S^* requires at least n weight vectors (vertices), belonging to single-vector categories, with the same prediction as the input pattern. If there are n vectors of this kind, OPTAM creates

a new single-simplex category C_{N_c+1} with just the simplex S^* ($N^s_{N_c+1} = 1$ and $S_{(N_c+1)1} = S^*$). If there are not n vectors of this kind, OPTAM creates a new single-vector category C_{N_c+1} with zero simplexes ($N^s_{N_c+1} = 0$) and with just one vector equal to the input pattern $\mathbf{w}_{N_c+1} = \mathbf{I}$.

3.2 Processing phase

During the processing phase, the output of OPTAM for an input pattern is the prediction associated to the category with the highest choice function $T^p_i(\mathbf{I})$. However, since the categories in OPTAM can overlap, the category size must be taken into account in the calculation of $T^p_i(\mathbf{I})$. Similarly to FAM, $T^p_i(\mathbf{I})$ is decreasing with the category size. We define the function $\chi_i(\mathbf{I})$ for category C_i as:

$$\chi_i(\mathbf{I}) = \begin{cases} \max\limits_{j=1,\ldots,N^s_i} \{T_{ij}(\mathbf{I})\} & N^s_i \geq 1 \\ 1 - \dfrac{\|\mathbf{I} - \mathbf{w}_i\|}{\sqrt{n}} & N^s_i = 0 \end{cases} . \tag{3.6}$$

where $T_{ij}(\mathbf{I})$ is defined in eq. 3.2. In the eq. 3.6, $\chi_i(\mathbf{I})$ is equal to $T^t_i(\mathbf{I})$ except for single-vector categories, for which $\chi_i(\mathbf{I})$ is decreasing with the distance between $\mathbf{I}$ and $\mathbf{w}_i$. When the input pattern falls inside a region covered by several overlapped categories, for which $\chi_i(\mathbf{I}) = 1$, the category with the lowest size S_i has the highest activation function $T^p_i(\mathbf{I})$. Using the function $\Theta(x)$, defined as $\Theta(x) = 1$ if $x \geq 0$ and $\Theta(x) = 0$ if $x < 0$, we define the category activation function $T^p_i(\mathbf{I})$ in the processing phase by:

$$T^p_i(\mathbf{I}) = [1 - \psi(\mathbf{I})]\chi_i(\mathbf{I}) + \psi(\mathbf{I}) \frac{\Theta[\chi_i(\mathbf{I}) - 1]}{\alpha + S_i} \tag{3.7}$$

$$\psi(\mathbf{I}) = \Theta\left[\sum_{l=1}^{N_c} \Theta[\chi_l(\mathbf{I}) - 1] - 2 \right] . \tag{3.8}$$

where $\alpha \gtrsim 0$ and S_i is defined in eq. 3.5. The function $\psi(\mathbf{I})$ verifies $\psi(\mathbf{I}) = 1$ if several categories exist with $\chi_i(\mathbf{I}) = 1$ (i.e., $\mathbf{I}$ falls in an overlap region). In such a case, $T^p_i(\mathbf{I}) = 1/(\alpha + S_i)$ for the categories with $\chi_i(\mathbf{I}) = 1$ (overlapped categories), and $T^p_i(\mathbf{I}) = 0$ for the remaining ones, so that $T^p_i(\mathbf{I})$ is decreasing with the category

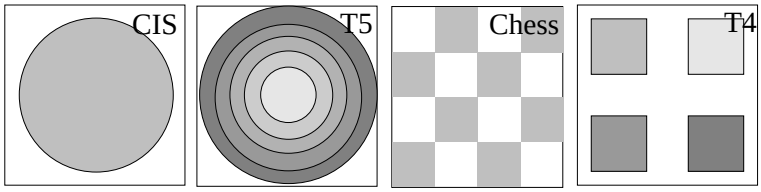


Figure 3.2: Data sets CIS, T5 (circular) and Chess, T4 (rectangular), with 2, 6, 2 and 5 output predictions respective

size S_i. If there is only one category with $\chi_i(\mathbf{I}) = 1$ ($\mathbf{I}$ falls inside only one category), then $\psi(\mathbf{I}) = 0$ and $T_i^p(\mathbf{I}) = \chi_i(\mathbf{I})$. The winner category C_I is that with the highest $T_i^p(\mathbf{I})$:

$$T_I^p(\mathbf{I}) = \max_{i=1,\ldots,N_c} \{T_i^p(\mathbf{I})\} \ . \tag{3.9}$$

The output of OPTAM is the prediction $P(C_I)$ associated to the winner category C_I.

3.3 Results

Experiments were developed comparing OPTAM with PTAM, SVM and the rectangular and ellipsoidal ART networks which achieved the best results in the experiments described in the previous chapter: Distributed ARTMAP (DAM) [36] (rectangular categories) and Gaussian ARTMAP (GAM) [144] (hyperellipsoidal categories). We used two data sets with circular geometry ("Circle-in-the-Square" (CIS) [143] and T5 [120]) and two data sets with rectangular geometry (Chess [66] and T4 [120]), in order to evaluate each network in data sets with the same and different geometries (fig. 3.2).

DAM, GAM and OPTAM require the optimization of the vigilance ρ for each data set. So, we developped cross-validation trials using 20 training, 20 validation and 20 test sets. Each set contains 10,000 randomly generated patterns with equiprobable predictions. We trained a version of each network in each training set using vigilance values in the range $0.00 : 0.05 : 0.95$. We proved each version in its validation set, and selected the vigilance value with the best tradeoff between the average classification error and number of categories over the 20 validation sets. The

figs. 3.3 and 3.4 show the classification error against the number of categories, averaged over the validation sets, for OPTAM, DAM, GAM, SVM and PTAM, varying the vigilance in the data sets CIS, Chess, T4 and T5. Finally, we evaluated each network, trained with its best value of ρ, on the 20 test sets. The table 3.1 shows the average test results over the 20 test sets. OPTAM uses $\rho = 0$ for data sets CIS, T4 and T5, and $\rho = 0.75$ for Chess.

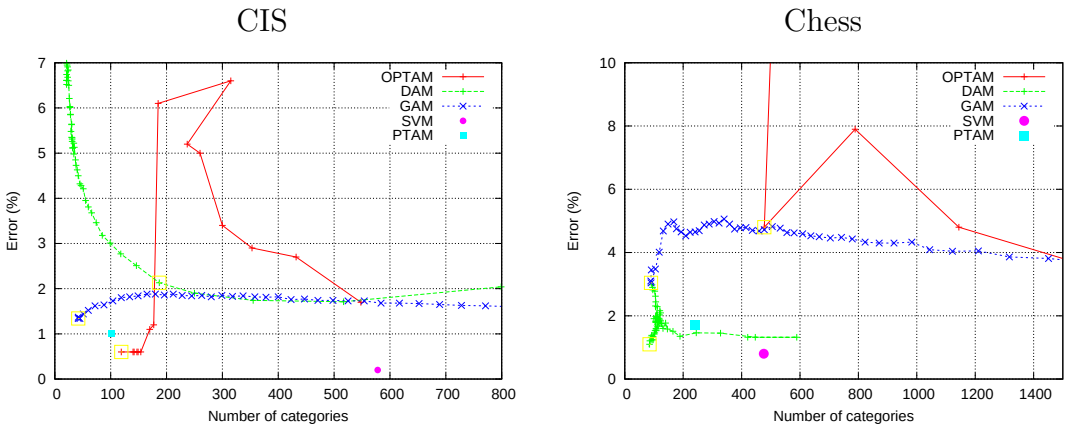


Figure 3.3: Average classification error against the number of categories over the validation sets in data sets CIS (left panel) and Chess (right panel) varying vigilance. The best tradeoff between error and categories is marked with an empty square for each network.

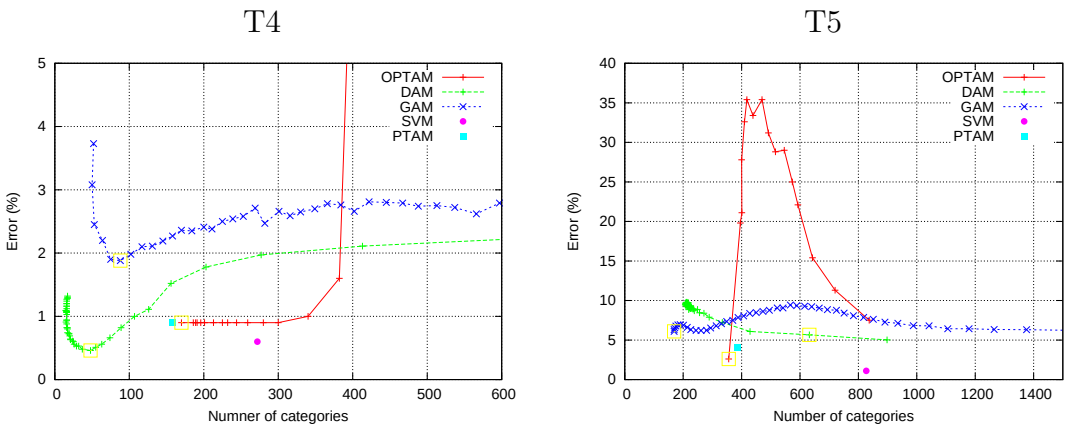


Figure 3.4: The same in data sets T4 and T5.

Table 3.1: Test error and number of internal categories (vectors for OPTAM-PTAM and support vectors for SVM) for each network and data set, and average values (the best values are in bold).

	CIS		Chess		T4		T5		Average	
	Error	No. Cat	Error	No. Cat	Error	No. Cat	Error	No. Cat	Error	No. Cat
OPTAM	**0.6**	119	4.8	478	0.9	170	**2.6**	356	2.2	281
PTAM	1.0	102	1.7	242	1.0	158	4.1	387	**2.0**	222
GAM	1.3	**43**	3.0	90	1.9	89	6.1	**169**	3.1	**98**
DAM	1.3	238	**1.1**	**89**	**0.4**	**48**	5.6	630	2.1	251
SVM	**0.2**	578	**0.8**	475	0.6	271	**1.1**	827	**0.7**	538

3.4 Discussion

The table 3.1 shows that OPTAM does not achieve less average error than SVM, which is again the best classifier. The behavior of OPTAM is variable: it achieves slightly lower error (0.9%) than PTAM (1.0%) in T4 and more error (4.8% against 1.7% using PTAM) in Chess, which are the two rectangular data sets. In the circular data sets OPTAM clearly overcomes PTAM (0.6% against 1.0% in CIS and 2.6% against 4.1% in T5), DAM and GAM. The behavior of OPTAM is also variable in the number of categories created, which is higher than PTAM in data sets CIS, Chess and T4, and less in T5. The figs. 3.3 and 3.4 also show huge variations in the validation error and number of vectors achieved by OPTAM varying vigilance, with respect to the other ART networks.

On the average, PTAM achieves the lowest classification error (2.0%) among the ART networks, slightly lower than DAM and OPTAM (2.1% and 2.2% respectively), and clearly lower than GAM (3.1%)[1]. On the other hand, GAM achieves the lowest number of categories (98), but the highest error, while PTAM creates less categories (222) than DAM (251) and OPTAM (281). Therefore, the use of overlapped categories in OPTAM does not reduce the error with respect to PTAM or to DAM and, in addition, it increases the number of categories compared with these two networks.

It is interesting to analyse the behavior of OPTAM in the data set Chess. Using $\rho = 0$, the maximum category size is $S_{max} = n$ (subsection 3.1.3) and OPTAM

[1]We also developped additional experiments with OPTAM using *Match-Tracking*, and we achieved higher average test error (4.2%) and number of categories (384), so they are not included in table 3.1.

creates only two overlapped categories, that cover all the input space. Since the two predictions are distributed among several non-adjacent regions, these two categories are not able to discriminate among the regions associated to each prediction, so the error is very high ($\sim$50%, left end of fig. 3.3, left panel). Therefore, OPTAM needs a high vigilance value ($\rho = 0.75$) to achieve an error comparable to PTAM, DAM and GAM, and it creates a high number of categories (478). Nevertheless, in the Chess data set, when a pattern falls in an overlap region during the processing phase, the smallest category is not necessarily the right one, so that OPTAM achieves a relatively high error (4.8%) compared with the other networks, even with high vigilance values (right end of fig. 3.3, left panel). On the contrary, in the data sets CIS, T4 and T5 the value $\rho = 0$ provides the smallest error and number of categories, because each prediction has only one associated region, which can be correctly covered by only one category using $\rho = 0$. For example, in T5 the predictions are concentric, so the categories created by OPTAM for the different predictions overlap. If a pattern falls inside an overlap region during the processing phase, it is assigned to the smallest category, which is the right one in T5. The same behavior can also be observed in CIS and T4.

This variability in the results, which depend on the number of regions associated to each prediction and on their relative positions, shows that the behavior of OPTAM is not very stable. In fact, the error achieved by PTAM varies less among data sets, since its standard deviation is the lowest among the ART networks (1.5 against 1.9 for OPTAM, 2.1 for DAM and 2.4 for GAM). These results suggest that polytope categories are more effective– both in error and number of categories– if they do not overlap. In addition, for an accurate learning of data sets with complex geometries, it turns out contradictory to use irregular polytope categories and, on the other hand, to give more relevance– as OPTAM does– to the category size compared with the category geometry when patterns fall inside several overlapped categories.

Conclusions

In the present thesis we propose several neural models oriented to extend the adaptive nature of the ART neural networks to the own geometry of the internal categories. These proposals try to evaluate how much generic geometry, learnt during the training process, contributes to the performance of the ART networks in supervised classification tasks.

The first consequence of internal categories with generic geometry is that each category can not be exclusively defined by a weight vector, acting as the expected value, and a choice function which defines a non-adaptive category geometry. On the contrary, the generic categories are defined by several selected training patterns which act as the category vertices[2]. Besides, the choice function is complex and it depends on the category vertices, implying the calculation of n-order determinants in PTAM and OPTAM. This function is one of the reasons of the high computational complexity of the proposed models.

Simplex ARTMAP (SAM) uses simplex-shaped categories defined by selected training patterns. Each category has a Gaussian-based choice function with non-zero values only inside the category support. The Gaussian functions are centered in points inside the simplex and they have the same spread, which determines the number of centers required. The regions associated to the output predictions are built as ensembles of adjacent, non-overlapped simplexes, since the simplex is the volume in $\mathbb{R}^n$ with linear borders and the lowest number of vertices. SAM does not allow prediction overlap, so that category expansion is limited by the other categories, and the vigilance parameter is not necessary. The learning proceeds creating a new simplex between the existing categories and the training pattern, or including the training pattern to an existing category if it has less than $n+1$ vertices.

[2]The number of vertices of a category SAM is bounded by $n+1$, but this number is not bounded in PTAM and OPTAM.

The experiments show the ability of SAM to build prediction borders and to obtain a performance level similar or higher than FAM and DAM, but SAM creates more categories. SAM does not need the vigilance parameter, which must be fine-tuned and has strong influence on the performance of FAM/DAM. However, the Gaussian spread replaces vigilance as a tuning parameter. On the other hand, the number of Gaussian centers raises exponentially with the dimension n of the input space, so its applicability to high-dimensional problems is hard.

These issues leaded us to define the category choice function based on the category border, and not on its inner volume, as in SAM. PolyTope ARTMAP (PTAM) uses hyperplanes as category borders to discriminate between training patterns falling inside and outside the simplex volume. Also, the irregular polytope categories of PTAM are defined as an ensemble of adjacent, non-overlapped simplexes. This geometric representation is more flexible, and potentially richer to model data sets, than the existing ART networks. The polytope categories can expand creating a new simplex between then and the training pattern, or replacing a vertex by the training pattern. PTAM uses an overlap test, based on geometric calculations to avoid category overlap. This test replaces the vigilance test, but it removes the vigilance, so PTAM has no tuning parameter, and it operates in a completely automatic way. A simplex is removed when a training pattern with a different prediction falls inside it. When no category with the desired prediction can expand towards the input pattern without overlap, PTAM creates a new single-vector category.

PTAM achieves less error than the leading ART networks in the two-dimensional data sets without noise and prediction overlap. As well, its behavior is less dependent on the data set geometry than the other ART networks. The number of vectors created by PTAM is comparable to or lower than the number of categories in the ART networks. Hence, PTAM is clearly better than the best rectangular and circular ART networks on a data set with irregular geometry. We would like to emphasize that PTAM achieves these results without the vigilance parameter. The absence of tuning parameters reports the following advantages: i) The non-expert user is not concerned about the internal operation of PTAM, and he does not need to know the meaning of each tuning parameter, so it is easier to use than most classifiers, which require parameter tuning. Although ART networks can use zero vigilance, it may not be the best value for a given data set. ii) The input patterns devoted to cross validation sets can not be used for training: this fact may be important for real data sets, which often have a reduced number of input patterns.

During the thesis, the results were achieved using several training epochs. The objective was to evaluate the best efficiency achieved by each classifier, which is normally higher using several epochs than using only one epoch, specially, in order to compare with SVM, which uses several epochs. However, since ART networks learn on-line, they are able to achieve acceptable results with only one epoch, as opposed to the networks with off-line training (MLP, SVM, etc.). Thus, they do not require several epochs, although in this case they may achieve better results.

Another relevant issue is the influence of the presentation order of the training patterns in the results. PTAM only creates a new simplex if the resonant category is not the first winner in the competition for the current training pattern (subsection 2.1.5). This criterion is oriented to reduce the number of training patterns learnt, but it increases the influence of the presentation order in the results. The reason for this criterion is that vectors are replaced only when condition 2.9 (subsection 2.1.4) is met, which is somehow hard, so that many training patterns create new simplexes. This fact limits the efficiency in the selection of training patterns, which leads to impose this criterion in order to limit the creation of new simplexes. We are working to better select the category vertices, in order to avoid this limiting criterion and to reduce the dependency with respect to the pattern presentation order.

The efficiency of the category expansion-contraction algorithm of PTAM is not so high with high noise levels and prediction overlap, which suggests that the category adjustment step creates noisy single-vector categories, so that both error and number of vectors is high. An objective of future work is to contract the wrong category in the category adjustment atep without the creation of new categories and without processing more training patterns.

Local computation a basic property of every neural model. Local computation means distributed processing over many neurons, connected in a parallel operation, being the output of each neuron a function of its inputs only. This issue is very important for the hardware implementation of a neural model. The training algorithm can be non-local in networks with off-line training (e.g., the SVM algorithm can be used to train a feedforward network). However, in the ART networks the training algorithm must be local, because they must be able to switch between training and processing modes, so the weight calculation must be done by the network itself. The models proposed in this thesis do not exhibit local computation, and intentionally we did not include any architecture diagram. The reason is that their complexity

(e.g., the weight vectors are vertices instead of prototypes, and the implementation of the overlap test, among other issues) difficults its implementation on a simple architecture, as in the classical ART networks. These models were proposed in an algorithmic style, without analyzing its possible hardware implementation, and highlighting its ART concepts (choice functions, expected values, prediction test, resonance, reject, ...). We delayed to the future work the local formulation, and we are now working in an efficient implementation of several stages (category expansion and contraction, vector replacement and overlap test), in order to allow a implementation with local computation.

The main contribution of this thesis is to enable irregular, non-predefined geometry categories in the ART networks. This feature introduces significant changes on its operation, e.g., the removal of the vigilance parameter. Therefore, we considered that the best way to measure the contributions of the proposed models was to compare them with the standard classifiers, specially ART-based, using several benchmark data sets. From this point of view, we delayed for the future work the application of the proposed models in classification problems of practical interest (real applications).

The main drawbacks of PTAM are related with its high computational complexity, due to the n-order determinants of the category choice funtions, and to the n-order systems of linear equations which must be solved to evaluate the category overlap (overlap test). We analyzed the interest of the polytope categories when category overlap is allowed, in order to determine the real contribution of irregular categories in the traditional ART networks– which do allow overlap–. The resulting model, called Overlapping PTAM, saves the cost of overlap test, but it uses again the vigilance parameter, which must be tuned using cross-validation trials. The results achieved by OPTAM in the two-dimensional data sets are worse than PTAM and the other ART networks, both in error and number of categories, and its behavior is clearly unstable. These results suggest that irregular geometries, on its own, do not raise the learning ability of the ART networks, nor create less categories. The irregular geometries seem to achieve better results if categories do not overlap, as in PTAM.

Our experiments lead us to conclude that non-overlapped irregular categories raise the learning ability of the ART networks, but the method used to select the category vertices and to expand and contract categories is very important to increase the performance and eficiency of the network. Consequently, the main objective of

future work is to change the category expansion and adjustment steps, in order to increase the efficiency in the selection of the category vertices, while avoiding the problem of acute simplexes, and to correct the wrong category in a more efficient way than PTAM, without simplex breaking. We are also working to replace simplexes by hyperplanes in the definition of the polytope categories, in order to overcome some simplex limitations in the category expansion. As well, we work to reduce the computational complexity of PTAM simplifying the calculation of the choice function and the overlap test, which difficult the use of PTAM in high-dimensional data sets.

Apêndice A

Máquinas de Suporte Vetorial (SVM)

As Máquinas de Suporte Vetorial (*Support Vector Machine*, SVM) [41] foram propostas nos anos 90 como método para a classificação de padrões e de regressão não-linear. Seu potencial encontra-se na forte conexão que possui com a Teoria da Aprendizagem Estatística [135], de modo que uma SVM é uma implementação aproximada do método de minimização do risco estrutural. Entre as propriedades mais atrativas da SVM, figuram a sua capacidade para uma elevada generalização e para evitar a super-aprendizagem, problema muito sério em outras aproximações (MLP, RBF, FAM). No contexto da classificação de padrões, a SVM opera como um classificador de duas categorias que constrói um hiperplano como uma superfície de decisão, de modo que se maximize a margem de separação entre os exemplos pertencentes a ambas as categorias. Esta classificação linear se efetua no espaço oculto, transformado a partir do espaço de entrada, mediante um núcleo (*kernel*) não linear. A SVM proporciona normalmente bons resultados de generalização em problemas de classificação, em comparação com os outros classificadores como o MLP, apesar de não incorporar o conhecimento relativo ao domínio do problema. Além do mais, a SVM não está afetada pela "Maldição da Dimensionalidade"(*curse of dimensionality*) [11], de modo que opera melhor que os outros classificadores com padrões de dimensionalidade elevada.

A SVM treina eficientemente a um classificador linear em um espaço de características de dimensão superior ao espaço original das características (espaço oculto), induzido por núcleos, que realizam uma transformação não-linear, desde o espaço de

entrada ao espaço oculto. Esta transformação trata de converter o problema original, que em geral é não-linearmente separável, em um problema separável no espaço oculto, baseando-se no espaço oculto do teorema de *Cover* [42]. A determinação deste discriminador linear (hiperplano) se realiza maximizando a capacidade de generalização. Para isso, a SVM utiliza os resultados da aprendizagem estatística e as teorias de otimização para maximizar sua capacidade de generalização sobre o conjunto de validação, e assim, minimizar os problemas de super-aprendizagem (*overfitting*).

O objetivo da SVM [43], num problema de classificação, é aprender a função $f : \mathbb{R}^n \rightarrow \{\pm 1\}$ (discriminador de categoria) definida por um conjunto de treinamento $\{(\mathbf{x}_i, y_i) \in \mathbb{R}^n \times \{\pm 1\}, i = 1, \ldots N\}$, onde o n é a dimensão do padrão de entrada, o N é o tamanho do conjunto de treinamento e o y_i é a saída desejada para o padrão de treinamento $\mathbf{x}_i$. A função f satisfaz a $f(\mathbf{x}_i) = 1$, para os padrões $\mathbf{x}_i$ pertencentes a esta categoria, e a $f(\mathbf{x}_i) = -1$, para os $\mathbf{x}_i$ que não pertencem a mesma. Num problema de classificação multi-categoria bastaria utilizar uma SVM para cada categoria, que discrimine entre os padrões dessa categoria e os demais.

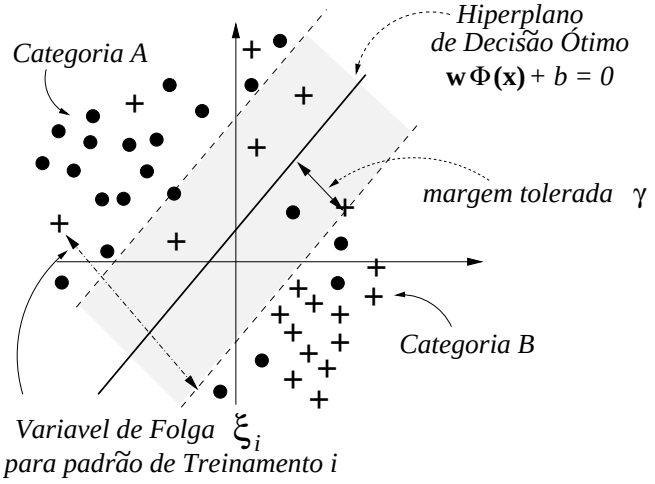


Figura A.1: Hiperplano de decisão ótimo para um problema de classificação não-linearmente separável.

O teorema de *Cover* expõe que, dado um conjunto de padrões não separável linearmente, existe uma transformação não-linear $\Phi(\mathbf{x})$, de $\mathbb{R}^n$, a um espaço de maior dimensão (espaço oculto), no qual o conjunto de padrões tem uma maior probabilidade de ser linearmente separável. Neste espaço de maior dimensão, a SVM seleciona o discriminador linear $f(\mathbf{x}) = sgn(\mathbf{w}\mathbf{x} + b)$ associado ao hiperplano

(definido pelo vetor $\mathbf{w}$) que melhor separa as amostras de treinamento (figura A.1). Portanto, a função de classificação implementada pela SVM no espaço de entrada é:

$$f(\mathbf{x}) = sgn(\mathbf{w}\mathbf{\Phi}(\mathbf{x}) + b) \tag{A.1}$$

A margem γ deste hiperplano no espaço oculto se define como a distância entre o hiperplano e o padrão de treinamento mais próximo ao mesmo. Pela geometria elementar, sabe-se que dado um hiperplano com a equação $\mathbf{w}\mathbf{x} + b = 0$, então a distância de um ponto de treinamento $\mathbf{x}_0$ ao plano (margem) é dada por:

$$\gamma = \frac{|\mathbf{w}\mathbf{x}_0 + b|}{\|\mathbf{w}\|} \tag{A.2}$$

A Teoria da Aprendizagem Estatística enfoca que, para maximizar a capacidade de generalização do classificador linear e minimizar os problemas da superaprendizagem, tem que selecionar o hiperplano com a máxima margem γ. Dado que a equação do hiperplano tem um grau de liberdade inerente (se $\mathbf{w}\mathbf{x} + b = 0$, então $\lambda(\mathbf{w}\mathbf{x} + b) = 0 \forall \lambda \in \mathbb{R}$), se impõe que $|\mathbf{w}\mathbf{x}_i + b| = 1$ para o padrão de treinamento $\mathbf{x}_i$ mais próximo ao hiperplano (denominados vetores de suporte), de modo que a margem seja $\gamma = \frac{1}{||\mathbf{w}||}$, de modo que maximizar γ implica minimizar $||\mathbf{w}||$.Este processo de otimização se deve compatibilizar com a minimização do erro de treinamento. Com este objetivo, para cada padrão de treinamento $\mathbf{x}_i$ se define a sua variável de folga (*slack variable*) denotada por ξ_i, que avalia a medida na qual este padrão está mal classificado pela função de classificação $f(\mathbf{x})$, implementada pela SVM:

$$\xi_i = max\{0, 1 - y_i f(\mathbf{x}_i)\}, i = 1, \ldots, N \tag{A.3}$$

onde a $f(\mathbf{x})$ se definiu na equação A.1. Se o padrão se classifica corretamente e, se encontra fora da margem tolerada (parte sombreada na figura A.1), então a $y_i f(\mathbf{x}_i) \geq 1$, de modo que, a $\xi_i = 0$. Se o padrão se classifica corretamente, mas encontra-se dentro da margem tolerada, então a $0 \leq y_i f(\mathbf{x}_i) < 1$ e, portanto, a $0 < \xi_i \leq 1$. Finalmente, para os padrões classificados incorretamente, a $y_i f(\mathbf{x}_i) < 0$, de modo que, a $\xi_i > 1$.

Para maximizar a capacidade de generalização da SVM, esta tem que maximizar a margem γ, que é inversamente proporcional a norma $\|\mathbf{w}\|$ do vetor que define o hiperplano. Por outro lado, também tem que minimizar o erro de treinamento,

que é dado pela soma das variáveis de folga, para todo os padrões de treinamento. Finalmente, pela própria definição da ξ_i (equação A.3), deve-se cumprir que a $\xi_i \leq 1 - y_i f(\mathbf{x}_i)$ e, portanto:

$$y_i f(\mathbf{x}_i) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, N \tag{A.4}$$

Os padrões de treinamento $\mathbf{x}_i$ mais próximos ao hiperplano separador (denominados vetores de suporte) são os que verificam $\xi_i = 0$ e $y_i f(\mathbf{x}_i) = 1$, de modo que $|\mathbf{w}\Phi(\mathbf{x}_i) + b| = 1$. Portanto maximizar γ implica maximizar $\frac{1}{\|\mathbf{w}\|}$, ou seja, minimizar $\|\mathbf{w}\|$. Por conseguinte, o processo de otimização deve minimizar a soma:

$$\frac{\|\mathbf{w}\|^2}{2} + C \sum_{i=1}^{N} \xi_i, \quad C > 0 \tag{A.5}$$

(onde N é o número de padrões de treinamento) sujeito às restrições (substituimos a $f(\mathbf{x}_i) = \mathbf{w}\mathbf{\Phi}(\mathbf{x}_i) + b$):

$$y_i[\mathbf{w}\mathbf{\Phi}(\mathbf{x}_i) + b] \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, N \tag{A.6}$$

Neste problema de otimização, a C é a denominada "constante de regularização", que controla o peso concedido a ambas componentes da quantidade a minimizar na equação A.5: a norma do vetor do hiperplano (primeiro somando), associado ao erro de generalização, e a soma das variáveis de folga, associada ao erro de treinamento (segundo somando). O fato de minimizar a $\|\mathbf{w}^2\|$ ao invés do $\|\mathbf{w}\|$, é devido as técnicas matemáticas de otimização que estão especialmente orientadas à otimização de funções quadráticas. Para resolver este problema de minimização, condicionada por uma série de inequações lineares, emprega-se o método dos Multiplicadores de *Lagrange* e o "Teorema da Dualidade", transformando-se num problema de maximização da seguinte função:

$$Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{\Phi}(\mathbf{x}_i) \mathbf{\Phi}(\mathbf{x}_j) \tag{A.7}$$

sob às condições:

$$\sum_{i=1}^{N} \alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C, i = 1, \ldots, N \tag{A.8}$$

e sob às condições de *Karush-Kuhn-Tucker* (KKT):

$$\alpha_i[y_i(\mathbf{w}\mathbf{\Phi}(\mathbf{x}) + b) - 1 - \xi_i] = 0, \quad i = 1, \ldots, N \tag{A.9}$$

O processo de otimização deve obter os valores para $\alpha_i, i = 1, \ldots, N$ (multiplicadores de *Lagrange*) que maximizam $Q(\alpha)$. A transformação não-linear $\Phi(\mathbf{x})$, que transforma o espaço original das características no oculto, implicitamente se realiza através de um núcleo $K(\mathbf{x}, \mathbf{y})$ que cumpra as condições do teorema de *Mercer* e, portanto, se pode escrever como $K(\mathbf{x}, \mathbf{y}) = \mathbf{\Phi}(\mathbf{x})\mathbf{\Phi}(\mathbf{y})$. Finalmente, a função discriminadora da SVM pode ser escrita como:

$$f(\mathbf{x}) = sgn\left[\sum_{i=1}^{N} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b\right] \tag{A.10}$$

Os vetores de suporte são os únicos padrões de treinamento que satisfazem a $y_i(\mathbf{w}\mathbf{\Phi}(\mathbf{x}) + b) = 1 - \xi_i$ e, portanto, usando a equação A.9, também são os únicos que satisfazem que $\alpha_i \neq 0$. Porém, $\alpha_i = 0$ para todo os padrões de treinamento que não são vetores de suporte. Portanto, na soma da equação A.10 apenas intervêm os vetores de suporte:

$$f(\mathbf{x}) = sgn\left[\sum_{i \in SV} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b\right] \tag{A.11}$$

onde SV é o conjunto de vetores de suporte. Além do mais, são necessários exatamente $N_s = card(SV) < N$ vetores de treinamento para calcular a $f(\mathbf{x})$. Por outro lado:

$$\mathbf{w} = \sum_{i \in SV} \alpha_i y_i \Phi(\mathbf{x}_i) \tag{A.12}$$

é o vetor diretor do hiperplano de decisão ótimo no espaço oculto, espaço com o qual não se tem que tratar diretamente, porque se encontra implícito através do uso do núcleo K. Por último, o termo independente b calcula-se como:

$$b = \frac{1}{|SV|} \sum_{i \in SV} \left[y_i - \sum_{j \in SV} \alpha_j y_j K(\mathbf{x}_i, \mathbf{x}_j) \right] \tag{A.13}$$

A determinação dos multiplicadores de *Lagrange* α_i, realiza-se através das técnicas interativas de maximização da função objetivo dual $Q(\alpha)$, sendo a otimização seqüencial mínima (*Sequential Minimal Optimization*, SMO) [123] uma das mais usadas.

A SVM se aplica em muitas áreas, entre elas a classificação de padrões, e, normalmente, proporciona resultados muito competitivos comparada com os outros classificadores, algumas vezes, especificamente definidos para o problema em questão. De fato, a SVM se converteu em um dos algoritmos de classificação de referência pelos seus bons resultados. Por esta razão a incluimos em nossos experimentos, para obter uma referência com a qual comparar os resultados obtidos pelas nossas propostas e pelas redes ART.

Apêndice B

Complexidade de problemas de classificação supervisionados

Nesta secção, resumimos brevemente as medidas que utilizamos para avaliar a complexidade dos conjuntos de dados em problemas de classificação, empregados no trabalho experimental. Estas medidas foram obtidas a partir do artigo [89], e nesta secção faremos uma breve descrição das mesmas. As três primeiras ($f1$, $f2$ e $f3$) são medidas das características individuais. As medidas $n2$ e $n3$ avaliam a separabilidade entre as classes. Finalmente, a medida $t1$ é do tipo topológico e, $t2$, é relativa à "maldição da dimensionalidade". Nestas seções, n designa a dimensão do espaço de entrada; é N_c o número de classes e N é o número total de padrões.

B.1 Inversa da razão discriminante de *Fisher* ($f1$)

Para um problema de classificação multi-classe, a razão discriminante de *Fisher* f_{ij} entre as classes i e j se define como:

$$f_{ij} = \max_{k=1,\ldots,n} \frac{(\mu_{ik} - \mu_{jk})^2}{\sigma_{ik}^2 + \sigma_{jk}^2} \tag{B.1}$$

onde o μ_{ik} e o σ_{ik} designam, respectivamente, o valor médio e a variança da classe i, na dimensão k. Em um problema de duas classes, somente se necessita um valor f_{12}, no entanto, para problemas com $N_c > 2$ classes, necessitaremos de uma matriz quadrada de ordem N_c, na qual teremos interesse somente nos valores do seu

triângulo superior ($f_{ij}, j > i$).

No artigo [89], define-se a medida de complexidade $f1$ como o valor máximo de f_{ij} sobre todas as classes i e j. Porém, este é um valor de simplicidade, e não de complexidade, dado que quanto maior seja a razão discriminante, o problema está mais próximo a ser linearmente separável e, portanto, é menos complexo. Por esta razão, nós definiremos $f1$ como a inversa deste máximo, de modo que $f1$ seja efetivamente uma medida de complexidade:

$$f1 = \frac{1}{\max\limits_{i=1,\ldots,N_c} \left\{ \max\limits_{j=i+1,\ldots,N_c} \{f_{ij}\} \right\}} \tag{B.2}$$

Os valores de $f1$ pertencem ao intervalo de $[0, +\infty)$ e, idealmente deveriam ser menores que 1.

B.2 Máxima sobreposição por dimensão entre as classes ($f2$)

Esta medida avalia a sobreposição entre as classes, em cada dimensão. Se denotamos por M_{ik} e m_{ik} ao máximo e mínimo, respectivamente, a característica k para a classe C_i, então a medida $f2_{ij}$ se define em [89] como:

$$\prod_{k=1}^{n} \frac{min[M_{ik}, M_{jk}] - max[m_{ik}, m_{jk}]}{max[M_{ik}, M_{jk}] - min[m_{ik}, m_{jk}]} \tag{B.3}$$

Onde n é a dimensão do espaço de entrada. O volume da sobreposição da $f2_{ij}$, entre as classes i e j, é nulo, se existe uma única dimensão na qual os intervalos das duas classes não se sobreponham. Nos problemas em que a dimensionalidade n é muito elevada, o volume assume o valor quase que nulo, uma vez que a multiplicação de muitos termos, todos eles menores que 1, também alcança um resultado quase que nulo. Por esta razão, ao invés da definição anterior, usaremos a sobreposição mínima sobre todas as dimensões:

$$f2_{ij} = \min_{k=1,\ldots,n} \left\{ \frac{min[M_{ik}, M_{jk}] - max[m_{ik}, m_{jk}]}{max[M_{ik}, M_{jk}] - min[m_{ik}, m_{jk}]} \right\} \tag{B.4}$$

Os valores da $f2_{ij}$ oscilam no intervalo de $[0, 1]$, idealmente deveriam ser muito menores que 1. Em um problema de classificação com mais de duas classes, usaremos como medida de complexidade a máxima sobreposição entre as classes:

$$f2 = \max_{i=1,\ldots,N_c} \left\{ \max_{j=i+1,\ldots,N_c} \{f2_{ij}\} \right\} \tag{B.5}$$

B.3 Inversa da máxima eficiência da característica ($f3$)

Para determinar a eficiência de uma característica na discriminação entre as classes, tem-se que calcular, previamente, os intervalos aos que pertencem os padrões das distintas classes em cada dimensão. Depois, tem-se que calcular o número de padrões n_k para os quais a característica k cai somente no intervalo da classe a que pertence o padrão e que não cai no intervalo das outras classes na dimensão k. A eficiência da característica k é o quociente entre este número n_k e, o número total de padrões (N). A eficiência máxima da característica é a máxima sobre todas as eficiências. Como, quanto maior a eficiência máxima da característica menos complexo é o problema, a medida de complexidade $f3$ é a inversa da eficiência máxima da característica:

$$f3 = \frac{N}{\max\limits_{k=1,\ldots,n} \{n_k\}} \tag{B.6}$$

O valor de $f3$ pertence ao intervalo de $[1, +\infty)$, e é recomendável um valor em torno de 1.

B.4 Quociente entre as distâncias médias inter- e intra-classe ($n2$)

Para determinar $n2$ é calculado, para cada padrão $\mathbf{x}_l$, a distância d_l^+ ao seu vizinho mais próximo pertencente à sua mesma classe $C(\mathbf{x}_l)$ (distância intra-classe), e a distância d_l^- ao seu vizinho mais próximo não pertencente à sua classe (distância

inter-classe). A medida $n2$ é o quociente entre a média dos valores d_l^+ e a média dos d_l^-:

$$d_l^+ = \min_{p=1,\ldots,N}\{\|\mathbf{x}_l - \mathbf{x}_p\| : C(\mathbf{x}_p) = C(\mathbf{x}_l)\} \tag{B.7}$$

$$d_l^- = \min_{p=1,\ldots,N}\{\|\mathbf{x}_l - \mathbf{x}_p\| : C(\mathbf{x}_p) \neq C(\mathbf{x}_l)\} \tag{B.8}$$

$$n2 = \frac{\sum_{l=1}^{N} d_l^+}{\sum_{l=1}^{N} d_l^-} \tag{B.9}$$

Quanto maior sejam as distâncias intra-classes em relação às distâncias inter-classes, ambas em valores médios, mais complexo é o problema. A $n2$ assume valores no intervalo de $[0, +\infty)$, e os valores recomendáveis são muito menores que 1.

B.5 Erro de um classificador do vizinho mais próximo ($n3$)

Esta medida é simplesmente o erro (em tanto por 1) de classificação cometido por um classificador de vizinho mais próximo (1−*Nearest-Neighbour*). Portanto, o intervalo é de $[0, 1]$ e o valor desejável é muito inferior a 1.

B.6 Inverso da ordem média das aderências dos padrões ($t1$)

Esta medida realiza uma descrição interior da classe, ao invés de uma descrição das suas fronteiras. Para isto está baseada no conceito de aderência de um padrão. Para esta definição usa a relação entre dois padrões $\mathbf{x}$ e $\mathbf{y}$:

$$\mathbf{x}\mathcal{R}\mathbf{y} \Leftrightarrow \|\mathbf{x} - \mathbf{y}\| < \epsilon \tag{B.10}$$

e, para cada padrão $\mathbf{x}$, define o seu conjunto fronteira:

$$\Gamma(\mathbf{x}) = \{\mathbf{y} : \mathbf{y}\mathcal{R}\mathbf{x}\} \tag{B.11}$$

Deste modo, a aderência de um conjunto de padrões é uma função $\phi : \mathbb{R}^n \rightarrow \mathbb{R}^n$ definida por:

$$\phi(\varnothing) = \varnothing \tag{B.12}$$

$$\phi(\{\mathbf{x}\}) = \{\mathbf{x}\} \cup \Gamma(\mathbf{x}) \tag{B.13}$$

$$\phi(A) = \bigcup_{\mathbf{x} \in A} \phi(\{\mathbf{x}\}) \tag{B.14}$$

onde A é um conjunto qualquer de padrões. Esta é a aderência de ordem 1. A partir dela, a aderência $\phi^n(\mathbf{x})$ de ordem n de um padrão $\mathbf{x}$ define-se indutivamente como:

$$\phi^n(\mathbf{x}) = \begin{cases} \phi(\{\mathbf{x}\}) & n = 1 \\ \phi(\phi^{n-1}(\{\mathbf{x}\})) & n > 1 \end{cases} \tag{B.15}$$

Para cada padrão $\mathbf{x}_l$, calcula-se a ordem n_l da aderência mais alta que não contém nenhum padrão de outra classe:

$$n_l = \max\{n : \nexists \mathbf{x}_p \in \phi^n(\mathbf{x}_l), p \neq l, C(\mathbf{x}_p) \neq C(\mathbf{x}_l)\} \tag{B.16}$$

onde $C(\mathbf{x})$ é a classe a que pertence o padrão $\mathbf{x}$. Uma ordem n_l mais alta, significa que a classe se ajusta mais a uma estrutura de hiper-esfera, e portanto o problema é mais simples: a ordem é, então, uma medida de simplicidade– e não de complexidade– do problema. Determina-se a ordem máxima média sobre todos os padrões, e a $t1$ se calcula como a inversa dessa ordem máxima média:

$$t1 = \frac{N}{\sum_{l=1}^{N} n_l} \tag{B.17}$$

onde N é o número de padrões. Dado que $n_l \geq 1$ (a ordem mínima é 1 para cada padrão), os valores da $t1$ variam no intervalo de $(0, 1]$, sendo recomendável um valor muito inferior a 1.

B.7 Quociente entre o número de entradas e os padrões ($t2$)

Este valor mede o problema da "Maldição da Dimensionalidade" [11], avaliando o quociente entre o número de entradas n e o número de padrões N:

$$t2 = \frac{n}{N} \tag{B.18}$$

O valor da $t2$ varia no intervalo de $[0, +\infty)$. Idealmente, N deveria ser muito superior a n, e portanto a $t2$ deveria ser muito inferior a 1.

A tabela B.1 resume as medidas implementadas, com os seus intervalos e valores recomendáveis.

Tabela B.1: Medidas de complexidade, descrição, intervalo e valor recomendável.

Medida	Descrição	Intervalo	Recomendável
$f1$	inversa da razão discriminante de *Fisher* máxima entre as classes	$[0, +\infty)$	< 1
$f2$	máxima sobreposição por dimensão entre as classes	$[0, 1]$	$\ll 1$
$f3$	inversa da máxima eficiência de característica	$[1, +\infty)$	~ 1
$n2$	quociente entre as distâncias médias intra- e inter-classes	$[0, +\infty)$	$\ll 1$
$n3$	erro de um classificador de vizinho mais próximo	$[0, 1]$	$\ll 1$
$t1$	inverso da ordem média das aderências dos padrões	$(0, 1]$	$\ll 1$
$t2$	quociente entre o número de entradas e o número de padrões	$(0, +\infty)$	$\ll 1$

Apêndice C

Cálculo do erro de *Bayes*

O erro de *Baye*s para um problema de classificação com C classes (predições) $C_1, \ldots, C_{N_c}$ é dado pela seguinte expressão [142]:

$$E_B = \sum_{i=1}^{N_c} p(C_i) \int_{\mathbb{R}^n \setminus \Omega_i} p(\mathbf{x}|C_i) d\mathbf{x} \tag{C.1}$$

Na qual:

- $p(C_i)$ é a probabilidade a priori da classe C_i. Esta probabilidade pode ser calculada como $p(C_i) = N_i/N$, onde N_i é o número de padrões pertencentes a C_i e, N é o número total de padrões.
- Ω_i é o recinto de C_i, quer dizer, o conjunto de pontos $\mathbf{x} \in \mathbb{R}^n$ que o classificador de *Bayes* codifica por C_i. O classificador de *Bayes* codifica cada padrão $\mathbf{x}$ pela classe que maximiza a sua probabilidade a posteriori $p(C_i|\mathbf{x})$, que se calcula como:

  $$p(C_i|\mathbf{x}) = \frac{p(\mathbf{x}|C_i)p(C_i)}{p(\mathbf{x})} \tag{C.2}$$

  Usualmente, $p(\mathbf{x}) = 1$ e portanto $p(C_i|\mathbf{x}) = p(\mathbf{x}|C_i)p(C_i)$. Então Ω_i é a região do espaço na que a probabilidade a posteriori de C_i é máxima:

  $$\Omega_i = \{\mathbf{x} \in \mathbb{R}^n : p(C_i|\mathbf{x}) > p(C_j|\mathbf{x}), \forall j \neq i\} \tag{C.3}$$

- $\mathbb{R}^n \setminus \Omega_i$, é a parte de $\mathbb{R}^n$ externa a Ω_i, quer dizer, o conjunto complemento de Ω_i.

A integral:

$$\int_{C(\Omega_i)} p(\mathbf{x}|C_i)d\mathbf{x} \tag{C.4}$$

deve ser calculada a partir dos padrões do conjunto de teste, conhecendo as classes às que pertencem. Esta integral representa a população dos padrões pertencentes à classe C_i (ou seja, padrões $\mathbf{x}$ para os quais $p(\mathbf{x}|C_i) > 0$) que se localizam fora de Ω_i, a região codificada pelo classificador de *Bayes* à C_i. Se denotamos por $\mathbf{x}_j$ a cada padrão de teste, e por $C(\mathbf{x}_j)$ à classe a que este pertence, a integral C.4 pode ser calculada como o cardinal do conjunto $\mathcal{B}_i$, definido como:

$$\mathcal{B}_i = \{\mathbf{x}_j : C(\mathbf{x}_j) = i, \mathbf{x}_j \notin \Omega_i\} \tag{C.5}$$

ou equivalentemente:

$$\mathcal{B}_i = \left\{\mathbf{x}_j : C(\mathbf{x}_j) = i, p(C_i|\mathbf{x}_j) < \max_k\{p(C_k|\mathbf{x}_j)\}\right\} \tag{C.6}$$

Portanto, o erro de *Bayes* pode ser escrito como:

$$E_B = \sum_{i=1}^{N_c} p(C_i)|\mathcal{B}_i| \tag{C.7}$$

onde $|\mathcal{B}_i|$ designa o cardinal ou o número de elementos de $\mathcal{B}_i$. No caso particular de que não exista sobreposição entre as diversas classes, então não haverá nenhum padrão situado fora de Ω_i pertencente à C_i, de modo que $\mathcal{B}_i = \varnothing, \forall i$ e portanto $E_B = 0$.

Para obter o conjunto $\mathcal{B}_i, i = 1, \ldots, C$, temos que calcular:

$$p(C_i|\mathbf{x}) = \frac{N_i}{N} p(\mathbf{x}|C_i) \tag{C.8}$$

(onde se utilizou a equação C.2 e que $p(C_i) = N_i/N$) e, portanto, necessitamos conhecer $p(\mathbf{x}|C_i)$. Nos problemas 4G1-4G3, as classes são 4 distribuições gaussianas, de modo que $p(\mathbf{x}|C_i)$ é determinada pela equação:

$$p(\mathbf{x}|C_i) = exp\left[-\frac{\|\mathbf{x}-\mathbf{c}_i\|^2}{\sigma_i^2}\right] \quad i = 1,\ldots,4 \tag{C.9}$$

onde $\mathbf{c}_i$ é o centro da classe i. No caso do problema CIS-noise, $p(\mathbf{x}|C_1)$ é dada pela equação:

$$p(\mathbf{x}|C_1) = \begin{cases} 1 & \|\mathbf{x}-\mathbf{c}\| \leq r \\ 0 & \text{em outro caso} \end{cases} \qquad p(\mathbf{x}|C_2) = 1 - p(\mathbf{x}|C_1)$$

onde r é o raio do círculo e $\mathbf{c} = (1/2, 1/2)$ é o centro do quadrado unidade. No caso do problema T5-noise, $p(\mathbf{x}|C_i)$ é determinada pela equação:

$$p(\mathbf{x}|C_i) = \begin{cases} 1 & r_{i-1} \leq \|\mathbf{x}-\mathbf{c}\| < r_i \\ 0 & \text{em outro caso} \end{cases} \qquad i = 1,\ldots,6$$

onde r_i é o raio da coroa circular correspondente à classe C_i para $i = 1,\ldots,5$, sendo $r_0 = 0$ e $r_6 = 1/2$.

Apêndice D

Equação do hiperplano definido por $n+1$ vetores

Considere h_{ijk} ser o hiperplano em $\mathbb{R}^n$ definido pelos vetores de pesos $\mathbf{w}_{ijk1}, \ldots, \mathbf{w}_{ijkn}$, e $g_{ijk}(\mathbf{x}) = 0$ a equação de h_{ijk}. Da geometria elementar, sabe-se que todos os pontos (padrões de entrada) $\mathbf{x} \in h_{ijk}$ são uma combinação de vetores no conjunto $\{\mathbf{w}_{ijk1}\} \cup \{\mathbf{w}_{ijkl} - \mathbf{w}_{ijk1}, l = 2, \ldots, n\}$, de modo que $\exists \lambda_2, \ldots, \lambda_n \in \mathbb{R}$ que satisfazem:

$$\mathbf{x} = \mathbf{w}_{ijk1} + \sum_{l=2}^{n} \lambda_l (\mathbf{w}_{ijkl} - \mathbf{w}_{ijk1}) \tag{D.1}$$

Conseqüentemente, o conjunto de vetores $\{\mathbf{x} - \mathbf{w}_{ijk1}, \mathbf{w}_{ijk2} - \mathbf{w}_{ijk1}, \ldots, \mathbf{w}_{ijkn} - \mathbf{w}_{ijk1}\}$ é dependente linearmente, e eles compõem uma matriz singular, cujo determinante vale 0:

$$\begin{vmatrix} x_1 - w_{ijk1,1} & \ldots & x_n - w_{ijk1,n} \\ w_{ijk2,1} - w_{ijk1,1} & \ldots & w_{ijk2,n} - w_{ijk1,n} \\ \ldots & & \\ w_{ijkn,1} - w_{ijk1,1} & \ldots & w_{ijkn,n} - w_{ijk1,n} \end{vmatrix} = 0 \tag{D.2}$$

Esta é a equação do hiperplano h_{ijk}. Nesta equação, o $w_{ijkl,m}$ designa ao m-ésimo componente de $\mathbf{w}_{ijkl}$. A eq. D.2 coincide com a definição de $\phi_{ijk}(\mathbf{x})$ na eq. 3.4 (subseção 3.1.1). Por outro lado, $g_{ijk}(\mathbf{x}) > 0$ deve implicar que $\mathbf{x}$ cai dentro do simplex S_{ij}, e $g_{ijk}(\mathbf{x}) < 0$ que $\mathbf{x}$ cai fora do S_{ij}. Considere $\widehat{\mathbf{w}}_{ijk}$ ser o vértice de S_{ij} que não pertence a h_{ijk}, e $sgn(x)$ a função sinal de $\mathbf{x}$, definida pela eq. 3.3

(subseção 3.1.1). Então, deve-se verificar que $sgn(\phi_{ijk}(\mathbf{x})) = sgn(\phi_{ijk}(\widehat{\mathbf{w}}_{ijk}))$, se o padrão de entrada $\mathbf{x}$ cai dentro do simplex, e $sgn(\phi_{ijk}(\mathbf{x})) \neq sgn(\phi_{ijk}(\widehat{\mathbf{w}}_{ijk}))$, quando $\mathbf{x}$ cai fora do simplex. Para cumprir esta condição, a equação do hiperplano h_{ijk} pode ser definida como:

$$g_{ijk}(\mathbf{x}) \equiv sgn(\phi_{ijk}(\widehat{\mathbf{w}}_{ijk}))\phi_{ijk}(\mathbf{x}) = 0 \tag{D.3}$$

que coincide com a eq. 3.5. Deste modo, se $\mathbf{x}$ cai dentro do simplex, $sgn(\phi_{ijk}(\mathbf{x})) = sgn(\phi_{ijk}(\widehat{\mathbf{w}}_{ijk}))$ e, $g_{ijk}(\mathbf{x}) > 0$, sendo $g_{ijk}(\mathbf{x}) < 0$, se $\mathbf{x}$ cai fora do simplex.

Apêndice E

Cálculo do vetor diretor de um hiperplano

A equação do hiperplano h_{ijk} é $g_{ijk}(\mathbf{x}) = 0$ (eq. 3.5 na secção 3.1.1). Para calcular seu vetor diretor, que denotaremos por $\mathbf{w}^*_{ijk}$, esta equação deve se escrever da seguinte forma:

$$\mathbf{w}^*_{ijk}\mathbf{x} + b^*_{ijk} = 0 \tag{E.1}$$

Portanto, deve ser a $g_{ijk}(\mathbf{x}) = \mathbf{w}^*_{ijk}\mathbf{x} + b^*_{ijk}$, e temos que obter o $\mathbf{w}^*_{ijk}$ e o b^*_{ijk} a partir da $g_{ijk}(\mathbf{x})$. A equação $g_{ijk}(\mathbf{x}) = 0$ implica que $\phi_{ijk}(\mathbf{x}) = 0$ (se $\phi_{ijk}(\mathbf{x}) \neq 0$, então $sgn(\phi_{ijk}(\mathbf{x})) \neq 0$ e, $g_{ijk}(\mathbf{x}) \neq 0$ pela eq. 3.4). Desenvolvendo o determinante $\phi_{ijk}(\mathbf{x})$ da eq. 3.4, pelos adjuntos da primeira linha obtemos que:

$$\begin{aligned} \phi_{ijk}(\mathbf{x}) &= \sum_{l=1}^{n}(-1)^{l+1}(x_l - w_{ijk1,l})D_{ijkl} = \\ &= \sum_{l=1}^{n}(-1)^{l+1}D_{ijkl}x_l - \sum_{l=1}^{n}(-1)^{l+1}w_{ijk1,l}D_{ijkl} \end{aligned} \tag{E.2}$$

onde D_{ijkl} é o determinante da matriz na eq. 3.4, depois de remover a linha 1 e a coluna l:

$$D_{ijkl} \equiv \begin{vmatrix} \alpha_{21} & \dots & \alpha_{2,l-1} & \alpha_{2,l+1} & \dots & \alpha_{2n} \\ & \dots & & \dots & & \dots \\ \alpha_{n1} & \dots & \alpha_{n,l-1} & \alpha_{n,l+1} & \dots & \alpha_{nn} \end{vmatrix} \tag{E.3}$$

$$\alpha_{pq} \equiv w_{ijkp,q} - w_{ijk1,q} \quad p = 2, \dots, n; q = 1, \dots, n, q \neq l \tag{E.4}$$

A expressão E.2 deve se igualar ao lado esquerdo da eq. E.1, de modo que obtemos:

$$\sum_{l=1}^{n} (-1)^{l+1} D_{ijkl} x_l - \sum_{l=1}^{n} (-1)^{l+1} w_{ijk1,l} D_{ijkl} = \mathbf{w}^*_{ijk}\mathbf{x} + b^*_{ijk} \tag{E.5}$$

Nesta equação, igualando os termos, obtemos que:

$$\mathbf{w}^*_{ijk} = \sum_{l=1}^{n} (-1)^{l+1} D_{ijkl} \mathbf{e}_l \tag{E.6}$$

$$b^*_{ijk} = -\sum_{l=1}^{n} (-1)^{l+1} w_{ijk1,l} D_{ijkl} \tag{E.7}$$

onde o $\mathbf{e}_l$ designa o vetor unitário ($\|\mathbf{e}_l\| = 1$) na dimensão l. Finalmente, a norma de $\mathbf{w}^*_{ijk}$ é dada por:

$$\|\mathbf{w}^*_{ijk}\| = \sqrt{\sum_{l=1}^{n} D^2_{ijkl}} \tag{E.8}$$

A norma do hiperplano do vetor diretor $\mathbf{w}^*_{ijk}$ é usada para calcular a distância entre um padrão de entrada $\mathbf{I}$ e o hiperplano h_{ijk} (eq. 3.10 na subseção 3.1.1).

Apêndice F

Implementação geométrica do Teste de Sobreposição

A sobreposição entre os simplexes e os segmentos de linhas, conectando o padrão de entrada e os vértices das categorias politopo, é testado avaliando as intersecções entre eles, mediante as técnicas de geometria analítica. Para o teste, se um vetor de peso é "conectável" ao padrão de entrada, o segmento de linha entre eles não pode se sobrepor com outros simplexes. Por outro lado, para criar um simplex novo, ou substituir um vetor de peso pelo padrão de entrada, o simplex novo ou modificado, não se pode sobrepor com outras categorias. Nas seguintes seções descreveremos os dois casos.

F.1 A sobreposição entre o segmento de linha e o simplex

O segmento de linha $\overrightarrow{\mathbf{w}_1, \mathbf{w}_2}$ e o simplex S_{ij} na categoria C_i são sobrepostos, somente se o segmento cruza com algum hiperplano do simplex (observe que os vetores de pesos podem ser somente os vértices do simplex, logo eles não podem cair dentro de um simplex). Podemos testar se existe a intersecção entre o segmento $\overrightarrow{\mathbf{w}_1, \mathbf{w}_2}$ e o hiperplano h_{ijk} do simplex S_{ij}, definido pelos vetores $\{\mathbf{w}_{ijk1}, \ldots, \mathbf{w}_{ijkn)}\}$, resolvendo para $\alpha^s, \alpha_2^h, \ldots, \alpha_n^h$, a equação vetorial seguinte (obtida a partir da equação paramétrica da linha reta associada ao segmento e da equação paramétrica do hiperplano):

$$\mathbf{w}_1 + \alpha^s(\mathbf{w}_2 - \mathbf{w}_1) = \mathbf{w}_{ijk1} + \sum_{l=2}^{n} \alpha_l^h(\mathbf{w}_{ijkl} - \mathbf{w}_{ijk1}) \tag{F.1}$$

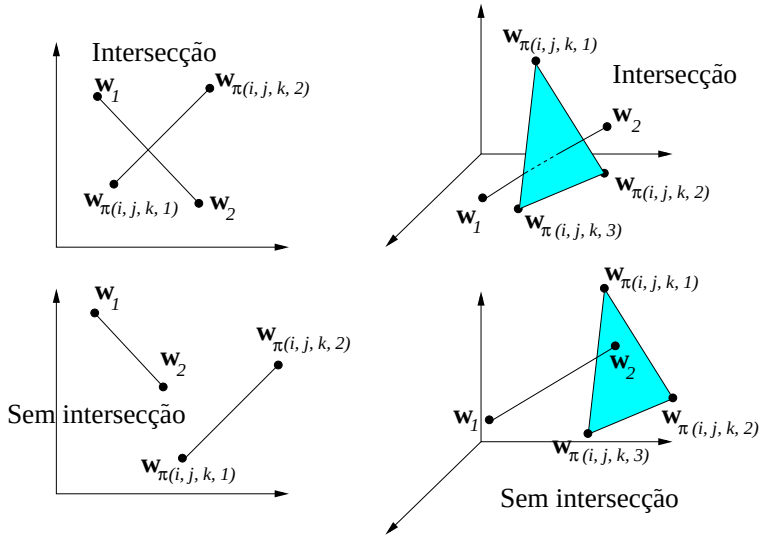


Figura F.1: Exemplos em $\mathbb{R}^2$ e $\mathbb{R}^3$ da intersecção e não-intersecção entre os segmentos $\mathbf{w}_1 - \mathbf{w}_2$ e os hiperplanos $(\mathbf{w}_{ijk1}, \mathbf{w}_{ijk2})$ em $\mathbb{R}^2$ e $(\mathbf{w}_{ijk1}, \mathbf{w}_{ijk2}, \mathbf{w}_{ijk3})$ em $\mathbb{R}^3$.

A solução $(\alpha^s, \alpha_2^h, \ldots, \alpha_n^h)$ do sistema de n equações é um ponto de intersecção válido, somente se ele cai dentro da área demarcada pelo hiperplano, cuja fronteira é definida pelos n vetores $\mathbf{w}_{ijk1}, \ldots, \mathbf{w}_{ijkn}$ de h_{ijk}, e se ele cai no segmento entre $\mathbf{w}_1$ e $\mathbf{w}_2$ (figura F.1). Isto impõe as seguintes condições para a solução da eq. F.1:

$$0 < \alpha^s < 1 \tag{F.2}$$

$$0 < \alpha_p^h + \alpha_q^h < 1 \qquad \forall p, q = 2, \ldots, n; p \neq q \tag{F.3}$$

Se $\exists \alpha^s, \alpha_2^h, \ldots, \alpha_n^h \in \mathbb{R}$ satisfazendo as eqs. F.1-F.3, o segmento $\overrightarrow{\mathbf{w}_1, \mathbf{w}_2}$ e o hiperplano h_{ijk} se cruzam. Podemos definir a função I_{sh} (intersecção segmento-hiperplano), da seguinte maneira:

$$I_{sh}(\mathbf{w}_1, \mathbf{w}_2, h_{ijk}) = \begin{cases} 1 & \text{eqs. F.1-F.3 têm uma solução} \\ 0 & \text{senão} \end{cases} \tag{F.4}$$

$I_{sh}(\mathbf{w}_1, \mathbf{w}_2, h_{ijk}) = 1$, se o segmento de linha e o hiperplano se cruzam e, $I_{sh}(\mathbf{w}_1, \mathbf{w}_2, h_{ijk}) = 0$, em caso contrário. Finalmente, a sobreposição entre o segmento de linha $\overrightarrow{\mathbf{w}_1, \mathbf{w}_2}$ e o simplex S_{ij} é dado pela função $O_{ls}(\mathbf{w}_1, \mathbf{w}_2, S_{ij})$, definida da seguinte forma:

$$O_{ls}(\mathbf{w}_1, \mathbf{w}_2, S_{ij}) = \begin{cases} 1 & \exists h_{ijk} \in S_{ij} : \\ & \quad I_{sh}(\mathbf{w}_1, \mathbf{w}_2, h_{ijk}) = 1 \\ 0 & \quad \text{senão} \end{cases} \tag{F.5}$$

Finalmente, é possível determinar se um vetor de peso $\mathbf{w}$ é conectável com o padrão de entrada $\mathbf{I}$, sem sobreposição com as outras categorias, usando a função $C(\mathbf{w}, \mathbf{I})$, definida como:

$$C(\mathbf{w}, \mathbf{I}) = \begin{cases} 0 & \nexists i, j : O_{ls}(\mathbf{w}, \mathbf{I}, S_{ij}) = 1 \\ 1 & \quad \text{senão} \end{cases} \tag{F.6}$$

De acordo com esta definição, $C(\mathbf{w}, \mathbf{I}) = 1$, se o segmento $\overrightarrow{\mathbf{w}, \mathbf{I}}$ não intersecciona com nenhum outro simplex S_{ij} e, portanto, $\mathbf{w}$ e $\mathbf{I}$ são conectáveis sem sobreposição. Se, ao contrário, existe algum simplex S_{ij} que se intersecciona com o segmento $\overrightarrow{\mathbf{w}_1, \mathbf{w}_2}$, então o $\mathbf{w}$ e o $\mathbf{I}$ não são conectáveis e $C(\mathbf{w}, \mathbf{I}) = 0$.

F.2 A sobreposição entre o simplex e a categoria

Se a categoria C_j é uma categoria mono-vetor ($N_j^s = 0$) dada pelo vetor $\mathbf{w}_j$, o simplex S_{ik} e a categoria C_j se sobrepõem se a $T_{ik}(\mathbf{w}_j) = 1$. Por sua vez, se a C_j é uma categoria politopo ($N_j^s \geq 1$), existe sobreposição somente se existem simplexes $S_{ik} \in C_i$ e $S_{jl} \in C_j$ sobrepostos. Considere a função O_{ss} (a sobreposição simplex-simplex) ser definida por $O_{ss}(S_{ik}, S_{jl}) = 1$, se os simplexes S_{ik} e S_{jl} se sobrepõem e, senão, $O_{ss}(S_{ik},\ S_{jl}) = 0$. Como os vetores de pesos não podem estar dentro de um simplex, os simplexes S_{ik} e S_{jl} se sobrepõem somente se um segmento de linha de S_{ik} e um hiperplano de S_{jl} se cruzam. Especificamente, se $\exists p, q \in \{1, \ldots, n+1\}, p < q$, tais que, $O_{ls}(\mathbf{w}_{ikp}, \mathbf{w}_{ikq},\ S_{jl}) = 1$ (O_{ls} é definida na eq. F.5), então o segmento de linha $\overrightarrow{\mathbf{w}_{ikp}, \mathbf{w}_{ikq}}$ e o simplex S_{jl} se cruzam, logo $O_{ss}(S_{ik}, S_{jl}) = 1$. Observe que se um segmento de linha do S_{jl} e um hiperplano do S_{ik} se cruzam, então algum segmento de linha do S_{ik} se cruzam com um hiperplano do S_{jl}, que é o caso anterior. A função

de sobreposição $O_{ss}(S_{ik}, S_{jl})$ entre os simplexes S_{ik} e S_{jl} é dado por:

$$O_{ss}(S_{ik}, S_{jl}) = \begin{cases} 1 & \exists p, q = 1, \ldots, n+1, p < q : \\ & \qquad O_{ls}(\mathbf{w}_{ikp}, \mathbf{w}_{ikq}, S_{jl}) = 1 \\ 0 & \qquad \text{senão} \end{cases} \tag{F.7}$$

Em resumo, a sobreposição entre o simplex S_{ik} e a categoria C_j é dado pela função $O_{sc}(S_{ik}, C_j)$, definida por:

$$O_{sc}(S_{ik}, C_j) = \begin{cases} \psi(T_{ik}(\mathbf{w}_j)) & N_j^s = 0 \\ \max\limits_{l=1,\ldots,N_j^s} \{O_{ss}(S_{ik}, S_{jl})\} & N_j^s \geq 1 \end{cases} \tag{F.8}$$

onde a função $\psi(x)$ é definida como:

$$\psi(x) = \begin{cases} 1 & x = 1 \\ 0 & x < 1 \end{cases} \tag{F.9}$$

Deste modo, se a C_j é uma categoria mono-vetor, ela se sobrepõe com o simplex S_{ik}, se a $T_{ik}(\mathbf{w}_j) = 1$, de modo que $\psi(T_{ik}(\mathbf{w}_j)) = 1$. Se, ao contrário, a C_j não é uma categoria mono-vetor ($N_j^s \geq 1$), então ela se sobrepõe com o simplex S_{ik}, se algum dos seus simplexes se sobrepõe com o S_{ik}.

Apêndice G

Número de parâmetros em PTAM

Como cada categoria em PTAM tem um número variável de simplexes, o seu número de vetores de peso (vértices) é também variável. Se PTAM tiver N_c categorias, N_s simplexes e N_w vetores de peso, o número N_p de parâmetros que ele armazena é o seguinte:

- Para cada categoria, os índices do seus simplexes. Considerando que a categoria C_i tem N_i^s simplexes, o número de parâmetros para armazenar para as N_c categorias são $\sum_{i=1}^{N_c} N_i^s = N_s$, porque cada simplex pertence a exatamente uma categoria e existem N_s simplexes.
- Para cada simplex, os índices dos seus vetores de peso. Como cada simplex tem $n+1$ vetores de peso, e existem N_s simplexes, o número de parâmetros é $(n+1)N_s$ (observe que $(n+1)N_s > N_w$, uma vez que os vetores podem ser compartilhados entre simplexes).
- Para cada vetor de peso, as suas n componentes. Como existem N_w vetores de peso, o número de parâmetros é nN_w.

Finalmente, o número total de parâmetros fornecidos por PTAM é:

$$N_p^{PTAM} = (n+2)N_s + nN_w \tag{G.1}$$

Comparativamente, o número de parâmetros usados por FAM é $N_p^{FAM} = 2nN_c$ (cada categoria interna é definida por um vetor de peso de $2n$-dimensão, devido a

codificação em complemento). Finalmente, cada categoria interna em DAM ([36], p. 804) é definido por $4n$ pesos $\{\tau_{ij}, \tau_{ji}; i, j = 1, \ldots, 2n\}$ e um contador de instância c_j, portanto o número de parâmetros em DAM é $N_p^{DAM} = (4n + 1)N_c$.

Apêndice H

Pseudocódigo do algoritmo SAM

Neste apêndice apresentamos o pseudocódigo do algoritmo *Simplex* ARTMAPnas etapas de treinamento e processamento usando a notação simplificada da tabela H.1.

n	Dimensão do espaço de entrada
$\mathbf{I}$	Padrão de entrada $\in [0,1]^n$
P_d	Predição desejada para o padrão de entrada $\mathbf{I}$
C_i	i-ésima categoria politopo
$P(C_i)$	Predição associada a i-ésima categoria
$\mathbf{w}_{ij}$	Vetor de pesos j-ésimo da categoria C_i
T_i^t	Função de Escolha de treinamento da i-ésima categoria
T_i^p	Função de Escolha de processamento da i-ésima categoria
N_c	Número de categorias internas
Γ	Limiar para as Funções de Escolha de Categoria no treinamento

Tabela H.1: Notação adotada para o pseudocódigo do algoritmo SAM.

H.1 Etapa de Treinamento

Inicio
`inicializa()`
Calcular $T_i^t, i = 1, \ldots, N_c$ (**Funções de escolha de categoria**, eq. 2.5)
Repetir

Seja I tal que $T_I^t \geq T_i^t, i = 1, \ldots, N_c, i \neq I, T_i^t > \Gamma, r_I = 0$; $I = -1$ se $\nexists i$ nestas condições
Se $I \neq -1$
Se $P(C_I) == P_d$ (**Teste de Predição**):
Codificar **I** a C_I (**Ressonância**)
`fim_classificacao` $= 1$
Senão
Dividir C_I em N_I categorias, cada uma com um vetor $\mathbf{w}_{Ij}, j = 1, \ldots, N_I$
Rejeitar categoria I: $r_I = 1$
Fim se
Senão
`seleciona_n_vetores_proximos()`
`expansao_categoria()`
Fim se
Ate que `fim_classificacao == 1`
Fim

Subprograma `inicializa()`
`fim_classificacao = 0`
Repetir para $i = 1, N_c$:
$r_i = 0$
Fim repetir
Fim subprograma

Subprograma `seleciona_n_vetores_proximos()`
Seja $E = \{\mathbf{w}_{ij} : P(C_i) == P_d\}$
$F = \varnothing$, $k = 0$, `mesma_categoria` $= 1$, $I = -1$
Repetir:
$d_{min} = \infty$
Repetir para $\mathbf{w}_{ij} \in E$:
$d = \|\mathbf{w}_{ij} - \mathbf{I}\|$
Se $d < d_{min}$:
$d_{min} = d$
$\mathbf{z} = \mathbf{w}_{ij}$
Se $I == -1$
$I = i$
Fim se

$I' = i$
Fim se
Fim repetir
$F = F \cup \{\mathbf{z}\}$
$E = E \setminus \{\mathbf{z}\}$
Se `mesma_categoria` $==$ 1 e $I' \neq I$
`mesma_categoria` $= 0$
Fim se
$k = k + 1$
Ate que $E == \varnothing$ ou $k == n$
$E = F$
Fim subprograma

Subprograma `expansao_categoria()`
Se $card(E) == 0$
`cria_categoria_monovetor()`
Senão
Se `mesma_categoria` $==$ 1 e $N_I < n + 1$
Adicionar um vetor $\mathbf{w}_{I(N_I+1)} = \mathbf{I}$ a C_I
Se `comprova_sobreposicao_predicao()` $==$ 1:
Eliminar vetor $\mathbf{w}_{I(N_I+1)}$ da categoria C_I
`cria_categoria_monovetor()`
Fim se
Senão
`cria_nova_categoria()`
Fim se
Fim se
`fim_classificacao = 1`
Fim subprograma

Subprograma `cria_nova_categoria()`
Sexa $I = N_c + 1$
Criar uma categoria nova C_I com os vetores do conjunto $E \cup \{\mathbf{I}\}$
Se `comprova_sobreposicao_predicao()` $==$ 1
Eliminar categoria C_I
`cria_categoria_monovetor()`

Fim se
Fim subprograma

Subprograma `comprova_sobreposicao_predicao()`
Repetir para $j = 1, \ldots, N_c; j \neq I$
  Se $P(C_j) == P(C_I)$ e $O(C_I, C_j) == 1$ (eq. 2.8):
    Retornar 1
  Fim se
Fim repetir
Retornar 0
Fim subprograma

Subprograma `cria_categoria_monovetor()`
Criar uma categoria nova C_I com $I = N_c + 1$
$N_I = 1$
$\mathbf{w}_{I1} = \mathbf{I}$
$P(C_I) = P_d$
Fim subprograma

H.2 Etapa de Processamento

Inicio
Calcular $T_i^p(\mathbf{I}), i = 1, \ldots, N_c$ (**Função de escolha de categoria**, eq. 2.11)
Codificar o padrão de entrada pela categoria interna C_I com $I = argmax\{T_i^p\}_{i=1}^{N_c}$
Proporcionar como saida a predição $P(C_I)$
Fim

Apêndice I

Pseudocódigo do algoritmo PTAM

Neste apêndice apresentamos o pseudocódigo do algoritmo PolyTope ARTMAP nas etapas de treinamento e processamento usando a notação simplificada da tabela I.1.

n	Dimensão do espaço de entrada
$\mathbf{I}$	Padrão de entrada $\in [0,1]^n$
P_d	Predição desejada para o padrão de entrada $\mathbf{I}$
$\mathbf{w}_k$	k-ésimo vetor de pesos
C_i	i-ésima categoria politopo
$P(C_i)$	Predição associada a i-ésima categoria
T_i	Ativação da i-ésima categoria
S_{ij}	j-ésimo simplex da i-ésima categoria politopo
T_{ij}	Ativação do simplex S_{ij}
$\mathfrak{C}$	Conjunto de toda as categorias C_i
N_c	Número de categorias = $card(\mathfrak{C})$
$\mathbf{w}_i$	Vetor de pesos da categoria mono-vetor C_i

Tabela I.1: Notação adotada para o pseudocódigo do algoritmo PTAM.

I.1 Etapa de Treinamento

Inicio
`inicializa()`
Calcular $T_i^t, i = 1, \ldots, N_c$ (**Funções de escolha de categoria**, eqs. 3.1, 3.13)
Repetir
  Seja I tal que $T_I \geq T_i, i = 1, \ldots, N_c, r_I = 0$; $I = -1$ se $\nexists i$ nestas condições
  Se $I \neq -1$
    Se $P(C_I) \neq P_d$ (**Teste de Predição**):
      Se $T_I == 1$
        Seja S_{Ij} o simplex de C_I tal que $T_{Ij}(\mathbf{I}) = 1$
        `suprime_simplex()`
      Fim se
      Rejeitar categoria I: $r_I = 1$
    Senão:
      Se `teste_sobreposicao() == 1` (**Teste de sobreposição**)
        Rejeitar categoria I: $r_I = 1$
      Senão
        `ressonancia()` (**Ressonância**)
      Fim se
    Fim se
  Senão
    `cria_nova_categoria()`
  Fim se
Ate que `fim_classificacao == 1`
Finalizar

Subprograma `inicializa()`
`fim_classificacao = 0`
`tipo_expansao = expansao_nula`
Repetir para $i = 1, N_c$
  $r_i = 0$
Fim repetir
Se $\mathbf{I}$ e um vetor pendente de classificar
  Suprimir a categoria mono-vetor C_i tal que $\mathbf{w}_i = \mathbf{I}$
Fim se
Fim subprograma

Subprograma `suprime_simplex()`
Se **I** não e um vetor pendente:
  Repetir para $\mathbf{w}_k \in S_{Ij}$:
    Se $\nexists S_{Il}, l \neq j : \mathbf{w}_k \in S_{Il}$:
      Criar categoria nova mono-vetor C_{N_c+1} com $\mathbf{w}_{(N_c+1)} = \mathbf{w}_k$
      Adicionar o vetor $\mathbf{w}_k$ a lista de vetores pendentes de classificar
    Fim se
  Fim repetir
Fim se
Suprimir o simplex S_{Ij}
Fim subprograma

Subprograma `teste_sobreposicao()`
`tipo_expansao = expansao_nula`
`sobreposicao = 1`
Se $T_I == 1$
  `sobreposicao = 0`
Senão
  $A =$ `busca_vetores_conectaveis`(C_I)
  Se $card(A) == n$
    `cria_novo_simplex()`
  Senão se $card(A) > n$
    `substitui_vetor()`
  Fim se
Fim se
Retornar `sobreposicao`
Fim subprograma

Subprograma `busca_vetores_conectaveis`(X)
Seja $Y = \varnothing$
Repetir para $\mathbf{w}_j \in X$:
  Se `comprova_sobreposicao_segmento`($\mathbf{I}$, $\mathbf{w}_j$, $\mathfrak{C}$) `== 0` :
    $Y = Y \cup \{\mathbf{w}_j\}$
  Fim se
Fim repetir
Retornar Y

Fim subprograma

Subprograma `cria_novo_simplex()`
Se `descarta_simplex_agudo(` $A \cup \{\mathbf{I}\}$ `) == 1`
  `sobreposicao = 1`
Se não
  Se `comprova_sobreposicao(`$A \cup \{\mathbf{I}\}$, $\mathfrak{C}$`) == 0`:
    Criar um simplex novo S^* com os vetores do conjunto A
    `sobreposicao = 0`
    `tipo_expansao = criacao_novo_simplex`
  Senão
    `sobreposicao = 1`
  Fim se
Fim se
Fim subprograma

Subprograma `substitui_vetor()`
$J = -1$; $d_{min} = \infty$
Repetir para $\mathbf{w}_j \in A$
  Se `criterio_substituicao_vetor(`$\mathbf{w}_j$`) == 1`
    $d = \|\mathbf{I} - \mathbf{w}_j\|$
    Se $d < d_{min}$
      $d_{min} = d$
      $J = j$
    Fim se
  Fim se
Fim repetir
Se $J == -1$
  `sobreposicao = 1`
Senão
  Seja $X = \{S_{Il} \in C_I : \mathbf{w}_J \in S_{Il}\}$
  `sobreposicao = 0`
  Repetir para $S_{Il} \in X$:
    Criar simplex novo $S^* = S_{Il} \setminus \{\mathbf{w}_J\} \cup \{\mathbf{I}\}$
    Se `comprova_sobreposicao(`S^*, $\mathfrak{C} \setminus C_I$`) == 1`:
      `sobreposicao = 1`

```
            Fim repetir
        Fim se
        Suprimir simplex S*
    Fim repetir
    Se sobreposicao == 0
        tipo_expansao = substituicao_vetor
    Fim se
Fim se
Fim subprograma
```

```
Subprograma descarta_simplex_agudo(S)
Seja S = {w_1, ... w_{n+1}}
Repetir para k = 1, n + 1
    Repetir para j = 1, n + 1
        Se j ≠ kel ≠ j
            Repetir para l = 1, n + 1
                Se l ≠ k
                    θ = arccos|(w_k − w_j)(w_k − w_l)|/(‖w_k − w_j‖‖w_k − w_l‖)
                    Se θ < θ_min
                        Retornar 1
                    Fim se
                Fim se
            Fim repetir
        Fim se
    Fim repetir
Fim repetir
Retornar 0
Fim subprograma
```

```
Subprograma criterio_substituicao_vetor(w)
Seja X = {S_Ij : w ∈ S_Ij}
Repetir para S_Ij ∈ X:
    exterior = 1
    Repetir para w_k ∈ S_Ij, w_k ≠ w
        Se I e interior ao hiperplano definido pelo conjunto de vetores S_Ij \ {w_k}
            exterior = 0
```

```
            Fim repetir
        Fim se
    Fim repetir
    Se exterior == 1
        Retornar 1
    Fim se
Fim repetir
Retornar 0
Fim subprograma
```

```
Subprograma ressonancia()
Se tipo_expansao == criacao_novo_simplex
    Se I ≠ argmax_i{T_i}
        Adicionar simplex S* a C_I
    Fim se
Senão Se tipo_expansao == substituicao_vetor
    Seja X = {S_Il ∈ C_I : w_J ∈ S_Il}
    Repetir para S_Il ∈ X:
        Substituir w_J por I em S_Il
    Fim repetir
    Eliminar o vetor w_J
Fim se
fim_classificacao = 1
Fim subprograma
```

```
Subprograma cria_nova_categoria()
Seja X = {w_k : w_k ∈ C_i, P(C_i) == P_d}
A = busca_vetores_conectaveis(X)
B =seleciona_n_vetores_proximos(A)
Se card(B) < n
    cria_categoria_monovetor()
Senão
    Criar um simplex novo S* com os vetores do conjunto B ∪ {I}
    Se descarta_simplex_agudo(S*) == 1
        Eliminar simplex S*
        cria_categoria_monovetor()
```

Senão
Se `comprova_sobreposicao`(S^*, $\mathfrak{C}$) == 0
`cria_categoria_monosimplex()`
Senão
Eliminar simplex S^*
`cria_categoria_monovetor()`
Fim se
Fim se
Fim se
`fim_classificacao = 1`
Fim subprograma

Subprograma `comprova_sobreposicao_segmento`($\mathbf{w}_1$, $\mathbf{w}_2$, X)
Repetir para $C_i \in X$
Repetir para $S_{ij} \in C_i$
Se existe sobreposição entre o segmento $\overline{\mathbf{w}_1, \mathbf{w}_2}$ e o simplex S_{ij}:
Retornar 1
Fim se
Fim repetir
Fim repetir
Retornar 0
Fim subprograma

Subprograma `comprova_sobreposicao`(C, X)
Repetir para $C_i \in X$
Repetir para $S_{ij} \in C_i$
Se existe sobreposição entre C e S_{ij}:
Retornar 1
Fim se
Fim repetir
Fim repetir
Retornar 0
Fim subprograma

Subprograma `seleciona_n_vetores_proximos`(X)
$Y = \varnothing$, $i = 0$

Repetir:
$d_{min} = \infty$
Repetir para $\mathbf{w}_k \in X$:
$d_k = \|\mathbf{w}_k - \mathbf{I}\|$
Se $d_k < d_{min}$:
$d_{min} = d_k$
$\mathbf{z} = \mathbf{w}_k$
Fim se
Fim repetir
$Y = Y \cup \{\mathbf{z}\}$
$X = X \setminus \{\mathbf{z}\}$
$i = i + 1$
Ate que $X == \varnothing$ ou $i == n$
Retornar Y
Fim subprograma

Subprograma `cria_categoria_monovetor()`
Criar uma categoria nova C_{N_c+1}
$N^s_{N_c+1} = 0$
$\mathbf{w}_{N_c+1} = \mathbf{I}$
$P(C_{N_c+1}) = P_d$
Fim subprograma

Subprograma `cria_categoria_monosimplex()`
Criar uma categoria nova C_{N_c+1}
$N^s_{N_c+1} = 1$
$S_{(N_c+1)1} = S^*$
$P(C_{N_c+1}) = P_d$
Fim subprograma

I.2 Etapa de Processamento

Inicio
Calcular $T_i^p(\mathbf{I}), i = 1, \ldots, N_c$ (**Funções de escolha de categoria**, eq. 3.18)
Codificar o padrão de entrada pela categoria interna C_I com $I = argmax\{T_i^p\}_{i=1}^{N_c}$
Proporcionar como saida a predição $P(C_I)$
Fim

Apêndice J

Pseudocódigo do algoritmo OPTAM

Neste apêndice apresentamos o pseudocódigo do algoritmo *Overlapping PolyTope* ARTMAP nas etapas de treinamento e processamento usando a notação simplificada da tabela J.1.

n	Dimensão do espaço de entrada
$\mathbf{I}$	Padrão de entrada $\in [0,1]^n$
P_d	Predição desejada para o padrão de entrada $\mathbf{I}$
C_i	i-ésima categoria politopo
$P(C_i)$	Predição associada a i-ésima categoria
N_i^s	Número de simplexes na categoria C_i
$\mathbf{w}_{ij}$	Vetor de pesos j-ésimo da categoria i
T_i^t	Função de Escolha de treinamento da i-ésima categoria
T_i^p	Função de Escolha de procesamento da i-ésima categoria
S_{ij}	j-ésimo simplex da i-ésima categoria politopo
N_c	Número de categorias internas

Tabela J.1: Notação adotada para o pseudocódigo do algoritmo OPTAM.

J.1 Etapa de Treinamento

Inicio
`inicializa()`
Calcular $T_i^t, i = 1, \ldots, N_c$ (**Funções de escolha de categoria**, (eqs. 4.1 a 4.2)
Repetir
  Seja I tal que $T_I^t \geq T_i^t, i = 1, \ldots, N_c, r_I = 0$; $I = -1$ se $\nexists i$ nestas condições
  Se $I \neq -1$
    Se $P(C_I) \neq P_d$ (**Teste de Predição**):
      Rejeitar categoria I: $r_I = 1$
    Senão:
      `expansao_categoria()`
      Se `teste_vigilancia()` $== 0$
        `rejeita_categoria()`
      Senão
        `fim_classificacao` $= 1$ (**Ressonância**)
      Fim se
    Fim se
  Senão
    `cria_nova_categoria()`
  Fim se
Ate que `fim_classificacao == 1`
Finalizar

Subprograma `inicializa()`
`fim_classificacao = 0`
Repetir para $i = 1, N_c$
  $r_i = 0$
Fim repetir
Fim subprograma

Subprograma `expansao_categoria()`
`tipo_expansao = expansao_nula`
Se $T_I^t < 1$

`busca_vetores_conectaveis`()
Se $card(A) == n$
`cria_novo_simplex()`
Senão se $card(A) > n$
`substitui_vetor()`
Fim se
Fim se
Fim subprograma

Subprograma `teste_vigilancia()`
Calcular S_I (tamanho da categoria C_I) (eq. 4.5)
Se $S_I \geq S_{max}$ (eq. 4.6)
Retornar 0
Senão
Retornar 1
Fim se
Fim subprograma

Subprograma `rejeita_categoria()`
Se `tipo_expansao` == `criacao_novo_simplex`
Suprimir simplex novo S^* de C_I
Senão se `tipo_expansao` == `substituicao_vetor`
Repetir para $S_{Il} \in B$
Suprimir simplex S^*_{Il} de C_I
Adicionar simplex S_{Il} a C_I
Fim repetir
Fim se
$r_I = 1$
Fim subprograma

Subprograma `busca_vetores_conectaveis`()
Seja $A = \varnothing$
Repetir para $\mathbf{w}_{Ij} \in C_I$:
Se `comprova_sobreposicao_segmento`($\mathbf{I}$, $\mathbf{w}_{Ij}$, C_I) == 0:
$A = A \cup \{\mathbf{w}_{Ij}\}$
Fim se

Fim repetir
Fim subprograma

Subprograma `cria_novo_simplex()`
Criar um simplex novo S^* com os vetores do conjunto $A \cup \{\mathbf{I}\}$
Adicionar simplex novo S^* a C_I
`tipo_expansao = criacao_novo_simplex`
Fim subprograma

Subprograma `substitui_vetor()`
$J = -1$; $d_{min} = \infty$
Repetir para $\mathbf{w}_j \in A$
  Se `criterio_substituicao_vetor(`$\mathbf{w}_j$`) == 1`
    $d = \|\mathbf{I} - \mathbf{w}_j\|$
    Se $d < d_{min}$
      $d_{min} = d$
      $J = j$
    Fim se
  Fim se
Fim repetir
Se $J \neq -1$
  Seja $B = \{S_{Il} \in C_I : \mathbf{w}_J \in S_{Il}\}$
  Repetir para $S_{Il} \in B$:
    Criar um simplex novo $S^*_{Il} = S_{Il} \setminus \{\mathbf{w}_J\} \cup \{\mathbf{I}\}$
    Adicionar simplex S^*_{Il} a C_I
    Suprimir simplex S_{Il} de C_I
  Fim repetir
  `tipo_expansao = substituicao_vetor`
Fim se
Fim subprograma

Subprograma `criterio_substituicao_vetor(`$\mathbf{w}$`)`
Seja $X = \{S_{Ij} : \mathbf{w} \in S_{Ij}\}$
Repetir para $S_{Ij} \in X$:
  `exterior = 1`
  Repetir para $\mathbf{w}_k \in S_{Ij}, \mathbf{w}_k \neq \mathbf{w}$

Se **I** e interior ao hiperplano definido pelo conjunto de vetores $S_{Ij} \setminus \{\mathbf{w}_k\}$
`exterior = 0`
Fim repetir
Fim se
Fim repetir
Se `exterior == 1`
Retornar 1
Fim se
Fim repetir
Retornar 0
Fim subprograma

Subprograma `cria_nova_categoria()`
Seja $X = \{\mathbf{w}_{ij} : N_i^s = 0, P(C_i) == P_d\}$
$Y =$`seleciona_n_vetores_proximos(`X`)`
Se $card(Y) == n$
Criar un simplex novo S^* com os vetores do conjunto $Y \cup \{\mathbf{I}\}$
`cria_categoria_monosimplex()`
Se `teste_vigilancia()` $== 0$
Suprimir simplex S^*
Suprimir categoria C_I
`cria_categoria_monovetor()`
Fim se
Senão
`cria_categoria_monovetor()`
Fim se
`fim_classificacao = 1`
Fim subprograma

Subprograma `comprova_sobreposicao_segmento`($\mathbf{w}_1$, $\mathbf{w}_2$, X)
Repetir para $C_i \in X$
Repetir para $S_{ij} \in C_i$
Se $O_{ls}(\mathbf{w}_1, \mathbf{w}_2, S_{ij}) == 1$ (eq. F.5)
Retornar 1
Fim se
Fim repetir

Fim repetir
Retornar 0
Fim subprograma

Subprograma `seleciona_n_vetores_proximos(`X`)`
Seja $Y = \varnothing$, $i = 0$
Repetir:
    $d_{min} = \infty$
    Repetir para $\mathbf{w}_k \in X$:
        $d = \|\mathbf{w}_k - \mathbf{I}\|$
        Se $d < d_{min}$:
            $d_{min} = d$
            $\mathbf{z} = \mathbf{w}_k$
        Fim se
    Fim repetir
    $Y = Y \cup \{\mathbf{z}\}$
    $X = X \setminus \{\mathbf{z}\}$
    $i = i + 1$
Ate que $X == \varnothing$ ou $i == n$
Retornar Y
Fim subprograma

Subprograma `cria_categoria_monovetor()`
Criar uma categoria nova C_I com $I = N_c + 1$
$N_I^s = 0$
$\mathbf{w}_{I1} = \mathbf{I}$
$P(C_I) = P_d$
Fim subprograma

Subprograma `cria_categoria_monosimplex()`
Criar uma categoria nova C_I com $I = N_c + 1$
$N_I^s = 1$
$S_{I1} = S^*$
$P(C_I) = P_d$
Fim subprograma

J.2 Etapa de Processamento

Inicio
Calcular $T_i^p(\mathbf{I}), i = 1, \ldots, N_c$ (**Função de escolha de categoria**, eq. 4.10)
Codificar o padrão de entrada pela categoria interna C_I com $I = argmax\{T_i^p\}_{i=1}^{N_c}$
Proporcionar como saida a predição $P(C_I)$
Fim

Appendix K

Equation of the hyperplane defined by $n+1$ vectors

Let h_{ijk} be the hyperplane in $\mathbb{R}^n$ defined by weight vectors $\mathbf{w}_{ijk1}, \ldots, \mathbf{w}_{ijkn}$, and $g_{ijk}(\mathbf{x}) = 0$ the equation of h_{ijk}. From elementary geometry, we know that every point $\mathbf{x} \in h_{ijk}$ is a linear combination of vectors in the set $\{\mathbf{w}_{ijk1}\} \cup \{\mathbf{w}_{ijkl} - \mathbf{w}_{ijk1}, l = 2, \ldots, n\}$, so $\exists \lambda_2, \ldots, \lambda_n \in \mathbb{R}$ which verify:

$$\mathbf{x} = \mathbf{w}_{ijk1} + \sum_{l=2}^{n} \lambda_l (\mathbf{w}_{ijkl} - \mathbf{w}_{ijk1}) \tag{K.1}$$

Thus, the set of vectors $\{\mathbf{x} - \mathbf{w}_{ijk1}, \mathbf{w}_{ijk2} - \mathbf{w}_{ijk1}, \ldots, \mathbf{w}_{ijkn} - \mathbf{w}_{ijk1}\}$ is linearly dependent, and they compose a singular matrix:

$$\begin{vmatrix} x_1 - w_{ijk1,1} & \ldots & x_n - w_{ijk1,n} \\ w_{ijk2,1} - w_{ijk1,1} & \ldots & w_{ijk2,n} - w_{ijk1,n} \\ \ldots & & \\ w_{ijkn,1} - w_{ijk1,1} & \ldots & w_{ijkn,n} - w_{ijk1,n} \end{vmatrix} = 0 \tag{K.2}$$

where $w_{ijkl,m}$ is the m-th component of $\mathbf{w}_{ijkl}$. Eq. K.2 is the definition of $\phi_{ijk}(\mathbf{x})$ in eq. 2.3 (subsection 2.1.1). On the other hand, $g_{ijk}(\mathbf{x}) > 0$ must imply that $\mathbf{x}$ falls in the side of h_{ijk} inside the simplex S_{ij}, and $g_{ijk}(\mathbf{x}) < 0$ must imply that $\mathbf{x}$ falls outside S_{ij}. Let $\widehat{\mathbf{w}}_{ijk}$ be the vertex of S_{ij} which does not belong to h_{ijk}, and $sgn(x)$ the sign function (subsection 2.1.1). Then, it must be $sgn(\phi_{ijk}(\mathbf{x})) = sgn(\phi_{ijk}(\widehat{\mathbf{w}}_{ijk}))$ if $\mathbf{x}$ falls inside the simplex, and $sgn(\phi_{ijk}(\mathbf{x})) \neq sgn(\phi_{ijk}(\widehat{\mathbf{w}}_{ijk}))$

when $\mathbf{x}$ falls outside the simplex. In order to meet this condition, the equation of hyperplane h_{ijk} can be written as $sgn(\phi_{ijk}(\widehat{\mathbf{w}}_{ijk}))\phi_{ijk}(\mathbf{x}) = 0$ (eq. 2.2), so that $g_{ijk}(\mathbf{x}) \equiv sgn(\phi_{ijk}(\widehat{\mathbf{w}}_{ijk}))\phi_{ijk}(\mathbf{x})$.

Appendix L

Calculation of the direction vector of a hyperplane

Let be $g_{ijk}(\mathbf{x}) = 0$ (eq. 2.2 in subsection 2.1.1) the equation of hyperplane h_{ijk}. In order to calculate its direction vector $\mathbf{w}^*_{ijk}$, this equation must be transformed to the following form:

$$\mathbf{w}^*_{ijk}\mathbf{x} + b^*_{ijk} = 0 \tag{L.1}$$

Equation $g_{ijk}(\mathbf{x}) = 0$ implies $\phi_{ijk}(\mathbf{x}) = 0$ (if $\phi_{ijk}(\mathbf{x}) \neq 0$, $sgn(\phi_{ijk}(\mathbf{x})) \neq 0$ and $g_{ijk}(\mathbf{x}) \neq 0$ by eq. 2.3). Developping the determinant $\phi_{ijk}(\mathbf{x})$ in eq. 2.3 by the first row and equaling to eq. L.1 yields:

$$\begin{aligned} \phi_{ijk}(\mathbf{x}) = \sum_{l=1}^{n}(-1)^{l+1}(x_l - w_{ijk1,l})D_{ijkl} = \\ = \sum_{l=1}^{n}(-1)^{l+1}D_{ijkl}x_l - \sum_{l=1}^{n}(-1)^{l+1}w_{ijk1,l}D_{ijkl} = \mathbf{w}^*_{ijk}\mathbf{x} + b_{ijk} \end{aligned} \tag{L.2}$$

where D_{ijkl} is the determinant of matrix in eq. 2.3 after removing row 1 and column l. Equaling the two sides of the last equality and denoting by $\mathbf{e}_l$ the unitary vector ($\|\mathbf{e}_l\| = 1$) in dimension l yields:

$$\mathbf{w}^*_{ijk} = \sum_{l=1}^{n} (-1)^{l+1} D_{ijkl} \mathbf{e}_l \quad \Rightarrow \quad \|\mathbf{w}^*_{ijk}\| = \sqrt{\sum_{l=1}^{n} D^2_{ijkl}} \tag{L.3}$$

The norm of the hyperplane direction vector $\mathbf{w}^*_{ijk}$ is used to calculate the distance between an input pattern $\mathbf{I}$ and the hyperplane h_{ijk} (eq. 2.4).

Appendix M

Geometric implementation of the Overlap Test

The overlap between a simplex and a line segment is tested using geometric techniques. In order to test if a weight vector is "connectable" from the input pattern, the line segment between them can not overlap with other simplexes. On the other hand, in order to create a new simplex, or to replace a weight vector by the input pattern, the new or modified simplex can not overlap with other categories. The following subsections will analyze the two cases.

M.1 Overlap between a line segment and a simplex

The line segment $\overrightarrow{\mathbf{w}_1\mathbf{w}_2}$ and simplex S_{ij} in category C_i overlap only if the former intersects with some hyperplane h_{ijk} of the latter (note that weight vectors can only be simplex vertices, so they can not fall inside a simplex). The intersection between the segment $\overrightarrow{\mathbf{w}_1\mathbf{w}_2}$ and the hyperplane h_{ijk} defined by vectors $\{\mathbf{w}_{ijk1}, \ldots, \mathbf{w}_{ijkn}\}$ can be tested solving for $\alpha^s, \alpha_2^h, \ldots, \alpha_n^h$ the following vector equation (obtained from the parametric equations of the straight line and the hyperplane):

$$\mathbf{w}_1 + \alpha^s(\mathbf{w}_2 - \mathbf{w}_1) = \mathbf{w}_{ijk1} + \sum_{l=2}^{n} \alpha_l^h(\mathbf{w}_{ijkl} - \mathbf{w}_{ijk1}) \tag{M.1}$$

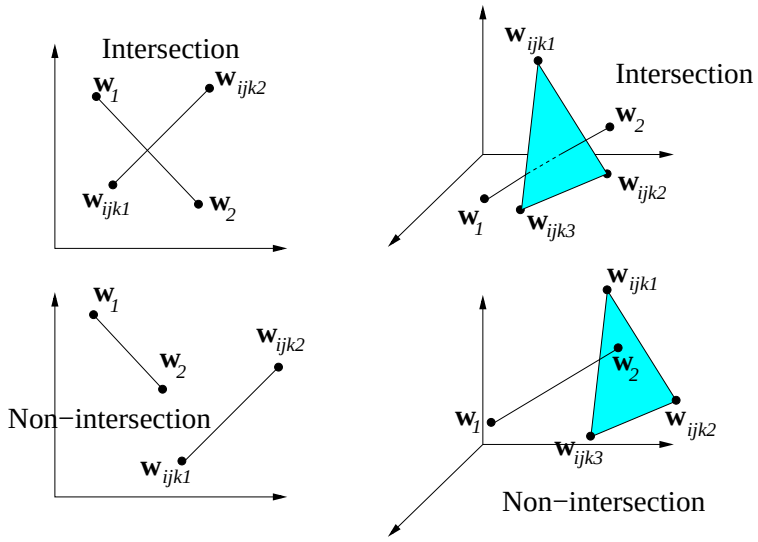


Figure M.1: Examples in $\mathbb{R}^2$ and $\mathbb{R}^3$ of intersection and non-intersection between the segment $\mathbf{w}_1 - \mathbf{w}_2$ and hyperplanes $(\mathbf{w}_{ijk1}, \mathbf{w}_{ijk2})$ in $\mathbb{R}^2$ and $(\mathbf{w}_{ijk1}, \mathbf{w}_{ijk2}, \mathbf{w}_{ijk3})$ in $\mathbb{R}^3$.

The solution $(\alpha^s, \alpha_2^h, \ldots, \alpha_n^h)$ of this system of n equations is a valid intersection point only if it falls inside the bounded hyperplane area whose border is defined by the n vectors $\mathbf{w}_{ijk1}, \ldots, \mathbf{w}_{ijkn}$ of h_{ijk}, and if it falls in the segment between $\mathbf{w}_1$ and $\mathbf{w}_2$ (figure M.1). This imposes the following conditions to the solution of eq. M.1:

$$0 < \alpha^s < 1 \tag{M.2}$$

$$0 < \alpha_p^h + \alpha_q^h < 1 \qquad \forall p, q = 2, \ldots, n; p \neq q \tag{M.3}$$

If $\exists \alpha^s, \alpha_2^h, \ldots, \alpha_n^h \in \mathbb{R}$ satisfying eqs. M.1-M.3, the segment $\overrightarrow{\mathbf{w}_1\mathbf{w}_2}$ and the hyperplane h_{ijk} intersect. We can define the function I_{sh} (intersection segment-hyperplane) in the following way:

$$I_{sh}(\mathbf{w}_1, \mathbf{w}_2, h_{ijk}) = \begin{cases} 1 & \text{eqs. M.1-M.3 have a solution} \\ 0 & \text{otherwise} \end{cases} \tag{M.4}$$

The function $I_{sh}(\mathbf{w}_1, \mathbf{w}_2, h_{ijk}) = 1$ if the line segment and the hyperplane intersect and $I_{sh}(\mathbf{w}_1, \mathbf{w}_2, h_{ijk}) = 0$ otherwise. Finally, the overlap between the line segment $\overrightarrow{\mathbf{w}_1, \mathbf{w}_2}$ and the simplex S_{ij} is given by:

$$O_{ls}(\mathbf{w}_1, \mathbf{w}_2, S_{ij}) = \begin{cases} 1 & \exists h_{ijk} \in S_{ij} : I_{sh}(\mathbf{w}_1, \mathbf{w}_2, h_{ijk}) = 1 \\ 0 & \text{otherwise} \end{cases} \tag{M.5}$$

M.2 Overlap between a simplex and a category

If C_j is a single-vector category ($N_j^s = 0$) defined by the vector $\mathbf{w}_j$, the simplex S_{ik} and C_j overlap if $T_{ik}(\mathbf{w}_j) = 1$. If C_j is a polytope category ($N_j^s > 0$), they overlap only if exists a simplex $S_{jl} \in C_j$ that overlaps with S_{ik}. Let function O_{ss} (overlap between two simplexes) be defined by $O_{ss}(S_{ik}, S_{jl}) = 1$ if simplexes S_{ik} and S_{jl} overlap and $O_{ss}(S_{ik}, S_{jl}) = 0$ otherwise. Since the vertices of a simplex can not be inside another simplex, the simplexes S_{ik} and S_{jl} overlap only if a line segment of S_{ik} and a hyperplane of S_{jl} intersect. Specifically, if $\exists p, q \in \{1, \ldots, n+1\}, p < q$, verifying $O_{ls}(\mathbf{w}_{ikp}, \mathbf{w}_{ikq}, S_{jl}) = 1$ (O_{ls} is defined in eq. M.5), then the line segment $\overrightarrow{\mathbf{w}_{ikp}, \mathbf{w}_{ikq}}$ and the simplex S_{jl} intersect, so $O_{ss}(S_{ik}, S_{jl}) = 1$. Note that if a line segment of S_{jl} and a hyperplane of S_{ik} intersect, then some line segment of S_{ik} intersects with a hyperplane of S_{jl}, which is the previous case. The overlap function $O_{ss}(S_{ik}, S_{jl})$ between the simplexes S_{ik} and S_{jl} is given by:

$$O_{ss}(S_{ik}, S_{jl}) = \begin{cases} 1 & \exists p, q = 1, \ldots, n+1, p < q : \\ & \quad O_{ls}(\mathbf{w}_{ikp}, \mathbf{w}_{ikq}, S_{jl}) = 1 \\ 0 & \quad \text{otherwise} \end{cases} \tag{M.6}$$

Finally, the overlap between the simplex S_{ik} and the category C_j is given by the function $O_{sc}(S_{ik}, C_j)$, defined by:

$$O_{sc}(S_{ik}, C_j) = \begin{cases} \psi[T_{ik}(\mathbf{w}_j)] & N_j^s = 0 \\ \max\limits_{l=1,\ldots,N_j^s} \{O_{ss}(S_{ik}, S_{jl})\} & N_j^s > 0 \end{cases} \tag{M.7}$$

where the function $\psi(x)$ is defined as:

$$\psi(x) = \begin{cases} 1 & x = 1 \\ 0 & x < 1 \end{cases} \tag{M.8}$$

Appendix N

Number of parameters in PTAM

Since each category in PTAM has a variable number of simplexes, its number of weight vectors (vertices) is also variable. If PTAM has N_c categories, N_s simplexes and N_w weight vectors, the number N_p of parameters which it stores is the following:

- For each category, the indexes of their simplexes. Given that category C_i has N_i^s simplexes, the number of parameters to store for the N_c categories is $\sum_{i=1}^{N_c} N_i^s = N_s$, because each simplex belongs to exactly one category and there are N_s simplexes.
- For each simplex, the indexes of their weight vectors. Since each simplex has $n+1$ weight vectors, and there are N_s simplexes, the number of parameters is $(n+1)N_s$ (note that $(n+1)N_s > N_w$, since vectors may be shared between simplexes).
- For each weight vector, its n components. Since there are N_w weight vectors, the number of parameters is nN_w.

Finally, the total number of parameters stored by PTAM is:

$$N_p^{PTAM} = (n+2)N_s + nN_w \tag{N.1}$$

Comparatively, the number of parameters used by FAM is $N_p^{FAM} = 2nN_c$ (each internal category is defined by a $2n$-dimensional weight vector, due to complement-coding). Finally, each internal category in DAM ([36], p. 804) is defined by $4n$

weights $\{\tau_{ij}, \tau_{ji}; i, j = 1, \ldots, 2n\}$ and an instance counter c_j, so the number of parameters in DAM is $N_p^{DAM} = (4n + 1)N_c$.

Referências Bibliográficas

[1] R. K. Aggarwal, Q. Y. Xuan, A. T. Johns, F. Li, e A. Bennett. A novel approach to fault diagnosis in multicircuit transmission lines using fuzzy ARTMAP neural networks. *IEEE Trans. on Neural Networks*, vol 10(5), 1999, páginas 1214–1221.

[2] G. C. Anagnostopoulos e M. Georgiopoulos. Hypersphere ART and ARTMAP for unsupervised and supervised incremental learning. Em *Proceedings of 2000 IEEE Int. Joint Conf. on Neural Networks*, vol 6. 2000, páginas 59–64.

[3] G. C. Anagnostopoulos e M. Georgiopoulos. Ellipsoid ART and Ellipsoid ARTMAP for incremental unsupervised and supervised learning. Em *Proceedings of 2001 IEEE Int. Joint Conf. on Neural Networks*, vol 2. 2001, páginas 1221–1226.

[4] G. C. Anagnostopoulos e M. Georgiopoulos. Category regions as new geometrical concepts in Fuzzy-ART and Fuzzy-ARTMAP. *Neural Networks*, vol 15(10), 2002, páginas 1205–1221.

[5] G. C. Anagnostopoulos e M. Georgiopoulos. Putting the Utility of Match Tracking in Fuzzy ARTMAP Training to the Test. Em *Proceedings of 2003 International Conference on Knowledge-Based Intelligent Information & Engineering Systems*, vol 2. 2003, páginas 1–6.

[6] G. C. Anagnostopoulos, M. Georgiopoulos, S. Verzi, e G. Heileman. Reducing generalization error and category proliferation in Ellipsoid ARTMAP via tunable misclassification error tolerance: Boosted Ellipsoid ARTMAP. Em *Proceedings of 2002 IEEE Int. Joint Conf. on Neural Networks*, vol 3. 2002, páginas 2650–2655.

[7] Y. R. Asfour, G. A. Carpenter, S. Grossberg, e G. W. Lesher. Fusion ARTMAP: A neural network architecture for multichannel data fusion and classification. Em *Proceedings of 1993 World Congress on Neural Network*, vol II. 1993, páginas 210–215.

[8] S. Barro, M. Fernández-Delgado, J. Vila, R. C.V., e E. Sánchez. Classifying multichannel ECG patterns with an adaptive neural network. *IEEE Engineering in Medicine and Biology Magazine*, vol 17(1), 1998, páginas 45–55.

[9] G. Bartfai. An ART-based modular architecture for learning hierarchical clustering. *Neurocomputing*, vol 13, 1996, páginas 31–45.

[10] G. Bartfai e R. White. Adaptive Resonance Theory-based Modular Networks for incremental learning of Hierarchical clusterings. *Connection Science*, vol 9(1), march 1997, páginas 87–112.

[11] R. Bellman. *Adaptive control processes: A guided tour*. Princeton University Press, 1961.

[12] C. Blake e C. Merz. UCI Repository of machine learning databases, 1998. http://www.ics.uci.edu/∼mlearn/MLRepository.html.

[13] L. B. Booker, D. E. Goldberg, e J. H. Holland. Classifier Systems and Genetic Algorithms. *Artificial Intelligence*, vol 40, 1989, páginas 235–282.

[14] J. Brezmes, M. L. L. Fructuoso, E. Llobet, X. Vilanova, I. Recasens, J. Orts, G. Saiz, e X. Correig. Evaluation of an electronic nose to assess fruit ripeness. *IEEE Sensors Journal*, vol 5, 2005, páginas 97–108.

[15] J. Cano-Izquierdo, Y. Dimitriadis, M. Araúzo-Bravo, e J. Coronado. FasArt: A new neuro-fuzzy architecture for incremental learning in system identification. Em *Proceedings of the IFAC World Congress, IFAC96*, vol F. 1996, páginas 133–138.

[16] J. Cano-Izquierdo, Y. Dimitriadis, e J. Coronado. FasBack: Matching-error based learning for automatic generation of fuzzy logic systems. Em *Proceedings of the sixth IEEE International Conference on Fuzzy Systems*, vol 3. 1997, páginas 1561–1566.

[17] J. Cano-Izquierdo, Y. Dimitriadis, E. Gómez-Sánchez, e J. López-Coronado. Learning from noisy information in FasArt and FasBack neuro-fuzzy systems. *Neural Networks*, vol 14(4/5), 2001, páginas 407–425.

[18] G. A. Carpenter. A distributed outstar network for spatial pattern learning. *Neural Networks*, vol 7(1), jan 1994, páginas 159–168.

[19] G. A. Carpenter. Distributed ART networks for learning, recognition, and prediction. Em *Proceedings of the World Congress on Neural Networks (WCNN96)*. 1996, páginas 333–344. Technical Report CAS/CNS TR-96-005, Boston, MA: Boston University.

[20] G. A. Carpenter. Distributed learning, recognition, and prediction by ART and ARTMAP neural networks. *Neural Networks*, vol 8(10), 1997, páginas 1473–1494.

[21] G. A. Carpenter. Default ARTMAP. Em *Proceedings of 2003 IEEE Int. Joint Conf. on Neural Networks*. Portland, Oregon, 2003, páginas 1396–1401.

[22] G. A. Carpenter, S. Gopal, S. Macomber, S. Martens, C. Woodcock, e J. Franklin. A neural network method for efficient Vegetation Mapping. *Remote Sens. Environ.*, vol 70, 1999, páginas 326–338.

[23] G. A. Carpenter e S. Grossberg. A massively parallel architecture for a self-organization neural pattern recognition machine. *Computer Vision, Graphics, and Image Processing*, vol 37, 1987, páginas 54–115.

[24] G. A. Carpenter e S. Grossberg. ART2: Self-organization of stable category recognition codes for analog input patterns. *Applied Optics : Special Issue on Neural Networks*, vol 26, 1987, páginas 4919–4930.

[25] G. A. Carpenter e S. Grossberg. The ART of adaptive pattern recognition by a self-organizing neural network. *Computer*, vol 21(3), March 1988, páginas 77–88.

[26] G. A. Carpenter e S. Grossberg. ART3: Hierarchical search using chemical transmitters in self- organizing pattern recognition architectures. *Neural Networks*, vol 3, 1990, páginas 129–152.

[27] G. A. Carpenter e S. Grossberg. Normal and amnesic learning recognition, and memory by a neural model of cortico-hippocampal interactions. *Trends in Neuroscience*, vol 16(4), 1993, páginas 131–137.

[28] G. A. Carpenter, S. Grossberg, N. Markuzon, J. H. Reynolds, e D. Rosen. Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions on Neural Networks*, vol 3(5), september 1992, páginas 698–713.

[29] G. A. Carpenter, S. Grossberg, e J. H. Reynolds. ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network. *Neural Networks*, vol 4(5), 1991, páginas 565–588.

[30] G. A. Carpenter, S. Grossberg, e D. Rosen. ART2-A: An adaptive resonance algorithm for rapid category learning and recognition. *Neural Networks*, vol 4(4), 1991, páginas 493–504.

[31] G. A. Carpenter, S. Grossberg, e D. B. Rosen. Fuzzy ART: An adaptive resonance algorithm for rapid, stable classification of analog patterns. Em *Proceedings of 1991 IEEE Int. Joint Conf. on Neural Networks*, vol 2. 1991, páginas 411–416.

[32] G. A. Carpenter, S. Grossberg, e D. B. Rosen. Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks*, vol 4(6), 1991, páginas 759–771.

[33] G. A. Carpenter e N. Markuzon. ARTMAP-IC and medical diagnosis: Instance counting and inconsistent cases. *Neural Networks*, vol 11(2), 1998, páginas 323–336.

[34] G. A. Carpenter, S. Martens, e O. Ogas. Self-organizing information fusion and hierarchical knowledge discovery: A new framework using ARTMAP neural networks. *Neural Networks*, vol 18(3), 2005, páginas 287–295.

[35] G. A. Carpenter, S. Martens, e O. J. Ogas. Self-organizing Hierarchical Knowledge Discovery by an ARTMAP Image Fusion System. Em *Proceedings of the 7th International Conference on Information Fusion (Fusion 2004)*. Portland, Oregon, 2004, páginas 235 – 242. Technical Report CAS/CNS TR-2004-001, Boston, MA: Boston University.

[36] G. A. Carpenter, B. Milenova, e B. Noeske. Distributed ARTMAP: A neural network for fast distributed supervised learning. *Neural Networks*, vol 11(5), 1998, páginas 793–813.

[37] G. A. Carpenter e W. D. Ross. ART-EMAP: A neural network architecture for object recognition by evidence accumulation. *IEEE Transactions on Neural Networks*, vol 6(4), july 1995, páginas 805–818.

[38] T. P. Caudell, S. D. G. Smith, R. Escobedo, e M. Anderson. NIRS: large scale ART-1 neural architectures for engineering design retrieval. *Neural Networks*, vol 7(9), 1994, páginas 1339–1350.

[39] D. Clark, Z. Schreter, e A. Adams. A Quantitative Comparison of Dystal and Backpropagation. Em *Proceedings of 1996 Australian Conference on Neural Networks*. 1996.

[40] R. Collobert, S. Bengio, e J. Mariéthoz. Torch: A modular machine learning software library, 2003. Http://www.idiap.ch/∼bengio/publications/pdf/rr02-46.pdf. Library Torch available from http://www.torch.ch.

[41] C. Cortes e V. Vapnik. Support-Vector Networks. *Machine Learning*, vol 20(3), 1995, páginas 273–297.

[42] T. M. Cover. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE Trans. on Electronics Computers*, vol 14, 1965, páginas 326–334.

[43] N. Cristianini e J. Shawe-Taylor. *An introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.

[44] I. Dagher, M. Georgiopoulos, e G. Bebis. An ordering algorithm for pattern presentation in Fuzzy ARTMAP that tends to improve generalization performance. *IEEE Transactions on Neural Networks*, vol 10(4), 1999, páginas 768–778.

[45] B. V. Dasarathy. *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques*. Las Alamitos, California. IEEE Computer Society Press, 1991.

[46] R. Duda, H. P.E., e D. Strock. *Pattern classification*. John Wiley & Sons, Inc., 2001.

[47] D. Enke, K. Ratanapan, e C. Dagli. Machine-part family formation utilizing an ART1 neural network implemented on a parallel neuro-computer. *Comput. Ind. Eng.*, vol 34(1), 1998, páginas 189–205.

[48] D. Enke, K. Ratanapan, e C. Dagli. Large machine-part family formation utilizing a parallel ART1 neural network. *Journal of Intelligent Manufacturing*, vol 11(6), Dec 2000, páginas 591–604.

[49] D. Fan, A. Waxman, M. Aguilar, D. Ireland, J. Racamato, W. Ross, W. Streilein, e M. Braun. Fusion of multi-sensor imagery for night vision: color visualization, target learning and search. Em *Proceedings of the Third International Conference on Information Fusion*, vol 1. July 2000, páginas TUD3/3–TUD310.

[50] J. Fan e G. Wu. A Foremost-Policy Reinforcement Learning Based ART2 Neural Network and Its Learning Algorithm. Em *Advances in Neural Networks - ISNN 2005, Second International Symposium on Neural Networks*, vol Part I de *Lecture Notes in Computer Science*. Springer, May-June 2005, páginas 634–639.

[51] M. Fernández Delgado e S. Barro. A multi-channel ART-based neural network. *IEEE Transactions on Neural Networks*, vol 9, 1998, páginas 139–150.

[52] M. Fernández-Delgado, J. Presedo, e S. Barro. Multichannel pattern recognition neural network. Em *5th International Work-Conference on Artificial and Natural Neural Networks (IWANN99)*. 1999, páginas 719–729.

[53] M. Fernández Delgado, C. Regueiro, E. Sánchez, e S. Barro. MART: Una red neuronal para la clasificación de patrones multicanal. *Rev. Iberoamericana de Inteligencia Artificial*, vol 1, jan 1997, páginas 8–15.

[54] J. A. Freeman e D. M. Skapura. *Neural Networks: Algorithms, applications and programming techniques*. Addison-Wesley, 1991.

[55] W. Fung e Y. Liu. Adaptive categorization of ART networks in robot behavior learning using game-theoretic formulation. *Neural Networks*, vol 16(10), 2003, páginas 1403–1420.

[56] K. W. Gan e K. T. Lua. Chinese character classification using adaptive resonance network. *Pattern Recognition*, vol 25(8), 1992.

[57] P. García-Sevilla e M. Petrou. Analysis of Irregularly Shaped Texture Regions. *Computer Vision and Image Understanding*, vol 84, 2001, páginas 62–76.

[58] M. Georgiopoulos, I. Dagher, G. L. Heileman, e G. Bebis. Properties of learning of a Fuzzy ART Variant. *Neural Networks*, vol 12(6), 1999, páginas 837–850.

[59] M. Georgiopoulos, J. Huang, e G. L. Heileman. Properties of learning in ARTMAP. *Neural Networks*, vol 7(3), 1994, páginas 495–596.

[60] M. Georgiopoulos, A. Koufakous, G. Anagnostopoulos, e T. Kasparis. Overtraining in Fuzzy ARTMAP: myth or reality? Em *Proceedings of 2001 IEEE Int. Joint Conf. on Neural Networks*, vol 2. 2001, páginas 1186–1190.

[61] D. Gomes, M. Fernández Delgado, e S. Barro. Simplex ARTMAP: Building general borders among predictions with simplex-shaped classes. Em *Proceedings of 2004 IASTED Artificial Intelligence and Soft. Computing*, vol 1. 2004, páginas 232–237.

[62] D. Gomes, M. Fernández Delgado, e S. Barro. A vigilance-free ART network with general geometry internal categories. Em *Proceedings of 2005 IEEE Int. Joint Conf. on Neural Networks*. 2005, páginas 463–468.

[63] D. Gomes, M. Fernández Delgado, e S. Barro. Internal Categories with Irregular Geometry and Overlapping in ART networks. Em *Lectures Notes in Artificial Intelligence (LNAI)*. 2006, páginas 291–300.

[64] D. Gomes, M. Fernández Delgado, e S. Barro. Polytope ARTMAP: pattern classification without vigilance based on general geometry categories. *IEEE Trans. on Neural Networks*, 2006, página Em segunda revisão.

[65] E. Gómez-Sánchez, Y. Dimitriadis, J. M. Cano-Izquierdo, e J. Coronado. Safe μARTMAP: a new solution for reducing category proliferation in Fuzzy ARTMAP. Em *Proceedings of 2001 IEEE Int. Joint Conf. on Neural Networks*, vol 2. 2001, páginas 1197–1202.

[66] E. Gómez-Sánchez, Y. A. Dimitriadis, J. M. Cano-Izquierdo, e J. Coronado. MicroARTMAP: use of mutual information for category reduction in Fuzzy ARTMAP. Em *Proceedings of 2000 IEEE Int. Joint Conf. on Neural Networks*, vol 6. 2000, páginas 47–52.

[67] E. Gómez-Sánchez, Y. A. Dimitriadis, J. M. Cano-Izquierdo, e J. Coronado. μARTMAP: use of mutual information for category reduction in Fuzzy ARTMAP. *IEEE Transactions on Neural Network*, vol 13(1), jan 2002, páginas 58–69.

[68] E. Gómez-Sánchez, G. J. A. González, Y. A. Dimitriadis, J. M. Cano-Izquierdo, e J. Coronado. Experimental study of a novel neuro-fuzzy system fo on-line handwritten UNIPEN digit recognition. *Pattern Recognition Letters*, vol 19, 1998, páginas 357–364.

[69] S. Gopal, C. Woodcock, e A. Strahler. Fuzzy ARTMAP classification of global land cover from the 1 degree AVHRR data set. *Remote sensing of Environment*, vol 67, 1999, páginas 230–243.

[70] E. Granger, S. Grossberg, P. Lavoie, e M. A. Rubin. Comparison of Classifiers for Radar Emitter Type Identification. *Intelligent Engineering Systems Through Artificial Neural Networks*, vol 9, 1999, páginas 3 –12.

[71] E. Granger, M. A. Rubin, S. Grossberg, e P. Lavoie. Classification of Incomplete Data Using the Fuzzy ARTMAP Neural Network. *IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN'00)*, vol 6, 2000, página 6035.

[72] N. Griffith e P. M. e. Todd. *Musical Networks: Parallel Distributed Perception and Performance*. MIT Press, Cambridge, MA, 1999.

[73] P. Gritzmann e V. Klee. *Polytopes: Abstract, Convex and Computational*, capítulo On the Complexity of Some Basic Problems in Computational Convexity II. Volume and Mixed Volumes. T. Bisztriczky and P. McMullen and R. Schneider and A. W. Weiss (Eds), Kluwer, 1994.

[74] J. R. Grossberg, S. Williamson. A self-organizing neural system for learning to recognize textured scenes. *Vision Research*, vol 39(7), 1999, páginas 1385–1406.

[75] S. Grossberg. Adaptive pattern classification and universal recoding,I: Parallel development and coding of neural feature detectors. *Biological Cybernetics*, vol 23, 1976, páginas 121–134.

[76] S. Grossberg. Adaptive pattern recognition and universal recoding,II: Feedback, expectation, olfaction, and illusions. *Biological Cybernetics*, vol 23, 1976, páginas 187–202.

[77] S. Grossberg. How does a brain build a cognitive code? *Psychological Review*, vol 87, 1980, páginas 1–51.

[78] S. Grossberg. A theory of human memory: Self-organization and performance of sensor-motor codes, maps, and plans. Studies of mind and brain. *Boston - MA,Reidel*, 1982, páginas 498–639.

[79] S. Grossberg. Behavioral contrast in short-term memory: Serial binary memory models or parallel continous memory models. Studies of mind and brain. *Boston - MA,Reidel*, 1982, páginas 425–447.

[80] S. Grossberg. Nonlinear neural networks: Principles, mechanisms, and architectures. *Neural Networks*, vol 1(1), 1988, páginas 17–61.

[81] S. Grossberg. The link between brain, learning, attention, and consciousness. *Consciousness and Cognition*, vol 8, 1999, páginas 1–44.

[82] S. Grossberg. How does he cerebral cortex work? Development, learning, attention and 3-D vision by laminar circuits of visual cortex. *Behavioral and Cognitive Neuroscience Reviews*, vol 2(1), 2003, páginas 47–76.

[83] S. Grossberg, K. K. Govindarajan, L. L. Wyse, e M. A. Cohen. ARTSTREAM: a neural network model of auditory scene analysis and source segregation. *Neural Networks*, vol 17(4), 2004, páginas 511–536.

[84] S. Haykin. *Neural Networks: A comprehensive Foundation*. Upper Saddle River, New Jersey, USA. Prentice-Hall, 1994.

[85] J. He, A. Tan, e C. Tan. ART-C: A neural architecture for self-organization under constraints. Em *Proceedings of 2002 IEEE Int. Joint Conf. on Neural Networks*, vol 3. 2002, páginas 2550–2555.

[86] M. J. Healy e T. P. Caudell. Guaranteed two-pass convergence for supervised and inferential learning. *IEEE Transactions on Neural Networks*, vol 9(1), 1998, páginas 195–204.

[87] M. J. Healy, T. P. Caudell, e D. G. S. Smith. A neural architecture for pattern sequence verification through inferencing. *IEEE Transactions on Neural Networks*, vol 4(1), 1993, páginas 9–20.

[88] J. Hertz, A. Krogh, e R. G. Palmer. *Introduction to the Theory of Neural Computation. Reading.* MA:Addison-Wesley, 1991.

[89] T. K. Ho e M. Basu. Complexity measures of supervised classification problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol 24(3), 2002, páginas 289–300.

[90] J. Hopfield. Pattern recognition using action potential timing for stimulus representation. *Nature*, vol 376, 1995, páginas 33–36.

[91] J. Huang, M. georgiopoulos, e G. L. Heileman. Fuzzy ART properties. *Neural Networks*, vol 8(2), 1995, páginas 203–213.

[92] C. A. Hung e S. F. Lin. Adaptive Hamming net: a fast-learning ART1 model without searching. *Neural Networks*, vol 8(4), 1995, páginas 605–618.

[93] C. A. Hung e S. F. Lin. Supervised Adaptive Hamming Net for classification of multiple-valued patterns. *International journal of Neural Systems*, vol 8(2), 1997, páginas 181–200.

[94] C. A. Hung e S. F. Lin. An incremental learning neural network for pattern classification. *International Journal of Pattern Recognition and Artificial Intelligence*, vol 13(6), 1999, páginas 913–928.

[95] E. Kaltofen e G. Villard. On the complexity of computing determinants. *Computation complexity*, vol 13(3–4), 2004, páginas 91–130.

[96] T. Kasuba. Simplified Fuzzy ARTMAP. *AI Expert*, vol 8(11), 1993, páginas 18–25.

[97] E. S. Kim, J. W. Cha, e C. S. Ryu. Modular mART for 3D target recognition. *Electronics Letters*, vol 33(16), 1997, páginas 1396–1398.

[98] M.-H. Kim, D.-S. Jang, e Y.-K. Yang. A robust-invariant pattern recognition model using Fuzzy ART. *Pattern Recognition*, vol 34(8), 2001, páginas 1685–1696.

[99] Y. S. Kim e S. Mitra. An adaptive integrated fuzzy clustering model for pattern recognition. *Fuzzy Sets and Systems*, vol 65, 1994, páginas 297–310.

[100] G. A. Klotz e D. A. Stacey. ART2 based classification of sparse high dimensional parameter sets for a simulation parameter selection assistant. Em *Proceedings of 2005 IEEE Int. Joint Conf. on Neural Networks*. 2005, páginas 1081–1085.

[101] N. Kopco, P. Sincak, e L. Molinár. Evaluation of dependence on cluster representation in ARTMAP classifiers. Em *Proceedings of the 3rd IEEE International Conference on Intelligent Engineering Systems INES-99*. 1999, páginas 377–381.

[102] A. Koufakou, M. Georgiopoulos, G. Anagnostopoulos, e T. Kasparis. Cross-validation in Fuzzy ARTMAP for large databases. *Neural Networks*, vol 14(9), 2001, páginas 1279–1291.

[103] A. Leon-Garcia. *Probability and random Processes for Electrical Engineering*. Addison-Wesley - 2ed., 1994.

[104] C. P. Lim e R. Harisson. An Incremental Adaptive Network for On-line Supervised Learning and Probability Estimation. *Neural Networks*, vol 10(5), 1997, páginas 925–939.

[105] C.-P. Lim, J.-H. Leong, e M.-M. Kuan. A Hybrid Neural Network System for Pattern Classification Tasks with Missing Features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol 27(4), 2005, páginas 648 –653.

[106] P. Lisboa. Industrial use of safety-related artificial neural networks. *John Moores University*, 2001.

[107] S. Marriott e R. F. Harrison. A modified Fuzzy ARTMAP architecture for the approximation of noisy mappings. *Neural Networks*, vol 8(4), 1995, páginas 619–641.

[108] L. Massey. On the quality of ART1 text clustering. *Neural Networks*, vol 16(5–6), 2003, páginas 771–778.

[109] L. Massey. Evaluating and Comparing Text Clustering Results. Em *Computational Intelligence*. 2005, páginas 85–90.

[110] L. Massey. Real-World Text Clustering with Adaptive Resonance Theory Neural Networks. Em *Proceedings of 2005 IEEE Int. Joint Conf. on Neural Networks*, vol 5. 2005, páginas 2748–2753.

[111] H. E. Mendoza, J. Santos, A. N. C. Santa Rosa, e N. C. Silva. Land use/land cover mapping in brazilian amazon using neural network with ASTER/TERRA data. Em *Proceedings of 2004 ISPRS Congress- Int. Soc. for Photogrammetry and Remote sensing and Spacial Inform. Sciences*, vol XXXV-B7. july 2004, página 123.

[112] J. Moody e C. J. Darken. Fast learning in networks of locally-tuned processing units. *Neural Computation*, vol 1, 1989, páginas 281–294.

[113] B. Moore. ART1 and pattern clustering. Em *Proceedings of the 1988 connectionist models summer school*. 1989, páginas 174–185. San Mateo, CA: Morgan Kauffmann Publisher.

[114] D. Muchoney e J. Williamson. A Gaussian Adaptive Resonance Theory Neural Network Classification Algorithm Applied to Supervised Land Cover Mapping Using Multitemporal Vegetation Index Data. *IEEE Transactions on Geoscience and Remote Sensing*, vol 39(9), 2001, páginas 1969–1977.

[115] A. V. Nandedkar, K. Venishetti, e A. K. Rathod. Fuzzy Min-Max Neural Network Based Translation, Rotation and Scale Invariant Character Recognition Using RTSI Features. Em *Proceedings of the Fourth International Conference on Computer and Information Technology (CIT'04)*, vol 1. 2004, páginas 159–164.

[116] A. M. Nepomuceno, D. M. Valeriano, C. C. Freitas, A. N. C. S. Rosa, N. C. Silva, J. R. Santos, e L. V. Dutra. Classificação de dados de radar na banda-P utilizando rede neural artificial para mapeamento de cobertura da terra na região de Santarém, Pará. Em *Anais XI Simpósio Brasileiro de Sensoriamento Remoto (SBSR-2003)*, vol 1. apr 2003, páginas 2249–2256.

[117] S. C. Newton, S. Pemmaraju, e S. Mitra. Adaptive fuzzy leader clustering of complex data sets in pattern recognition. *IEEE Transactions on Neural Networks*, vol 3(5), 1992, páginas 784–800.

[118] P. R. Oliveira, J. Vicentini, e R. Romero. Combining fuzzy ART network clustering and sparse code shrinkage for image segmentation. Em *Proceedings of*

the 9th International Conference on Neural Information Processing - ICONIP '02, vol 5. Nov 2002, páginas 2435– 2439.

[119] M. Page. Connectionist modelling in psychology: A localist manifesto. *Behavioral and Brain Sciences*, vol 23, 2000, páginas 443–512.

[120] E. Parrado-Hernández, E. Gómez-Sánchez, e A. Y. Dimitriadis. Study of distributed learning as a solution to category proliferation in Fuzzy ARTMAP based neural systems. *Neural Networks*, vol 16(7), 2003, páginas 1039–1057.

[121] E. Parrado-Hernández, E. Gómez-Sánchez, Y. A. Dimitriadis, e J. Coronado. A neuro-fuzzy system that uses distributed learning for compact rule set generation. Em *IEEE International Conference on Systems, Man, and Cybernetics (SMC99)*, vol 3. 1999, páginas 441–446.

[122] O. Parsons e G. A. Carpenter. ARTMAP neural networks for information fusion and data mining: map production and target recognition methodologies. *Neural Networks*, vol 16(7), 2003, páginas 1075–1089.

[123] J. Platt. Fast training of support vector machines using sequential minimal optimization. Em B. Schölkopf and C.J.C. Burges and A.J. Smola, editor, *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1999, páginas 185–208.

[124] T. Poggio e F. Girosi. A theory of networks for approximation and learning. Em *A.I. Memo N 1140*. Massachusets Institute of Tecnology, 1989.

[125] M. J. D. Powell. Radial basis functions for multivariable interpolation: a review. Em J. C. M. . M. G. C. (Eds.), editor, *Algorithms for approximation*. Oxford: Clarenclon Press, 1987.

[126] S. Rajasekaran e R. Amal Raj. Image recognition using analog-ART1 architecture augmented with moment-based feature extrator. *Neurocomputing*, vol 56, 2004, páginas 61–77.

[127] E. Sapozhnikova, V. Lunin, L. Ludwig, e W. Rosentiel. The use of dARTMAP and fuzzy ARTMAP to solve the quality testing task in semiconductor industry . Em *Proceedings of 1999 International Conference on Knowledge-Based Intelligent Information & Engineering Systems*, vol 1. 1999, páginas 361–364.

[128] E. P. Sapozhnikova e W. Rosentiel. AFC: ART-based Fuzzy Classifier. Em *Proceedings of 2003 International Conference on Knowledge-Based Intelligent Information & Engineering Systems*, vol 2. 2003, páginas 30–36.

[129] B. Shock, G. A. Carpenter, S. Gopal, e C. E. Woodcock. ARTMAP Neural network classification of land use change. Em *Proceedings of the World Congress on Computers in agriculture and Natural Resources*. sept 2001, páginas 22–28. Iguaçu Falls, Brazil, Tech. Rep. CAS/CNS/TR-2001-009, Boston, MA: Boston University.

[130] P. K. Simpson. Fuzzy Min-Max neural networks, part 1: classification. *IEEE Trans. on Neural Networks*, vol 3(5), 1992, páginas 776–786.

[131] D. F. Specht. Probabilistic Neural Networks. *Neural Networks*, vol 3(1), 1990, páginas 109–118.

[132] A. H. Tan. Adaptive Resonance Associative Map. *Neural Networks*, vol 8(3), 1995, páginas 437–446.

[133] A. H. Tan. Cascade ARTMAP: Integrating neural computation and symbolic knowledge processing. *IEEE Transactions on Neural Networks*, vol 8(2), 1997, páginas 237–250.

[134] S. Theodoridis e K. Koutroumbas. *Pattern Recognition*. Academic Press -2ed., 2003.

[135] V. Vapnik. *The nature of statistical learning theory*. Springer-Verlag, 1995.

[136] V. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

[137] P. Venkumar e A. Noorul Haq. Fractional cell formation in group technology using modified ART1 neural networks. *The International Journal of Advanced Manufacturing Technology*, vol 28(7–8), April 2006, páginas 761–765.

[138] S. Verzi, G. L. Heileman, M. Georgiopoulos, e M. J. Healy. Boosting the performance of ARTMAP. Em *Proceedings of 1998 IEEE Int. Joint Conf. on Neural Networks*. 1998, páginas 396–401.

[139] P. R. Villela e J. H. S. Azuela. Improving Pattern Recognition Using Several Feature Vectors. Em *MICAI '02: Proceedings of the Second Mexican International Conference on Artificial Intelligence*. Springer-Verlag, 2002, páginas 282–291.

[140] J. Wachs, H. Stern, e M. Last. Color Face Segmentation Using a Fuzzy Min-Max Neural Network. *Int. J. Image Graphics*, vol 2(4), 2002, páginas 587–602.

[141] L. Wang. *Adaptive fuzzy system and control*. Prentice Hall, 1994. New Jersey.

[142] A. Webb. *Statistical Pattern Recognition*. John Wiley and Sons Ltd, 2002. 2nd ed.

[143] G. Wilensky. Analysis of neural network issues: scaling, enhanced nodal processing, comparison with standard classification. *DARPA Neural Network Program Review*, oct. 1990, páginas 29–30.

[144] J. Williamson. Gaussian ARTMAP: A neural network for fast incremental learning of noisy multidimensional maps. *Neural Networks*, vol 9(5), 1996, páginas 881–897.

[145] R. Xu, G. C. Anagnostopoulos, e D. C. I. I. Wunsch. Tissue classification through analysis of gene expression data using a new family of ART architecture. Em *Proceedings of 2002 International Joint Conference on Neural Networks*, vol 1. 2002, páginas 300–304.

[146] R. Xu, G. C. Anagnostopoulos, e D. C. I. I. Wunsch. Multi-class cancer classification by semi-supervised ellipsoid ARTMAP with gene expression data. Em *Proceedings of 2004 International Conference of the Engineering in Medicine and Biology Society,EMBC*, vol 1. 2004, páginas 188–191.

[147] M.-F. Yeh e S.-S. Chian. GreyART network for data clustering. *Neurocomputing*, vol 67, 2005, páginas 313–320.

[148] L. Zadeh. Fuzzy sets. *Information and control*, vol 8, 1965, páginas 338–353.

[149] N. Zhang e M. Kexunovic. Coordinating fuzzy ART neural networks to improve transmission line fault detection and classification. Em *Proceedings of the Power Engineering Society General Meeting, 2005-IEEE*, vol 1. 2005, páginas 734–740.

[150] J. Zhou e S. Bennett. A supervised learning network based on adaptive resonance theory. *International Journal of Neural Systems*, vol 8(2), 1997, páginas 239–246.

[151] Z. Zhou, S. Chen, e Z. Chen. FANNC: A Fast Adaptive Neural Network Classifier. *Knowledge and Information Systems*, vol 2(1), 2000, páginas 115–129.