FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Semi-automatic classification: using active learning for efficient class coverage

Nuno Filipe Fonseca Vasconcelos Escudeiro

U. PORTO

FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Programa Doutoral em Engenharia Informática

Supervisor: Prof. Doutor Alípio Mário Jorge

Second Supervisor: Prof. Doutor Rui Carlos Camacho

October, 2012

Semi-automatic classification: using active learning for efficient class coverage

Nuno Filipe Fonseca Vasconcelos Escudeiro

Programa Doutoral em Engenharia Informática

October, 2012

Resumo

Alguns problemas de classificação automática, tais como a classificação de texto, exigem um grande esforço para a etiquetação dos exemplos necessários para treinar um classificador apesar da facilidade e baixo custo envolvidos na recolha de exemplos não etiquetados. Contrariamente à aprendizagem supervisionada, que exige que os exemplos do conjunto de treino sejam todos previamente etiquetados, a aprendizagem ativa é um paradigma em que os exemplos são etiquetados em função da sua utilidade para o fim em vista. Para além da seleção criteriosa dos exemplos a etiquetar, a aprendizagem ativa é um processo iterativo que pode ser interrompido quando o valor acrescentado dos exemplos ainda não etiquetados fôr baixo. De uma forma geral, a aprendizagem ativa requer um esforço de etiquetagem inferior ao da aprendizagem supervisionada.

A maioria das abordagens correntes da aprendizagem ativa aplicadas a problemas de classificação assume a existência de um conjunto de exemplos previamente etiquetados, cobrindo todas as classes de interesse. O processo de aprendizagem é inicializado a partir deste conjunto. O esforço necessário para a etiquetação destes exemplos não é, de uma forma geral, contabilizado para efeitos do cálculo do esforço total de etiquetação.

No entanto, a identificação de exemplos representativos de todas as classes pode exigir um esforço significativo, em particular, no que refere à identificação de exemplos representativos de classes minoritárias. Acresce que, em alguns domínios, tais como a deteção de fraude e o diagnóstico de doenças raras, por exemplo, estas classes minoritárias podem ser as mais críticas. Nestas circunstâncias, conduzir e avaliar o processo de aprendizagem com base exclusivamente em critérios de precisão, como é comum em problemas de classificação, pode não ser suficiente. De facto, dependendo do enviesamento da distribuição das classes, um classificador pode apresentar uma taxa de erro baixa mesmo desconhecendo por completo as classes minoritárias.

O tratamento adequado destes casos requer uma abordagem diferente que assegure, para além da precisão, também a capacidade de reconhecimento de todas as classes independentemente da sua distribuição. Entendemos que é possível desenvolver uma estratégia de aprendizagem ativa que permita construir classificadores precisos, com conhecimento de todas as classes, com um esforço de etiquetação (custo) inferior ao das abordagens atuais.

Nesta tese propomos uma estratégia de aprendizagem ativa que inclui um critério de seleção dos exemplos a etiquetar e um critério de paragem que interrompe o processo de aprendizagem quando o valor acrescentado dos exemplos disponíveis para etiquetar é baixo. Esta estratégia promove a eficiência do processo de aprendizagem, focando-se nos exemplos mais informativos e unicamente enquanto o seu valor acrescentado o justifique.

O critério de seleção proposto, d-Confidence, agrega a confiança do classificador com a distância entre os exemplos não etiquetados e as classes conhecidas. É um critério que tende a selecionar exemplos de classes desconhecidas, em que o classificador tenha confiança reduzida, que se localizam em regiões inexploradas do espaço de exemplos, a uma grande distância das classes conhecidas. O critério de paragem proposto, hcw, combina dois indicadores do valor acrescentado do conjunto dos exemplos ainda não etiquetados: o gradiente de classificação e um indicador da estabilidade da distribuição da entropia das previsões. O gradiente de classificação fornece informação sobre a diferença nas predições entre duas iterações consecutivas. O indicador de estabilidade da distribuição da entropia das previsões fornece indicações sobre a igualdade das medianas dessas distribuições entre duas iterações consecutivas.

Espera-se que esta estratégia permita identificar exemplos de todas as classes, independentemente da sua distribuição, sendo capaz de gerar classificadores precisos com um esforço de etiquetação inferior ao das abordagens atuais.

Os resultados da avaliação efetuada mostram que o d-Confidence supera outras abordagens na identificação de exemplos cobrindo todas as classes. Os ganhos são particularmente notórios em presença de distribuições enviesadas. Os classificadores construídos com o d-Confidence apresentam também ganhos ao nível da precisão na maioria dos casos. No entanto, em algumas situações, a redução do esforço de etiquetação necessário para cobrir todas as classes é obtida à custa de uma precisão mais baixa.

O critério de paragem aqui proposto apresenta um desempenho superior às restantes abordagens analisadas. É um critério robusto que gera indicações de paragem de forma consistente quando a utilidade dos exemplos não etiquetados ainda disponíveis é baixa.

A estratégia de aprendizagem ativa proposta nesta tese, como um todo, incluíndo o critério de seleção e o critério de paragem, gera classificadores precisos, capazes de reconhecer todas as classes, a um custo reduzido em comparação com outras abordagens atuais.

Abstract

In some classification tasks, such as those related to the automatic building and maintenance of text resources, it is expensive to obtain labeled instances to train a classifier although it is common to have massive amounts of data available at low cost. Unlike supervised learning, that requires a fully pre-labeled training set, active learning allows asking an oracle to label only the most informative instances given the specific purpose of the learning task and the available data. Moreover, active learning is an iterative process that may be halted when the potential utility of the unlabeled instances remaining in the working set is low. Active learning generally requires a lower labeling effort to build accurate classifiers than supervised learning.

However, common active learning approaches assume the availability of a pre-labeled set, covering all the target classes, to initialize the learning process. The labeling effort required to build this initialization set is not generally considered when analyzing the performance of the learning process. When in presence of imbalanced class distributions, identifying labeled instances from minority classes might be very demanding, requiring extensive labeling, if queries are randomly selected. Nevertheless, these minority classes are the most critical to certain classification tasks, such as, detection of fiscal fraud and rare diseases diagnosis. In such circumstances, evaluating the performance and building a classifier based exclusively in accuracy might not be appropriate since an accurate classifier might still fail to identify minority classes – the critical ones – with a little impact in accuracy.

A novel approach to active learning is required in order to comply with these cases. Besides accuracy, it is also important to assure that the classifier being built is aware of all target classes irrespectively of their distribution. It is our belief that it is possible to develop an active learning strategy that builds accurate classifiers being aware of all the target classes at a reduced labeling effort – that is, at low cost – when compared to current approaches.

In this thesis we propose a strategy for active learning that comprises an active learning criterion to select queries and a stopping criterion to halt the learning process when the utility of the remaining unlabeled instances is low. D-Confidence, our query selection approach, is based on a query selection criterion that aggregates the posterior classifier confidence and the distance between unlabeled instances and known classes. This criterion is biased towards instances belonging to unknown classes – low confidence – that are located in unexplored regions in the input space – high distance to known classes. The stopping criterion in our strategy, hcw, is an ensemble of classification gradient and steady entropy mean, two base indicators of the utility of unlabeled instances. Classification gradient provides evidence on the differences of the predicted labels between two consecutive iterations of the learning process. Steady entropy mean provides information on the stability of the distribution of the entropy of predictions between two consecutive iterations.

This strategy is expected to identify exemplary instances from all the target classes, independently of their frequency, being able to train an accurate classifier while requiring a reduced labeling effort when compared to common active learning approaches.

The main results from our evaluation show that d-Confidence outperforms state-of-the-art approaches in the identification of exemplary instances from all classes. The improvements are mainly evident in imbalanced data. The accuracy of the classifiers built with d-Confidence improves over other approaches in most situations. However, in some cases, the improved representativeness is obtained at the cost of accuracy.

The hcw stopping criterion significantly outperforms other state-of-the-art approaches used for evaluation. It is a robust criterion, triggering consistent stop signs when the utility of the remaining unlabeled instances is low.

The d-Confidence strategy as a whole – including both query selection and stopping criteria – generates accurate classifiers, being able to recognize all target classes, with a reduced cost when compared to state-of-the-art approaches.

Acknowledgements

The last but not the least. One of the first thoughts that came into my mind a few years ago, when I began my PhD, was how nice it must be to have the research concluded, the thesis written and, finally, just have to write the acknowledgments. By that time, I had not a clue on the amount of contributions without which I would have never reached this point.

Now, that I am, finally, really writing the acknowledgments, I am experiencing an intense mixture of feelings. I am thrilled for having finished writing, I am nostalgic because one important period in my life is coming to an end and I am anxious waiting for the jury day. For all this, these last pages are as important as the rest (except that I am more relaxed since it will not be subject to review from my supervisors nor will it be subject to discussion with my jury).

My name is written in the front page of this thesis. That is a great responsibility. I am just the lucky one representing the efforts from many people who made this possible.

My deep thanks, my deep recognition forever are due:

To Alípio Jorge, my supervisor, for the opportunity, for all the advice, patience, unconditional support and sharing.

To Rui Camacho, my co-supervisor, for all the effort and availability to promptly answer to any request.

To Professor Eugénio Oliveira, the Director of the Doctoral Program in Informatics Engineering, for welcoming me and for being always available to solve any problem.

To the School of Engineering of the Polytechnic of Porto, ISEP, for granting the funds and the working conditions required to support this thesis.

To the Laboratory of Artificial Intelligence and Decision Support, LIAAD-INESC Porto L.A., for hosting me as a researcher and supporting this thesis.

To Fernanda Moura, and Embrapa, Empresa Brasileira de Pesquisa Agropecuária, in general for organizing and supporting my stay there, in October 2008, and for the invitation to join the TIENA research project.

To Sarabjot Anand and Tao Li, from the Warwick University, for supporting my stay there, in June 2008, and for all our debates and sharing.

To Robson Motta, from the Instituto de Ciências Matemáticas e de Computação of the University of São Paulo, for the fruitful cooperation and for being always available to discuss and to share.

To Zélia Prior, from the Academic Services of the Faculty of Engineering of the University of Porto, for all the precious help with the bureaucratic hurdles.

To Paulo Ferreira, my colleague, for the providential help when I was about to desperate with LaTeX.

To my wife, for the constant encouragement, for making me believe when things look unattainable, for teaching me setting priorities, for sustaining everything despite my absence during many important moments.

To my daughter, for managing herself and understanding when I was not able to provide all the time she needed and deserved, for her encouragement, for putting up with all my “windy” reactions with a smile in her face, for her care and good mood.

To my mother, for the many kilometers she drove to assist us whenever needed, for the constant concern and care.

To Dias Henriques, for all the immediate help whenever needed and for assuring the conditions for my serenity.

To my grandmother, for her constant care since ever, for being there.

To my friends Fernando and Mário, for keeping inviting me over and over again to Friday lunches despite being continuously declined.

To François Carneiro, Alexandre Rousseaux, Vânia Lopes, Sophie Chapman, Pedro Ribeiro, Carlos Correia, Marcelo Bateira, Els Bockaerts, Matthias Vermeiren, Joachim Seminck, Ricardo Lapa, Bruno Queirós, Alexandra Alves, Daniel Gonçalves, Augusto Cruz, Luiza Gabriel and Tiago Oliveira, my MSc and BSc students, for their commitment and their hard work on projects related to my research during the last few years.

Nuno Escudeiro

Em memória de Mário Augusto Escudeiro

Contents

List of Figures

List of Tables

List of Algorithms

Abbreviations and Symbols

A	Cost of querying the oracle for one label
A^2	Agnostic Active learning
AL	Active learning
α	Significance level for statistical tests
B	Utility of one unlabeled instance in U, the value of the improvement in the performance of the classifier induced by adding a new query to the training set. Opportunity cost for not querying
balee	BalancedEE
C	Set of target classes
c	Refers to confidence in Tables and Figures
C4.5	Algorithm to generate decision trees
CART	Algorithm to generate decision trees
cgr	Classification gradient stopping criterion
CHAD	Algorithm to generate decision trees
C_i	Set of target classes known at iteration i, $C_i \subseteq C$
c_k	A specific target class, an element of C
$ClassDist(x_j, c_k)$	Aggregation function. Computes d_j^k, the distance between unlabeled instance x_j and all labeled instances belonging to class c_k
$conf_i(u_j, c_k)$	Posterior confidence on class c_k given u_j
cvar	Variance model stopping criterion
d-Confidence	Refers to both the active learning criterion and the active learning strategy proposed in this thesis
dc	Refers to d-Confidence in Tables and Figures
$dconf_i(x_j, c_k)$	Marginal posterior d-Confidence of class c_k given x_j
$dConf_i(x_j)$	D-Confidence of instance x_j at iteration i
d_j^k	Distance between one unlabeled instance x_j and all labeled instances belonging to class c_k
E	Oracle, domain expert providing labels on request
e	Refers to error in Tables and Figures
ECOC	Error Correcting Output Code
EM	Expectation Maximization
ff	Refers to farthest-first in Tables and Figures
fh_k	Number of queries required to identify the first instance belonging to class c_k; first-hit of c_k
h	Hypothesis generated by the learning algorithm, a classifier learned for C
HCAC	Hierarchical Confidence-based Active Clustering
hck	Hybrid Classification gradient Kolmogorov-smirnov stopping criterion

hcw	Hybrid Classification gradient Wilcoxon stopping criterion
HKLD	History Kullback-Leibler Divergence
hs	Hierarchical Sampling
HUS	History Uncertainty Sampling
i	Learning process iteration index
IBL	Instance Based Learning
ID3	Algorithm to generate decision trees
IE	Information Extraction
IR	Information Retrieval
ISC	Intrinsic Stopping Criterion
K	Number of target classes, #C
kc	Number of known classes; number of classes that have representative instances in the labeled set, L
kff	Kernel Farthest First
KNN	K-Nearest-Neighbors
KQBC	Kernel Query By Committee
L	Labeled set, set of labeled instances
L_1	Labeled set at iteration 1, set of pre-labeled instances used to initialize the learning process
LDC	Label Disclosure Complexity
ldc	Refers to LDC in Tables and Figures
L_i^k	Set of labeled instances known at iteration i that belong to class c_k
maxc	Max-confidence stopping criterion
MI	Exploitation mode
mine	Min-error stopping criterion
MMC	Maximal loss reduction with Maximal Confidence
MR	Exploration mode
N	Number of instances in the working set, $\#W$
N_1	Number of pre-labeled instances used to initialize the learning process
NER	Named Entity Recognition
NG	Newsgroup dataset
n_k	Number of instances in the training set that belong to class c_k
NNET	Neural Network
ovru	Overall uncertainty stopping criterion
PBAC	Prototype Based Active Classification
POS	Part-Of-Speech
QBC	Query By Committee
q_i	Query selected at iteration i
R52	Reuters-21578 dataset
RBF	Radial Basis Function
RPART	Decision tree base classifier
sek	Steady Entropy Kolmogorov-smirnov stopping criterion
sew	Steady Entropy Wilcoxon stopping criterion
simple	Query selection criterion that selects the query lying in the SVM margin closest to the dividing hyperplane
SVM	Support Vector Machine
TF$\times$IDF	Term Frequency Inverse Document Frequency weighting
U	Unlabeled set, set of unlabeled instances
UCI	UCI machine learning repository

$V()$	Variance
W	Working set, set of all instances available to learn
x_j	Instance in W
y_j	True label of instance x_j
$\hat{y}_j$	Predicted label of instance x_j

Chapter 1

Introduction

The Web is a comprehensive, dynamic, permanently up to date repository of information – readily available for automatic processing by digital means – regarding most of the areas of human knowledge (Hu, 2002) and supporting an increasingly important part of commercial, artistic, scientific, public and personal transactions, rising a very strong interest from individuals, as well as from private and public institutions, at a universal scale. The lack of any solid editorial control over Web publishing is probably one of its main advantages contributing to its success but it is also one of its main pitfalls demanding for costly validation efforts in order to assure the quality of the information being retrieved.

But the Web is not the only source of valuable information being widely used on behalf of a better life quality. In our daily lives data is produced, gathered, processed and used by a ever growing number of equipments and systems. Mobile devices and communication systems in general, cars and other vehicles, labor equipment, sensors for a variety of purposes like pollution or disabled people monitoring, surveillance systems, financial and banking monitoring systems, health equipment and energy distribution are just a few examples.

Information is a core asset to improved quality of life and social progress (Atkinson and Castro, 2008).

However, as the potential benefit of information increases with the amount of information available, also does the cost of retrieving useful information for a particular purpose. Investigating how to conduct this retrieval process such that the cost-benefit ratio is favorable becomes more important as more information is available. The ability to use information efficiently is a key issue to innovation and improvement in our daily lives. Information availability is a necessary condition but not a sufficient one. The efficient use of information depends on its proper organization for a specific need[1]. There must be a close mapping between organizational and domain concepts that conforms to specific purposes.

Assigning previously defined classes to instances is a well established and effective approach to organize collections of instances – the Dewey Decimal System (Dewey, 2004), for instance, is a system of library classification in use since 1876. This task is known by *classification* in the machine learning field. When done by hand, classification is effective for small collections. However, this manual process is not scalable. Automatic classification fixes this scalability problem allowing people to take advantage of the amount and ubiquitousness of digital information available these days.

Nevertheless, automatic classification systems require exemplary instances – previously classified by experts – that are representative of the target concepts. Automatic classification will only be useful after building a classification model that is aligned with the target classes given the specific purposes of the classification task. Building classification models in machine learning is usually based on a set of pre-labeled exemplary instances. Classification models are then able to assign categories to new instances replacing the domain expert at a lower cost – although with a certain loss in accuracy. Automatic classification has a cost that is proportional to the amount or pre-labeled instances required to build the classification model. This cost is related to the effort that is required from domain experts to retrieve and classify a set of exemplary instances.

1.1 The problem

Effective classification requires a set of pre-labeled instances covering the target domain in breadth – all target classes are represented thus contributing to class-completeness – and in depth – the pre-labeled instances from each target class constitute a sound representative sample thus contributing to accuracy. The work reported in this thesis is related to the construction of classification models, addressing class-completeness and accuracy, at low cost.

[1]A traditional phone list, sorted by alphabetic order of names is very efficient when looking for the phone number of a person whose name is known. But, what if we were searching for the name of a person whose phone number is known? In such case, the former organization is as useless as a random list. Organizing information according to specific needs is a core issue regarding its usefulness.

The specific problem being addressed is the effective classification of collections of objects. Effectiveness in this sense means (a) awareness of all the target classes – class-completeness, (b) accuracy and (c) low cost. Our aim is to build effective classification models that can be used to organize collections of objects according to specific needs at low cost in the absence of any prior description of the target concept.

1.2 Hypotheses

We assume that there is no pre-labeled set of instances from which to initialize a classification model but only a pool of instances that is representative of the target concept. Unlike most current approaches, that assume the availability of such a pre-labeled set at no cost, we consider that the task of labeling representative instances to initialize the learning process is an indispensable part of the construction process itself. Our purpose is to build accurate and class-complete classification models, irrespectively of class distribution, with less effort from domain experts (lower cost) than that required by state-of-the-art approaches.

Active learning (AL) is an adequate learning setting for our purposes. Unlike supervised learning, which requires a fully pre-labeled set in advance to learning a classification model, AL builds the pre-labeled set iteratively as it learns the classification model. The instances to label are selected at each iteration of the learning process based on the current evidence which is incorporated at each iteration. AL selects the most informative instances, given current evidence, potentially reducing the number of pre-labeled instances required to build the classification model when in comparison to the supervised setting. The selection criterion to be used may be tailored according to specific purposes.

A core concern in AL relates to the compromise between exploration and exploitation that is implemented by each query selection criterion. Query selection may be biased towards exploration – selecting queries that will contribute to explore unknown regions in input space – or exploitation – selecting queries that will contribute to fine tune the current learning function. Balancing exploration and exploitation has impact on the performance of the learning strategy.

Another relevant characteristic of AL addressing our needs relates to the fact that AL processes can be stopped when no further improvements are expected, something that is not possible in supervised learning since the pre-labeled set is prepared in advance. All of these characteristics make AL an adequate setting given our purpose – build accurate classification models being aware of all the target concepts at low cost, that is, demanding a reduced labeling effort.

Current AL approaches to classification are mainly focused on improving accuracy. The availability of a pre-labeled set including representative instances from all the target classes is generally assumed. Under such an assumption, accuracy is the natural concern since class-completeness is not an issue. However, it is not usually described how the class-complete pre-labeled set required to initialize the learning process is obtained. The cost of such a class-complete pre-labeled set may be high – depending on the target domain and its typical class distribution – and its feasibility should be questioned. Moreover, when in presence of imbalanced class distributions, we may generate accurate classifiers that do not cover under-represented classes since those have a marginal contribution to error. Nevertheless, those minority classes might be critical. Failing to learn them may jeopardize all the efforts in building the classifier.

We are concerned with building low cost classifiers that are aware of all of the target classes. The cost of instantiating the pre-labeled set required to initialize the learning process must also be accounted for since it might be a significant part of the total cost. Unlike generally accepted in the AL literature, we do not assume the availability of a pre-labeled set including representative instances from all the target classes at zero cost. Our classifiers are initialized with a minimal pre-labeled set having two labeled instances from two distinct classes selected from the initial pool at random. During the AL process we keep querying with a focus on the early – low cost – identification of exemplary instances from all the target classes. Accuracy is also a concern but it is not the only one. We aim to build class-complete accurate classifiers.

1.3 Thesis proposal and expected results

We propose an AL strategy aimed at building class-complete accurate classifiers at low cost that includes a query selection criterion, *d-Confidence*, and a stopping criterion, *hcw*.

D-Confidence selects queries – instances to label – based on a criterion that aggregates confidence and distance to known classes. The query selected by d-Confidence at each iteration is one unlabeled instance for which the current classifier is not certain about its class – low confidence – and that is simultaneously located far apart from the classes that are known – high distance.

At an initial stage of the learning process, when in presence of a sparse training set, d-Confidence is expected to query regions in the input space that are far from what is already known thus contributing to a fast coverage of the input space. At this initial stage, the distance factor of d-Confidence is expected to be predominant introducing a bias favoring exploration of the input space and contributing to the early retrieval of representatives of all classes to learn.

As the input space gets populated with labeled instances and target classes become represented in the training set, the confidence factor is expected to become predominant. At this stage, d-Confidence is expected to shift to an exploitation bias querying unlabeled instances contributing to sharpen decision boundaries.

This nature of d-Confidence is expected to assure a dynamic compromise between exploration and exploitation that will contribute to build accurate classifiers being aware of all target classes at low cost.

AL processes run until some stopping condition is met. A trivial stopping condition is the exhaustion of the pool of instances available to learn. However, efforts to build a classifier at low cost – with few queries – may be useless if such a naive stopping condition is applied. If the learning process keeps querying the oracle yet not improving the classifier, we are adding costs at no benefit. From the point of view of cost-benefit, the decision on when to stop querying is as important as selecting useful queries.

We propose three base stopping criteria plus two hybrid ones having the aforementioned aspects into consideration. One of the base stopping criteria is based on changes in predictions between consecutive iterations. The other two are based on changes in the distribution of the entropy of predictions between iterations, one being evaluated by a Wilcoxon test for equality of medians and the other by a Kolmogorov-Smirnov test for equality of distributions.

The hybrid stopping criteria are ensembles of the base criteria merging changes in predictions and changes in the distribution of the entropy of predictions. Changes in the predictions provide evidence on the predictions themselves but do not inform on their confidence. Changes in the entropy of predictions provides evidence on the distribution of the current classifier confidence however, with no connection to the predictions themselves. Each of these conditions on its own is a necessary stopping condition but not a sufficient one. The ensemble of the two is expected to be sufficient.

Our hypotheses were investigated through an empirical research methodology.

1.4 Main results

The main results from our experimental evaluation show that d-Confidence exhibits a significant potential to the early coverage of input space. D-Confidence retrieves exemplary instances from all the target classes at a lower cost than its baseline criteria and other state-of-the-art AL approaches. This improvement, in general, has no negative impact on accuracy. In fact, in many cases, there is an improvement in accuracy. In some other cases, the improvement in class coverage is made at the cost of accuracy.

D-Confidence is characterized by a dynamic shift between exploration and exploitation that arises from its nature and does not require any tuning, that is, no overhead cost. The compromise between exploration and exploitation is balanced by the geometrical properties of the input space itself. This characteristic of d-Confidence gears a faster coverage of the input space while simultaneously generating accurate models.

Concerning the stopping criteria, our evaluation indicates that the hybrid criteria proposed in this thesis outperform the other stopping criteria under evaluation. There is a clear dominance of hybrid criteria, notably *hcw*, regarding both cost and predictive ability.

1.5 Thesis structure

The remaining of this thesis is organized in seven chapters. Chapters 2 and 3 refer to the state-of-the-art of the AL field. In Chapter 2 we review this field of machine learning analyzing its evolution from its inception in the 80s. Then, in Chapter 3 we describe in more detail two aspects of AL that are fundamental to our work – query selection strategies and stopping criteria.

Chapter 4 reviews the main techniques in text classification. We refer to the pre-processing phase, reviewing the preliminary aspects required for the automatic processing of text documents. These include text preparation and models for text representation. Next, we review common text classification settings, text classifiers and performance indicators.

Chapter 5 elaborates a formal description of the problem studied in this thesis, providing a general setting to promote discussion and further developments.

Chapters 6 and 7 are core to this thesis. They describe in detail our main contributions and the evaluation of d-Confidence and the stopping criteria. The reader is assumed to be familiar with the formal description of d-Confidence provided in Chapter 5.

In Chapter 8 we review the main contributions and describe opportunities for further research arising from our work.

Chapter 2

Active Learning Retrospective

Machine learning (Mitchell, 1997) is a scientific field of artificial intelligence. It covers the research and development of models and computer algorithms based on patterns extracted from empirical data. In this thesis we address one of the major areas of machine learning: classification problems.

In a classification problem, labels – the classes – are assigned to instances. Class labels come from a predefined set, previously established by the user. The assignment of classes to a given instance is done by a classifier, based on a classification model. The classifier is inferred by a learning algorithm from the empirical data. The models[1] are constructed from a subset of the instance space – the training set. Classification models are hypotheses of the target concept that are consistent with the observed training instances – as perceived by the learner given evidence on the target concept. In the context of this thesis we assume that instances are described by a set of features.

2.1 Machine learning settings

There are several machine learning settings suited for classification each with its own pros and cons given the specific problem at hand.

[1]Instance based learning (IBL) is a specific learning setting that does not require the induction of a model. IBL just stores the training set that, in classification problems, is somehow used to assign labels to instances.

Supervised learning This is a setting where the learner generates hypotheses mapping the input features – the set of features describing instances – to a pre-established set of classes (Cunningham P, 2008). Supervised learning algorithms require a set of pre-classified (pre-labeled) instances from which the learner generates the hypotheses. The need to have a fully labeled training set is a major drawback of such a setting, especially when the cost of labeling is high.

Unsupervised learning This setting does not require any pre-labeling; learning is achieved exclusively from the input features (Ghahramani, 2004). Unsupervised algorithms seek to realize what is the underlying structure governing the input features' space. Unsupervised learning generates models relying on the most salient patterns found in the input feature space which may not properly map the target concept. Unsupervised classification algorithms (Karakos et al., 2005; Sona et al., 2006; Sigogne and Constant, 2009) are commonly based on clustering techniques which, in the specific field of text categorization, assume the clustering hypothesis (van Rijsbergen, 1979) – documents having similar content are also relevant to the same topic. Unsupervised learning does not require any labeling but users have no chance to tailor clusters to their specific needs and there is no guarantee that the induced clusters will be aligned with the classes to learn. This lack of guidance towards user needs during the training phase is a major drawback of unsupervised algorithms from the point of view of the work presented in this thesis.

Semi-supervised learning The learning settings above rely exclusively on a fully labeled dataset – in the case of supervised learning – or on a completely unlabeled one – unsupervised learning. Combining both labeled and unlabeled data to take advantage of this combination, and to leverage the information contained in unlabeled data, is the goal of semi-supervised learning (Nigam et al., 2000; Chapelle et al., 2006; Zhu, 2008; Zhu and Goldberg, 2009). Semi-supervised algorithms usually rely on a small set of labeled data and a large set of unlabeled data. A simple heuristic approach to semi-supervised learning consists in a two step learning process. In the first step a classifier is trained based on the labeled data only. This classifier is then used to classify unlabeled data. The instances where the current classifier is most confident about are added to the labeled set assuming the predicted labels are correct.

Active learning In supervised learning the training set can be obtained by some method, such as random sampling, without any arbitration by the learning algorithm, in which case we refer to *passive learning*, or by some specific sampling criterion under control of the learning algorithm, biased according to some desirable properties of the training set, in which case we refer to *active learning*. Active learning (Angluin, 1988; Cohn et al., 1994; Roy and McCallum, 2001; Muslea et al., 2006) is a particular form of supervised learning where instances to label are selected by the learner through some criteria aimed at reducing the labeling complexity (Hanneke, 2007). Labeling complexity is defined as the number of label requests that are necessary and sufficient to learn the target concept.

2.2 Active learning

Several classification tasks – for instance those involving unstructured data, such as, speech recognition, text and Web pages (Settles, 2009) categorization, images and music retrieval and filtering – require efficient classification algorithms due to the high labeling cost, on one side, and the vast amount of available, but unlabeled, data, on the other. Efficiency in such circumstances refers to a trade-off solution between high accuracy and comprehensiveness, on one hand, and low labeling effort, on the other. Active learning (AL) is an appropriate learning setting for this scenario given the chance to develop learning strategies aiming at a desirable trade-off.

In AL, the learner is allowed to ask an oracle (typically a human) to label instances – these requests are called *queries*. The most informative queries, given the goals of the classification task, are selected by the learning algorithm unlike passive learning where training instances are selected at random. AL can be performed in several distinct settings which will be covered in Chapter 3. The core idea in AL is to estimate the value of labeling unlabeled instances. The general learning process in Algorithm 2.1 is the basis for AL classification.

Referring to Algorithm 2.1, W is the working set, a representative sample of instances from the problem space. L_i is a subset of W. Members of L_i are the instances in W whose labels are known at iteration i. At iteration i, U_i is the (set) difference between W and L_i, $U_i = W \setminus L_i$, i.e., the set of unlabeled instances in the working set; h_i represents the classifier learned at iteration i; q_i is the query selected by the active learner at iteration i. A specific instance is represented by

Algorithm 2.1 General AL algorithm

```
 1: Input: $W$, set of unlabeled instances $x_j$; $f_q()$ query utility function
 2: Output: $h_i$, learned classifier
 3:
 4: Initialize $L_1$, $U_1$ from $W$
 5: $i = 1$
 6: while stopping criteria does not hold do
 7:     $h_i = learn(L_i)$, generate classifier $h_i$ using current labeled set $L_i$
 8:     Use $h_i$ to classify instances in the current unlabeled set $U_i$
 9:     $q_i = \underset{x_j}{\text{argmax}} f_q(x_j), x_j \in U_i$, select $q_i = x_j \in U_i$ maximizing query utility
10:     Ask the oracle for the label of $x_j$, $y_j$
11:     $L_{i+1} = L_i \cup < x_j, y_j >$
12:     $U_{i+1} = U_i \setminus x_j$
13:     $i++$
14: end while
15: return $h_i$
```

$< x_j, y_j >$ where x_j is the set of descriptive features and y_j is its true class (label).

All AL approaches analyze unlabeled instances and select the most useful ones once labeled. The general idea of AL is to estimate the value of labeling unlabeled instances, i.e., the value of queries. Query selection may be based either on a generative strategy (Angluin, 1988) or on a discriminative strategy (Li et al., 2010).

In a generative strategy – the *query construction paradigm* – queries are artificially synthesized (Angluin, 1988; Baum, 1991). Generative strategies are suitable for the case where artificially synthesized instances make sense to the oracle providing labels. This assumption makes generative AL strategies unsuited for the generality of unstructured data domains as is the case of text corpora.

Discriminative strategies – the *query filtering paradigm* – select queries from a given distribution. These strategies are suitable when the distribution of the available data might be different from the target distribution and also when artificially synthesized instances are not meaningful to the oracle as is usual in text categorization relying on the bag-of-words model (Harris, 1954). Two approaches are common under the query filtering paradigm: *pool based active learning* (Lewis and Gale, 1994; McCallum and Nigam, 1998) – where queries are selected from a static pool of data – and *stream based active learning* (Zhu et al., 2007, 2010c; Chu et al., 2011) – processing data streams and deciding online whether or not to query each new incoming instance.

Applying AL techniques to classification involves a set of specific challenges that add to the common issues arising in general classification problems. In general, automatic classification involves a number of distinct tasks, including the definition of the main goal of the learning process, setting the evaluation procedure, gathering training and test sets, defining the data representation model, selecting and tuning the most adequate learner. Specific AL challenges include: retrieving an initial set of labeled instances, establishing the query selection criteria, establishing the stopping criteria, agreeing on a compromise between exploration – finding representative samples in the dataset that are useful to label, focusing on completeness – and exploitation – sharpening the classification boundaries, focusing on accuracy. Decisions on these issues are directed by the goals of the classification problem at hand. These aspects of AL are addressed in the following sections.

2.3 A retrospective view

Research in AL became popular in recent years. The massive quantity of digital information that has become widely available during the last years triggered its popularity. Nevertheless, the AL paradigm, applied to machine learning, has been in use for over 30 years.

In 1984, Valiant describes *machine learning* – the process of "knowledge acquisition in the absence of explicit programming" – as consisting of (i) an information gathering mechanism and (ii) a process to explore the concept space that can be learned in a reasonable (polynomial) number of steps (Valiant, 1984). Performance and learnability are concerns already perceived from this remark on "reasonable" complexity. It is worthwhile noting the core role assumed by the information gathering mechanism in machine learning, from its inception. AL contributes to the feasibility of the learning process by reducing the extent of the input needed to learn. In this work from Valiant, the learning paradigm is extended to include queries, in the current sense of the term in AL – the learner supplies a set of feature's values and asks for an output that is provided by an oracle.

In the following we will describe the most relevant landmarks of AL in the machine learning field.

2.3.1 Inception, 1980's

The term *active learning* has been originally coined in the educational field in 1991, as a corollary of the broad discussion around instructional paradigms that occurred during the 80's, referring to the instructional activities involving students in doing things and thinking about what they are doing (Bonwell and Eison, 1991). In the educational field, AL has always been associated to seeking new information, organizing it in meaningful ways and further exploiting it (Allen D., 2005), the very same concerns of the machine learning field.

Learning from queries, 1981 A few years earlier, during the 80's, the paradigm had already been applied to machine learning, although not explicitly tagged as AL. In 1988, Dana Angluin (Angluin, 1988) proposes a formal framework to organize and study several types of queries and their value for machine learning tasks. Six distinct types of queries were established: membership, equivalence, subset, superset, disjointness, and exhaustiveness queries. The answer for each one of these queries, except for the membership type which returns a single Boolean (True/False), is composed by a Boolean result and a counterexample in case of a negative answer.

Before that, a few learning systems based on queries had been proposed. For instance, in 1981, Shapiro (Shapiro, 1981) and, in 1986, Sammut et al. (Sammut and Banerji, 1986) both propose generative approaches to learn new concepts from previous knowledge. However, Dana Angluin did the first formal description of the AL paradigm in the machine learning field.

Farthest-first, 1985 One of the baseline criteria in our work, farthest-first was introduced in 1985 (Hochbaum and Shmoys, 1985) to find an efficient sub-optimal solution to the k-center problem[2]. In this approach, an initial instance is selected at random. From there on, we select the instance that is farther apart from the previously fetched instances until we have k instances. These k instances – collectively known as the farthest-first traversal of the data – are set as cluster centers. The remaining instances are then assigned to the closest center. The distance between

[2]The k-center problem is defined as follows (Mihelič and Robič, 2005): Let G = (V, E) be a complete undirected graph with edge costs satisfying the triangle inequality, and k be a positive integer not greater than $|V|$. For any set $S \subseteq V$ and vertex $v \in V$, define d(v, S) to be the length of a shortest edge from v to any vertex in S. The problem is to find such a set $S \subseteq V$, where $|S| \leq k$, which minimizes $max_{v \in V} d(v,S)$

an instance and a set is the minimum distance between the instance at hand and each of the instances belonging to the set. Farthest-first traversal may provide a set of seeds to build hierarchical clustering with certain performance guarantees (Dasgupta and Long, 2005).

Learnability, 1988 In 1998, Pitt et al. focus on learnability issues, stating that some concepts are not learnable just by instances when we have no prior knowledge on the base distribution (Pitt and Valiant, 1988). This same problem is discussed by Eisenberg and Rivest (Eisenberg, 1991), that set a bound on the degree to which membership queries (Angluin, 1988) may improve generalization when the underlying distribution is unknown. 1988 was also the year of the seminal paper from Dana Angluin (Angluin, 1988) setting for the first time a formal framework for AL in the field of machine learning.

In 1990, Kinzel et al. (Kinzel and Ruján, 1990) show evidence on the ability of simple perceptron learners[3] to strongly enhance generalization by allowing the network itself to select the training examples.

In 1991, Baum (Baum, 1991) proposes a hybrid algorithm that learns a binary classifier from pre-labeled instances and artificially synthesized queries. Two groundwork instances – one positive and one negative example – are randomly selected from the pre-labeled set to start with. Then, a query is generated half way between those two. This generated instance is labeled by the oracle and replaces the previous groundwork instance with the same label. The process iterates reducing the distance between the selected instances and the separating hyperplane in each and every iteration.

2.3.2 Rudiments, early 1990's

Selecting queries based on the distance to labeled instances, farthest-first, or on the posteriors generated by the current classifier, uncertainty sampling, are, together with Query By Committee, among the main groundwork approaches to AL. In the early 90's, AL is explicitly assumed as a research area in the machine learning field.

[3] A perceptron is a simple type of neural network, developed in late 50's and early 60's mainly by Frank Rosenblatt, that deploys a linear binary classifier.

Query By Committee, 1992 Seung et al. (Seung et al., 1992) propose *Query By Committee* (QBC) in 1992, a new paradigm for AL, that relies on an ensemble of learners – the committee – to select a query as the instance with the highest disagreement among the committee. While most of the previous work in the field had been focused on improving generalization error, Seung et al. clearly assume their focus on minimizing the number of queries that are required to learn a concept. This same indicator – the minimum number of queries required to learn a concept – will lately be named *labeling complexity* by Steve Hanneke in 2007 (Hanneke, 2007).

Understanding the general principles that govern, or should govern, the query selection criteria is another major issue in AL, until then overlooked, that was also brought up by the authors. Their approach is supported on entropy and information gain (Shannon, 1948). The authors claim that the information value of a query can be estimated from the disagreement among the committee and that by maximizing the disagreement the information gain can be high. They prove the advantages of QBC when compared to random sampling (Cochran, 1977) in two toy classification problems. When using QBC, the information gain of a query approaches a finite value, other than 0, when the number of queries goes to infinity. This asymptotically finite information gain leads to an exponentially decreasing generalization error. When relying on random sampling, the information gain tends to 0, as the number of queries increases, and generalization error decreases as an inverse power law in the number of queries, therefore performing poorer than QBC. In 1997, Freund confirms the exponentially decreasing trend in generalization error (Freund et al., 1997).

Further work on QBC, by Freund et al. (Freund et al., 1993) in 1993, showed a sharp increase in the utility of the selected queries when the size of the committee increases above two.

In 1992, Mackay proposes a query selection strategy based on the estimated utility of particular sub-areas within the uncertainty region (MacKay, 1992). Queries come from the most informative sub-areas that are dynamically bounded and evaluated for utility in each iteration.

Liere et al. (Liere and Tadepalli, 1997) apply QBC in a text categorization task relying on Winnow as a base classifier. Winnow classifiers (Littlestone, 1988) are particularly suited to high-dimensional feature spaces with many irrelevant features as is common in text corpora. The empirical results show significant reduction in the labeling effort – by one to two orders of magnitude – when comparing QBC to a single Winnow classifier. The authors experimented with several strategies to select queries. The best AL strategy achieves the same accuracy as a supervised learner while requiring only less than 3% of the labeled instances. An interesting finding relates to the committee dimension. In this work, the committee is composed by seven members, which was determined by trial and error. The authors claim that an excessively large number of committee members – committees of as much as 1000 members have been tested – imply high computational costs with no significant improvements in accuracy while a too small committee gets dominated by one or two of its members and the overall performance of the committee approaches that of a single member.

In 1998, McCallum et al. (McCallum and Nigam, 1998) merge Expectation-Maximization (EM) (Dempster et al., 1977) with QBC reporting significant reductions in the labeling effort when compared to its baseline approaches, EM and QBC, alone. EM is used to provide labels for unlabeled data that are used to support the query selection process of AL. Committee disagreement is measured using Kullback-Leibler divergence to the mean (Pereira et al., 1993) that takes into consideration not only the top ranked class by each committee member – as in vote entropy – but also the certainty of disagreement computed from the committee members class distributions.

Uncertainty sampling, 1994 Lewis and Gale (Lewis and Gale, 1994) propose *uncertainty sampling* in 1994, a sequential sampling approach (Ghosh, 1991). In a sequential sampling method the sample size is not fixed in advance and the decision to add, or not, a new query is influenced by previously labeled instances. Lewis and Gale demonstrate that uncertainty sampling reduces the number of labeled instances that are required to generate an effective classifier. In uncertainty sampling the set of queries in each iteration is the set of unlabeled instances whose class membership is sufficiently uncertain by the current classifier.

Uncertainty sampling proved to be effective on text corpora both with logistic regression (Lewis and Gale, 1994) and decision trees (Lewis and Catlett, 1994).

Lewis and Catlett were one of the first to address the computational cost of training online while collecting training data. Training the best suited learner for a given task while simultaneously selecting the instances to train on might be too demanding. In their work, the authors propose a heterogeneous approach that consists in using a low cost classifier – a highly efficient probabilistic classifier – to select the instances that will be used to train the actual learner – a C4.5 decision tree (Quinlan, 1993).

In 1995, Lewis (Lewis, 1995) reinforces the reduction in labeling effort that might be granted by AL when building text classifiers and draws our attention to the risk of biasing the resulting probability estimates.

Relevance sampling, 1994 Relevance feedback (Salton and Buckley, 1990), a common strategy in information retrieval (Mooers, 1950) that significantly increases retrieval effectiveness (Croft, 1995), may be seen as a particular form of AL. Lewis et al. (Lewis and Gale, 1994) call this approach *relevance sampling*. In relevance sampling, users are requested to confirm, or not, the relevance of a given document under their current information need. The documents that users are requested to label are those that the current classifier considers most likely to be relevant.

Active learning is coined, 1994 Although extensive research has been performed on AL during the late 80's and early 90's, the term *active learning* has been explicitly used to refer to the field in machine learning only from 1994 on, in a paper published at the Machine Learning journal by Cohn et al. (Cohn et al., 1994). A preliminary version of this paper had been presented in 1990 (Cohn et al., 1990).

Before that, the AL setting in the machine learning field was commonly referred to by *selective sampling* (Cohn et al., 1994), *learning from queries*, *query learning* and *active data selection*. In statistics the problem of selecting samples to label is typically referred to by *experimental design* (Pronzato, 2008) – a field of study concerned with the design of experiments aiming to minimize the variance of a parameterized model.

Cohn et al., in their seminal paper from 1994 (Cohn et al., 1994), define AL as any form of learning where the learner has some control over the inputs on which it trains. This work, however, is restricted to membership queries. The authors define *region of uncertainty* as the areas that are not determined by the available information, that is, the set of instances in the working set such that there are two hypotheses that are consistent with all training instances yet disagree on the classification of those.

Batch mode active learning, 1994 Batch mode AL – selecting a batch of queries instead of a single one before retraining – is also discussed by Cohn et al. (Cohn et al., 1994), mainly the influence of the batch size in selective sampling. The authors conclude that the queries selected by the selective sampling algorithm are more valuable for smaller batch sizes. The process to select the instances to include in the batch at each iteration does not consider members' diversity and so, it seems natural to have redundant instances present in the batch which leads to low efficiency. Batch mode AL adds a new non-trivial challenge: the need to assure diversity among batch members to avoid redundancy and querying unlabeled instances of low utility – wasted queries.

Iyengar et al. (Iyengar et al., 2000) use an ensemble of classifiers and their predictions to select new queries. A set of weights are computed for unlabeled instances. These weights are proportional to the classification error made by the ensemble, assuming the validity of the predicted labels. A set of unlabeled instances is then selected by sampling using the normalized version of these weights.

2.3.3 Maturity, late 1990's

In the late 90's, the AL field is established in machine learning. The ever growing amount of digital data easily available to all promotes AL and creates the conditions for an increasing interest from researchers.

Optimal solution for pool-based active learning, 1996 AL is quite adequate to analyze unstructured and high-dimensional datasets where the labeling cost is usually high. It soon began to be applied to text classification. Text documents, however, are hard to synthesize by a computer algorithm if we want them to make sense to a human reader. In such a scenario, the most common approaches in the early days of AL, relying on query synthesis, were not adequate – an artificial text constructed by a learning algorithm will seldom be interpretable by a human oracle. The need to filter queries out of a pool of existing unlabeled instances – the pool-based paradigm – instead of the common generative query construction paradigm (Shapiro, 1981; Sammut and Banerji, 1986; Angluin, 1988; Pitt and Valiant, 1988; Baum, 1991; Plutowski and White, 1993; Cohn et al., 1996) arose.

In 1996, Cohn et al. prove that the optimal solution for pool-based AL (Cohn et al., 1996) is possible when using models of mixtures of Gaussians (Titterington et al., 1985) or locally weighted regression (Cleveland et al., 1988). According to their findings, the instance that produces the minimum expected error is the one that minimizes the expected variance of the predictions.

Learning with noise, 1998 It is generally assumed in AL that the oracle is able to provide true labels for any query. Nevertheless, we may be learning from pre-labeled instances that are corrupted by noise.

Kearns (Kearns, 1998) addresses these circumstances proposing the *statistical query model*. In the statistical query model, the oracle provides the probability that an example belongs to a particular class instead of providing actual class labels. This model restricts the way in which a learning algorithm may use a random sample and creates conditions to construct hypotheses that are based on the statistical properties of the working set rather than on the specificities of a particular sample. The author claims that statistical query model of learning is a robust, noise-tolerant learning process allowing to obtain efficient algorithms for several classes of problems.

Text categorization and SVM, 2000 In 2000, Schohn et al. (Schohn and Cohn, 2000) describe a simple AL heuristic, based on SVM classifiers applied to text classification that, surprisingly, assures better performance w.r.t. accuracy and computational cost than a supervised classifier trained on all available data. This is a key finding, showing that, for the essayed text corpora, the accuracy of AL improves as new queries are added until it achieves a maximum that overcomes the maximum that is achievable using passive learning with all the available data. From that point on, the accuracy of AL degrades as new queries are added. Their query selection heuristic estimates the expected change in error from adding a given instance to the training set.

Tong et al. (Tong and Koller, 2002), working with SVM classifiers applied to text, propose selecting queries from the SVM margin that halves the version space (M. and Mitchell, 1982; Mitchell, 1997) at each iteration thus maximizing the reduction in version space – the set of hypotheses that are consistent with all labeled instances. Three methods to select queries – simple margin, MaxMin margin and radius margin – are evaluated as to their efficacy and computational cost. These are distance based approaches, under the pool-based setting of AL, that significantly reduce – over an order of magnitude – the need to label training instances when compared to random selection (the supervised setting). Performance improvements were observed at both inductive and transductive settings[4] (Vapnik, 1998; Joachims, 1999).

[4]In a traditional inductive setting, general rules are inferred from training instances and then applied to any test instances. In a transductive setting, instead of inferring general rules, the patterns observed in the training instances are transfered to specific test instances.

Active learning performance validation, 2000 In 2000, Schohn et al. (Schohn and Cohn, 2000) claim that cross validation does not provide reliable error estimates for an active learner. The rationale for this conviction is that cross validation assumes that the distribution in the training set is representative of the distribution in the test set which does not hold in AL due to its inherent sampling bias, mainly when the labeled set is reduced. However, this assumption is violated whenever the training set is not sufficiently large, independently of which learning setting, active or passive, is being used.

The main concern of Schohn et al. is to estimate when generalization error has reached peak performance. They propose an indirect method that can be applied with SVM base classifiers. Peak performance is assumed to occur when the unlabeled instance closest to the decision hyperplane is no closer than any of the actual support vectors. In such circumstances the SVM margin has been exhausted.

Active learning and instance deletion, dual approaches, 2002 Dataset reduction by deleting instances from the training set is a general solution to the problem of instance selection. Instance deletion may be seen as the dual of AL. In both settings the objective is to retain only the few instances with high utility, ignoring the rest. In instance deletion, this goal is achieved by deleting the redundant, low-utility, instances and keeping the remaining while AL selects the most informative instances and ignores the remaining low-utility instances. Brighton et al., report good performance with reduced data in instance based learning settings (Brighton and Mellish, 2002). One important conclusion from their work is that the performance of the deletion/selection scheme in classification tasks depends on the class structure of the input feature space. Distinct schemes are required to deal with homogeneous and non-homogeneous class structures – homogeneous classes are defined by homogeneous regions in feature space, i.e., instances from the same class are located close to each other.

No universal top performer, 2002 In 2002, Brighton et al. (Brighton and Mellish, 2002) refer the need to select the best AL scheme for a specific problem since there is no universal top performer. Another concern regarding the dominance of one scheme over the others concerns the selection of the best performer at each step of a given AL process – a vertical view as opposed to the horizontal view discussed by Brighton et al. (Brighton and Mellish, 2002). Instead of relying on a single base classifier, as is common, Baram et al. (Baram et al., 2004) propose an approach that relies on an ensemble of base classifiers. The AL process is governed by an algorithm that combines these base learners by evaluating their individual performance and dynamically switching to the best performer at each iteration.

Batch mode revisited, batch diversity, 2003 Batch mode AL is a learning approach that is adequate when the training cost is high. Another motivation to add more than one query to the training set at each iteration is to avoid annoying the user with many consecutive iterations of a single query that might have a negligible improvement in the inferred classification model, not really observable by the user (Chen et al., 2010). In batch mode, instead of retraining at each single query – the traditional AL setting, retraining is done after having queried a batch of instances. This approach poses a new problem. The issue is that diversity within the batch of instances to query is required to avoid querying redundant instances. This desirable diversity is not assured by just selecting the top m most informative instances as seen by the current learner (Schohn and Cohn, 2000; Warmuth et al., 2002). Specific care is needed to assure that all batch members add value on top of the rest.

In 2003, Brinker (Brinker, 2003) proposes a new batch mode approach to AL that incorporates a diversity measure while selecting batch members. This approach selects the queries lying close to the decision boundary that have the largest angles to previously selected candidates. Brinker's approach outperforms previous AL approaches using SVM base classifiers.

Hoi et al. (Hoi et al., 2006) suggest a batch mode approach relying on the Fisher information matrix (Papathanasiou, 1993) to reduce redundancy among selected instances. Li et al. (Li and Sethi, 2006) compute diversity within selected instances from their conditional error. Hoi et al. (Hoi et al., 2009) propose *semi-supervised SVM batch mode*, a new batch mode approach with two objectives in mind: to increase the number of training instances and to assure their diversity to improve SVM performance. Semi-supervised SVM batch mode first learns a kernel function from labeled and unlabeled data. Then, this function is used to identify the most informative and diverse instances to query.

Rare category detection, 2004 Pelleg (Dan Pelleg, 2004) describes a novel AL scenario that addresses a very similar problem to our own. Pelleg explores AL with the purpose of identifying rare categories – experiments are reported with rare categories having as few as 0.002% instances in the data pool – in a setting where no labeled instances from these classes are present in the initial training set. Our work is somehow more general than this one. We focus on gathering exemplary instances for all the classes to learn – irrespective of their frequency – in the absence of any pre-labeled instances, at low labeling cost. The authors propose an interleaving strategy based on a mixture model to fit the data and relying on the degree of ownership of each mixture components w.r.t. unlabeled instances. In each iteration, a batch of 50 instances is selected based on four selection criteria, one of which is the proposed interleaving strategy. No care to avoid redundancy in the batch composition is suggested.

Hierarchical sampling, proposed by Dasgupta et al. (Dasgupta and Hsu, 2008) in 2008, was also applied to the detection of rare categories. The authors report significant gains in the number of queries that are required to discover at least one instance from each class. This latter work is also in line with our own efforts for devising a method capable to swiftly identify instances from unknown classes. Preliminary results have been published by us also in 2008 in a workshop paper (Escudeiro and Jorge, 2008).

Class probability estimates, 2004 The most common use of AL is probably in classification problems, aiming at maximizing accuracy. In many situations, however – for instance, when in presence of unequal misclassification costs – computing class probability estimates is more useful than aiming at high accuracy. Class probability estimates are used to evaluate the expected utility of a set of alternatives, assuming particular relevance in decision making. A few works on the application of AL to class probability estimates show empirical evidence on the improvements assured by active selection of queries. BOOTSTRAP-LV (Saar-Tsechansky and Provost, 2004) requires less labeled instances to produce accurate class probability estimates when compared to the traditional uncertainty sampling strategy (Lewis and Gale, 1994). Melville et al. (Melville et al., 2005) improve over the previous work by using the Jensen-Shannon divergence as a measure of the utility of new queries.

2.3.4 Exploring new directions, early 2000's

Application of AL in stream-based settings and the scalability of QBC are among the new directions addressed by AL research in the early 2000's. Theoretical essays, trying to provide a sound theoretical ground for AL, were also a concern.

Clustering approaches, 2004 Clustering has also been considered to provide an initial structure to data or to suggest valuable queries.

In 2004, Basu et al. (Basu et al., 2004) used AL based on farthest-first to provide labeled data in a semi-supervised clustering setting (Basu et al., 2002). They actively acquire a set of must-link and cannot-link constraints to improve the clustering process. Empirical results over UCI data (Frank and Asuncion, 2010) and text corpora, show that the proposed AL strategy improves over random pairwise queries.

In 2004, Nguyen et al. (Nguyen and Smeulders, 2004) incorporate clustering into AL by learning a classification model from the set of the cluster representatives, and then propagate the classification decision to the other instances via a local noise model. The proposed model allows to select the most representative instances as well as to avoid repeatedly labeling instances in the same cluster.

Adami et al. (Adami et al., 2005) merge clustering and oracle labeling to bootstrap a predefined hierarchy of classes. Although the original clusters provide some structure to the input, this approach still demands for a high validation effort, especially when these clusters are not aligned with class labels.

Huang et al. (Huang et al., 2008) explore the Wikipedia[5] as a background knowledge base to create a concept-based representation of a text document enabling the automatic grouping of documents with similar themes. The semantic relatedness between Wikipedia concepts is used to find constraints for supervised clustering using AL.

In 2008, Dasgupta et al. (Dasgupta and Hsu, 2008) propose *hierarchical sampling*, a cluster-based method that consistently improves label complexity over supervised learning by detecting and exploiting clusters that are loosely aligned with class labels. Their method starts with a hierarchical clustering of the unlabeled pool. Then, random samples from each node of a partition of

[5]http://en.wikipedia.org, accessed on October 2012

the data, given by a pruning of the tree, are queried. Their labels are used to compute the purity of each node in the partition. Nodes with low levels of purity are replaced by their child nodes. This process can be halted whenever required, for instance when a given purity level is achieved at each node. At each iteration, the sampling strategy favors less pure nodes.

Hu et al. (Hu et al., 2009), motivated by similar concerns, propose an AL schema, based on graph-theoretic clustering algorithms. Their approach aims to suppress the lack of ability of common AL approaches to query instances that belong to new classes, that have not yet appeared in the training set. This motivation is common to our own.

The AL setting in general is considered to be an appropriate setting for learning from imbalanced data (Kapoor et al., 2010a).

Xu et al. (Xu et al., 2003) proposed *representative sampling* in 2003. Representative sampling is an AL method that applies k-means clustering to the unlabeled instances lying within the SVM margin and select the cluster centroids for labeling. Their proposal significantly outperforms SVM AL - selecting the unlabeled instance that is closer to the separating hyperplane (Tong and Koller, 2002) – and random sampling at the initial stages of the learning process. However, after a number of iterations, in some cases, SVM AL outperforms representative sampling. According to the authors, this poor performance is probably due to the poor clustering structure of the unlabeled instances within the SVM margin when the margin is getting exhausted.

Donmez et al. (Donmez et al., 2007) noticed that uncertainty based approaches tend to disagree with density based approaches when selecting queries for AL. Due to their nature, uncertainty based approaches perform well when in presence of a large labeled set and poorly when in presence of few labeled instances. The opposite behavior is observable in density based approaches. Donmez et al. (Donmez et al., 2007) propose a method, in 2007, that mixes density and uncertainty components to take advantage of their best performance at each particular situation. Their method dynamically updates the selection strategy parameters based on the estimated future residual error reduction.

Scalable QBC, 2005 Gilad-Bachrach et al. (Gilad-bachrach et al., 2005) introduce *Kernel Query By Committee*, (KQBC), a novel QBC algorithm that is able to learn large scale problems by using

AL. KQBC projects the input feature space into a low dimensional space to reduce the cost of query selection. The authors reported improved performance over traditional QBC.

Instance deletion, filtering out irrelevant instances, 2005 The former work of Lewis et al. on uncertainty sampling (Lewis and Gale, 1994) was extended by Becker et al. (Becker and Osborne, 2005) in 2005. In this later work, a two-stage method is proposed. In the first stage the instances that cannot be reliably selected using uncertainty sampling are filtered out. At the second stage, uncertainty sampling is applied to the remaining reliable instances to select queries. Empirical results support better performance of this method when compared to pure uncertainty sampling.

A similar motivation – filtering out irrelevant instances that will most probably generate wasted queries – is present in Mazzoni et al. (Mazzoni et al., 2006). They propose *relevance bias* that combines the query selection criteria in use with the output of a relevance classifier, trained in parallel, to favor instances that are likely to be both relevant and informative. Queries are ranked by the product of the output of the active learner, normalized to [0,1], by the probability of relevance, the output of the relevance classifier. Three query selection criteria have been evaluated – simple margin (Tong and Koller, 2002), MaxMin margin (Tong and Koller, 2002) and a batch mode strategy assuring diversity among the selected instances (Brinker, 2003) – and compared to random query selection. The authors define *probabilistic AL*, a variant of the base query selection criteria that sorts unlabeled instances by the query selection criteria and then selects a random sample among the top 10% instead of selecting the top-ranked query. The rationale for this procedure is based on the heuristic nature of the query selection criteria in use that does not assure optimal query selection. Besides, the differences in the value of the selection criteria among the top-ranked queries might be non-significant. For these reasons, the top-ranked query might not be the optimal one and a random sample of the most probable candidates might be valuable. Despite the fact that there is no justification for the threshold that the authors use – 10% of the top-ranked instances – and that this threshold is independent of the data and, particularly, of the variance of the utility of those 10% top-ranked instances, improvements in performance are observed when compared to strict base criteria.

Stream-based active learning, 2005 We must go back a few years to review one of the former approaches to stream-based AL. In 1997, Helmbold et al. (Helmbold and Panizza, 1997) discussed the trade-off between the cost of requesting a query and the cost of errors in a stream-based setting – referred by *label efficient learning*. Further approaches to stream-based, or online, AL include recent research efforts like the work from Bianchi et al. in 2005 and 2006 (Cesa-bianchi et al., 2005; Cesa-Bianchi et al., 2006) and more recent work from Dasgupta et al. in 2009 (Dasgupta et al., 2009) and Chu et al. in 2011 (Chu et al., 2011).

The dynamic nature of data streams – increasing data volumes and evolving decision concepts – poses new challenges to stream-based AL. Between 2007 and 2010, Zhu et al. (Zhu et al., 2007, 2010c) introduce a *minimum-variance* principle to guide instance labeling from data streams based on an ensemble of classifiers. A weight updating rule is derived to ensure a proper adjustment to drifting concepts in the data stream.

Theoretical essays, 2005 In 2005, Dasgupta (Dasgupta, 2005) defined theoretical bounds showing that AL has exponentially smaller label complexity than supervised learning under some particular and restrictive constraints. Kääriäinen extended this work by relaxing some of those constraints (Kääriäinen, 2006). An important conclusion of this later work is that the gains of AL are much more evident in the initial phase of the learning process, after which these gains degrade and the speed of learning drops to that of passive learning.

In 2006, Balcan et al. propose *Agnostic Active learning* (Balcan et al., 2006), A^2. A^2 achieves an exponential improvement over the usual label complexity of supervised learning in the presence of arbitrary forms of noise. This model is further studied by Hanneke (Hanneke, 2007), in 2007, who sets general bounds on label complexity.

In 2007, Castro et al (Castro and Nowak, 2007) show theoretically that, in a classification task, AL outperforms passive learning achieving a faster rate of classification error decay irrespective of the behavior of the posterior probability in the vicinity of the decision boundary and of the complexity of the Bayes decision boundary.

2.3.5 Modern active learning, late 2000's to early 2010's

Relaxing the base assumptions of AL sets the ground for proactive learning, a generalization of the former. AL can be particularly beneficial to applications depending on unstructured, high-dimensional data, like text and video. The compromise between exploration and exploitation is also being addressed.

Variable labeling costs, 2005 Cullota et al. (Culotta and McCallum, 2005) introduce variable labeling costs applied to information extraction (IE) (Cowie and Lehnert, 1996). They propose a new AL paradigm which reduces not only the number of instances to label but also the difficulty in labeling each one of them. The proposed strategy provides a way to quantify the number of actions a user must perform to label each training example, distinguishing between boundary annotations – boundary or segmentation in IE is the task performed to define the limits of an entity in a sequence of text – and classification annotations – classifying in IE is the task performed to assign a class to an entity in a pre-segmented text. Boundary annotations are usually more demanding than classification annotations.

SVM base classifiers, 2006 Most of the AL methods relying on SVM as base classifiers query for instances based on their distance to the current separating hyperplane. Mitra et al. (Mitra et al., 2004) extended this common approach by introducing to SVM an adaptive confidence factor estimated from local information using k-nearest-neighbor principles. Their method, motivated by the statistical query model of learning (Kearns, 1998), selects a batch of queries according to a distribution that is determined by the current separating hyperplane and by this adaptive confidence factor, enabling more robust and efficient learning capabilities.

Cesa-Bianchi et al. (Cesa-Bianchi et al., 2006) introduced, in 2006, a label efficient method of selective sampling for linear classifiers that requests for the label of a given instance with a probability that is a function of its distance to the dividing hyperplane. This probability is higher when the distance to the hyperplane is lower. When this distance is 0, the current model has a confidence of 0 on the instance label, and the probability of asking a query is 1.

Sculley (Sculley, 2007) proposes *logistic margin sampling*, a similar heuristic to the previous work from Cesa-Bianchi et al. (Cesa-Bianchi et al., 2006) but that models the sampling probability using a logistic regression of the distance to the dividing hyperplane. Both these heuristics are compared to the so-called *fixed margin sampling*, proposed by the authors, that simply requests a label for an instance when its distance to the dividing hyperplane is below a given preset threshold.

Multi-label classification, 2006 AL is most frequently used in single-label classification tasks. Uncertainty sampling, for instance, focuses on measuring the confidence of the most probable class. Expected error reduction strategies are based on an error estimate for one single class. QBC select the instances where the current committee has the biggest disagreement w.r.t. the top class. There are some previous approaches of AL to multi-label classification (Brinker, 2006), frequently applied to image retrieval (Li et al., 2004; Qi et al., 2009). However these approaches are not adequate for text classification, as reported by Yang et al. (Yang et al., 2009) in 2009, either for exhibiting poor performance when applied to text corpora or for requiring reading and labeling the same document several times, which might be reasonable for images but very costly for text documents. Yang et al. (Yang et al., 2009) propose an AL approach to deal with multi-label text classification. The AL component selects queries based on the *Maximum loss reduction with Maximal Confidence* (MMC) criteria as defined by the authors.

Esuli et al. (Esuli and Sebastiani, 2009) propose several AL strategies for multi-label classification tasks, evaluating them on text corpora, each one combining the outputs returned by individual binary classifiers as a result of classifying a given unlabeled document.

Both of these approaches to multi-label text classification (Yang et al., 2009; Esuli and Sebastiani, 2009) are based on a set of binary classifiers, one for each class to learn.

Exploration versus exploitation, 2006 Kai Yu et al. (Yu et al., 2006) propose *transductive experimental design*, an AL approach applied to regression (Hastie et al., 2003). Their work is focused on experimental design, and denotes concerns related to the exploratory aspects of AL (Thrun, 1998). Transductive experimental design searches for queries that are simultaneously hard to predict – addressing exploitation – and representative of the unexplored data – addressing exploration.

In 2009 Cebron et al. (Cebron and Berthold, 2009), introduce *Prototype Based Active Classification* (PBAC), an AL algorithm for classification that evaluates the potential of each instance based on a combination of its representativeness and the uncertainty of the classifier. Their motivation – the datasets need to be explored first to generate a coarse model and then the model can be adapted to further fine-tune the classification accuracy – is aligned with our own and arises from the need to classify large datasets without any a-priori information on the target classes. PBAC takes into account the density of the feature space and the uncertainty of the classifier combined to form one single criterion for the selection of queries. The transition between exploration and exploitation occurs as the learning process evolves. This transition is achieved at each iteration by decreasing the influence in the selection criterion of the exploration term whilst exploitation influence increases.

This concern, an explicit sense of exploration versus exploitation, is also present in the work of Osugi et al. (Osugi et al., 2005) that apply a Kernel-Farthest-First algorithm for exploration in AL with SVM. The decision to go for an exploration step is made at each iteration based on the change that is induced by the newly added labeled instance on the hypothesis space.

Our proposal dynamically shifts between exploration and exploitation modes without requiring any tuning thus, avoiding overhead costs. This automatic shifting is guided by both the geometric properties of the working set and the knowledge on the target concept embedded in the current hypothesis.

Transfer learning, 2008 AL and *transfer learning* (Caruana, 1997; Dai et al., 2007) are distinct strategies to obtain labeled instances for learning. AL asks domain experts to label particularly informative queries while transfer learning intends to leverage the knowledge from a given domain to learn in a different one. Shi et al. (Shi et al., 2008) work on the application of AL to the transfer of knowledge across domains. The authors propose a framework to actively transfer knowledge from one domain in order to help labeling the instances in the target domain. They extend the standard procedure by including an uncertainty based AL component that queries an oracle when the out-of-domain example is classified with low confidence.

Videos may come from many different sources or domains. For instance, we may find a video showing an airplane – let's say, the semantic concept to learn – that comes from a movie or from a news channel. Each of these sources has its distinctive class distribution so, transfer of knowledge across different sources becomes relevant. AL can be used to reduce the labeling effort required to build a classifier for a new source by reusing a classifier previously trained for the same semantic concept but under a different source or domain.

Li et al. (Li et al., 2010) proposed an hybrid approach to select queries in a cross-domain video semantic concept classification task. Their approach selects a batch of queries of fixed length. Queries are selected by an ensemble of a discriminative query strategy, SVM AL, and a generative query strategy that selects the sample that is most unlikely to have been generated by the source domain distribution. The percentage of queries in the batch that come from each of these strategies is defined by a parameter of the model. This parameter is initialized at 50% and is dynamically updated according to the number of positive instances that have been selected by the strategy in the previous iteration. The strategy that selects more positive instances will be assigned a bigger share in the batch of queries.

Proactive learning, 2008 Recent work on AL is focused on relaxing some of the base assumptions underlying this learning paradigm, such as, the existence of a single omniscient oracle who is assumed to be infallible (never wrong), indefatigable (always answers), individual (only one oracle) and insensitive to costs (always free or always charges the same). *Proactive learning* (Donmez and Carbonell, 2008) is a generalization of AL designed to relax these unrealistic assumptions.

Recent approaches and applications of active learning, 2008 Robson Motta et al. (Motta et al., 2009) propose a novel approach to support AL in classification tasks. They explore the use of complex network properties (Newman, 2003; Luciano et al., 2007) – mainly vertex centrality measures such as closeness and betweenness – to improve the performance of AL algorithms. The authors discussed and evaluated how these measures can be explored to guide query selection.

Xiaofei He (He, 2010) showed improvements in optimal experimental design when applying AL. He proposed a novel AL algorithm based on the intrinsic geometry of the input space, grasped from the graph structure inferred from the data, to apply in image retrieval. Empirical results show improvements over SVM AL and Laplacian regularized least squares (Belkin et al., 2006).

Despite the large volume of research on AL its methods are being slowly adopted in practical applications (Attenberg and Provost, 2011). Text classification and movie filtering are among the fields that may directly benefit from AL to a great extent. In general, any field characterized by unstructured abundant data will possess the features required to benefit from AL: high labeling cost and data availability. Meta-learning applied to predict the performance of learning algorithms is a field that may also benefit from AL. Labeling instances in this field may be expensive since it is necessary to evaluate all the candidate algorithms and record the appropriate descriptors: meta-features describing the problem and information on the performance of each learning algorithm being evaluated.

Prudêncio et al. (Prudêncio and Ludermir, 2008) applied AL techniques to the meta-learning problem. Their proposal is developed in two stages. In a first stage, an outlier detection technique, based on distance, is applied to delete eventual outlier instances. Then, uncertainty sampling (Lewis and Gale, 1994) is applied to select the next meta-instance to label.

The interest on AL for realistic robotics applications arises naturally given its intensive need for labeling unstructured data with severely imbalanced class priors. Dima et al. (Dima and Hebert, 2005) concluded that when AL techniques are applied to obstacle detection in outdoor robotic vehicles they significantly reduce the required labeling effort. Further successful application of AL to classification tasks commonly arising in autonomous navigation in outdoor, off-road environments, such as obstacle detection, road following and terrain classification are also described (Dima et al., 2004).

Research on AL applied to multimedia – video, image and music – annotation and retrieval is being very active in the last decade (Tong and Chang, 2001; Ferecatu et al., 2004; Li et al., 2004; Goh et al., 2004; Dagli, 2005; Mandel et al., 2006; Wang et al., 2009; Kapoor et al., 2010b; Chen et al., 2010). A comprehensive survey is available in Wang et al. (2011) (Wang and Hua, 2011)

Recent improvements on QBC, 2010 *Adaptive informative sampling*, proposed by Lu et al. (Lu et al., 2010) in 2010, is a QBC approach that dynamically fine tunes the composition of an heterogeneous ensemble, composed by multiple instances of different classifier types. In the reported work, they explore neural networks and decision trees. The purpose is to reach the most suitable committee for a given dataset. Their motivation is that certain types of classifiers may prove more suited to a given dataset than others.

A core issue in QBC is assuring that the committee is composed by consistent hypotheses that are very distinct from each other. Melville et al. propose DECORATE (Melville and Mooney, 2003) – a method that builds such committees using artificial data – and, later on, ACTIVE-DECORATE (Melville and Mooney, 2004) – which uses DECORATE committees to select queries – focusing their work on this core issue.

Liu et al. (Liu and Agrawal, 2011) applied AL to the frequent itemset mining scenario (Agrawal and Srikant, 1994) achieving more than 95% accuracy in estimating the support of frequent itemsets with less than 10% of all the instances being labeled.

Active learning hierarchical clustering, 2012 *Hierarchical Confidence-Based Active Clustering*, (HCAC), is a new semi-supervised clustering method based on agglomerative hierarchical clustering (Nogueira et al., 2012). HCAC uses cluster-level constraints, provided by a human oracle, along the iterations of an agglomerative hierarchical clustering algorithm.

This AL approach is based on a new concept of merged confidence in agglomerative clustering. When there is low confidence in a cluster merge the oracle is queried and provides a cluster-level constraint indicating whether or not to merge the clusters being queried.

2.4 General assumptions in active learning classification

The majority of AL approaches for classification assume the availability of an initial labeled set covering all the classes of interest. However, this assumption does not necessarily hold. In fact, collecting and labeling instances is a critical stage in classification. Being one of the first stages in a learning process, it might limit the performance of the following. Moreover, it is also a demanding stage requiring domain specialists to retrieve and label exemplary instances for all target classes (Li and Sethi, 2006). The effort in finding these exemplary instances is proportional not only to the number of target classes (Adami et al., 2005) but also to their distribution in the working set. On a highly imbalanced class distribution, it is particularly demanding to identify examples from minority classes. These, however, may be important in terms of representativeness. Imbalanced class distribution is a particular concern in machine learning in general (Kubat and Matwin, 1997; Japkowicz, 2000; Novak et al., 2006; Ertekin et al., 2007; Lu et al., 2008).

This assumption might excel the cold start problem (Attenberg and Provost, 2011) but it is misleading since it neglects the labeling effort that is required to setup the pre-labeled set covering all classes which might be high. The cold start problem is a key difficulty in building effective classifiers quickly and cheaply via AL, particularly when in presence of highly skewed class distributions. Data selection directly depends on the perception of the input space as provided by the current classifier, which might be deficient at the early stage of the learning process when no pre-labeled set is available. This may generate costly wasted queries.

Traditional AL assumes that instances are freely available and a cost is incurred when asking for labels. In 2007, Lomasky et al. discussed *active class selection* (Lomasky et al., 2007), a setting where the learner knows the labels and retrieves exemplary instances representative of particular labels.

In AL it is common to assume that the oracle is able of answering any query, that is, labeling any unlabeled instance w.r.t. the target concept. This assumption is questioned, in 2008, by Harpale (Harpale and Yang, 2008) that proposes a personalized AL approach applied to collaborative filtering. The baseline query selection criterion, Bayesian selection (Jin and Si, 2004), is multiplied by a personalization term estimated as the probability of getting a rating for a given object from a given user. Collaborative filtering (Su and Khoshgoftaar, 2009) is an important setting to recommend items whose relevance cannot be evaluated from its content/features. It is a popular technique for retrieving unstructured objects, such as video, music and text, that are relevant to a particular user, based on its relevance to other users exhibiting the same interests' pattern. To perform collaborative filtering it is required to have a global model of the interest or information need, which is instantiated for each interest group, and a user model, which is instantiated for each user. For new users, this user model is very weak and the learning system needs the user feedback to fill it in. AL for collaborative filtering (Boutilier et al., 2003; Jin and Si, 2004) is an important technique for the instantiation of these user models avoiding annoying the user with too many queries.

Chapter 3

Active Learning Approaches

Many distinct heuristics have been proposed for selecting queries in active learning (AL). In this chapter, we review the most relevant approaches to our work, organized into five groups according to the underlying selection strategy – uncertainty based criteria, version space reduction, expected utility gain, density based and hybrid approaches. In the last section of the chapter we review initialization and stopping criteria.

3.1 Uncertainty reduction

Uncertainty sampling (Lewis and Gale, 1994) is a query selection criteria that aims to reduce the classifier uncertainty by querying the unlabeled instances on which the current classifier is less confident. Many AL approaches rely on this procedure: query those instances whose labels are not perceptible with enough confidence. A key point, not always addressed by these approaches is the meaning of "enough" in this sense. We will discuss this joint in Section 3.6.2. Uncertainty sampling requires a base classifier that outputs posterior class membership probabilities. In (Lewis and Gale, 1994) a binary probabilistic text classifier is used.

Lewis et al. (Lewis and Gale, 1994) claim that *b* in Algorithm 3.1 should ideally be set to 1, although other values might be appropriate when scoring and selecting, or retraining, is expensive. This algorithm can be applied with any base classifier that predicts a class and outputs a measurement of its certainty – assuming that the certainty measure is a probability or an estimate of a probability (without loss of generality).

Algorithm 3.1 Uncertainty sampling (adapted from (Lewis and Gale, 1994))

```
1: Create an initial classifier $h$
2: while not stopping criteria do
3:     Run $h$ to classify unlabeled instances
4:     Find the $b$ instances for which the classifier $h$ is least certain of class membership
5:     Ask the oracle for the labels of the $b$ instances in the subsample
6:     Train a new classifier $h$ on all labeled instances
7: end while
8: Return $h_i$
```

The least reliable prediction of a binary classifier occurs when its confidence on any one of the classes is 0.5 – assuming that the confidence is measured in the [0,1] range. The subsample selected by uncertainty sampling in each iteration is composed by the $b/2$ instances with uncertainty w.r.t. one of the classes closer to 0.5 and above it plus the $b/2$ instances with uncertainty w.r.t. the same class closer to 0.5 and below it. This heuristic is based on the assumption that, when in batch mode ($b > 1$), training on pairs of instances lying on both sides of the decision boundary is useful. When $b = 1$, as recommended by the authors, this subsample is restricted to the instance with uncertainty closer to 0.5. The results of this work from Lewis et al. demonstrate that uncertainty sampling can significantly reduce the labeling effort in text classification tasks. In some cases, random sampling requires 500 times more labels than uncertainty sampling.

Uncertainty sampling operates in batch mode when $b > 1$. In batch mode it is crucial to assure an heterogeneous batch composition. Although not explicitly mentioned, this concern is addressed by uncertainty sampling when selecting queries from both sides of the classification boundary.

AL query strategies are commonly based exclusively on the current hypothesis and its predictions. Former predictions are neglected although they might add some evidence on the utility of unlabeled instances. Davy et al. (Davy and Luz, 2007) describe an approach based on uncertainty sampling that explores historical predictions to improve the query selection mechanism. They propose two novel criteria, *History Uncertainty Sampling* (HUS) and *History Kullback-Leibler divergence* (HKLD) that evaluate the utility of unlabeled instances from the variance of the former predictions. HUS extends uncertainty sampling by redefining the uncertainty of a given instance $x_j \in U_i$ as the sum of the uncertainty computed at the last m iterations (Equation 3.1). Unlabeled instances are ranked according to their HUS value and the unlabeled instance with the lowest HUS is selected to query. In Equation 3.1, $\hat{\Phi}_i(x_j)$ is the maximum a posteriori membership probability assigned by the hypothesis learned at iteration i to the unlabeled instance x_j.

$$q_i = \operatorname*{argmin}_{x_j \in U_i} \sum_{l=1}^{m} \left(\left| \hat{\Phi}_{i-l}(x_j) - 0.5 \right| \right) \tag{3.1}$$

HKLD is an alternative to HUS that evaluates the m former hypotheses as a committee and measures uncertainty as the disagreement among committee members using Kullback-Leibler divergence to the mean (McCallum and Nigam, 1998). Although the authors do not provide information on statistical significance neither on the experimental procedure, the results presented on binary classification tasks over text indicate that HKLD outperforms uncertainty sampling w.r.t. accuracy. The impact of the depth of the history used to compute uncertainty, m, is mentioned by the authors as future work.

Another common standing in uncertainty based AL strategies is to consider only the information of the most probable class for each unlabeled instance, neglecting any other information about the label distribution. *Margin sampling* (Scheffer et al., 2001) is an approach that selects queries based on the difference between the posterior probabilities of the two most probable classes, c_1 and c_2, given the instance x_j (Equation 3.2).

$$q_i = \operatorname*{argmin}_{x_j \in U_i} \left(P_{h_i}(c_1|x) - P_{h_i}(c_2|x) \right) \tag{3.2}$$

Instances with a large margin are not as informative as those with a lower margin, in which case the classifier has a greater difficulty in distinguishing between the two most likely classes.

Many more AL approaches select queries based on uncertainty (Freund et al., 1997; Scheffer et al., 2001; Mitra et al., 2004; Hwa, 2004; Li and Sethi, 2006; Shi et al., 2008; Settles and Craven, 2008; Prudêncio and Ludermir, 2008).

3.2 Version space reduction

This paradigm is focused on querying the instances that will further reduce version space (Mitchell, 1997). The optimum split, achieving the greatest reduction, is obtained when halving the version space (Tong and Koller, 2002).

Query By Committee (Seung et al., 1992), for example, uses a set of classifiers – the committee – to identify the instance with the highest disagreement. Each unlabeled instance is evaluated by a committee with an even number of members, $2m$, each one assigning a label. Queries are selected by the principle of maximal disagreement among committee members. In a binary classifier, disagreement maximization is achieved when half of the committee classifies the input as positive while the other half classifies it as negative. In such circumstances, each query bisects the version space when $m \to \infty$, maximizing information gain (Shannon, 1948). Monte Carlo simulations using a two-member committee confirm the improved performance of Query By Committee (QBC) over random sampling as prescribed by the theoretical study. The information gain of a QBC query tends asymptotically to a finite non-zero value leading to an exponentially decreasing generalization error as the number of queries increases. When using random sampling this information gain tends to 0 and generalization error decreases as an inverse power law in the number of queries thus, performing poorer than QBC. The authors refer that, in a filtering approach, where queries are selected from the available dataset, the learner may have to evaluate many instances before finding one where the committee disagrees. This drawback does not stand in the query-construction setting where one can generate specific artificial instances to promote disagreement in the committee. QBC is normally used in stream-based learning where a query is made for each incoming instance generating committee disagreement. The results from this seminal paper on QBC (Seung et al., 1992) are generalized and further discussed by Freund et al. (Freund et al., 1997).

Lu et al. (Lu et al., 2010) sustain that specific datasets demand for specific ensembles of base classifiers. The “one-size-fits-all” approach, underlying the original QBC approach, can be improved by dynamically adapting the ensemble of base classifiers to the dataset at hand. The authors propose *adaptive informative sampling*, extending the base QBC technique to accommodate their claim. Adaptive informative sampling is initialized with a balanced ensemble composed by an equal number of classifiers from two distinct types – their work relies on neural networks and decision trees. Then, at each iteration, the percentage of committee members from each type is updated based on the classifiers’ performance in the previous iteration in a way to favor best performers.

The performance of a given classifier in the committee is evaluated by two distinct fitness functions depending on the number of classes to learn. In binary classification tasks, committee members, h_i, are evaluated by the fitness function, f, in Equation 3.3, where tp_k is the number of true positives identified for class c_k and n_k is the total frequency of class k in the training set.

$$f(h_i) = \frac{tp_0}{n_0} \times \frac{tp_1}{n_1} \tag{3.3}$$

For multi-class classification tasks the fitness function is defined as the number of misclassified instances in the training set. Experimental results show that adaptive informative sampling consistently outperforms homogeneous ensembles.

3.3 Expected utility gain

The AL approaches that we have considered under this category select queries based on an estimate of the utility of unlabeled instances. The utility measure depends on the concrete task but expected error and error variance are usual.

Cohn et al. (Cohn et al., 1996) describe an optimal solution for pool-based AL that selects the instance that, once labeled and added to the training set, produces the minimum expected error. The expected error of the learner is decomposed into three terms: noise in the class distribution (independent from the learner), the learner's bias and the learner's variance (reflecting the sensitivity of the learner to the training set). Direct estimation of both bias and variance, is not possible for inductive learning since it requires knowing the actual class distribution which is not available. However, variance estimation is computed without reference to the underlying probability distribution. Moreover, when in presence of low variance, accurate classification is achievable regardless of bias (Friedman, 1997). Cohn et al. assume approximately unbiased learners, i.e., learners whose average prediction for a given instance is equal to its true class. Under such circumstances error is reduced to the variance term thus, minimizing the learner's variance is equivalent to minimizing error. This approach, however, requires high computational effort.

Schohn et al. (Schohn and Cohn, 2000) worked on AL, with SVM base classifiers, defining a query selection heuristic that estimates the expected change in error when adding a given instance to the training set. Their results are, in some cases, better than if all available data is used to train. One optimal approach, in a binary classification task, estimates the expected error, E_j, after adding a new instance, x_j, to the training set as the mean of the expected error when the new query being added belongs either to class 1, $E_{(x_j,1)}$, or to class -1, $E_{(x_j,-1)}$ (Equation 3.4).

$$E_j = P(y_j = 1|x_j) \cdot E_{(x_j,1)} + P(y_j = -1|x_j) \cdot E_{(x_j,-1)} \tag{3.4}$$

$E_{(x_j,c_k)}$ might be computed by Monte Carlo simulation (Cohn et al., 1996) or, as suggested by the authors, using a simpler non-probabilistic approach that defines $E_{(x_j,c_k)}$ as the volume spanned by the SVM margin and then computes $E_j = max\left(E_{(x_j,1)}, E_{(x_j,-1)}\right)$. In such case, E_j sets a lower bound to the decrease in uncertainty that is achievable when adding x_j to the training set.

However, both these greedy approaches are unfeasible in practice since computing $E_{(x_j,c_k)}$ requires building a large number of classifiers for each iteration – twice the number of unlabeled instances for a binary classification task. The same drawback – high computational cost – is experienced by Roy et al. (Roy and McCallum, 2001) whose approach requires retraining the classifier several times at each iteration, one for each class to learn.

To overcome this practical drawback, the authors claim that queries can be selected without the need to estimate the expected change in error, thus avoiding extensive retraining. Selecting the next query such that the expected generalization error is minimized can be accomplished by narrowing the existing margin as much as possible, that is, by querying the unlabeled instance that is closer to the dividing hyperplane. The computational cost of this heuristic is incomparably lower than the one required by the optimal algorithms described above. The results reported by the authors are also favorable to their proposal. High accuracy is achieved very quickly, in some cases requiring four times fewer labeled instances than competing methods.

Chapelle (Chapelle, 2005) proposes an approach that directly computes an estimate of the expected generalization error, performing the optimal AL strategy described above (Equation 3.4) in a feasible way. This approach relies on a simple classifier, the Parzen window classifier (Devroye et al., 1996), that gives direct estimates of the posterior probabilities avoiding a costly retraining process at each iteration.

Lindenbaum et al. (Lindenbaum et al., 2004) apply AL to instance based learning exploring nearest-neighbor classifiers[1]. They propose two accuracy based utility functions that are maxi-

[1]The nearest-neighbor classifier stores all previously labeled instances that are then used to classify unlabeled instances according to the label of its nearest labeled neighbor. Variations of this scheme include k-nearest-neighbor classifiers (Duda and Hart, 1973) that assign classes to unlabeled instances by majority vote of the k-nearest labeled neighbors (Cover and Hart, 1967).

mized after labeling and adding the new query to the training set: an *absolute-accuracy* utility function, that estimates the absolute expected accuracy of the future hypothesis regardless of the current one and a *gain-based* utility function that estimates the accuracy gain of the future hypothesis relative to the current one.

Using random field models (Wong and Hajek, 1985), Wong and Hajek estimate the probability of the possible labels of an unlabeled instance, x_j, and then compute its expected utility from those.

Berardi et al. (Berardi et al., 2012) propose *inspection gain*, an approach to text classification that ranks automatically labeled documents by the expected improvement in the classification effectiveness that is achievable by retraining after reviewing the automatically labeled documents. The ranking function used to sort automatically labeled documents acts as a query selection criteria based on utility.

3.4 Density based

Under this category we have considered AL approaches that are based either on the distance between instances or on the density of the data in input space or on clustering. Distance based approaches are common in AL. The frequent use of SVM as the base classifier may have contributed to this since uncertainty in SVM classifiers is related to distance to the dividing hyperplane. The simplest distance based approach, *farthest-first* (Hochbaum and Shmoys, 1985), selects the next query as the unlabeled instance that is farther apart from all labeled instances. This approach, favoring exploration over exploitation, relies exclusively on distance measures in the input space that are independent from the base learner.

SVM AL (Tong and Koller, 2002) selects the instance lying within the SVM margin closest to the dividing hyperplane. This approach assumes a clear focus on exploitation. From the point of view of the exploration-exploitation trade-off these two opposing strategies, farthest-first and SVM AL, both based on distance, take extreme positions.

Representative sampling (Xu et al., 2003), also a SVM based approach, goes a little further beyond SVM AL and tries to capture the structure underlying the unlabeled instances within the SVM margin. Representative sampling selects a batch of *m* queries in each iteration following

Algorithm 3.2 until some stopping criteria is satisfied. This batch is composed by the medoids[2] of the clusters formed by the unlabeled instances lying inside the SVM margin.

Algorithm 3.2 Representative sampling (adapted from (Xu et al., 2003))

```
1: while not stopping criteria do
2:     Train a linear SVM based on the labeled instances
3:     Cluster the unlabeled instances lying in the margin of the newly trained SVM into $m$ groups
       using k-means clustering and inner product as a similarity measure
4:     Identify the $m$ medoids of the $m$ clusters
5:     Ask the oracle for the labels of these $m$ instances
6: end while
7: Return the current SVM classifier
```

The clustering step (step 3 in Algorithm 3.2) is expected to preserve the density distribution by allowing to query the most important uncertain instances. Batch diversity is achieved by selecting the medoid of each cluster as the batch members.

Wang et al. (Wang et al., 2009) select a preliminary batch of instances that lie in the SVM margin and then enforce diversity by selecting those instances from this preliminary batch that explicitly maximize the distance between each other in the original input feature space. To assure diversity, Chen et al. (Chen et al., 2010) use distance diversity and set density in the SVM feature space to evaluate the heterogeneity of the selected batch.

Clustering has also been explored to provide an initial structure to data or to suggest valuable queries (McCallum and Nigam, 1998; Nguyen and Smeulders, 2004; Hu et al., 2009; Zhu et al., 2010a).

Adami et al. (Adami et al., 2005) merge clustering and oracle labeling to bootstrap a predefined hierarchy of classes. Although the original clusters provide some structure to the input, this approach still demands for a high validation effort, especially when these clusters are not aligned with class labels.

This concern on misalignment is also present in (Dasgupta and Hsu, 2008). Dasgupta et al. propose a cluster-based method that consistently improves label complexity – the number of queries that is sufficient to learn a concept – over supervised learning. Their method detects and exploits clusters that are loosely aligned with class labels.

[2] A medoid is a representative object of a given set. It is the instance whose average dissimilarity to all the instances in the set is minimal. Medoids are similar in concept to centroids, but they are always real instances of the dataset. The medoid is the dataset instance that is closest to the centroid.

Jiang et al. (Jiang and Ip, 2007) propose a novel dynamic distance-based approach to AL with SVM, named *dynamic distance-based active learning*. The authors claim that their approach outperforms the standard SVM AL approach (Tong and Koller, 2002). The dynamic distance-based strategy is implemented in two steps. In the first step, the nearest instance to the current decision boundary is queried – the standard SVM AL approach. Then, its neighbors are sorted by increasing distance to the current decision boundary. The second step involves the oracle that must label the instances in this ranked list, in sequence, from top to bottom – the closest instances to the decision boundary are queried first – until reaching an instance whose label is opposite to the previous. The last positive instance is added to the training set.

3.5 Hybrid approaches

Hybrid approaches combine several distinct strategies in an attempt to take advantage of the benefits from each one. Pure strategies tend to favor either exploration or exploitation competences. While exploitation concerns seem to have been dominating AL research, exploration seems to have been gaining relevance. The issue is that exploration and exploitation are correlated. Acting on one has influence on the other, usually requiring a compromise solution. Focusing on instances near the decision boundary, favoring exploitation, prevents exploring regions in the feature space that might contain instances being misclassified by the current hypothesis (Baram et al., 2004). On the other hand, focusing on exploration, i.e., selecting queries from regions in the feature space away from the decision boundary, reduces the chances to sharpen current decision boundaries.

Several hybrid approaches, like *Prototype Based Active Classification* (PBAC) (Cebron and Berthold, 2009), try to combine exploration and exploitation capabilities in a unique strategy in search for a good compromise solution.

PBAC, motivated by the need to classify large datasets without any a-priori information, selects queries based on the *uncertainty distribution*, a novel criterion proposed by the authors. Uncertainty distribution estimates the utility of an instance from an aggregation of its representativeness potential, which is evaluated from density estimates on the unlabeled data, and classifier uncertainty, based on labeled data. PBAC (Algorithm 3.3) starts by exploring a dataset to generate a coarse model; then, this preliminary model is exploited to tune classification accuracy. The transition from exploration to exploitation occurs as the learning process evolves by decreasing at each iteration the influence of the exploration term while increasing the influence of the exploitation term.

Algorithm 3.3 Prototype Based Active Classification (adapted from (Cebron and Berthold, 2009))

```
1: Set threshold $T$
2: $GlobalPotential = 0$
3: for all $x_j \in U$ do
4:     Compute the potential $P(x_j)$
5:     $GlobalPotential = GlobalPotential + P(x_j)$
6: end for
7: while $GlobalPotential > T$ do
8:     for all $x_j \in U$ do
9:         Compute the classifier uncertainty $C(x_j)$
10:        Compute the uncertainty distribution $D(x_j)$ according to Equation 3.5
11:    end for
12:    Select $q_i$ the instance $x_j$ with the highest potential
13:    Obtain a class label $y_i$ for $q_i$
14:    Create a new prototype with values $q_i$ and class label $y_i$
15:    Classify the datasets with the current set of prototypes
16:    Reduce the potentials
17: end while
```

The uncertainty distribution, D, of an unlabeled instance, x_j, combines its potential, $P(x_j)$, computed on the unlabeled data, and the classification uncertainty, $C(x_j)$, computed on the labeled data (Equation 3.5). In Equation 3.5, $\varepsilon \in [0,1]$ controls the influence of the exploitation term.

$$D(x_j) = (1-\varepsilon)P(x_j) + \varepsilon C(x_j) \tag{3.5}$$

The potential of x_j, $P(x_j)$, is computed from the distance between x_j and its closest neighbors. Instances that have more neighbors in their close vicinity have a higher potential. Having the potentials computed, the instance with higher potential, x_j^*, is selected and the potentials of x_j^* and their close neighbors are reduced to avoid having these instances selected in the next iteration. This potentials' reduction step also reduces the overall influence of exploration as the learning process iterates. The reduction of potentials is also used to define a stopping criterion. The learning process stops when the total sum of all potentials drops under a predefined threshold, T.

The classifier uncertainty for a given instance x_j, $P(x_j)$, is computed as the entropy of its membership probabilities for all classes. These class probabilities are computed from a weighted k-nearest-neighbor classifier based only on the labeled instances, called prototypes. The class label for a given unlabeled instance, x_j, is assumed to be the class label of the prototype with the largest prototype weight, i.e., the closest prototype to x_j.

An explicit concern of exploration versus exploitation, is also discussed by Osugi et al. (Osugi et al., 2005). They propose an AL strategy that decides at each iteration whether to explore or exploit. This decision is based on a binary random variable, a “coin flip” as the authors put it, assigning a probability p to explore (and $1-p$ to exploit).

If the decision is to explore, the Kernel Farthest First (Baram et al., 2004) algorithm is applied to select the next query – select the unlabeled instance that is further away from all labeled instances in the feature space induced by the kernel function used by the classifier. Otherwise the next query is selected with Simple (Tong and Koller, 2002) – select the unlabeled instance that is closest to the current decision boundary. This new query is labeled and added to the training set. To determine how successful an exploration step was, the authors compute $d(h_{i-1},h_i) \in [-1,+1]$, the change induced from the previous hypothesis, h_{i-1}, to the current, h_i. If $d(h_{i-1},h_i)$ is positive, implying significant change from h_{i-1} to h_i, the previous exploration step is assumed to be successful and the probability p is kept high encouraging further exploration. If $d(h_{i-1},h_i)$ is negative, p is reduced.

The exploration probability p is updated from iteration $i-1$ to iteration i using Equation 3.6.

$$p_i = max(min(p\lambda e^{d(h_{i-1},h_i)}, 1-\varepsilon), \varepsilon) \tag{3.6}$$

where ε is a parameter that bounds the value of p (so there is always a chance of exploring and exploiting) and λ is the learning rate for updating p. The function $d(h_{i-1},h_i)$, used to measure the change induced by the query just added to the training set, is a linear transformation (Equation 3.7) of the cosine, $s(h,h')$, between the vectors of the real-valued labels, h and h', predicted by both hypothesis h_i and h_{i-1} for the working set (including the predictions for labeled and unlabeled instances).

$$d(h_{i-1},h_i) = 3-4s(h,h') \tag{3.7}$$

3.6 Initialization and stopping criteria

AL classification is an iterative approach that evolves a base classifier until a certain performance level which is assumed to be adequate given the task at hand. During the learning process the same strategy is executed in every step of a loop. This loop is preceded by an initialization step – providing an initial labeled set required to learn the first instance of the classifier – and halted when a given stopping criterion is met. A simple way to perform the initialization step is by random sampling while a simple stopping rule is the exhaustion of the unlabeled set. Some research has been done to improve these naive approaches. We are particularly concerned with the stopping criteria enabling us to halt the learning process when in presence of a good "enough" classifier, thus avoiding to ask for labels that add little or no value.

3.6.1 Initialization

Initializing the labeled set in some proper way might reduce the number of wasted queries – queries that produce useless labels – and improve the performance of the classifiers learned at the initial stage of the learning process. Selecting a proper labeled set aims at early grasping the distribution of the data to classify, thus creating conditions to select valuable queries in the following iterations. This, however, is a task to be performed in the absence of any prior evidence on the concept to learn. Several approaches, based on chance alone or exploiting somehow the available working set, are available in the literature.

Some straight approaches to the initialization of the labeled set are common, such as using a set of instances previously labeled by some mean (Sun and Hardoon, 2010) or random sampling. Initializing the labeled set by randomly selecting training instances from every class to learn is probably the most common approach (Tong and Koller, 2002; Zhang and Chen, 2002; Warmuth et al., 2003; Xu et al., 2004; Schütze et al., 2006).

However, random sampling might be very demanding mainly when in presence of a severely imbalanced dataset. This is one of the main concerns in (Dima and Hebert, 2005) who define an initialization algorithm, based on the density of the input space, that discards redundant instances – those lying in regions of the feature space that are densely populated – while keeping instances from sparse regions available for querying. Their assumption is that instances from densely populated regions are representatives of the same class with high probability and repeated queries on these regions will miss under-represented classes while increasing the number of wasted queries. This assumption leads to a behavior which ranks exploration higher than exploitation.

An opposite reasoning – assuming that rare or borderline cases that do not occur very often are not interesting for classification and, therefore, discarding them from the initial labeled set – motivates the work described in (Cebron et al., 2007), where the seed queries are selected on the basis of the so called potential of each unlabeled instance. This potential, as defined by the authors, is a measure of the density of the input space in a given predefined neighborhood of the instance being evaluated. Any instance that lies within this neighborhood has a large influence on the potential of the instance at hand. Seed queries are the instances with the highest potential score, that is, those that lie in densely populated regions. This reasoning boosts exploitation over exploration.

Clustering is also a common approach to the initialization phase that tries to explore the intrinsic structure of the working set. K-means clustering is used to select initial training instances in (Kang et al., 2004). The authors propose a method that divides the unlabeled instances into clusters and then selects the clusters's medoids which are assumed to be the most representative instances from each cluster. The centroid itself may be difficult to label because it will be most likely a synthetic instance mainly when working with high dimensional input spaces as is the case with text corpora. Nevertheless, the cluster synthetic centroids themselves may also be used as training instances at no extra labeling cost since they will be assigned the same label that the oracle has assigned to the representative instance. In such a case, the centroids are named *model examples*. Experiments performed on various text datasets have shown that the active learner starting from the initial training set selected by this method reaches higher accuracy faster than when initialized by random sampling. The inclusion of the model examples in the training set further improves learning performance.

Nguyen et at. (Nguyen and Smeulders, 2004) use a simplified version of the K-medoid algorithm (Kaufman and Rousseeuw, 1990) that finds K representatives of the dataset to initialize the labeled set. These representative instances are those minimizing the sum of the distances from the data samples to the nearest representative. To overcome the high computational cost of the

K-medoid algorithm, the authors start by splitting the working set in smaller subsets and then applying the K-medoid algorithm to cluster every subset into a limited number of clusters.

Han et al. (Han and Zhao, 2010) propose a method that firstly uses dynamic clustering to select representative samples and labels them. These labeled samples are put in the training set. Then, a linear classifier is trained to maximize the area under the Receiver Operating Characteristic (ROC) curve (Bradley, 1997) following the *AUC maximization* algorithm proposed by the authors. This procedure is repeated until AUC converges.

The concept of *Representative Objects* (RO), discussed by Tao Li et al. (Li and Anand, 2007) in the field of relational data mining is another clustering approach that might be used to initialize AL.

A linear discriminant approach (Liere and Tadepalli, 1997), which does not require any actual data, computes the initial positions of the dividing hyperplanes so that they approximately bisect the space of all possible data points. These hyperplanes can be computed only from the number of attributes describing the data. The authors use QBC having individual committee members randomly initialized to slightly different hyperplanes so that they represent different initial hypotheses.

Yuan et al. (Yuan et al., 2011) propose three initialization methods – center-based, border-based and hybrid selection – based on fuzzy clustering (Bezdek, 1981). The initial training set is a batch of m instances to be selected and queried. The authors start by performing fuzzy c-means clustering on the unlabeled set, obtaining a class membership matrix containing for each unlabeled instance a measure of its membership to each induced cluster – the number of induced clusters is assumed to be r. There is no explanation on how to tune the required parameters, m and r. This matrix is then used to provide the initial training instances, assuming that induced clusters are aligned with the classes to learn, according to the following procedure:

- *Center-based selection* selects the instances with a high degree of membership. The same number of instances is to be selected from each cluster, so, the top $\frac{m}{r}$ instances w.r.t. cluster membership degree are selected from each cluster. This procedure assumes a uniform distribution of the data which contradicts the aim of the authors to find a initial training sample that is representative of the (unknown) underlying data.

- *Border-based selection* selects the m instances standing closer to the borders of the clusters, i.e., those instances having the most similar top-two class membership degrees. This strategy always selects the instances with the minimum difference between their top-two class membership degrees. This selection mechanism does not assure that all clusters will be represented in the initial training set.
- *Hybrid selection* assumes that the samples from both the methods above are representative, providing complementary evidence on the target concept, and generates the initial training set by merging the samples obtained above. The top $\frac{m}{2}$ instances are selected from each of the above samples.

The effective selection of instances to initialize AL based on centrality measures from complex networks, such as closeness and betweenness, is being investigated by Motta et al. (Motta et al., 2009, 2012). Such measures enable identifying instances that characterize prototypical or critical regions in the input space.

3.6.2 Stopping criteria

The main goal in AL is to select an accurate hypothesis from the version space at low cost, i.e., while requiring as few queries as possible. Asking the oracle to label more instances than those that are necessary has a negative impact on the performance (and cost) of the learning process. From this point of view, knowing when to stop might be as relevant as having a good query selection strategy.

The AL process should be stopped when the utility of new queries degrades below a given threshold and model quality stops improving. Specifying this utility, and the critical threshold, are task-dependent. Some simple stopping criteria, such as, exhausting the unlabeled set or predefining a desired size for the training set according to the available budget, are obvious but do not take into account efficiency concerns neither assure that the resulting learner is accurate enough for the task at hand.

Monitoring the learning process Setting stopping criteria for AL requires monitoring the learning process. AL monitoring is commonly based on the percentage of the candidate training instances that must be queried to achieve the same performance as when using all the available training instances or as the difference between the performance delivered by AL, given a number of queries, and that of supervised learning trained on a training set with equal size. These procedures require a comparison with the supervised setting which implies the need for a fully labeled training set. Stopping criteria for AL should not rely on a pre-labeled test set.

Convergence detection has been studied for the case of random sampling by estimating the slope of the learning curve (Provost et al., 1999). Special care might be required in the AL setting since the learning curve might be affected by the sampling bias.

A few approaches to monitoring the learning process, that depart from the graphical nature of the learning curve (Olsson, 2009), have been proposed to resume the performance of the learning process. *Data efficiency* (Abe and Mamitsuka, 1998) is defined as the ratio between the number of iterations required by a base learner to reach top performance in a supervised setting and the number of iterations required for the same base learner in an AL setting to reach the same performance. *Data utilization ratio* (Melville and Mooney, 2004) is defined as the ratio between the number of instances an active learner requires to reach a target error rate divided by the number of instances that are required by the base learner to reach the same error rate in a supervised setting. Both these measures reflect how good an active learner is at making use of the data.

Baram et al. (Baram et al., 2004) propose a performance measure aiming to quantify the deficiency of an AL algorithm, A, when compared to a supervised learning algorithm f, throughout the learning process. *Deficiency* is defined by Equation 3.8[3], where $acc_t(l)$ is the average accuracy achieved when using the learning function l trained with t labeled instances. f stands for a base learning function and a stands for the active learner under evaluation. N is the total number of available instances to learn.

$$def(a) = \frac{\sum_{t=2}^{N}(acc_N(f) - acc_t(a))}{\sum_{t=2}^{N}(acc_N(f) - acc_t(f))} \tag{3.8}$$

This deficiency measure captures the global performance of active learner a throughout the learning process. Smaller values indicate more efficient learning from a when compared to f.

Gradient-based approaches Laws et al. (Laws and Schätze, 2008) propose several gradient-based stopping criteria for AL. Absolute criteria, such as *minimum absolute performance* – stop-

[3]In the original proposal the sums on both denominator and numerator start at $t = 1$ which we have replaced by $t = 2$ to assure that the initial training set has at least one positive and one negative instance.

ping when the current classifier performance has reached a predefined minimum assumed to be good enough – or *maximum possible performance* – stopping when the maximum achievable performance is reached – demand for a validation set requiring an additional labeling effort, which is precisely what we want to avoid in AL in the first place. Thus, gradient-based approaches are more suited since they do not require a pre-labeled validation set and just compare performance indicators from one iteration to the next. Nevertheless, gradient-based approaches require the previous knowledge of the expected behavior of the performance indicator as the classifier performance reaches its optimal value. Laws et al. observe that the performance estimation rises along the learning process until it starts to slow down to an almost constant value at about the time when the true performance reaches its optimum. Based on this observation, they propose *uncertainty convergence*, a new stopping criterion that estimates the gradient of the performance indicator and stops when it approaches 0.

Bloodgood et al. (Bloodgood and Vijay-Shanker, 2009) follow this same gradient-based rationale to propose *Stabilizing Predictions*, a stopping criterion that checks for stabilization of predictions on a set of instances, called the stop set – that should be a representative sample of the target concept. Stabilization is evaluated by the agreement of the predictions made by the classifiers learned in two consecutive iterations. The authors claim that measuring this agreement by the percentage of instances on which both classifiers make the same predictions is not a robust approach because different datasets have different levels of agreement that can be expected by chance, which is not captured by percent agreement. They use a measure of agreement that takes chance into account (Artstein and Poesio, 2008).

Ghayoomi hypothesizes that there is a correlation between performance saturation of the classifier and the variability on the confidence of the selected instances and proposes a stopping criteria based on variance of confidence (Ghayoomi, 2010). He assumes that a pool-based AL process is divided in three stages w.r.t. the classifier performance – untrained, training and trained. In the initial untrained stage most of the class posteriors for unlabeled instances are low since the classifier is not trained enough. As a result its variance is also low since all observed posteriors are similarly low. As the classifier is training with more data, the class posteriors of some unlabeled instances will start increasing while other are still low. During this training stage the posteriors variance increases. When the classifier is well trained most of the posteriors will be high and concentrated around a (high) mean. As a consequence, posteriors variance decreases. Based on this assumption, the best stopping point is when the posteriors variance reaches its global peak and starts to decrease.

Classification gradient (Escudeiro, 2004; Escudeiro and Jorge, 2006) is a stopping criterion applied in the semi-supervised setting that is based on the differences of the predictions made at two consecutive iterations. This criterion is extended to AL in this thesis. Classification-change (Zhu et al., 2008), described by Zhu et al. in 2008, is a similar criterion to classification gradient. The work from Vlachos (Vlachos, 2008), may also be classified as gradient-based. These approaches are detailed below in this Section.

SVM based criteria Schohn et al. (Schohn and Cohn, 2000) noticed an interesting outcome when working with SVM base classifiers: classifier performance peaks at a level above that achieved when using all available data. This means that a better performance is achieved when training on a subset of data, than when using all available data. From this observation, they suggest to stop the AL process when the next query is no closer to the decision hyperplane than any of the support vectors. When this happens, the SVM margin has been exhausted and learning is terminated. The authors claim that this is a good approximation of true peak performance.

Mitra et al. (Mitra et al., 2004) select a batch of size m in each iteration. The stopping criterion proposed by them is based on the proportion of the batch corresponding to instances with margin greater than one, i.e., uninformative instances regarding the settlement of the decision boundary. The AL process is terminated when the proportion of instances in the batch exhibiting a margin greater than unity exceeds a given threshold, such as 90%. A high proportion implies that the current batch contains few instances able to impact the current hypothesis.

Criteria requiring a validation set Vlachos (Vlachos, 2008) claims that the confidence of uncertainty based sampling exhibits a rise-peak-drop pattern – previously observed empirically by Schohn (Schohn and Cohn, 2000) – due to the inclusion of queries with noisy features in the training set. Peak confidence is observed when the training set includes the minimum number of labeled instances covering all the unlabeled ones. From this point on, adding more queries to the training set promotes overfitting with a negative effect on the classifier performance. Besides this negative impact on the classifier performance, there is also no reason to continue querying the oracle beyond peak confidence since the rest of the unlabeled instances will probably be uninformative instances, already covered in the current training set. Supported by this reasoning, Vlachos suggests to use classifier confidence as a means to define a stopping criterion for uncertainty based sampling. The main idea is to stop querying when the confidence of the classifier remains at the same level or drops for a number of consecutive queries. In concrete, the learning process is terminated after having the confidence dropping for three consecutive iterations. Confidence is computed on a test set. To avoid the cost associated to a fully labeled test set, which contradicts the aims of AL, the authors suggest to compute confidence from the set of features of each instance in the test set.

In his PhD thesis, Olsson (Olsson, 2008) formulates *Intrinsic Stopping Criterion* (ISC) which is further developed by Olsson et al. (Olsson and Tomanek, 2009). ISC is a stopping criterion for committee based AL relying only on the characteristics of the base learner and the data at hand in order to decide when to stop the learning process. ISC builds upon a held-out validation set. The reasoning behind ISC is the following: if the committee agreement w.r.t. the most informative instance drawn from the unlabeled data pool, *selection agreement* (Tomanek et al., 2007), is larger than the agreement of the committee w.r.t. an instance drawn from the validation set, *validation set agreement* (Tomanek and Hahn, 2008), then the committee would learn more from a random sample extracted from the validation set, than it would from the unlabeled data pool. Under the ISC criterion, the learning process may be terminated when the selection agreement is greater than, or equal to, the validation set agreement.

Other approaches Zhu and colleagues have performed extensive work on AL stopping criteria. Several stopping criteria have been proposed as a consequence of this work, including *max-confidence* and *min-error* (Zhu and Hovy, 2007), *minimum expected error* (Zhu and Wang, 2008), *overall-uncertainty* and *classification-change* (Zhu et al., 2008) and *selected accuracy* (Zhu et al., 2010b).

Max-confidence is based on uncertainty measurement, considering whether the entropy of the unlabeled instance selected for querying is less than a predefined threshold close to zero (the authors suggest 0.001). Min-error is based on the labels provided by the oracle, considering whether the current classifier can correctly predict the true labels or the accuracy performance of predictions on queries is already larger than a predefined accuracy threshold (no benchmark is provided). Max-confidence and min-error are suggested, respectively, as the upper bound and the lower bound for stopping conditions. These criteria are based on the premises that if a classifier induced from the current training data has strong classification confidence on an unlabeled instance, then we can consider it as redundant.

It should be noticed that min-error is adequate for batch mode AL. In a different setting, where only one query is selected per iteration, the accuracy performance is either 1, if the classifier predicts the correct label, or 0, otherwise. In such settings, max-confidence and min-error might be used ensemble. Once both conditions are met, the current classifier is assumed to have enough confidence on the labels of all the remaining unlabeled data and the learning process terminates. This former work by Zhu et al. is extended to introduce the minimum expected error strategy that involves estimating the classification error on future unlabeled instances.

Overall-uncertainty is similar to max-confidence, but, instead of taking only the most informative instances into consideration, it is computed over all unlabeled instances.

Classification-change assumes that the most informative instance is the one which causes the classifier to change its predicted label. Thus, the learning process is terminated once no predicted label changes during two consecutive iterations.

Recently, in 2010, Zhu et al. (Zhu et al., 2010b) propose the selected accuracy method along with combinations of all the above strategies to estimate the required thresholds. The selected accuracy method is designed to apply in batch mode AL. In such a setting the learning algorithm has access, in each iteration, to the true labels of all the batch members. The unlabeled instances composing the batch are supposed to be the most informative given the current hypothesis and the unlabeled data pool. The learning process is terminated when the accuracy of the current classifier on these batch instances is above a given threshold.

Missed clusters Schutze et al. (Schütze et al., 2006) claim that there is no obvious procedure to decide conveniently when to stop querying due to the so called *missed clusters* – unexplored regions in the input space containing positive instances. Missing clusters can only be found by chance, demanding for random sampling. As a consequence of this reasoning the authors claim that instead of trying to set stopping criteria depending on the available data another alternative is to define a level of acceptable performance and stop the learning process when this level has been reached – they suggest using $F_1 = 80\%$.

Chapter 4

Text Classification

Classification tasks, in general, aim at assigning one or more classes, from a predefined set, to a given instance from the target domain. Text classification, also known as text categorization, refers to the classification task performed over text corpora. In text classification, classes, a.k.a. labels, are assigned to text documents.

Early text classification approaches, in use between the 60's and late 80's, were performed by domain experts deploying a set of rules to assign classes to documents in a specific domain. This expert system's approach has two major drawbacks: it is restricted to a specific target domain and requires a significant effort and time from the domain experts. These approaches are not scalable to the variety and volume of textual information that became available since the last decade of the 20th century with the widespread use of the Web and other Internet services. This mismatch between the chances offered by the amount of textual data widely available and the cost of text classification motivated the search for automatic text classification solutions. Research efforts focused on text classification in the area of machine learning arose naturally in the early 90's.

Nowadays, many of the text classification systems rely, to some extent, on automatic classifiers coming from the machine learning field. State-of-the-art text classification systems exhibit acceptable performance at a lower cost than that required by non-automatic systems relying exclusively on the knowledge of domain experts. Nevertheless, text classification has certain characteristics that make it a difficult task for machine learning. Text corpora, collections of text documents, are characterized by high-dimensional input spaces – frequently ranging over 10^4 dimensions – having many irrelevant features and containing high levels of noise. As a consequence, a large number of labeled instances is usually required to train. However, building classification rules by hand is certainly more demanding – concerning human effort and the required skills – than assigning labels to a set of text documents to be used to train a classifier (Hayes, 1992).

Text classification involves a set of tasks, including: (a) the proper preparation of text documents, which are by nature unstructured data objects, in order to extract the relevant features that are required by the classification process; (b) the proper representation of text documents in a format that is adequate for the classification process; (c) learning the target classes; (d) applying the learned model to classify new documents and (e) evaluate the performance of the classification process.

The pre-processing tasks, referred above as tasks (a) and (b), are discussed in Section 4.1, while tasks (c) to (e) are discussed in Sections 4.2, 4.3, 4.4 and 4.5.

4.1 Pre-processing

We consider a pre-processing stage comprising the tasks that are required to obtain a suitable document representation, valid for the subsequent automatic learning phase. This stage includes text preparation and text representation.

4.1.1 Text preparation

The text preparation phase takes a text document as input and returns a set of features describing it. This phase includes several steps that attempt to eliminate non-informative features and might involve some or all of the following (based on (Baeza-Yates and Ribeiro-Neto, 1999) and (Weiss et al., 2004)):

- tokenization – breaking the text document into tokens, commonly words; includes all sorts of lexical analysis steps, such as: eliminating punctuation, numbers, accents and extra spacing, converting to lower or upper case;
- stop-word removal – removing irrelevant terms; requires a list of stop-words (words to eliminate);
- stemming or lemmatization – reducing words to their semantic root; the Porter algorithm (Porter, 1980) is probably the most well known stemming algorithm; specific algorithms, such as the

one proposed by Orengo et al. (Orengo and Huyck, 2001) for the Portuguese language, are required for each language;

- feature selection – defining index terms, the features that will be used for document modeling. The full process of selecting features and computing their weights is known as *indexing*.

The application of these pre-processing tasks must be carefully done because the predictive power of words is highly dependent on the topic of interest (Chakrabarti et al., 1998a). Another essential aspect to consider is the language in which the document is written, which determines, at least, the list of stop-words and the stemming algorithm to use.

Stop-words removal is controversial. Removing words from a text, even those that in a linguistic sense have low semantic value, always reduces the information contained in the text document. Stop-word removal reduces the dimensionality of the feature space at a cost of loosing some information. A compromise solution must be set so that this information loss does not get counterproductive. What is left out from “to be or not to be” after stop-word removal? Recall that this phrase is completely made up of words that are frequently recognized as stop-words. To avoid this loss, some systems, like CiteSeer (Lawrence et al., 1999), for instance, do not remove any words from the documents to be indexed. Web documents, formatted in HTML or other markup language, still require splitting markup tags from content which is performed early in the tokenization step, before lexical analysis.

The words that appear in documents often have many morphological variants. Thus, pairs of terms such as “student” and “students”, will not be recognized as equivalent without some form of processing. Reducing morphological variants of words with the same semantics to a common root, or stem, is called *stemming*. A stem, by definition, is the portion of the word that is left after the removal of its affixes (prefixes and suffixes). Most frequently, several morphological variants of words have the same semantics and so they can be interpreted as the same for text categorization purposes. This way, not only the number of features gets reduced but also the topics described in the text get more noticeable to the learning algorithms since semantically similar words are conflated to a single representative form. For automatic processing purposes, it does not usually matter whether the stems generated are genuine words or not (for example: "computation" might be stemmed to "comput" instead of "compute") provided that different words with the same base meaning are conflated to the same stem and that words with distinct meanings are kept separate. Inflectional stemming, in linguistic terminology called morphological analysis, may be seen as a light stemming process limited to regularize grammatical variants such as singular/plural, genre and past/present. There are a number of stemming algorithms (known by stemmers or lemmatizers) available and widely used, such as, the Porter algorithm (Porter, 1980), the Krovetz algorithm (Krovetz, 1993) and the Lovins algorithm (Lovins) – the first stemmer with widespread dissemination, published in 1968.

In text classification the number of features is typically much larger than the number of training instances and, if care is not taken, undesirable overfitting may arise. Feature selection is desirable not only to avoid overfitting but also to reduce feature space dimension and, consequently, storage and processing cost. Feature selection (Yang and Pedersen, 1997) or feature reduction techniques may be heuristic – governed by linguistic principles or specific rules from the universe of discourse – or statistical. The procedure for feature selection usually comprises the following steps (Chakrabarti, 2003):

1. compute, for each feature, a measure that allows to discriminate target classes;
2. list features in decreasing order of that measure and
3. keep the subset of the features with the highest discriminative power.

The high feature space dimensionality, common in text corpora, can be reduced with techniques that might be categorized either as *feature selection* or *re-parameterization* techniques (Aas and Eikvil, 1999). Feature selection attempts to remove non-informative words from documents in order to improve categorization effectiveness and reduce computational complexity while re-parameterization is the process of constructing new features, as combinations or transformations of the original ones. Feature selection approaches are usually further classified as *wrapper* or *filter* techniques depending on whether they explore the learning algorithm to select the most appropriate features or not. Among common feature selection techniques we may include (Yang and Pedersen, 1997):

- *document frequency threshold* (Xu et al., 2008) relies on the inverse document frequency, i.e. the number of documents where the feature is present, and eliminates features whose inverse document frequency falls off some pre-defined threshold; the application of this technique is simple, inexpensive and has been providing good results, although it requires some care in the specification of the threshold value;

- *information gain* (Zheng et al., 2004) sorts features by decreasing order of their information gain; the most informative features are retained while the least informative are removed from the feature set;

- *mutual information* (Dumais et al., 1998; Novovičová et al., 2004; Peng et al., 2005) measures the association between features and classes based on a two way contingency table; features with the highest mutual information are selected;

- *chi-square* (Galavotti et al., 2000; Zheng et al., 2004) uses the same contingency table as mutual information but performs a chi-square statistical test to infer independence between features and target classes; the major advantage of this method, compared to mutual information, is that, since the test statistic is normalized, it allows for comparisons among features for the same class;

- *term strength* (Wilbur and Sirotkin, 1992; Liu et al., 2003) is significantly different from the above; this method computes each term strength independently from the document class. It assumes that documents sharing many common words are similar and, further, that the common words are informative. This method estimates term importance based on the conditional probability of a term appearing in a certain document given that it appears in another similar document;

- The *Markov blanket* criterion (Koller and Sahami, 1996; Aliferis et al., 2010) reduces the feature set by incrementally excluding the least relevant features until the reduced subset is satisfactory;

- *Latent semantic indexing* (Deerwester et al., 1990), LSI, is a re-parameterization technique, which uses the singular value decomposition of the document$\times$term matrix to reduce the dimension of feature space;

- *Part-Of-Speech (POS) tagging* assigns grammatical categories – such as verbs, nouns and adjectives – to terms in a text depending on their definition and context. Using POS tagging information for feature selection in text is a relevant approach to identify the most meaningful terms (Masuyama and Nakagawa, 2004; Gonçalves et al., 2006);

- *Named Entities (NE)* may also provide very relevant indexing terms for text in specific domains. NE are recognized as one of the most important indexing elements in biomedical text (Saha et al., 2009). NE Recognition (NER) aims to locate and classify terms into semantic categories, such as protein or gene, in the biomedical domain, or company or place in the business domain.;

- *Synonyms replacing* is another technique available to reduce the dimension of the features space in text (Bolshakov and Gelbukh, 2004). Synonyms may be replaced by a unique term. Synonyms may be obtained from lexical databases, like WordNet[1], or simpler synonyms dictionaries.

A different approach aimed at reducing the time and computational effort required to train a classifier is *sub-sampling* which uses only a reduced sample of the available data to train. *Text bundling* (Shih et al., 2003) is a sub-sampling technique that reduces the number of training instances by averaging together small groups of instances, such that important statistical information is retained. Text bundling organizes texts belonging to the same class into homogeneous groups. Each of these groups is averaged to generate a single bundled text that will replace all the group in the training set. This approach requires a pre-labeled set that might be used to organize the corpus in homogeneous groups corresponding to the target classes.

4.1.2 Text representation

Once the text preparation stage described above is concluded, each document is reduced to its representative features. Then, at the next step, text representation, this set of features is encoded into a specific format representing the document in an adequate manner for automatic processing.

[1]http://wordnet.princeton.edu/

Classic text models, view a document as a *bag-of-words*, describing text documents by the terms – words or phrases – appearing in it. In these models, each term in a document – known by *index term* – has a weight associated to it.

The vector space model (Salton et al., 1997), probably the most commonly used model for text representation, assigns real non-negative weights to index terms. In this model, documents are represented by vectors in a multi-dimensional Euclidean space. Each dimension in this space corresponds to an index term, a relevant term that is contained in the document collection and also part of the vocabulary in use. In the vector model, index term weights are usually obtained as a function of two factors:

- the *term frequency* factor, TF, a measure of intra-cluster similarity; computed as the number of times that the term occurs in the document, normalized in a way as to make it independent of document length and

- the *inverse document frequency* factor, IDF, a measure of inter-cluster dissimilarity; weights each term according to its discriminative power in the entire collection.

The degree of similarity of documents is evaluated as the correlation between the vectors representing the documents. This is usually quantified by the cosine of the angle between the two vectors.

Some proposals, distinct from the traditional vector space model, try to explore sequences of characters or words, known as *n-grams* (Cavnar and Trenkle, 1994; Lodhi et al., 2002; Zhang and Zhu, 2007; Rahmoun and Elberrichi, 2007).

Structured models (Baeza-Yates and Navarro, 1996), combining information on text content with information on the document structure, are also available although not as popular as bag-of-words models. At the end of the 80's and throughout the 90's, various structured text retrieval models have been proposed (Chakrabarti, 2003), such as non-overlapping list model, proximal nodes model (Navarro and Baeza-Yates, 1995), simple concordance list models (Dao et al., 1997), path prefixing and PAT expressions (Salminen and Tompa, 1994). These models, that explore the structural characteristics of the documents, are more directed for information retrieval (Baeza-Yates and Ribeiro-Neto, 1999), where the goal is to rank documents by their relevance to a given query, than for text classification, which is focused on assigning to documents the topics that describe their content.

Much textual data is extracted from hypertext documents retrieved from the Web. Hypertext documents contain other types of features besides content words (Ghani et al., 2000) that may hold significant predictive power. Hypertext features on Web pages include HTML tags, URLs, IP addresses, server names contained in URL, sub-strings contained in URLs, out-links (links from the current page pointing to other pages) and in-links (links from other pages pointing to the current page). Some models for hypertext documents include also the content of neighbors in the representation of a given document (Yang et al., 2002).

Pure text classifiers deal primarily with flat text documents and do not consider other features present in hypertext (Web) documents. Web page classifiers, relying on other features besides flat text, may be categorized according to the data they explore (Quek, 1998):

- Classifiers that use data on the Web page itself: typically text mining algorithms applied to the Web page content text and classifiers that try to explore HTML tags, like title and headings, which usually carry significant semantic value;
- Classifiers that rely on the hyperlinks between Web pages: the Web is a kind of social network and its structure naturally reveals clusters or groups of strongly interconnected pages, which are usually associated to some kind of common interest; this evidence can be leveraged in order to improve the classifier's performance;
- Classifiers that examine metadata about the page: metadata can be obtained from several sources, such as, specific HTML tags or the URL of the page that may contain some patterns that might help the classification task.

This is merely a conceptual partition of the evidential sources for the classification task, which does not prevent classifiers to combine information from several sources. In this thesis, we are concerned exclusively with flat text classification.

For the description of text models we will apply the following notation: $D = d_1, d_2, ..., d_N$ is the set of all documents, d_i, in the collection. N is the total number of documents in the collection. $T = t_1, t_2, ..., t_M$ is the set of all index terms, t_j, with cardinality M, the number of index terms in the collection or the dimension of the vocabulary in use. $w_{ij} \geq 0$ is the weight associated with index term t_j in document d_i; $w_{ij} = 0$ if term t_j is not present in document d_i. $\vec{d_i} = (w_{1j}, w_{2j}, ..., w_{NM})$ is the index term vector associated with the document d_i. g_j is a function that returns the j^{th} weight of $\vec{d_i}$, $g_j(d_i) = w_{ij}$. N_j is the number of documents in which the index term t_j appears, F_{ij} is the absolute frequency of term t_j in document d_i, f_{ij} is the normalized frequency of term t_j in document d_i, given by $f_{ij} = \frac{F_{ij}}{max_l(F_{il})}$ where $max_l(F_{il})$ is computed over all terms in document d_i. K is the number of classes to learn.

4.1.3 Bag-of-words models for text representation

Bag-of-words models do not take into consideration the structure of text documents. These text models assume that each document is described by a set of representative terms, called *index terms* (Baeza-Yates and Ribeiro-Neto, 1999). Index terms are words, or phrases, appearing in the document and usually assumed to be independent from each other. Each bag-of-words model assigns numeric weights to index terms, w_{ij}, computed by some specific method.

Boolean model The simplest bag-of-words model merely reflects the presence or absence of index terms in documents. Index terms' weights are binary, $w_{ij} \in 0, 1$. $w_{ij} = 1$ if the term t_j is present in document d_i and $w_{ij} = 0$ otherwise.

Vector space model The vector space model (Salton and Buckley, 1997) is an algebraic model that assigns real non-negative weights, $w_{ij} \geq 0$, to index terms. Documents are represented by vectors in a multi-dimensional Euclidean space. Each dimension in this space corresponds to a relevant index term contained in the document collection. A given document collection is then represented by a matrix W, with elements w_{ij} representing the weight of term t_j in document d_i.

These weights are usually obtained as a function of two factors: the term frequency factor, TF, and the inverse document frequency, IDF.

For the computation of w_{ij} weights there are several proposals (Aas and Eikvil, 1999):

- Boolean weighting: the simplest approach is to let the weight to be 1 if the term occurs in the document and 0 otherwise. In this case the vector space model becomes the Boolean model, which may be viewed as a particular case of the former (Equation 4.1).

$$w_{ij} = 1, \forall i, j : F_{ij} > 0; w_{ij} = 0, \forall i, j : F_{ij} = 0 \tag{4.1}$$

- Word frequency weighting: another simple approach is to use the frequency of the index term in the document (Equation 4.2). This approach, merely based on the TF factor, does not take into account the information value of the index term. One term might appear very frequently yet have a low information value.

$$w_{ij} = F_{ij} \tag{4.2}$$

- TF×IDF weighting: the previous schemes model a document independently of the collection where it is included. They do not take into account the frequency of the term throughout the entire collection which might be rather relevant for discriminative purposes. A common approach for computing w_{ij} is the TF×IDF weighting (Equation 4.3), which assigns weights to index terms that are proportional to the number of occurrences of the term in the document – the TF (term frequency) factor – and inversely proportional to the number of documents in the collection where the term occurs at least once – the IDF (inverse document frequency) factor.

$$w_{ij} = TF(d_i, t_j) \cdot IDF(t_j) \tag{4.3}$$

The coordinates of a document d_i are determined by two quantities:

Term Frequency, $TF(d_i, t_j)$ – the number of times that the term t_j occurs in document d_i normalized to make it independent of document length. There are many distinct ways to normalize $TF(d_i, t_j)$, such as dividing by the total number of terms in the document or dividing by the total number of occurrences of the most frequent term in the document.

Inverse Document Frequency, $IDF(t_j)$ – terms in the vocabulary are not equally important; $IDF(t_j)$ weighs the discriminative power of term t_j in the entire collection. The discriminative power of a term depends on its distribution in the document collection, not on its frequency. A very frequent term is not informative if its distribution in the collection is uniform. A common way to compute $IDF(t_j)$ is:

$$IDF(t_j) = log\frac{1+N}{N_j} \tag{4.4}$$

There are other ways of computing $IDF(t_j)$, in its majority as functions of N/N_j. The weight of index term t_j in document d_i, is then given by:

$$w_{ij} = f_{ij} \cdot log(\frac{N}{N_j}) \tag{4.5}$$

- TFC-weighting: the TF×IDF weighting does not take into account that documents may have different lengths. The TFC weighting is similar to the TF×IDF approach except that length normalization is used as part of the word weighting formula (Equation 4.6).

$$w_{ij} = \frac{f_{ij} \cdot log(\frac{N}{N_j})}{\sqrt{\sum_{j=1}^{M} \left(f_{ij} \cdot log(\frac{N}{N_j}) \right)^2}} \tag{4.6}$$

- Entropy weighting: inspired in information theory (Equation 4.7).

$$w_{ij} = log(f_{ij} + 1) \cdot \left(1 + \frac{1}{log(N)} \cdot \sum_{i=1}^{N} \left(\frac{f_{ij}}{N_j} \cdot log \frac{f_{ij}}{N_j} \right) \right) \tag{4.7}$$

  where $\frac{1}{log(N)} \sum_{i=1}^{N} \left(\frac{f_{ij}}{N_j} log \frac{f_{ij}}{N_j} \right)$ is the average entropy of term t_j. This quantity equals 1 if the word is equally distributed over all documents and 0 if the term occurs in a single document.

The vector space model is simple and fast, providing better or almost as good results as other known alternatives (Chakrabarti, 2003). This model ignores the context in which terms are included, considering them as isolated and independent features. It is assumed that the relative positioning of terms has a weak discriminative power. Although some models try to explore this relative positioning of words, by considering phrases as dimensions – instead of isolated terms – or statistical measures of co-occurrence, the underlying framework is based on a vector representation.

Some new proposals, distinct from the traditional vector space models, try to explore sequences of characters using kernel functions to measure similarity between documents; the *string kernel model*. Text can further be represented as sequences of words, which are linguistically more meaningful than characters, adapting string kernel functions to word kernel functions and significantly reducing the problem dimension (Lodhi et al., 2002; Cancedda et al., 2003).

The classic vector model assumes independence of index terms, which means that the set of vectors representing the t index terms forms a basis, of dimension t, for the indexing space. Usually this independence is interpreted in a more restrictive sense to mean pair-wise orthogonality between the index term vectors, i.e., meaning that for each pair of index term vectors, $\vec{t_r}$ and $\vec{t_p}$, we have $\vec{t_r} \otimes \vec{t_p} = 0$. In the *generalized vector space model*, the index term vectors are assumed linearly independent but are not pairwise orthogonal. In this model, index term vectors are composed as linear combinations of the orthogonal vectors that form the basis of the space, called the *minterms*, which are derived from the original index term vectors. The generalized vector space model adopts as a basic foundation the idea that co-occurrence of index terms inside documents induces dependencies among them. It is not clear whether this approach provides a clear advantage over the classic vector space model. Nevertheless, it is certainly more complex and computationally more expensive (Baeza-Yates and Ribeiro-Neto, 1999).

The degree of similarity between documents d_i and d_k, $sim(d_i, d_k)$, is evaluated by the correlation between vectors $\vec{d_i}$ and $\vec{d_k}$, which can be, and usually is, quantified by the cosine of the angle between the two vectors (Equation 4.8).

$$sim(d_i, d_k) = cosine(d_i, d_k) = \frac{\vec{d_i} \otimes \vec{d_k}}{\left\|\vec{d_i}\right\| \cdot \left\|\vec{d_k}\right\|} = \frac{\sum_{j=1}^{t} w_{ij} w_{kj}}{\sqrt{\sum_{j=1}^{t} w_{ij}^2}\sqrt{\sum_{j=1}^{t} w_{kj}^2}} \tag{4.8}$$

Since all weights are positive, $\forall i, j, w_{ij} \geq 0$, the similarity is always a positive quantity $0 \leq sim(d_i, d_k) \leq 1$.

4.2 Types of classification tasks

Document classification or categorization is the task of assigning one or more predefined categories to documents. Text classification tasks are frequently characterized by the properties of the set of classes to learn, C, and from the relationships between instances, i.e., text documents, and this set of classes. The number of classes to learn, K, is a common descriptor of a classification task. From this perspective, we refer to *binary* ($K = 2$) or *multi-class* ($K > 2$) classification (Kumar and Gopal, 2011). A complementary aspect, also very relevant, is the number of distinct classes that can be assigned to one single instance. From this point of view, *single-label*, a.k.a. *one of* or *multinomial* (Manning et al., 2008), refers to classification tasks assigning one single class to each instance while *multi-label* (Tsoumakas and Katakis, 2007; Moskovitch et al., 2006), a.k.a. *any of* or *multi-value* (Manning et al., 2008), refers to classification tasks where each instance may be associated with more than one class. The structure of the set of classes to learn is also used to describe the classification problem. *Flat* classification refers to unstructured class sets, having all classes at the same level, while *hierarchical* classification (D'Alessio et al., 2000; Sun and Lim, 2001; Silla and Freitas, 2011) refers to a set of classes organized as a hierarchy, a frequent setting in text categorization.

In single-label problems, each document belongs to just one of the K classes to learn. The probability of accurately classifying a given document by chance is $\frac{1}{K}$, assuming a uniform distribution. For multi-label problems each document may belong to just one or more than one of the K classes. In this case, each document might be assigned to any of the $2^K - 1 = \binom{K}{1} + \binom{K}{2} + ... + \binom{K}{K}$ subsets that can be obtained from the set of classes to learn, C. The probability of accurately classifying a given document by chance in multi-label problems is thus $\frac{1}{2^K-1}$, assuming a uniform distribution. These thresholds may be used as benchmarks for evaluation purposes.

Classification tasks may then be characterized by these dual aspects: binary or multi-class, referring to the cardinality of C, single-label or multi-label, referring to the number of classes that an instance may belong to and flat or hierarchical, referring to the internal structure of C. By nature, binary problems cannot be simultaneously multi-label neither hierarchical.

The probability of predicting the true class by chance in single-label classification, assuming a uniform class distribution, is $\frac{1}{K}$. In the special case of binary classification, where $K = 2$, this probability is 50%. Binary classification algorithms assume a particular relevance since they can be used to solve multi-class and multi-label problems by mapping the original problem to a set of independent binary sub-problems whose output is aggregated to produce an answer for the original (multi-class or multi-label) problem (Sebastiani and Ricerche, 2002).

This same strategy is frequently used in multi-label classification by breaking down the original problem into a set of single-label problems whose outputs are then combined to produce the answer for the original problem (Boutell et al., 2004; Diplaris et al., 2005). These problem transformation strategies are independent from the base classifier. Another strategy to deal with multi-label classification, known by algorithm adaptation (Tsoumakas and Katakis, 2007), consists in transforming the learning algorithms used in single-label classification to adapt them to the multi-label setting (Clare and King, 2001; Crammer and Singer, 2003; Godbole and Sarawagi, 2004; Zhang and Zhou, 2005).

In hierarchical classification it is also common to break down the original problem into a set of flat problems for simplicity reasons. These approaches, however, do not take into account the information contained in the hierarchical structure of the concept to learn (Koller and Sahami, 1997; Kiritchenko et al., 2006). Top-down, or level-based, approaches, take into account the hierarchical nature of the target concept at a local level (Sun and Lim, 2001). These approaches build flat classifiers to learn classification models that can predict the classes at each level of the hierarchy. These flat classifiers are then applied sequentially at all levels in the hierarchy that are deemed relevant for a given instance by the previous level classifier. The process stops when a certain level classifier does not find any relevant class for the instance at hand at its own level or when the process reaches a leaf node. Global approaches for hierarchical classification, known by *big-bang* (Sun and Lim, 2001; Kiritchenko et al., 2006), build a single classifier able of discriminating all the classes in the hierarchy taking into consideration the existing hierarchical relationships.

4.3 Learning settings

The application of machine learning techniques to classification problems generally requires two distinct stages: (1) the *learning stage* – when the classifier is learned, that is the algorithm builds a model of the concept to be learned based on training and test data – and (2) the *production stage* – when the previously learned model is applied to unseen instances in order to classify them. The learning stage demands for a sample of the target population that is to be partitioned in a training set and a test set. The need to label instances in this sample, according to the specific target concept, is probably one of the most expensive tasks experienced during all the classification process. Therefore, it becomes a core aspect to take into consideration.

Irrespectively of the learning setting in use, some target classes may not be learnable for several reasons (Schütze et al., 2006), such as, having few instances available in the corpus or using a text model that is not expressive enough for the purpose of the classification task – the bag-of-words model, for instance, does not take into consideration the relative order of words which might be of crucial importance.

When both training and test sets are fully labeled, we are in presence of a *supervised learning* setting. On the other extreme, if none of the training instances is labeled, we are in presence of *unsupervised learning*. When the training data is partially labeled, we are in presence of a *semi-supervised learning* setting. Bennet et al. (Bennett and Demiriz, 1998) further classify semi-supervised learning problems as either *semi-supervised clustering*, when the number of labeled instances is small when compared to the dimension of the training set, or *transduction* problems, when the number of labeled instances is large when compared to the training set dimension. The transduction problem refers to the estimation of the value of a classification function at a given instance, which opposes to the standard inductive learning problem of estimating the classification function for all possible instances and then using the fixed function to deduce its value at a given instance.

The supervised setting requires the full dataset, from where the training and test samples are obtained, to be labeled or, at least, that there is a large number of labeled instances from each class. This is one major drawback in this setting, concerning text classification, because of the high cost of labeling text documents. The process of manually assigning labels to text documents is both time consuming, inaccurate and subject to incontrollable factors arising from human nature (Macskassy et al., 1998) – two users with the same skills may classify the same page differently or the same user may classify the same page differently at different moments of time. In opposition to this passive learning process, *active learning* (Chapters 2 and 3) gives the learner the ability to select which instances should be included in the training set. Active learning (AL) reduces the amount of labeling that needs to be done through selective sampling of unlabeled data. In AL the learner examines a collection of unlabeled instances and selects the most informative ones, requiring an annotator to label the selected instances, and iteratively re-trains on the augmented set of labeled training examples.

In the unsupervised setting there is no prior knowledge on labels, neither on the labels of each document nor even on the labels themselves. Clustering algorithms organize documents in homogeneous groups, based on their similarity, forming partitions of the dataset that minimize intra-group variance and maximize inter-group variance.

Semi-supervised learning techniques are particularly interesting when the process of labeling training data is expensive and time consuming, as is the case of labeling text documents. In this setting, classifiers are wrapped by some method in order to take advantage of unlabeled documents. Several approaches have been proposed to solve the semi-supervised learning problem:

- *Bootstrapping* (Jones et al., 1999) is a simple iterative method. At each iteration, labeled instances are used to learn a classifier. This classifier is applied to label unlabeled instances; those where there is enough evidence in favor of a certain label against the others are added to the labeled set. The algorithm proceeds iteratively until convergence.

- Usage of *Expectation-Maximization* (Nigam et al., 2000) (EM) to estimate maximum a posterior parameters for a generative text classification model.

- *Co-training* (Blum and Mitchell, 1998) is a supervised learning method, particularly useful when it is required to combine sources of evidence originated form very distinct spaces – particularly if they have rather different dimensions and scales, which may bias the aggregation of measures from these distinct sources. With co-training distinct classifiers keep disjunctive, independent feature sets and their estimates are never directly consolidated; instead this method uses the estimates of one classifier to train others. The application of co-training requires the existence of distinct and independent sets of features. Blum et al (Blum and Mitchell, 1998) apply co-training to Web document classification, a field where the features are naturally separable into disjoint sets, such as text in the page itself and words appearing in the in-links to the page, and two classifiers, one for each feature set, can be built.

- *Error Correcting Output Code* (Dietterich and Bakiri, 1995) (ECOC) is a method that converts a K-multi-class problem in a set of L binary problems (Witten and Frank, 2000). Any binary classifier can then be used to learn these L problems. ECOC assigns to each class a unique binary string, the code word, where each bit is predicted by one of the binary classifiers. The predicted class is the one whose code word is closest to the code word produced by the set of the L binary classifiers. The distance between code words is computed by the Hamming distance, which is calculated as the number of different bits in both code words. Ghani (Ghani, 2001) describes a method for semi-supervised learning that uses the ECOC method bundled with co-training techniques in order to learn the binary classifiers.

- *Transduction* (Gammerman et al., 1998) is naturally related to instance based learning. In transduction we are interested in the classification of a particular instance rather than in a general rule for classifying any future instance.

- *Coaching* (Tibshirani and Hinton, 1998) is a technique that applies when we are in the presence of two distinct sets of predictive variables but only one of these will be available on the future examples to classify. Coaching techniques use one of the sets of predictive variables to coach the other set how to improve prediction in the absence of the former.

4.4 Text classifiers

Many classification problems are binary in nature: a given example either belongs to some specified concept or it does not. In text categorization we are usually interested in sets of classes with more than just two distinct classes: the classification problem is frequently a multi-class problem.

One common approach to multi-class problems, valid for some classifiers, is to use a set of binary classifiers, each one responsible for determining the relevance of the document as to one specific class. The relevance scores of each one of the individual binary classifiers are then combined in order to provide the final answer.

Several types of classifiers used in machine learning in tabular, structured data, are also applied to unstructured high-dimensional problems like text classification.

4.4.1 Rochio

Rochio's algorithm (Joachims, 1997) is a classic method for document categorization in Information Retrieval[2] (Manning et al., 2008). In this method the training examples are used to build a prototype vector for each class. The prototype vector for each class is computed as the average vector over all the training document vectors that belong to the class. A new document is classified according to the distance measured between the document vector and the class prototype vectors.

[2]The purpose of Information Retrieval is not to assign classes to documents but to rank documents according to their similarity to a given query document – usually a user query specified through a set of keywords.

4.4.2 K-Nearest-Neighbors

K-Nearest-Neighbors (KNN) is an instance based classifier which has been demonstrating good performance in pattern recognition and text categorization problems (Yang and Chute, 1994; Yang and Pedersen, 1997). This method classifies a document based on the characteristics of its closest k neighbor documents in the input space.

In applications to text categorization, documents are usually represented in the traditional vector space model and the cosine between document vectors is also frequently used as the similarity measure. Classes might be assigned by some voting scheme – the majority class in the k neighbors is assigned, for instance – or, the classes that have a relevance score above a given threshold are assigned to the document (Yang et al., 2002).

KNN is a local method based on instances that does not require any training stage. However, it demands for a fully pre-labeled set of instances.

4.4.3 Naive Bayes

Naive Bayes methods (Kibriya et al., 2005; Kim et al., 2006; Jiang et al., 2011) use the joint probability of terms, t_i, and categories, c_j, to estimate category probabilities given a document, $P(c_j|t_1,t_2,...,t_n)$. Dependencies between terms are ignored, i.e., Naive Bayes assumes that the conditional probability of a term given a category is independent of the conditional probability of any other terms given the category – the naive assumption.

When assuming term independence, the conditional probability of document d, given class c_j can be obtained by Equation 4.9:

$$P(d|c_j) = \prod_i P(t_i|c_j) \tag{4.9}$$

Given a document, the algorithm computes the posterior probabilities of each one of the classes and assigns to the document the most probable one, c_{NB} (Equation 4.10). Marginal, a priori, class probabilities, $P(c_j)$, may be estimated from the class distribution in the training set.

$$c_{NB} = \underset{c_j \in C}{\text{argmax}} P(c_j) \prod_i P(t_i|c_j) \tag{4.10}$$

Computing class posteriors requires to estimate the conditional probabilities $P(t_i|c_j)$, which may be obtained by Equation 4.11:

$$P(t_i|c_j) = \frac{n_i + 1}{n + |vocabulary|} \tag{4.11}$$

In Equation 4.11, n is the total number of terms in all training documents belonging to category c_j, n_i is the frequency of term t_i and $|vocabulary|$ is the total number of distinct terms in the training corpus – the lexicon cardinality (Fang et al., 2001).

The constant 1, added to the numerator, and $|vocabulary|$, added to the denominator, are both necessary to avoid the $0/0$ indeterminate that would arise for the terms, t_i, not appearing in the training documents belonging to class c_j, thus forcing $P(d|c_j) = 0$ in such cases.

Special care is required when applying Naive Bayes to high-dimensional data, as is the case of text corpora. In fact, when dealing with very large sets of attributes, problems relating to the limits of precision in computers may arise. By nature of probability, $P(x|y) <= 1$. It is also true that $\lim\limits_{n \to \infty} \prod_i P(t_i|c_j) = 0$. In practice, it may happen that n grows large enough for the value of $\prod_i P(t_i|c_j)$ to exceed below the limits of double precision floating point numbers in modern computers. Computing the logarithm of the conditional probabilities as follows, addresses this problem (Equation 4.12).

$$\underset{c_j \in C}{\text{argmax}} \left[-log(P(c_j) \prod_i P(t_i|c_j) \right] = \underset{c_j \in C}{\text{argmax}} \left[-log(P(c_j)) - \sum_i log(P(t_i|c_j)) \right] \tag{4.12}$$

4.4.4 Decision trees

Decision trees are decision structures built over a root node containing all the training instances (Witten and Frank, 2000; Lewis and Ringuette, 1994). The set of instances in any specific node is partitioned into its descendant nodes. This split is made with the objective of minimizing the diversity of categories present at each node and it is carried out until no further reasonable improvement is possible. At a given node, the split is made as to assure that the sum of the diversities at the child nodes is (much) less than the diversity at the present node without splitting. The goal is to maximize $diversity(node) - \sum diversity(childnodes)$.

Care must be taken to avoid overfitting which is usually done by pruning the decision tree.

There are two common pruning approaches: *post* or *backward* pruning and *pre* or *forward* pruning (Witten and Frank, 2000). In the pos-pruning approach the tree is expanded as much as possible at an initial stage and then it is pruned back by removing those nodes that do not significantly improve the homogeneity, hence the predictive power, of the tree. Pre-pruning approaches evaluate when to stop developing sub-trees during the tree construction process.

The nodes at the bottom of the tree are called the *leaf nodes*. Any training example belongs to a certain leaf node. Each leaf node is then assigned to a class and the error rate of the leaf is the probability of examples in that leaf node being misclassified. The global tree error rate is a weighted sum of all the leaf nodes error rates.

A key issue in decision trees is to decide which features allow for the best split at each node, the one that generates the most homogeneous partition, and the definition of the diversity measure to use. One of the most common diversity measures is entropy. The entropy of a given node, L, is given by Equation 4.13.

$$-\sum_{j=1}^{K} P(c_j|L)log(P(c_j|L)) \tag{4.13}$$

Where $P(c_j|L)$ is the probability of a training example belonging to class c_j given that it is located in node L, which can be estimated by the relative frequency of class c_j in node L (Equation 4.14):

$$P(c_j|L) = \frac{N_j(L)}{N(L)} \tag{4.14}$$

$N_j(L)$ is the number of instances of class c_j in node L and $N(L)$ is the total number of instances in node L. Several algorithms are available to grow decision trees – CART, ID3, CHAID and C4.5 are common approaches (Aas and Eikvil, 1999). The CART algorithm (Breiman et al., 1984) builds a binary decision tree by splitting the training examples at a given node. The splitting rule is a linear combination of features. The main task is to decide, at each node, which combination of features performs the best partition and what is the split value. ID3 (Quinlan, 1986) splits nodes based on the unused attribute exhibiting minimum entropy – i.e., maximum information gain. The C4.5 (Quinlan, 1993) algorithm builds decision trees that have two children per node, when splitting on numeric features, but might have more then two children per node when the splitting rule is based on a categorical attribute. In such cases, it is not limited to binary trees (two children per node) as is CART. CHAID (Kass, 1980) is also a popular algorithm but it is limited to categorical features; thus, if the domain under study has numeric attributes then they must be previously discretized.

4.4.5 Support Vector Machines

Support Vector Machines (SVM) (Joachims, 1998) are based on the intuition that a hyperplane that is close to several training examples will have a bigger chance of making erroneous decisions than one which is as far as possible from all training examples. The SVM algorithm is a binary classifier that defines a maximum margin hyperplane between the convex hulls formed by the training examples of each class. The maximum margin hyperplane is the one that is as far away as possible from both convex hulls – it is orthogonal to the shortest line connecting the hulls, intersecting it half way. This hyperplane may be defined as a function of the training examples that are closest to it, the *support vectors*.

SVMs, like all discriminative classifiers, are non-parametric. They do not assume any underlying data distribution besides the trivial assumption that training and testing instances all come from the same population, thus assuming identical distributions.

SVM implementations require some parameter tuning depending on the type of kernels in use. Linear kernels require no parameters. Radial Basis Function (RBF) kernels require setting γ, the width of the RBF kernel, and the cost parameter, C, that affects the trade-off between model complexity and training error, that is, the acceptable proportion of nonseparable instances. If C is too large, favoring model complexity, we have a high penalty for nonseparable instances which may lead to store many support vectors and overfit – $C = 1$ is a small value for C while $C = 1000$ is high.

4.4.6 Exploring other features besides content text

Hypertext (Web) documents might have some additional attributes besides content text. When compared to plain text this type of documents allows for a richer representation that might be explored in (hyper)text classification. Yang et al (Yang et al., 2002) define five types of regularities that might be present in hypertext collections:

- *no regularity* – the only place that has relevant information about the class of the document is the document itself,

- *encyclopaedia regularity* – documents with a given class only link to documents with the same class,
- *co-referencing regularity* – some, or all, of the neighboring documents belong to the same class but this class is distinct from the document class,
- *pre-classified regularity* – the regularity is present at the structural level where a single page (hub) points to several pages which belong to the same class and
- *metadata regularity* – metadata is available from external sources and can be explored as additional sources of evidence generating new features.

The authors then define the types of features that should be used in order to improve the classification task of documents belonging to each of these regularities. However, there is no suggestion as to how to previously determine the type of regularity that is present in a given document collection. According to their experiments different classifier designs should be considered, depending on which of the above regularities holds. With no regularity we would not expect any benefit from using hyperlinks and the suggestion is to use flat text classifiers, exclusively based on the text of the document itself. Encyclopaedia regularity suggests augmenting the text of each document with the text of its neighbors, thus increasing the number of words related to the topic that are present in the document representation. In the case of co referential regularity, the text of the document should also be augmented with the text from its neighbors but these imported words should be treated as if they came from a different vocabulary, for instance prefixing them with a specific tag. If the collection has a pre-classified regularity then there is no need to look at the document text, it suffices to look at the pages that link to it and determine their class. When external sources of information are available that can be used as metadata, metadata regularity, we can collect them – possibly relying on information extraction techniques.

Chakrabarti et al (Chakrabarti et al., 1998b) also test several feature sets, similar to the ones suggested by (Yang et al., 2002): local text, local text concatenated with all neighbors text, local text plus neighbors text prefixed with discriminative tags. They conclude that naive use of terms in the link neighborhood of a document can even degrade performance. Yang et al (Yang et al., 2002) have reached the same conclusion, which seems consensual. Although the use of extended sets of features available in hypertext collections – including text from hyperlink anchors, the full text from neighbor documents, HTML tags, category distribution over a linked neighborhood, metadata available from external sources – might provide rich information for the classification task, it is not guaranteed that the use of such features will improve performance, which in general is dependent of the specific document collection.

A folksonomy[3], a.k.a. social classification or collaborative tagging is a distributed unsupervised classification system created and maintained by a group of individual users. Folksonomies may suffer from common problems related to tag ambiguity, synonymous tags or multilingualism (Robu et al., 2009; Wetzker et al., 2010; Trattner et al., 2011). Nevertheless, an empirical analysis of the complex dynamics of tagging systems (Halpin et al., 2007) shows that coherent categorization schemes can emerge from unsupervised tagging by groups of users.

4.5 Performance measures

Performance evaluation is one of the most important issues in machine learning, in general. In classification tasks, this evaluation can be based on several measures. Common measures in text classification are recall, precision, F-measure – which aggregates recall and precision in a single measure – and accuracy or error – two complementary measures of the efficiency of the learner.

Recall is defined as the ratio between the number of documents correctly classified and the total number of documents in the category. *Precision* is defined as the ratio between the number of documents correctly classified and the total number of documents classified in the category.

Usually a classifier exhibits a trade-off between precision and recall. These measures are negatively correlated: improvement in recall is made at the cost of precision and vice-versa. It is frequent to have text classifiers operating at the break-even point – the operating point where recall and precision have the same value. The F-measure combines recall and precision in a unique indicator (Equation 4.15):

$$F_\beta = \frac{\left(\beta^2+1\right) \times precision \times recall}{\beta^2 \times precision + recall} \tag{4.15}$$

where β is a parameter allowing different weighting of precision and recall – precision and recall are equally weighted when $\beta = 1$. At the break-even point, recall, precision and F_1 all have the

[3]http://vanderwal.net/folksonomy.html

same value.

Accuracy and *error* are complementary measures of the error probability of the classifier given a certain category. Accuracy is defined as the ratio of the number of correctly classified examples by the total number of evaluated examples, while error is defined as the ratio of the number of incorrectly classified examples by the total number of evaluated examples.

These measures are defined for binary classifiers. To measure performance in multi-class problems there are two common approaches that aggregate the measures evaluated at each singular class: macro-averaging and micro-averaging. *Macro-averaged* measures are obtained by first computing the individual scores, for each individual class, and then, averaging these scores to obtain the global measure. *Micro-averaging* measures are obtained by first computing the total number of documents correctly and incorrectly classified, irrespectively of their classes, and then using these values to compute the global performance measure by applying its definition. There is an important distinction between these two aggregation algorithms: macro-averaging equally weighs all classes while micro-averaging gives equal weight to every document.

Chapter 5

Problem Statement

Organizing objects into a set of classes is probably one of the oldest and most effective ways for storing and retrieving information (Dewey, 2004). This cataloging process requires the specification of a model defining the classes in the target concept. Defining classes from instances is an adequate strategy for this purpose in machine learning: on the one hand, it is expressive enough – we may express almost any information need from a set of instances; on the other hand, it does not demand for specific domain knowledge – it does not require neither the explicit specification of the classes to learn nor any formal specification model. However, retrieving and labeling exemplary instances to get a full description of the target concept may still be very demanding.

5.1 Motivations

Active learning (AL) is an iterative process guided by specific criteria tailored to meet specific goals. Our research is focused on building accurate classifiers recognizing all target classes at a reduced cost when compared to current approaches. Our main motivation is to reduce human effort in text categorization of documents in large corpora.

5.1.1 Labeling cost

Collecting and annotating instances that fully describe the target concept is a critical and demanding stage in classification tasks (Li and Sethi, 2006). It is critical because it is one of the first stages of the whole classification process and limits the performance of all the following stages. It is demanding because it requires domain experts to retrieve and label exemplary instances for all classes to learn. Given the weight of this initial stage in the cost of the whole process – approximately 80% of the total effort (Li and Sethi, 2006) – any improvements at this stage may be very important to reduce editorial costs. The number of labeled instances that are required to learn the target concept may be reduced by selecting the most informative instances, instead of selecting instances to label at random. Knowing how to select the most informative instances to label is an important aspect regarding cost reduction.

AL implements iterative processes that select the most informative instances to query at each iteration. As a consequence, the utility of unlabeled instances decreases as the learning process proceeds. Knowing when this utility has dropped below the minimum acceptable prevents wasted queries and unnecessary costs. Defining a proper stopping criterion is another important aspect regarding cost reduction in AL.

This thesis is directed to reduce the total cost of the entire learning process.

5.1.2 Full class coverage assumption

AL seems adequate to our goals as it learns from instances that are selected by the learning algorithm according to ad-hoc criteria. These criteria may be biased to meet specific goals that go beyond error minimization – the traditional goal in classification tasks is to achieve low error. This does not mean that we are not concerned with classification error; instead, it means that we are not *exclusively* concerned with error. Besides error, representativeness – covering all the target classes – is also crucial in our research.

There is a variety of underlying approaches to apply AL to classification problems. Just to name a few, we may refer to AL classifiers based on uncertainty, see for example (Lewis and Gale, 1994; Becker and Osborne, 2005; Cebron and Berthold, 2009), others that rely on distance (Hochbaum and Shmoys, 1985; Tong and Koller, 2002; Brinker, 2003), on clustering techniques, such as (Xu et al., 2003; Nguyen and Smeulders, 2004; Dasgupta and Hsu, 2008), or on committees of classifiers (Seung et al., 1992; Iyengar et al., 2000; Lu et al., 2010). However, all the above mentioned solutions assume the availability of a pre-labeled set covering all the classes to learn. This assumption is not valid in our setting – we assume no previous knowledge on the target concept – which demands for a new approach.

5.1.3 Imbalanced class distributions

The effort required to retrieve and label representative instances is not only related to the number of target classes (Adami et al., 2005); it is also related to the class distribution in the available pool. Random sampling over a fairly balanced pool will provide exemplary instances from all classes with a high probability. On a highly imbalanced class distribution, however, it is particularly demanding to identify instances from minority classes. In such a setting, random sampling will not be efficacious in finding representatives for those classes. These, however, may be important in terms of representativeness (Attenberg and Provost, 2011). It is the case, for instance, of Web resources (Escudeiro and Jorge, 2006), news organization (Ribeiro and Escudeiro, 2008) and assisted assessment of written examinations (Escudeiro and Escudeiro, 2011) where minority classes may correspond to specific concepts which are as relevant as those corresponding to majority classes. If this specificity is not taken into account, many queries will be wasted before these special instances are retrieved. In many more situations, such as fraud detection and clinical diagnosis, we face the problem of imbalanced class distributions where minority classes are probably the most valuable to be properly identified (Ertekin et al., 2007). Failing to identify instances from under-represented classes may have costs and, in some cases, put at risk the goals of the classification problem under consideration.

5.1.4 Missed clusters

Common active learners focus on the uncertainty region asking queries that are expected to narrow it down. The issue is that the uncertainty region is determined by the labels that are known in advance. Focusing our search for queries exclusively on this region, while we are still looking for exemplary instances on some labels that are not yet known, is not effective. Under such circumstances, unknown classes hardly come by unless, by chance, they happen to be represented in the current uncertainty region. This myopic view of the uncertainty region has already been noticed in the early days of AL. Cohn et al. (Cohn et al., 1994) refer to the limitation generated by the inductive bias of some learning algorithms that tend to draw sharp distinctions in input space becoming overly confident in regions that are still unknown. As a result, the uncertainty region, as perceived by the learning algorithm, will in general be a small subset of the true region of uncertainty. This behavior may be more evident when in presence of imbalanced datasets. To our knowledge, there is no general procedure, so far, for determining missed clusters (Schütze et al., 2006) – unexplored regions of the input space that contain positive instances.

5.2 Research questions

The research problem investigated in this thesis – efficient AL strategies to build accurate classifiers aware of all the target classes – is closely related to the typical exploitation versus exploration compromise in AL. Exploitation bearing tends to improve the decision functions in known regions in instance space while a exploratory drift is likely to query instances in unknown regions. AL algorithms focused on exploitation tend to generate narrower classifiers but more sharp over known classes while focusing on exploration tends to generate broader classifiers but not as sharp. Favoring exploitation might be an adequate strategy when in presence of a pre-labeled set covering all the classes to learn but an exploratory bias is required, mainly at an initial stage of the learning process, when the training set does not cover all the target classes. Shifting between these two modes of operation during the learning process might contribute to improve efficiency.

Several research questions arise in this scenario addressing specific issues that are expected to contribute to reduce the labeling cost:

(i) How can we improve instance space coverage and early class exposure?

(ii) How to swap between exploration and exploitation lightly?

(iii) How to identify the most appropriate time to stop querying?

(iv) How can we assess the effectiveness of such approaches?

Answering these questions will guide our investigation towards the main research question:

(v) How can we reduce the labeling effort that is required to build classifiers that are aware of all the classes to learn and still accurate?

5.3 Hypotheses

The aim of our research is to produce a strategy that, with a reduced number of queries to the user, is able to generate a classifier that has a high predictive ability and covers the set of existing classes. Our hypotheses are:

1. an active learning strategy based on distance and confidence improves the results of existing strategies in terms of number of queries needed to cover the set of classes without compromising predictive ability,
2. a stopping criterion based on the combination of classification gradient and label distribution provides better results.

5.4 Formal problem setting

A formal description of the research problem provides a general setting supporting further developments and promoting discussion related to this investigation.

AL in general, and our proposal in particular, includes a set of data objects that evolves over time, as the learning process iterates, converging to the final solution. In general, AL classification is an iterative process where each cycle includes a learning stage, a predicting stage and a querying stage. The learning stage outputs a classifier from a set of labeled instances. This classifier is then used to predict the labels of unlabeled instances at the predicting stage. These predictions are sustained by the statistics that are output by the classifier, such as confidence, or computed from these, such as entropy. At the querying stage, the AL algorithm selects unlabeled instances to query relying on some ad-hoc selection heuristic based on current evidence. This selection heuristic is expected to maximize the utility of the queries given the goal of the learning task.

5.4.1 General setting

A target concept is characterized by a set C of classes – a.k.a. labels – c_k, $k \in \{1,2,...,K\}$. W is a set of N instances, x_j, $j \in \{1,2,...,N\}$, that is assumed to be representative of the target concept, i.e., that contains examples from all classes in C. The true class y_j of instance x_j is not known in advance for any instance $x_j \in W$. However, it can be requested to some domain expert, E, throughout the learning process. None of the classes to learn are previously specified with the exception of what can be inferred from W.

We are assuming an iterative learning process with queries being asked at each iteration. When a query is asked we assume that the true label y_j of one unlabeled instance x_j is always provided. At each iteration, i, during the learning process, L_i and U_i form a set partition of W. L_i is the subset of instances in W whose true label is known at iteration i, $L_i = \left\{ < x_j, y_j >: x_j \in W \wedge y_j = E(x_j) \in C \right\}$. U_i is the subset of instances in W whose label is not known at iteration i.

A domain expert, E, knowing the target concept and being aware of each of the classes in C, is always available. At each iteration, i, this expert may be queried for the label of a single unlabeled instance $x_j \in U_i$ – batch mode AL was not considered – at a certain cost, A. When queried for a label, the expert always provides its true label, $\forall j, E(x_j) = y_j, y_j \in C$ – the expert E is always available and always certain.

We assume the availability of a base classification algorithm that generates hypotheses – a.k.a. classifiers, h, from a set of labeled instances. The classifier, h_i, generated at each iteration, i, from L_i, predicts labels, $\hat{y}_j$, for the instances $x_j \in U_i$.

The utility of an instance at iteration i, B, is the value of the improvement in the performance of h that a query may induce if included in L_{i+1}.

5.4.2 Learning process

In general, AL is an iterative process. Each iteration has three phases: learn, predict and query. This learning process is initialized from a set of pre-labeled instances, L_1. This set must contain at least two labeled instances from W having distinct labels. This imposition stems from the fact that we need at least a positive and a negative example to boot up a classifier. Besides this imposition, there are no other constrains to the building process of L_1. A core concern, however, must be taken into consideration. Since we are focusing on cost reduction and the cost of L_1 is $N_1 A$, assuming N_1 is the number of pre-labeled instances in L_1, then N_1 should be small, ideally $N_1 = 2$ as we use. The instances in L_1 are randomly selected from W. Their labels are requested to the domain expert, E.

Once this initialization set, L_1, is built we enter the iterative learning process. At each iteration, i, the labeled set L_i is used to build a classifier, h_i, that predicts labels to all instances in U_i. Then, the AL criteria in applied to select a query, q_i from U_i. We are assuming that one single query is selected at each iteration. Batch mode AL was not considered.

The label of the selected query is requested to the expert, E. The query is then added to the labeled set, L_i and removed from the unlabeled set, U_i. The labeled set for the next iteration, L_{i+1}, is the union of the previous labeled set and the query selected at the current iteration $L_{i+1} = L_i \cup \{< q_i, E(q_i) >\}$. The unlabeled set for the next iteration, U_{i+1} is the set difference between the previous unlabeled set and the query selected at the current iteration $U_{i+1} = U_i \setminus \{q_i\}$. L_i and U_i always form a partition of W. This process iterates until a given stopping criterion is met.

The verification of the hypotheses under investigation depends on:

1. the early identification of instances whose labels fully cover C, i.e., there should be a small i such that $\forall c_k \in C, \exists x_j \in L_i : E(x_j) = c_k$:

2. simultaneously, a low error rate should be observed at the predictions made by the learned hypothesis, i.e., the ratio of the number of correct predictions made by h_i on U_i by the cardinality of U_i should be low when compared to current approaches;

3. the identification of effective stopping criteria preventing costly useless queries.

5.4.3 Evaluation of solutions

The evaluation of the solutions for our problem should be based on error, an essential performance dimension in classification, and on the number of target classes that are known, i.e., that have representative instances in L. Having representatives from all classes in C is a core concern in the research problem being investigated.

At each iteration, i, the *error rate* is evaluated on the set of unlabeled instances, U_i – generalization error. The number of *known classes* is evaluated on the set of labeled instances, L_i. Besides the number of known classes in itself, it is also important to record the first time that a given class is identified, i.e., the first iteration outputting a query being labeled with a given class, c_k. We will refer to this indicator as *first-hit*. From a broader perspective it is also important to evaluate the minimum number of queries that are required to identify representative instances of all the target classes. we define *label disclosure complexity* for this purpose. All these performenca indicators are defines in Section 6.3.1.

Chapter 6

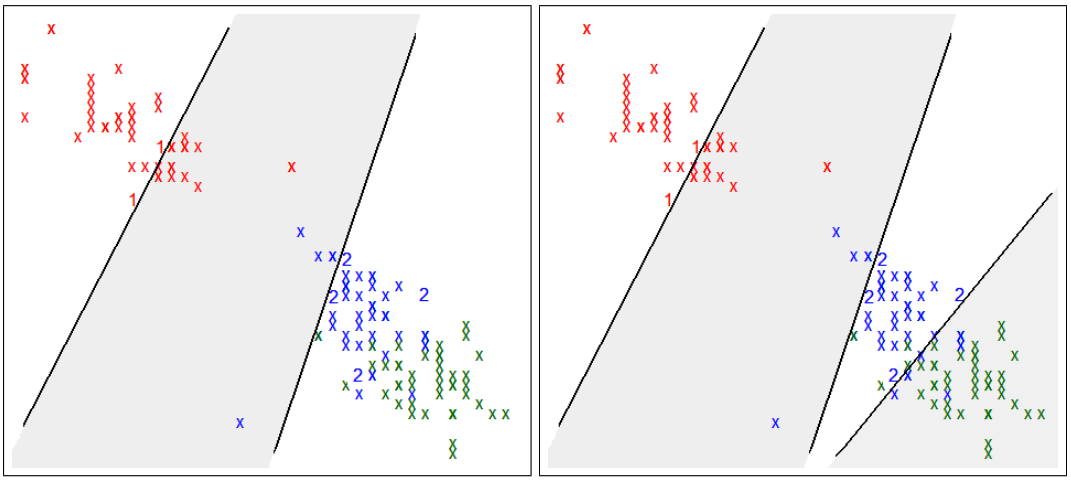
D-Confidence

Given a target concept with an arbitrary number of classes together with a sample of unlabeled instances from the target space – the working set, W – our purpose is to build an accurate classifier covering all target classes while posing as few queries as possible. A query consists of requesting the oracle, E, to provide the true label, y_j, to a specific instance, $x_j \in U$. U is the set of instances in W whose label is not known. Querying E for a label has a cost, A – the querying cost – assumed to be constant throughout the learning process. The working set is assumed to be representative of the class space – the representativeness assumption (Liu and Motoda, 2001).

Active learners commonly search for queries in the neighborhood of the decision boundary (Figure 6.1a), where class uncertainty is higher. However, the uncertainty region, as perceived given current evidence, might be unaligned with the real target concept. The (perceived) uncertainty region is defined (Cohn et al., 1994) as the area that is not determined by available information, that is, the set of instances in the working set such that there are two hypotheses that are consistent with all training instances yet disagree on the classification of those.

Limiting instance selection to the perceived uncertainty region seems adequate when the training set is a representative sample of the target concept in which case the perceived uncertainty region is probably consistent with the target concept. Class representativeness in the training set is assumed by the majority of active learning (AL) approaches. In such a scenario, selecting queries from the uncertainty region is effective in reducing version space.

But, what if the real uncertainty region is not correctly or fully perceived by the current hypothesis? Under such an assumption, favoring exploitation rather than exploration withholds the chances to achieve an early complete coverage of the target concept.



(a) Perceived uncertainty region (b) Real uncertainty region

Figure 6.1: Uncertainty region (shaded). n represents labeled instances from class c_n and x represents unlabeled instances. We assume that the concept to learn has three distinct classes, one of which has not yet been identified

6.1 The intuition

The initial stage of the learning process, when still searching for exemplary instances covering all target classes, is critical regarding the labeling cost. While still missing labeled instances of some target classes, the uncertainty region perceived by the active learner (Figure 6.1a) might be reduced to a portion of the real uncertainty region (Figure 6.1b) or might be severely biased. Being limited to this partial erratic view of the concept, the learner may be misled and is more likely to waste queries, thus increasing labeling cost at no benefit. The amount of the uncertainty region that the learner misses is related to the number of target classes that have not yet been identified.

Our intuition (Figure 6.2) is that query selection should be based not only on classifier confidence but also on distance to previously labeled instances. In the presence of two instances with equally low confidence – say, X_a and X_b in Figure 6.2 – we prefer to select the one that is farther apart from what we already know, i.e., from previously labeled instances – referring to Figure 6.2 we would prefer to query X_a than X_b.

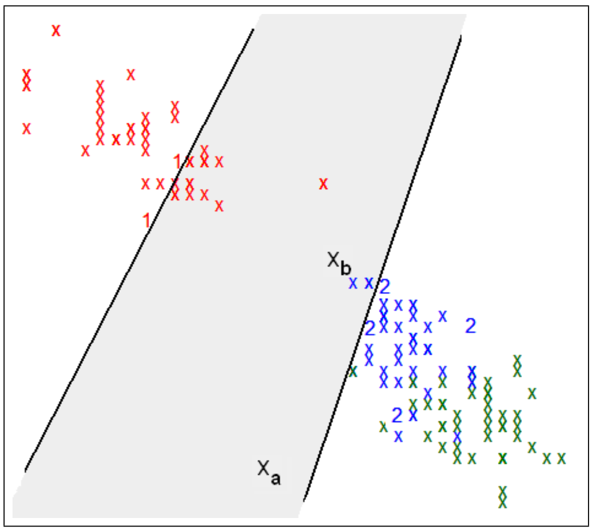


Figure 6.2: For equally confident instances prefer those that are far from previously explored regions in instance space

We expect that an AL approach that exhibits a high exploratory potential at the initial phase of the learning process – while still searching for exemplary instances for some of the target classes – and then smoothly shifts to a higher exploitation potential might reduce the amount of wasted queries, thus reducing the labeling cost. We search for a query selection criterion that favors exploration – tends to selects queries from unexplored regions in instance space – at an initial phase and then, as the input space is becoming explored, turns to favor exploitation.

6.2 The d-Confidence criterion

Many AL approaches rely on classifier confidence to select queries (Angluin, 1988) and assume that the pre-labeled set covers all the labels to learn. The performance of these approaches is focused on accuracy, favoring exploitation over exploration. Our scenario is somehow different: we do not assume that we have pre-labeled instances from all classes and, besides accuracy, we are mainly concerned with the fast identification of representative instances from all classes.

To achieve our goals we propose a new selection criterion, *d-Confidence* (Escudeiro and Jorge, 2012), which deals well with under-represented classes. Instead of relying exclusively on classifier confidence we propose to select queries based on the ratio between classifier confidence and the distance to known classes. D-Confidence, weighs the confidence of the classifier with the inverse of the distance between the instance at hand and previously known classes.

D-Confidence is expected to favor a faster coverage of instance space, exhibiting a tendency to explore unknown regions. As a consequence, it provides better exploratory behavior than confidence alone. This drift towards unexplored regions and unknown classes is achieved by selecting the instance with the lowest d-Confidence as the next query. Low d-Confidence combines low confidence – probably indicating instances from unknown classes – with high distance to known classes – pointing to unseen regions in instance space. This effect produces significant differences in the behavior of the learning process. Active learners focused on the uncertainty region, ask queries that are expected to narrow it down. The issue is that the portion of the uncertainty region that is perceived at a given moment is determined by the labels known at that moment. Focusing our search for queries exclusively in this region, while we are still looking for exemplary instances of some target classes that are not yet known, is not effective given our goals. Unknown classes hardly come by unless they are represented in the current uncertainty region.

Algorithm 6.1 presents d-Confidence, our AL proposal specially tailored to achieve a fast class representative coverage.

Algorithm 6.1 D-Confidence algorithm

```
 1: given $W$
 2: compute distance between instances in $W$
 3: $i = 1$
 4: initialize $L_1$
 5: while not stopping criteria do
 6:     $U_i = W - L_i$
 7:     $C_i =$ distinct class labels in $L_i$
 8:     learn $h_i$ from $L_i$
 9:     apply $h_i$ to $U_i$ generating $conf_i(x_j, c_k), \forall x_j \in U_i, C_k \in C_i$
10:     for $(x_j \in U_i)$ do
11:         for $(c_k \in C_i)$ do
12:             $d_j^k = ClassDist(x_j, c_k)$
13:             $dconf_i(x_j, c_k) = \frac{conf_i(x_j, c_k)}{d_j^k}$
14:         end for
15:         $dConf_i(x_j) = max_{c_k}(dconf_i(x_j, c_k))$
16:     end for
17:     $q_i = \underset{x_j}{\operatorname{argmin}}(dConf_i(x_j))$
18:     $L_{i+1} = L_i \cup < q_i, E(q_i) >$
19:     $i++$
20: end while
```

W is the working set, a representative sample of instances from the problem space. L_i is a subset of W. Members of L_i are the instances in W whose labels are known at iteration i. C_i is the set of the class labels that have representative instances in L_i. U, a subset of W, is the set of the unlabeled instances present in the working set. At iteration i, U_i is the (set) difference between W and L_i; h_i represents the classifier learned at iteration i; q_i is the query selected at iteration i; $conf_i(u_j, c_k)$ is the posterior confidence on class c_k given instance u_j, at iteration i.

The core of our proposal is the computation of the d-Confidence value of unlabeled instances. That is accomplished at the outer *for* cycle in Algorithm 6.1, at steps 10 to 16, as explained next.

6.2.1 Computing d-Confidence

D-Confidence is obtained as the ratio between confidence and distance between unlabeled instances and known classes (Equation 6.1). We may view d-Confidence as the confidence per unit distance.

$$dConf(x_j) = \max_k \left(\frac{conf(x_j, c_k)}{d_j^k} \right) \tag{6.1}$$

For a given unlabeled instance, $x_j \in U_i$, the classifier generates the posterior confidence w.r.t. known classes (step 9 in Algorithm 6.1). The distance between one unlabeled instance x_j and all labeled instances belonging to class $c_k \in C_i$, d_j^k, is computed by *ClassDist*() at step 12. At our current implementation this distance indicator, d_j^k, is the median of the distances between instance x_j and all labeled instances in L_i belonging to class c_k. We expect the median to soften the effect of outliers. The Euclidean metric was previously used, at step 2, to compute the distance between all pairs of instances in W. We compute $dconf_i(x_j, c_k)$, the marginal d-Confidence for each known class $c_k \in C_i$ given x_j, by dividing class confidence by the aggregated distance to that class (step 13).

The maximum d-Confidence on individual classes $c_k \in C_i$ for a given instance $x_j \in U_i$ is finally computed (step 15) as the d-Confidence of the instance at iteration i, $dConf_i(x_j)$.

If we now look at the iterative learning process as a whole, we see that at the initial phase there are few labeled instances – the instance space is barely explored – and the median of the distances to know classes is high for many unlabeled instances and low for others. This high variability will have a big influence in d-Confidence and will led it to select queries that lie far apart from known classes, thus exhibiting a high exploratory potential. When the instance space gets more and more explored, the median of the distances to known classes is expected to become more homogeneous among unlabeled instances and the confidence factor of d-Confidence exerts its influence rising the exploitation potential of d-Confidence.

D-Confidence is expected to dynamically shift between exploration and exploitation as the learning process iterates. This dynamic shifting is guided by the bond between the variance of the posteriors generated by h_i and the variance of the distance between members of U_i and L_i.

Being based on the distance between what is known and what has not been explored yet, d-Confidence is also a robust approach that applies independently of the specific geometric properties of the instance space. D-Confidence automatically adapts to the input space structure – having both U and L into consideration – without requiring any preliminary tuning effort. This characteristic of d-Confidence is expected to reduce any severe bias from the original data distribution that may occur in AL approaches that are exclusively based on confidence (Wang and Hua, 2011) and do not take into consideration the structural properties of the input space.

Also, this approach does not incur in the overhead cost that is imposed by the few AL approaches that are concerned with the exploration versus exploitation compromise. For instance, (Osugi et al., 2005; Cebron and Berthold, 2009) are two of these approaches, both requiring to tune two parameters guiding the transition between exploration and exploitation.

SVM classifiers – by default, we use SVM as the base classifier – can be unstable with a small training set (Dagli, 2005). This is probably due to the fact that SVM confidence is very high for any instances lying far from the decision margin. The decision hyperplane might change significantly from one iteration to the next when the training set is small and even more when we do not have an initial pre-labeled set covering all classes. As a consequence the set of instances where the classifier is highly confident may also change from iteration to iteration rather easily. D-Confidence merges two complementary aspects of the working dataset: distance, which is measured in the input features space, and confidence, which is computed in the base classifier features space. Adding the contribution of the distance measured in the input space is expected to improve the robustness of d-Confidence and contribute to improve the stability of the learning process when using SVM base classifiers.

6.2.2 Baseline criteria

D-Confidence aggregates two baseline AL criteria, confidence and distance (based on farthest-first). The confidence generated at each iteration by the current version of the base classifier, $conf_i(x_j, c_k)$, is the posterior probability of class c_k given x_j. The aggregated distance to known classes, d_j^k, is computed by $ClassDist(x_j, c_k)$ based on the individual distances between each pair of instances (Equation 6.2). Individual pair distances might be computed by any distance function – at the current implementation we are using the Euclidean distance. $ClassDist(x_j, c_k)$ is any aggregation function computed on the individual pair distances between one unlabeled instance $x_j \in U_i$ and every labeled instance from class $c_k \in C_i$ – at the current implementation we are using the median.

$$ClassDist_i(x_j, c_k) = d_j^k = median\left(dist\left(x_j, L_i^k\right)\right) \tag{6.2}$$

L_i^k is the set of labeled instances known at iteration i that belong to class c_k, that is, $L_i^k = \left\{< x_j, y_j > \in L_i : y_j = c_k\right\}$.

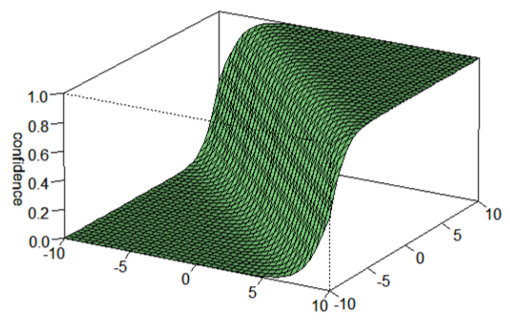
6.2.3 Effect of d-Confidence on SVM

The output of SVM classifiers is the signed distance to the decision boundary measured in terms of half margin width – an instance located on the decision boundary outputs 0 while an instance which is collinear with support vectors for class +1 generates an output 1 and an instance which is collinear with support vectors for class -1 generates an output -1. An instance with a distance to the decision boundary that is n times the distance between the boundary and a support vector outputs n. This distance, d, is transformed into $p \in [0,1]$, a measure of the posterior confidence of the learner on class $+1$.

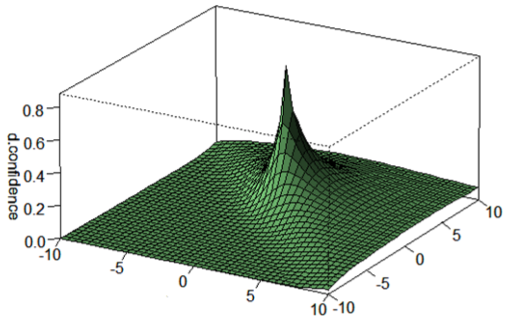
If, as is commonly the case, this transformation is based on logistic regression (Equation 6.3), the SVM classifier will be very confident on any instance located far from the decision boundary (Figure 6.3a), reducing the chances to select queries that are far from the current uncertainty region.

$$p = f(d) = \frac{1}{1+e^{-d}} \tag{6.3}$$

To prevent this behavior and to direct the learner to low confidence instances but also to unexplored regions in instance space, the d-Confidence value of an instance is high in the neighborhood of known instances – featuring high confidence and low distance to labeled instances – decreasing with the distance to those (Figure 6.3b).



(a) Confidence for class $+1$



(b) D-Confidence for class $+1$

Figure 6.3: Effect of d-Confidence for class $+1$ with an SVM classifier. We assume we have labeled instances near the point $(0,0)$ of the bi-dimensional input space. The decision boundary is the diagonal line from $(-10,10)$ to $(10,-10)$

6.3 Experimental setup

The evaluation of d-Confidence described in this chapter provides empirical evidence related to its ability as a query selection criterion. In particular, we investigate whether d-Confidence reduces the labeling effort needed to cover the set of target classes without compromising predictive ability. Our evaluation plan was designed to address the following issues:

(a) compare the performance of d-Confidence against its baseline and other state-of-the-art criteria regarding the ability to retrieve exemplary instances from all target classes at low cost;

(b) compare the performance of d-Confidence against its baseline and other state-of-the-art criteria regarding generalization error;

(c) assess the impact of the base classifier on the performance of d-Confidence;

(d) determine whether the performance of d-Confidence depends on the dimensionality of the input feature space. In particular, we want to determine whether d-Confidence is appropriate for high-dimensional unstructured datasets, mainly text corpora.

The evaluation of d-Confidence was performed over three base classifiers, 29 datasets and seven state-of-the-art query selection criteria, including d-Confidence and its baseline criteria – confidence and farthest-first. Each experimental trial is characterized by a dataset, a base classifier and a query selection criterion.

The following performance indicators were used:

- *error* and *known classes* (see Definition 1) that are evaluated at each iteration throughout the learning cycle and

- *first-hit* (see Definition 2) and *label disclosure complexity* (see Definition 3) that are evaluated once for each combination of dataset, base classifier and query selection criterion.

6.3.1 Performance indicators

To make the performance indicators referred above – error, known classes, first-hit and label disclosure complexity – clear, lets assume a generic classification task. C is the set of class labels to learn (refer to Section 5.4.1 in Chapter 5). $C_i \subseteq C$ is the set of class labels contained in the training set, L_i, available at iteration i.

AL is an iterative process requiring some prior initialization. C_1 is the set of labels that are represented in L_1, the initialization training set. L_1 contains two pre-labeled instances from W representing two distinct target classes, that is, C_1 contains two distinct classes from C. At each iteration a new labeled instance, called query, is added to the training set.

Error is a common assessment criterion for classification tasks. We have computed the progress of the generalization error – the error in the test set – over all iterations as new labeled instances are added to the training set.

Known classes is the number of classes that have representative labeled instances in the training set, L_i, at a given iteration, i.

Definition 1. *Known classes, kc_i is the cardinality of C_i, that is, the number of classes given for learning.*

First-hit is a performance indicator defined for each class, c_k, as the number of queries that are required to identify the first instance of the class for a given dataset, base classifier and query selection criterion. The two initial queries – the instances in L_1 – are not considered when computing first-hit. In each experimental trial first-hit is computed for all $c_k \in C$ such that $c_k \notin C_1$.

Definition 2. *For each $c_k \in C$, first-hit, fh_k, is the number of queries required to identify the first instance of class c_k.*

Label disclosure complexity aims to evaluate the ability of the learning process to reveal all the target classes. Label disclosure complexity (LDC) is inspired on label complexity (Hanneke, 2007). Label complexity is defined for the AL setting as the number of queries that are sufficient and necessary to learn the target concept. LDC is the minimum number of queries required to identify at least one instance from every class to learn. LDC equals the maximum first-hit computed over all the classes for a given combination of dataset, base classifier and query selection criterion.

Definition 3. *Label disclosure complexity, LDC, is the minimum number of queries that are required to identify at least one instance from every $c_k \in C$. LDC is equal to* $\max_k (fh_k)$.

6.3.2 Evaluation plan

The d-Confidence evaluation plan includes three phases that were performed in sequence. Besides these results we will also refer to an independent work from our colleagues Motta et al. (Motta et al., 2012) given its relevance for our investigation (Section 6.4.4).

Our first concern was the preliminary evaluation of the general competence of d-Confidence regarding an early coverage of the input space which is expected to contribute to the early discovery of all target classes. This competence was evaluated at the first phase, *Instance space coverage* (Section 6.4.1), over 16 artificial datasets simulating a broad range of two-dimensional input space topologies. This phase is a preliminary study aimed at realizing the potential of d-Confidence in addressing issues (a) and (b) (refer to Section 6.3).

In the second phase, *Class disclosure and accuracy* (Section 6.4.2), addressing issues (a), (b) and (c), we have evaluated on real datasets the impact of the base classifier on d-Confidence – issue (c) – and its performance regarding class disclosure and accuracy – issues (a) and (b). For this phase we have used five UCI datasets (Frank and Asuncion, 2010) and three distinct base classifiers.

The third evaluation phase, *Text* (Section 6.4.3), is focused on text corpora. At this phase we have used two text corpora to evaluate the performance of d-Confidence regarding class disclosure and accuracy on high-dimensional unstructured datasets. This phase addresses issues (a), (b) and (d).

Estimates were performed using 10-fold cross validation. Whenever possible, folds are stratified random samples comprising a partition of the working set. Stratified samples are not possible in extremely imbalanced datasets due to the big difference between the frequency of majority and minority classes. In such cases, we respect the original proportions whenever possible, i.e., when the classes are frequent enough. The remaining classes are represented by the minimum number of unlabeled instances, one or two, according to their relative frequency.

The labels in the training set are initially hidden from the classifier being revealed as the learning process iterates. In each iteration, the AL algorithm asks for the label of a single instance. In each cross validation fold, the AL process is initialized by revealing to the classifier two pre-labeled instances from two distinct classes. These are randomly selected from the training set. For a given dataset the initial pre-labeled instances in each fold, L_1, are invariant. The same L_1 is used to boot all classifiers at the same validation fold.

In all the experiments, in all assessment phases, we have compared d-Confidence against its baseline criteria: confidence – where query selection is solely based on low posterior confidence of the current classifier – and farthest-first – where query selection is based only on distance from training instances which is independent from the base classifier.

We have performed significance t-tests for the differences of the means observed when using farthest-first, confidence and d-Confidence. Statistically different means, assuming a significance level $\alpha = 5\%$, are presented in bold face.

In some cases, to avoid excessive computation time, we have used stratified random samples extracted from the whole dataset. There is no loss of generality arising from this fact since the learning process converges in respect to the indicators being measured, before those samples are exhausted. Sample dimension was previously estimated to assure acceptable computation costs as well as residual impact, if any, on the experimental results.

D-Confidence is tailored to use SVM as a base classifier. The main reason for this choice is the fact that we are concerned with text classification and SVM is commonly referred as being among the most accurate classifiers for high dimensional input spaces, in general, and text, in particular (Chakrabarti, 2003). Except for the second phase – class disclosure and accuracy – where we address the impact of other base classifiers, we always use SVM classifiers. We are using linear kernels for text corpora and Radial Basis Function (RBF) kernels for tabular data. Linear kernels do not require any parameter tuning while RBF kernels require to tune two parameters: C, the penalty cost for errors in the training set and γ, the width of the RBF kernel. We have used the SMV implementation provided by the e1071 package for R[1]. Parameter tuning in this SVM implementation – available through the $tune.svm()$ function – is unstable, depending on the cross validation folds used. Thus, to avoid uncontrollable effects we have decided to set C and γ to standard values for all datasets: $C = 10$ and $\gamma = 0.1$.

Datasets used in the first phase A preliminary analysis of instance space coverage was investigated during the first evaluation phase over bi-dimensional artificial datasets. These datasets were tailored to simulate certain geometric properties allowing to study the impact of global dataset meta-attributes on the performance of d-Confidence. We have complemented these with a (real) UCI dataset, Iris, with the aim of further illustrating the robustness of d-Confidence.

Artificial datasets were designed to simulate a set of geometric properties: cluster alignment, label distribution, cluster topology and cluster separability. All these properties are defined as binary.

Cluster alignment refers to non-collinear centroids (0) or collinear centroids (1). Collinear centroids means that the gravitational centers of data clusters lie in or close to a straight line in input features' space.

Label distribution may be balanced (0) or imbalanced (1). Balanced datasets have a uniform label distribution in the working set.

Cluster topology may be polymorphic (0) or isomorphic (1). Polymorphic datasets have clusters of instances belonging to the same class located in several distinct regions in input space.

[1]http://www.r-project.org, accessed on October 2012

Cluster separability may be separable (0) or overlapping (1). Separability refers to the input features space – on separable clusters it is possible to define linear decision boundaries between all pairs of clusters. It should be noted that linear separability of classes is not always considered in its strict sense. This would imply, for each pair of classes, the existence of some linear decision function being able to distinguish instances from those classes. However, in the polymorphic datasets, classes are constituted by several clusters which are apart from each other. In these cases when referring to separable classes we mean in fact linear separability in its strict sense but for each pair of clusters. At polymorphic datasets the strict sense of linear separability is verified between pairs of clusters, not classes.

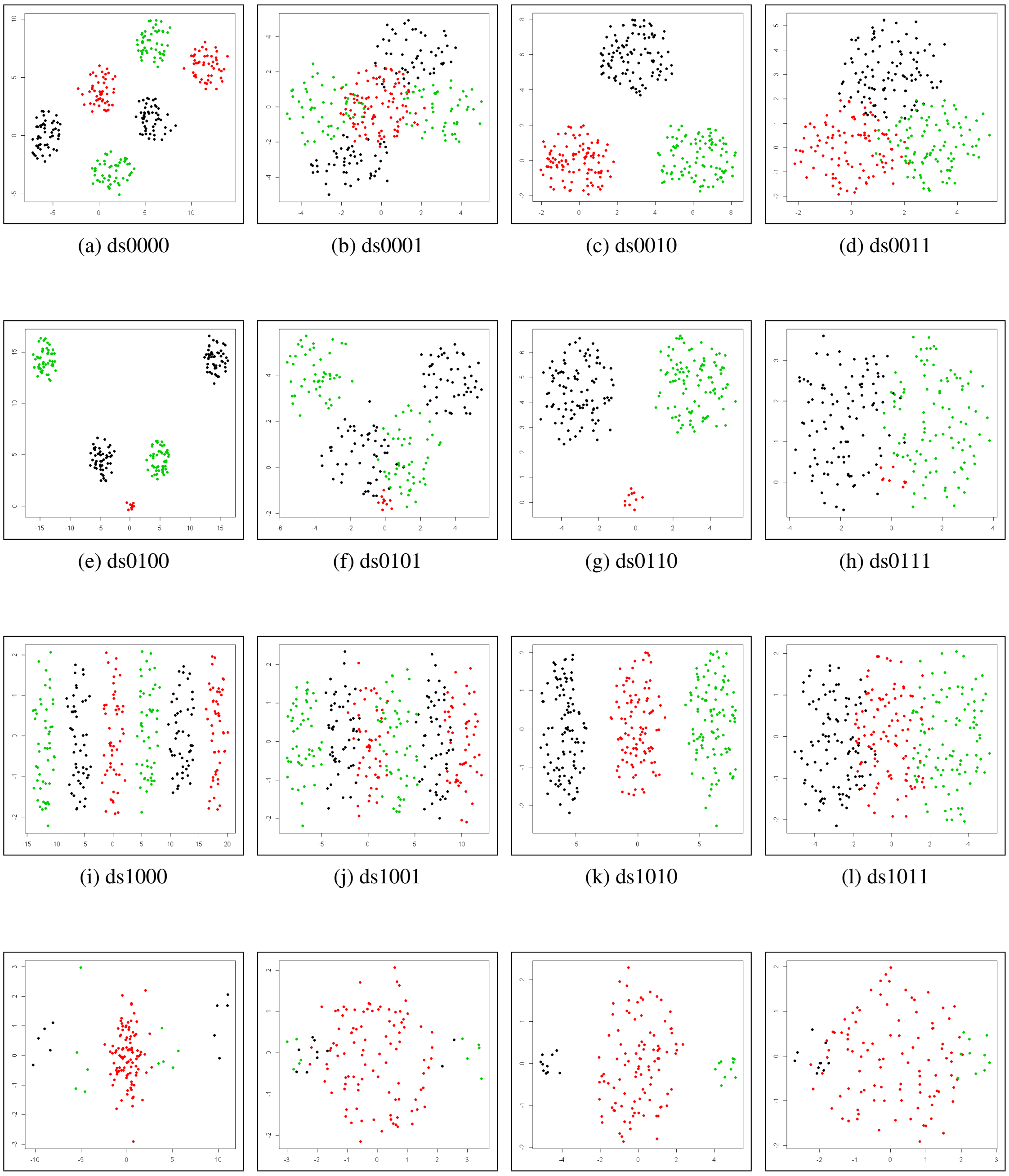
Sixteen artificial datasets covering the possible combinations of these four binary meta-descriptors were generated (Table 6.1). These datasets have been named with a four digit key where each digit refers to a given property as previously defined. For instance, the *ds0010* dataset has non-col-linear centroids, balanced label distribution, isomorphic structure and linearly separable classes.

We expect that a group of artificial datasets covering the possible combinations of these geometric properties will provide valuable information on the behavior of d-Confidence that can be extrapolated and further investigated in the following phases. These artificial datasets are simplistic – with only two attributes – but suitable for a simple graphical representation which is useful for a preliminary study.

Artificial instances are described by two numeric attributes plus the class attribute. Numeric attributes are random variables with uniform distribution centered at O with range $2R$. O and R are set according to the pattern we want to simulate in each dataset. Random noise with a normal distribution – with zero mean and $\sigma = 0.1R$ – is added to both attribute values. The class attribute has three distinct values, $C = \{1,2,3\}$. Regular classes have 100 instances and under-represented classes have 10 instances. A scatter plot of these datasets is presented in Figure 6.4.

Datasets used in the second phase The experiments in the second phase – class disclosure and accuracy – were performed over tabular data. We have used five datasets from the UCI repository (Frank and Asuncion, 2010):

- Iris (one class is separable while the other two are not),


(a) ds0000 (b) ds0001 (c) ds0010 (d) ds0011

(e) ds0100 (f) ds0101 (g) ds0110 (h) ds0111

(i) ds1000 (j) ds1001 (k) ds1010 (l) ds1011

(m) ds1100 (n) ds1101 (o) ds1110 (p) ds1111

Figure 6.4: Artificial datasets

Table 6.1: Artificial datasets and their properties

Dataset properties				Dataset
Alignment	Distribution	Topology	Separability	
non-collinear	balanced	polymorphic	separable	ds0000
			overlapping	ds0001
		isomorphic	separable	ds0010
			overlapping	ds0011
	imbalanced	polymorphic	separable	ds0100
			overlapping	ds0101
		isomorphic	separable	ds0110
			overlapping	ds0111
collinear	balanced	polymorphic	separable	ds1000
			overlapping	ds1001
		isomorphic	separable	ds1010
			overlapping	ds1011
	imbalanced	polymorphic	separable	ds1100
			overlapping	ds1101
		isomorphic	separable	ds1110
			overlapping	ds1111

- Cleveland heart disease (imbalanced class distribution),
- a random sample from Vowels (higher number of distinct classes than the others),
- a sample from Satlog (higher number of attributes than the others) and
- a sample from Poker (highly imbalanced class distribution).

These datasets were selected for their properties, mainly due to their distinct class distributions (Table 6.2).

Table 6.2: Class distribution in tabular datasets

Dataset	#instances	#features	1	2	3	4	5	6	7	8	9	10	11
Iris	150	4	50	50	50								
Cleveland	298	13	161	53	36	35	13						
Vowels	330	10	30	30	30	30	30	30	30	30	30	30	30
Satlog	500	36	125	118	96	67	48	46					
Poker	500	10	270	170	34	12	4	3	3	2	1	1	

The Poker dataset with a highly imbalanced class distribution causes some exceptions. The two classes with frequency 1 from the Poker dataset are never selected as initial classes. Two out of the 10 folds used for cross validation do not include all the 10 classes in the Poker dataset.

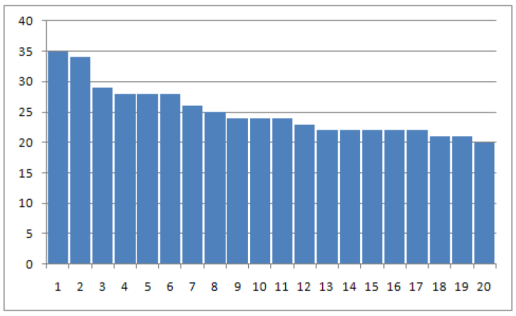
For this reason, the maximum number of classes found when using this dataset is below the total number of classes in the dataset, since this figure is estimated as a mean over all validation folds.

At this second evaluation phase we have investigated the performance of d-Confidence when using, besides SVM, neural networks (NNET) and decision trees (RPART) as base classifiers.

Datasets used in the third phase For the third phase we have selected two high-dimensional unstructured datasets. Two samples from traditional text corpora were used:

- a stratified sample from the 20 Newsgroups corpus (NG), containing 500 documents described by 10333 terms and
- a stratified sample from the R52 set of the Reuters-21578 collection (R52), containing 1000 documents described by 6019 terms.

The NG dataset has documents from 20 distinct classes while the R52 dataset has documents from 52 distinct classes. Text documents are modeled with TF$\times$IDF weighting. These datasets have been selected for their distinct class distributions. The class distribution in NG is fairly balanced (Figure 6.5a) with a maximum frequency of 35 and a minimum frequency of 20 while the R52 dataset presents an highly imbalanced class distribution (Figure 6.5b). The most frequent class in R52 has a frequency of 435 while the least frequent has only two instances in the dataset. This dataset has 42 classes, out of 52, with a frequency below 10 from which 31 have a frequency below five.


(a) NG corpus

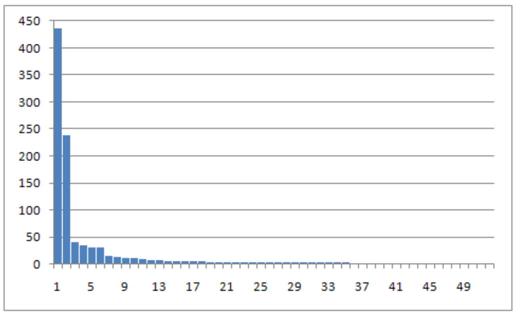

(b) R52 corpus

Figure 6.5: Class distributions in text corpora

We are relying on SVM as our base classifier by default. Although the performance of text classifiers depends heavily on the document collection at hand (Yang and Pedersen, 1997), some classifiers, particularly SVM and K-Nearest-Neighbors seem to outperform others in the majority of the domains (Joachims, 1998). A few properties of text documents – high dimensional feature spaces, many irrelevant features, document vectors' sparsity and the fact that most text categorization problems are linearly separable – justify the dominance of SVM in text categorization tasks (Joachims, 1998).

6.4 Evaluation

The evaluation of d-Confidence described in this chapter investigates its ability as a query selection criterion in comparison to its baseline and other state-of-the-art criteria. In particular, we investigate the ability of d-Confidence to reduce the labeling effort needed to cover all target classes without compromising accuracy. The results obtained in the three phases of our evaluation plan are discussed in Sections 6.4.1 to 6.4.3. The results from the work of our colleagues Motta et al. (Motta et al., 2012) are discussed in Section 6.4.4.

6.4.1 Instance space coverage

This phase aims to assess the ability of d-Confidence to achieve a fast coverage of input space and fast retrieval of representative instances from all target classes. Fast, in this sense, means with few queries which is equivalent to low cost. We also want to evaluate the accuracy of the classification models generated by d-Confidence. Are we trading accuracy for coverage? Another aim of these experiments is to investigate how the geometric structure of the dataset impacts the performance of d-Confidence.

In this phase we have used SVM with RBF kernels as the base classifier. We have recorded, at every iteration, the newly added query, the number of distinct labels known to the classifier and generalization error for all selection criteria under evaluation – farthest-first, confidence and d-Confidence. From these, we have computed, on each dataset, mean coverage, mean number of queries required to identify the hidden class – which in this case is equivalent to LDC since $\#C = 3$ and $\#C_1 = 2$ – and mean generalization error in each iteration over all cross validation folds (Table 6.3).

Table 6.3: Coverage (Cov), mean number of queries to identify one instance from the unknown class (LDC) and error (Err) with an SVM classifier on artificial data. Mean coverage and error are computed over all iterations in all cross validation folds for every artificial dataset. *ff* stands for farthest-first, *c* stands for confidence and *dc* stands for d-Confidence

Dataset	Cov (ff)	Cov (c)	Cov (dc)	LDC (ff)	LDC (c)	LDC (dc)	Err (ff)	Err (c)	Err (dc)
ds0000	0.745	0.967	**0.979**	46	20	**6**	0.209	0.038	**0.023**
ds0001	0.756	0.922	**0.937**	42	19	22	0.281	0.192	**0.174**
ds0010	0.716	**0.920**	0.908	71	19	**2**	0.104	0.032	**0.014**
ds0011	0.684	0.893	0.886	24	27	**3**	0.198	0.137	**0.104**
ds0100	0.838	0.897	**0.945**	180	9	10	0.185	**0.032**	0.046
ds0101	0.721	0.914	**0.933**	66	35	**13**	0.221	0.112	0.106
ds0110	0.725	0.877	**0.894**	147	28	**2**	0.149	0.052	**0.019**
ds0111	0.675	0.875	0.870	149	34	**11**	0.219	0.111	**0.086**
ds1000	0.767	0.908	**0.974**	89	29	**3**	0.352	**0.077**	0.088
ds1001	0.743	0.953	**0.976**	74	11	**7**	0.411	**0.240**	0.255
ds1010	0.771	0.893	**0.958**	180	24	**2**	0.238	0.039	**0.016**
ds1011	0.704	0.911	**0.933**	92	25	**6**	0.282	0.174	**0.144**
ds1100	0.769	**0.883**	0.819	104	55	**13**	0.222	**0.183**	0.198
ds1101	0.767	**0.852**	0.835	89	22	**11**	0.276	0.188	**0.178**
ds1110	0.766	0.862	**0.877**	18	29	**2**	0.153	0.052	**0.028**
ds1111	0.667	0.803	**0.827**	7	32	**3**	0.220	0.128	0.120
Iris	0.720	0.918	**0.949**	84	18	**3**	0.304	0.134	**0.082**

Instance space coverage is the percentage of instances in W that lie on a given neighborhood of any labeled instance. We assume that, at any iteration i, those instances yielding a distance to any labeled instance in L_i lower than $\frac{1}{10}$ of the maximum distance between instances in W are covered. The progress of instance space coverage is depicted in Figure 6.6 where we can see the percentage of covered instances after querying four, 16 and 64 instances.

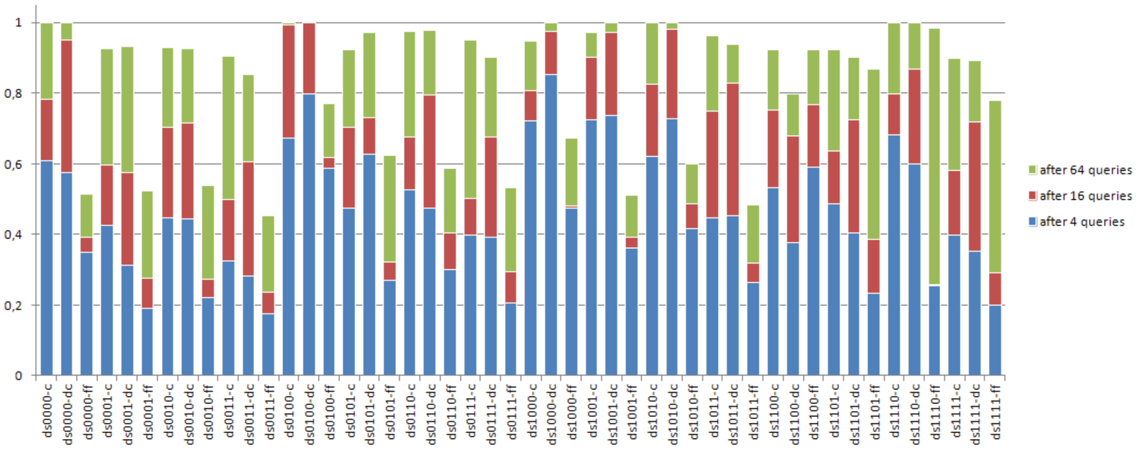


Figure 6.6: Progression of instance space coverage as new queries are added. *c* stands for confidence; *dc* stands for d-Confidence

On every dataset we have computed mean coverage and mean error over all iterations and over the 10 folds for farthest-first, confidence and for d-Confidence. This process generated three paired samples with the observed instance space coverage plus three paired samples with observed error. With these samples we have tested the significance of the differences of the means using paired t-tests.

The number of queries required to identify one instance from the unseen class is estimated as the average over the 10 folds for farthest-first, confidence and d-Confidence. These means have also been tested for equal means with paired t-tests. Statistically different means, at a significance level of 5%, are bold faced in Table 6.3.

D-Confidence consistently improves instance space coverage over both confidence and farthest-first. This behavior is observed irrespectively of dataset properties. There is a clear dominance of both d-Confidence and confidence when compared to farthest-first.

D-Confidence outperforms confidence on six out of eight collinear datasets – collinear datasets have the first numerical digit on their name set to *1*, *ds1???* (see Section 6.3.2). This same figure is observed on balanced datasets (*ds?0??*), on polymorphic (*ds??0?*) and also on separable (*ds???0*) datasets. On all the other groups of datasets – non-collinear (*ds0???*), imbalanced (*ds?1??*), isomorphic (*ds??1?*) and overlapping (*ds???1*) – d-Confidence outperforms confidence on five out of eight datasets.

The results on polymorphic datasets are particularly interesting since these contain classes having distinct clusters in different regions of instance space. Although the coverage efficiency of d-Confidence is not as clear as in isomorphic datasets, d-Confidence still outperforms both confidence and farthest-first.

From Figure 6.6 we observe that confidence generally achieves a better coverage than d-Confidence after the initial four queries – which happens in 10 out of 16 datasets. After these few initial queries this trend reverses and d-Confidence improves over confidence. After 16 queries d-Confidence outperforms confidence in 14 out of 16 datasets.

Label disclosure complexity When analyzing LDC – which, in this case is equivalent to the number of queries required to first hit an instance of the third class – we observe a clear dominance of d-Confidence against confidence and farthest-first. D-Confidence outperforms confidence and farthest-first in 14 out of 16 datasets. Confidence present a lower LDC at *ds0001* and *ds0100*. The overall mean LDC on these artificial datasets is 7 for d-Confidence, 26 for confidence and 86 for farthest-first – a clear advantage of d-Confidence. This is a core result addressing our purposes. The low LDC observed in these artificial datasets is a very promising indicator of the competence of d-Confidence to achieve low-cost disclosure of all target classes.

Error Somehow surprisingly, we observe that d-Confidence also improves on error. D-Confidence outperforms both confidence and farthest-first in 11 out of 16 datasets while confidence achieves the better performance in four out of 16. In other words, improved class coverage is not done at the cost of increasing error.

Cluster morphism seems to have impact on error. From all the isomorphic datasets, d-Confidence has a significant lower mean error than that of confidence and farthest-first on seven out of eight datasets. However, on polymorphic datasets, d-Confidence has similar results to those of confidence – d-Confidence outperforms confidence on three out of eight datasets, while the inverse occurs on four datasets.

Performance evaluation on Iris Such results on simulated data have been checked on a real dataset (Figure 6.7). We have applied this same experimental plan to the Iris dataset (Frank and Asuncion, 2010). The results we have achieved on Iris confirm the results on artificial data. Instance space coverage is more efficient when using d-Confidence and this is not achieved at the cost of increasing error which, in fact, also improves.

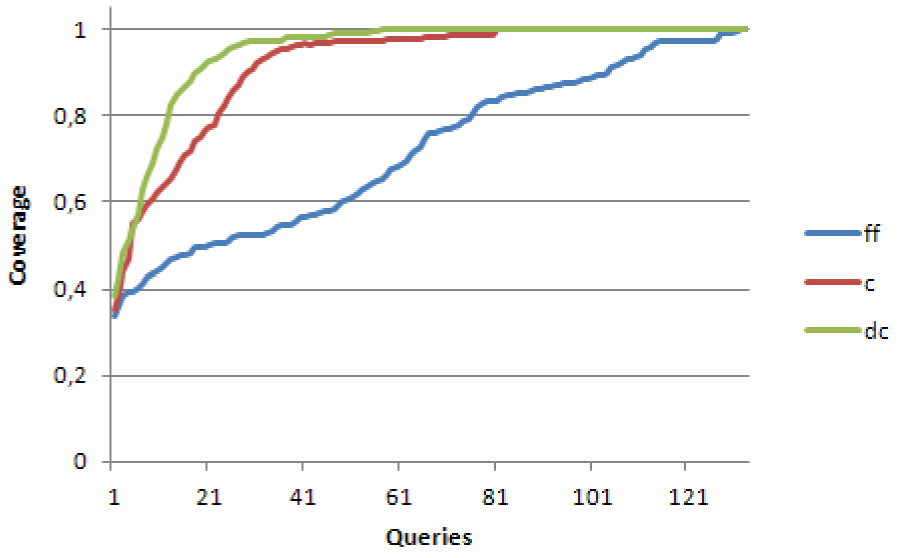


Figure 6.7: Instance space coverage on the Iris dataset as new queries are added

On the Iris dataset we have also recorded the number of queries required to get a full coverage of instance space. Instance space is assumed to be fully covered when all instances in the working set lie closer than a certain predefined distance from at least one labeled instance. This predefined distance has been set to $\frac{1}{10}$ – the initial setting – and then to $\frac{1}{8}$, $\frac{1}{6}$ and $\frac{1}{4}$ of the maximum distance between instances. It is expected that the number of queries required to achieve a full coverage decreases as the radius of the assumed covered neighborhood increases. This should be more evident when newly added queries belong to remote regions in instance space thus having reduced neighborhood intersections with previously covered instances. Our purpose is to evaluate whether d-Confidence is in fact exploring unseen regions in instance space more efficiently than confidence – its direct competitor.

We have observed that the number of queries required to get a 100% coverage of instance space with confidence decreases from 84 to 35 – a reduction of 58% in the labeling effort – when the neighborhood radius goes from $\frac{1}{10}$ to $\frac{1}{4}$. On this same scenario, d-Confidence labeling effort to get a full coverage is reduced from 51 to 8 queries – a reduction of 84%. These results confirm that d-Confidence selects queries from remote regions in instance space more efficiently than confidence.

The performance of d-Confidence on Iris supports the foreseen improvements over its baseline criteria. The Iris LDC for farthest-first is 84, for confidence it is 18 and for d-Confidence, three. The mean error is 30.4% for farthest-first, 13.4% when using confidence and 8.2% when using d-Confidence.

Impact of input space geometry The difference between farthest-first's performance and the other criteria is very significant. We have questioned whether farthest-first under-performance is related to some bias introduced by our artificial datasets and/or the indicator we are using to assess instance space coverage. Our hypothesis is that it is related to both the topology of the dataset – mainly with the relation between dense and sparse regions in instance space – and the indicator in use to measure coverage.

Being guided by distance only, farthest-first might be directed to distant regions that are sparse. This behavior does not contribute to instance space coverage the way we have defined it – number of instances lying in some neighborhood of all labeled instances in L_i.

Table 6.4: Artificial datasets sorted by decreasing order of farthest-first coverage

Dataset	Cov (ff)
ds0100	0.838
ds1010	0.771
ds1100	0.769
ds1000	0.767
ds1101	0.767
ds1110	0.766
ds0001	0.756
ds0000	0.745
ds1001	0.743
ds0110	0.725
ds0101	0.721
ds0010	0.716
ds1011	0.704
ds0011	0.684
ds0111	0.675
ds1111	0.667

Sorting the artificial datasets by decreasing order of farthest-first mean coverage (see Table 6.4) provides some evidence on this question.

We may observe that in the top eight datasets there are six separable datasets. Our separable datasets have dense clusters distant from each other (see Figure 6.4). This is an adequate topology for farthest-first that guides the learning process to select queries from distant regions that are simultaneously dense thus, contributing to improved coverage in our sense. In non-separable datasets there is no clear distance between clusters and the distribution of instances in input space is more homogeneous. When being directed to select queries in borderline regions, that are also less dense, farthest-first misses the chance to improve coverage as much as the other criteria.

Impact of cluster variance Analyzing cluster variance (Table 6.5) provides further evidence on this hypothesis. Inter-cluster and intra-cluster variance were computed for the number of clusters artificially generated in each dataset.

The correlation between the percentage of total variance that is explained by inter-cluster variance and instance space coverage for farthest-first (Table 6.6) is more than 10% higher than that of confidence and d-Confidence. There is also a high correlation between d-Confidence and confidence coverage rates.

Table 6.5: Percentage of inter-cluster to total variance

Dataset	Number of clusters	Inter-cluster/Total variance
ds0000	6	0.886
ds0001	5	0.785
ds0010	3	0.875
ds0011	3	0.670
ds0100	5	0.974
ds0101	5	0.874
ds0110	3	0.837
ds0111	3	0.664
ds1000	6	0.958
ds1001	6	0.947
ds1010	3	0.924
ds1011	3	0.765
ds1100	5	0.897
ds1101	5	0.827
ds1110	3	0.666
ds1111	3	0.651

Apparently confidence introduces a bias that leads the learning process to select queries from more dense areas in instance space – favoring exploitation. Farthest-first directs the learning process to query instances in unexplored regions irrespectively of how dense they are – favoring exploration. Despite the high correlation between the coverage rates of d-Confidence and confidence, d-Confidence outperforms confidence probably for its ability to take advantage of the merits of both its baseline criteria.

Main outcomes The results from these experiments provide evidence that d-Confidence outperforms the traditional confidence approach as well as farthest-first regarding instance space coverage, identification of unknown classes and error. D-Confidence improves instance space coverage and reduces the number of queries required to identify instances from unknown classes without

Table 6.6: Correlation between inter-cluster/total variance and coverage

Correlation	Inter/Total	Cover ff	Cover c
Cover ff	0.689	1	
Cover c	0.562	0.231	1
Cover dc	0.580	0.316	0.806

degrading accuracy – in fact, improving it on average – when compared to confidence and farthest-first. We are not trading coverage by accuracy in these artificial datasets.

Instance space is covered more efficiently when using d-Confidence, creating conditions to identify representative instances from unknown classes earlier. On average, d-Confidence requires almost four times less queries to identify instances from unknown classes than confidence and 12 times less than farthest-first.

Regarding the global properties of the datasets, d-Confidence is clearly better than confidence on well behaved datasets (balanced, collinear, isomorphic and separable). On not so well behaved datasets, d-Confidence is also better, but not as clearly, especially with respect to classification error. In general, d-Confidence also improves the results of its baseline criteria regarding the predictive ability.

6.4.2 Class disclosure and accuracy

In this experimental phase we have evaluated the performance of d-Confidence over tabular data w.r.t. LDC, accuracy and first-hit. This assessment was performed over a set of distinct base classifiers to evaluate their impact on the performance of the learning strategy.

We have recorded the number of distinct labels identified and the error on the test set for each iteration, for each combination of dataset, base classifier and query selection criteria. From these, we have then computed the mean number of known classes and mean generalization error in each iteration over all cross validation folds.

The evolution of the error rate and the number of known classes for each dataset, when using SVM as a base classifier, is shown in Figures 6.8a to 6.8e with curves for each selection criteria under evaluation[2].

For convenience of representation, the mean number of known classes was normalized to the total number of classes in the dataset thus being transformed into the percentage of known classes instead of the absolute number of known classes. This way the number of known classes and generalization error are both bounded in the same range – between 0 and 1 – and can be conveniently represent on the same chart.

[2]We will use the following notation to refer to results in tables and charts: *ff* stands for farthest-first, *c* stands for confidence and *dc* stands for d-Confidence. Generalization error will be referred by *e*, *kc* will refer to the mean number of known classes and *ldc* refers to LDC.

Means at each iteration (Table 6.7) are micro-averages – all the instances are equally weighted – over all cross validation folds for a given combination of dataset, classifier and selection criterion, providing a perspective of the average performance of the query strategy throughout the learning cycle.

The evolution of these indicators – generalization error and mean number of known classes – throughout all the learning cycle can be summed up to provide evidence on overall performance.

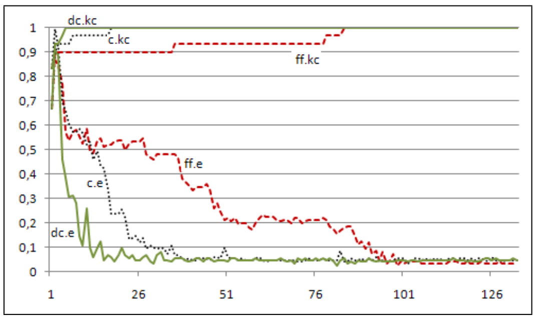
Table 6.7: Micro-averaged number of known classes and error. Means have been computed over all iterations from all cross validation folds for each combination of dataset, classifier and query selection criteria

Dataset	Classifier	ff.kc	c.kc	dc.kc	ff.e	c.e	dc.e
Iris	SVM	2.8	3.0	3.0	0.257	0.134	**0.082**
Iris	NNET	2.8	2.7	**3.0**	0.14	0.164	**0.05**
Iris	RPART	2.8	3.0	3.0	0.342	0.187	0.184
Cleveland	SVM	4.9	4.8	4.9	0.451	0.473	0.45
Cleveland	NNET	4.9	4.9	4.9	0.464	0.465	**0.447**
Cleveland	RPART	4.9	4.9	4.9	0.479	0.496	0.485
Poker	SVM	8.7	7.2	**8.8**	0.484	0.466	0.484
Poker	NNET	8.7	7.8	**8.8**	0.526	0.49	0.49
Poker	RPART	8.7	7.5	8.6	0.524	**0.495**	0.517
Satlog	SVM	5.6	5.8	**6.0**	0.349	0.186	**0.162**
Satlog	NNET	5.6	5.9	5.9	0.729	0.726	0.739
Satlog	RPART	5.6	5.9	6.0	0.430	**0.261**	0.28
Vowels	SVM	9.8	10.4	**10.5**	0.546	0.341	**0.322**
Vowels	NNET	9.8	10.7	10.6	0.661	0.601	0.623
Vowels	RPART	9.8	10.7	10.5	0.645	0.617	0.632

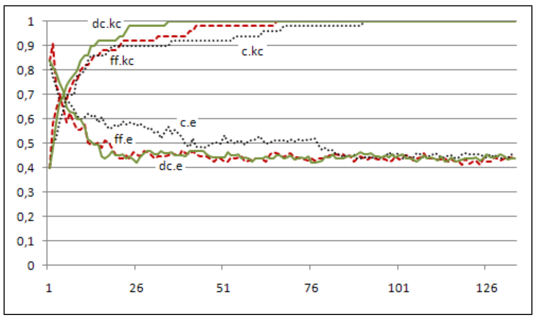
Besides the overall error and number of known classes we have also observed first-hit (Table 6.8). When computing first-hit for a given class we have omitted the experiments where the initialization labeled set, L_1, contains that class, following Definition 2.

From first-hit we compute LDC for each scenario (Table 6.9) – LDC is the maximum first-hit, that is, the first-hit of the last class being identified. LDC gives the minimum number of queries that are required by the AL strategy to fully cover the target concept, that is, to identify at least one instance from each target class.

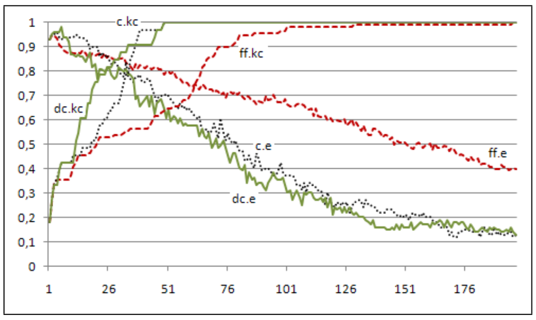
Support vector machine as base classifier If we focus on SVM, we can observe in Table 6.7 that, in general, d-Confidence outperforms confidence and farthest-first, both at labeling effort



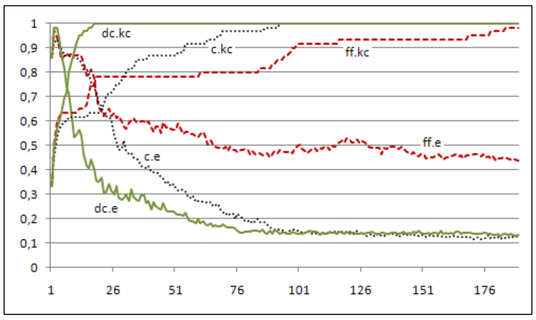
(a) Iris dataset



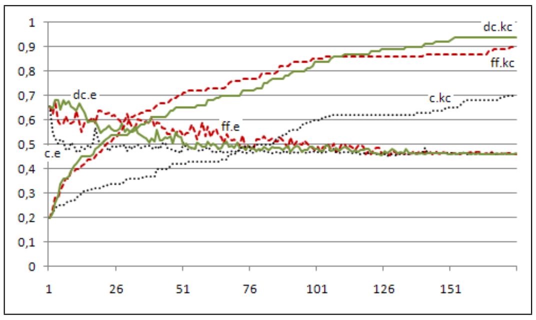
(b) Cleveland dataset



(c) Vowels dataset



(d) Satlog dataset



(e) Poker dataset

Figure 6.8: Known classes and generalization error in tabular data (when using SVM as the base classifier)

Table 6.8: Mean number of queries required to first hit unknown classes

Dataset	Classifier	AL	1	2	3	4	5	6	7	8	9	10	11
Iris	SVM	ff	1.0	65.3	1.0								
Iris	SVM	c	1.0	6.7	2.7								
Iris	SVM	dc	1.0	2.7	1.0								
Iris	NNET	ff	1.0	65.3	1.0								
Iris	NNET	c	37.5	1.0	83.0								
Iris	NNET	dc	1.0	1.3	1.0								
Iris	RPART	ff	1.0	65.3	1.0								
Iris	RPART	c	1.0	2.0	3.3								
Iris	RPART	dc	1.0	1.7	1.0								
Cleveland	SVM	ff	3.2	12.5	13.5	2.3	24.2						
Cleveland	SVM	c	2.5	7.0	8.3	19.0	39.8						
Cleveland	SVM	dc	2.7	14.5	8.3	4.8	8.0						
Cleveland	NNET	ff	3.2	12.5	13.5	2.3	24.2						
Cleveland	NNET	c	2.2	2.8	5.3	3.5	16.2						
Cleveland	NNET	dc	1.7	9.8	4.7	3.5	10.5						
Cleveland	RPART	ff	3.2	12.5	13.5	2.3	24.2						
Cleveland	RPART	c	3.0	1.0	17.7	4.3	16.2						
Cleveland	RPART	dc	2.2	13.2	3.5	4	5.3						
Poker	SVM	ff	4.5	2.0	2.9	17.2	27.6	85.1	39.1	63.5	200.4	63.7	
Poker	SVM	c	1.0	3.0	19.5	42.8	112.5	112.2	146.9	222.9	250.9	248.8	
Poker	SVM	dc	3.0	2.0	4.6	9.0	45.0	96.6	98.2	68.1	90.0	58.8	
Poker	NNET	ff	4.5	2.0	2.9	17.2	27.6	85.1	39.1	63.5	200.4	63.7	
Poker	NNET	c	2.5	1.0	12.5	41.2	74.7	145.3	177.5	67.0	70.6	311.6	
Poker	NNET	dc	2.0	2.0	7.0	26.1	49.2	38.7	74.0	63.6	114.2	95.1	
Poker	RPART	ff	4.5	2.0	2.9	17.2	27.6	85.1	39.1	63.5	200.4	63.7	
Poker	RPART	c	1.0	3.0	29.0	48.0	34.1	116.8	124.5	211.6	326.5	155.9	
Poker	RPART	dc	2.5	2.0	5.6	11.3	24.9	89.0	83.8	73.0	168.4	92.0	
Satlog	SVM	ff	68.0	107.0	20.9	1.6	1.1	95.8					
Satlog	SVM	c	11.5	5.2	34.1	31.6	28.1	23.1					
Satlog	SVM	dc	8.8	9.6	4.4	3.0	1.1	9.5					
Satlog	NNET	ff	68.0	107.0	20.9	1.6	1.1	95.8					
Satlog	NNET	c	4.8	6.6	7.9	2.5	24.4	6.2					
Satlog	NNET	dc	3.5	8.4	5.0	2.5	1.2	15.9					
Satlog	RPART	ff	68.0	107.0	20.9	1.6	1.1	95.8					
Satlog	RPART	c	5.8	1.0	2.3	10.9	16.0	7.8					
Satlog	RPART	dc	7.8	7.4	4.1	2.4	1.1	11.4					
Vowels	SVM	ff	1.1	13.0	22.6	52.2	60.5	71.4	66.4	8.2	62.1	3.9	88.6
Vowels	SVM	c	2.5	10.0	14.0	31.0	12.3	27.3	29.0	15.0	31.3	18.3	24.0
Vowels	SVM	dc	2.0	12.0	19.0	16.0	24.3	26.3	23.3	2.3	25.7	3.0	22.7
Vowels	NNET	ff	1.1	13.0	22.6	52.2	60.5	71.4	66.4	8.2	62.1	3.9	88.6
Vowels	NNET	c	27.3	13.0	4.5	7.1	13.8	7.5	9.5	14.8	5.6	11.8	5.9
Vowels	NNET	dc	3.6	8.4	15.9	7.8	21.6	15.5	9.5	7.5	11.9	4.8	24.3
Vowels	RPART	ff	1.1	13.0	22.6	52.2	60.5	71.4	66.4	8.2	62.1	3.9	88.6
Vowels	RPART	c	2.0	31.6	17.8	4.9	2.8	12.8	10.0	3.8	12.9	11.8	4.0
Vowels	RPART	dc	1.3	8.0	39.0	17.1	13.2	30.9	10.5	3.2	26.6	6.2	39.9

Table 6.9: LDC for tabular datasets

Dataset	Classifier	ff.ldc	c.ldc	dc.ldc	Best
Iris	SVM	65.3	6.7	2.7	dc
Iris	NNET	65.3	83.0	1.3	dc
Iris	RPART	65.3	3.3	1.7	dc
Cleveland	SVM	24.2	39.8	14.5	dc
Cleveland	NNET	24.2	16.2	10.5	dc
Cleveland	RPART	24.2	17.7	13.2	dc
Poker	SVM	200.4	250.9	98.2	dc
Poker	NNET	200.4	311.6	114.2	dc
Poker	RPART	200.4	326.5	168.4	dc
Satlog	SVM	107.0	34.1	9.6	dc
Satlog	NNET	107.0	24.4	15.9	dc
Satlog	RPART	107.0	16.0	11.4	dc
Vowels	SVM	88.6	31.3	26.3	dc
Vowels	NNET	88.6	27.3	24.3	dc
Vowels	RPART	88.6	31.6	39.9	c

and accuracy. The only exception occurs at the Poker dataset where error is lower when using confidence.

The dominance of d-Confidence throughout all the learning process is also observable from Figures 6.8. This dominance is clear, both in terms of error and known classes, at Iris, Vowels and Satlog (charts 6.8a, 6.8c and 6.8d). Iris and Vowels have uniform class distributions while Satlog has a fairly balanced class distribution with a coefficient of variation[3] equal to 42%. The same performance is also evident at the Cleveland dataset (Figure 6.8b). Here, however, while the gain of d-Confidence over confidence is clear it is not as salient over farthest-first. The Cleveland dataset has one majority class with a frequency over 50% and one under-represented class with frequency below 5%. The coefficient of variation of the class distribution in the Cleveland dataset is equal to 98%. At the highly imbalanced Poker dataset (Table 6.2) d-Confidence takes clear advantage over confidence w.r.t. known classes over all the learning process (chart 6.8e). We can also observe that d-Confidence is outperformed by farthest-first w.r.t. known classes at the initial quarter of the learning process – up to iteration 106 – but overcomes it from there on. At this dataset, however, the error of d-Confidence is clearly dominated by that of confidence at the initial stage of the learning process.

[3]The coefficient of variation is a measure of relative dispersion computed by the ratio of the standard deviation to the mean.

The differences in mean error gains are statistically significant at the Iris, Satlog and Vowels datasets in favor of d-Confidence. At the other datasets – Cleveland and Poker – the difference is not statistically significant. Besides confirming the ability of d-Confidence to find representatives of all classes early in the learning process, the most relevant evidence, when using SVM as the base classifier, is probably the fact that d-Confidence does not degrade error in general. In fact, the predictive ability of d-Confidence generally improves when compared to confidence and farthest-first.

Other base classifiers If we move now to the other base classifiers – NNET (neural networks) and RPART (decision trees) – we can observe a similar dominance. D-Confidence achieves higher or equal means of the number of known classes on all combinations except when using NNET and RPART over the Vowels dataset and RPART over Poker (Table 6.7).

When it comes to the mean error rate, d-Confidence does not perform as well as when relying on SVM as the base classifier. D-Confidence presents a lower mean error at the Iris dataset, when using NNET or RPART, and also at Cleveland and Poker when using NNET. At the other combinations, the differences in mean error observed when using d-Confidence in comparison with the other AL criteria are not statistically significant.

D-Confidence also outperforms confidence w.r.t. first-hit performance, in general. The same does not hold when comparing d-Confidence to farthest-first in which case there is no clear evidence on the best performer.

If we sum the number of classes over all datasets, we can find a total of 35 classes over the five tabular datasets (three from Iris, five from Cleveland, 10 from Poker, six from Satlog and 11 from Vowels). These datasets have been submitted to three distinct base classifiers – SVM, NNET and RPART. In total, for all the experiments, we have evaluated 105 classes. We can observe that confidence first-hits classes before d-Confidence only on 33 out of these 105 classes (Table 6.8). From these 33 cases, 8 happen when using SVM as a base classifier, 12 when using NNET and 13 when using RPART. It is worthwhile noting that 17 out of these 33 cases occur at the Vowels dataset. The Vowels dataset has a uniform class distribution. The added value of d-Confidence applied on these real datasets is more evident on imbalanced class distributions.

The Poker dataset – where two out of 10 classes occur only on a single case corresponding to a relative frequency of 0.2% and six other classes have a relative frequency below 1% – allows evaluating the early identification of under-represented classes. The average first-hit computed from Table 6.8 over under-represented classes – classes 5 to 10 – shows a weak performance of confidence in finding rare classes (Table 6.10).

Table 6.10: Average first-hit over under-represented classes at the Poker dataset

Classifier	ff	c	dc
SVM	80	182	76
NNET	80	141	72
RPART	80	162	89

D-Confidence outperforms both its baseline criteria w.r.t. the early identification of instances from under-represented classes when using SVM and NNET as base classifiers. Farthest-first however, improves over the other when using RPART.

LDC provides further evidence supporting the improved performance of d-Confidence over its baseline criteria. In fact, d-Confidence has the lowest LDC on all combinations of dataset and classifier that were evaluated on tabular data except on the Vowels dataset when using RPART as a base classifier (Table 6.9). The average gain on d-Confidence LDC for all pairs dataset/classifier when compared to confidence on tabular data is of 542%, meaning that confidence requires over six times more queries than d-Confidence to identify all target classes. This figure, however, is highly biased by the outlier observed on Iris/NNET. Nevertheless, if we remove this outlier from our data we still have a gain of 101% in LDC, meaning that, on average, confidence requires twice as many queries as d-Confidence to achieve a full coverage of the classes to learn on all tabular datasets.

Performance under different levels of class imbalance With the purpose of further investigating the ability of d-Confidence when in presence of imbalanced data, we have evaluated the AL strategies being studied under different levels of class imbalance. We have performed this evaluation on the datasets with uniform class distribution – Iris and Vowels – using SVM as the base classifier. The original training datasets were manipulated to assure imbalanced class distributions.

From each of those datasets we have extracted four samples with biased class distributions. At Iris, the number of instances from one of the classes – which will become the minority class – was reduced in those samples to 1, 3, 5 and 9, corresponding to a percentage of 2%, 6%, 11% and 19% relative to the frequency of each of the two remaining classes which kept their original frequency. At Vowels, the number of instances from four of its 11 classes – which will become the minority classes – was reduced in those samples to 1, 2, 3 and 6, corresponding to a percentage of 3%, 7%, 10% and 21% relative to the frequency of each of the remaining classes whose frequency was kept unchanged. Then we have repeated the same experiments as before but now on these biased training sets. The empirical results are presented in Table 6.11.

The LDC computed from these experiments (Table 6.11) confirms the ability of d-Confidence to retrieve rare instances in comparison to its baseline criteria.

Table 6.11: LDC under different imbalance levels. SVM as base classifier. Imbalance is the ratio of the frequency of the minority classes to the rest

Dataset	Imbalance	ff.ldc	c.ldc	dc.ldc	Best
Iris	19%	84	61	3	dc
Iris	11%	87	61	3	dc
Iris	6%	87	61	4	dc
Iris	2%	88	88	5	dc
Vowels	21%	99	26	23	dc
Vowels	10%	84	39	35	dc
Vowels	7%	98	69	55	dc
Vowels	3%	102	58	74	c

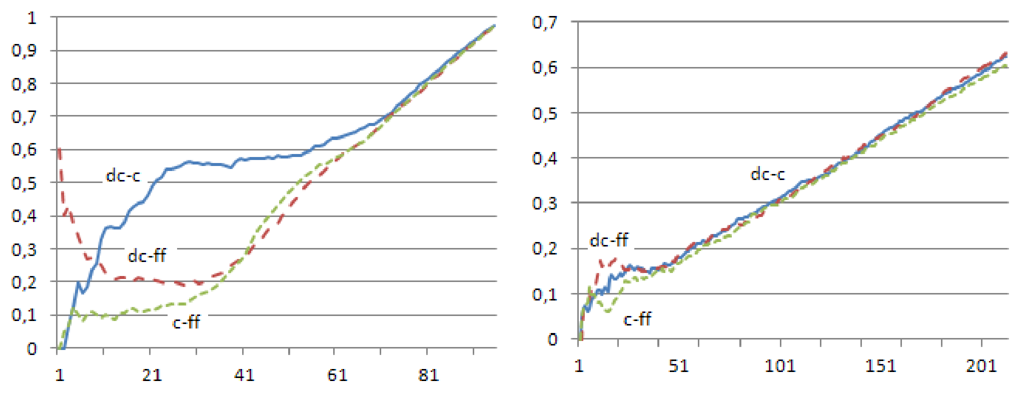
On average, d-Confidence presents lower LDC than its baseline criteria on all settings except at Vowels with 3% imbalance. We may observe a similar scenario, with a significant dominance by d-Confidence, when analyzing the number of known classes and error (Table 6.12). D-Confidence outperforms its baseline criteria with statistical significance at all settings except at the Vowels dataset with 21% imbalance.

Common queries selection Comparing the instances that are selected by each AL strategy adds relevant information to our discussion. Are all strategies selecting the same instances at the same stages of the learning cycle? We have investigated this question by measuring the percentage of common queries being selected by each criteria as the learning process iterates at Iris (Figure 6.9a) and Vowels (Figure 6.9b). Each curve in these charts represents the average, computed over all cross validation folds at each iteration, of the percentage of common instances observed in the labeled sets used to train the classifier under the referred strategies – d-Confidence (dc), confidence

Table 6.12: Micro-averaged number of known classes and error. Means have been computed over all iterations from all cross validation folds for each combination of dataset, imbalance level and query selection criteria. Bold faced values are statistically significant at 5%

Dataset	Imbalance	ff.kc	c.kc	dc.kc	ff.e	c.e	dc.e
Iris	2%	2.56	2.48	**2.96**	0.72	0.79	**0.67**
Iris	6%	2.60	2.58	**2.98**	0.63	0.59	**0.40**
Iris	11%	2.61	2.59	**2.98**	0.58	0.49	**0.26**
Iris	19%	2.64	2.62	**2.98**	0.52	0.41	**0.15**
Vowels	3%	8.11	8.87	**8.98**	0.94	0.92	**0.92**
Vowels	7%	8.40	9.36	**9.55**	0.92	0.92	**0.91**
Vowels	10%	8.65	9.77	**10.12**	0.91	0.89	**0.89**
Vowels	21%	8.83	10.27	10.28	0.81	0.77	**0.76**

(c) or farthest-first (ff). Instances in L_1 – which, given a dataset, are the same for all AL criteria – were not considered when computing these intersections. Only the instances that were in fact selected by each criteria from the first iteration on were accounted for.



(a) Iris (imbalance 19%) (b) Vowels (imbalance 21%)

Figure 6.9: Evolution of the percentage of common selected queries throughout the learning cycle. Each line represents the percentage of common instances for a given pair of strategies (dc-c, dc-ff, c-ff)

It is clear from Figure 6.9a that d-Confidence and confidence query many common instances during the initial stage of the learning process at the Iris dataset. In fact, after the first 29 queries, the labeled sets of both these strategies, L_{29}, have nearly 60% intersection. This level of overlapping then stabilizes to start increasing later as a consequence of the exhaustion of the unlabeled set which necessarily increases the interception between the labeled sets of all strategies.

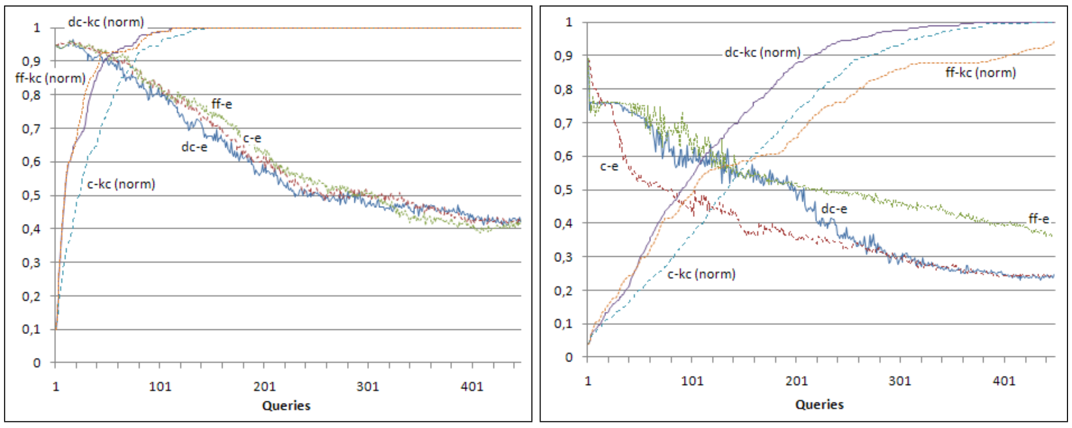
The opposite behavior is observed when comparing farthest-first with either confidence or d-Confidence. Despite the fact that d-Confidence and farthest-first share many common instances at the very first iterations (60%), this overlap drops fast getting close to 20% after 11 queries.

At the Vowels dataset, the overlap between the labeled sets being built by all AL strategies increases at a constant rate throughout the majority of the learning process. Only at the very beginning, during the initial 35 iterations, a distinct behavior is observed with d-Confidence and farthest-first querying more common instances than the other. As observed also at the Iris dataset, confidence and farthest-first are the strategies sharing fewer queries. This behavior is expected since d-Confidence is a combination of both confidence and farthest-first while these are independent from each other.

With the exception of the initial stage of the learning process for Iris w.r.t. dc-c, the percentage of common queries shared by d-Confidence and its baseline criteria is small. This is an indication that d-Confidence is promoting a new learning path.

6.4.3 Text

The evolution of the error rate and the number of known classes over text corpora is shown in Figures 6.10a and 6.10b with curves for each selection strategy under evaluation.



(a) NG corpus (b) R52 corpus

Figure 6.10: Known classes and generalization error

Similarly to what we have done at phase two, the evolution of error and mean number of known classes throughout all the learning cycle has been also summed up to summarize overall performance on text corpora (Table 6.13).

Table 6.13: Micro-averaged number of known classes and error. Means have been computed over all iterations from all cross validation folds for each combination of dataset, classifier and query selection criteria

Dataset	Classifier	ff.kc	c.kc	dc.kc	ff.e	c.e	dc.e
NG	SVM	19.2	18.6	19.1	0.631	0.629	**0.612**
R52	SVM	34.4	35.5	**39.7**	0.531	**0.383**	0.447

Besides the total number of queries required to retrieve labels from all classes and generalization error, we have also observed first-hit (Tables 6.14 and 6.15). When computing first-hit for a given class we have excluded the experiments where the initial labeled set, L_1, contains instances from that class.

Table 6.14: First-hit for the NG dataset

Class	Freq	ff-fh	c-fh	dc-fh
1	29	**29.8**	36.9	35.7
2	22	45.4	46.6	45.7
3	21	87.9	**63.7**	85.4
4	34	7.5	29.4	7.4
5	35	**22.2**	23.6	25.2
6	24	17.6	41.2	17.1
7	21	11.4	59.6	12.6
8	24	12.6	32.9	13.1
9	25	12.5	45.4	**11.4**
10	22	45.5	**41.1**	48.9
11	22	3.8	47.2	3.9
12	24	3.7	31.8	4.8
13	28	30.0	31.3	34.0
14	28	6.1	25.8	**5.4**
15	22	**5.4**	27.4	6.2
16	28	2.4	14.9	2.6
17	23	25.3	23.8	31.0
18	26	8.6	38.3	8.6
19	22	22.7	23.6	24.7
20	20	8.6	29.7	**7.7**
average		20.45	35.71	21.57

The learning process for the R52 dataset was halted after 600 iterations, before exploring the full unlabeled pool – the working set had 1000 instances, 900 of which were used for training in each fold. All the class labels to learn were identified after 600 iterations for all the selection

Table 6.15: First-hit for the R52 dataset

Class	Freq	ff-fh	c-fh	dc-fh
1	239	1.0	24.0	1.0
2	5	78.5	115.6	**64.7**
3	3	230.3	**118.6**	178.7
4	2	**98.7**	167.4	107.8
5	6	239.0	173.7	**110.6**
6	11	7.5	80.0	10.0
7	4	15.9	123.6	19.1
8	3	130.0	173.3	**102.9**
9	7	240.2	128.8	136.0
10	2	153.2	118.0	**99.5**
11	40	14.6	12.4	20.0
12	2	209.9	158.5	166.4
13	435	2.5	25.2	4.0
14	2	219.0	152.2	150.4
15	3	192.8	214.1	**123.9**
16	7	113.7	**91.9**	107.8
17	9	**33.1**	92.7	46.3
18	5	24.9	96.7	**16.8**
19	2	**93.1**	140.0	104.7
20	3	411.8	206.7	**184.9**
21	2	273.2	143.6	154.5
22	2	588.6	**188.9**	202.8
23	30	76.0	**28.9**	63.4
24	4	341.9	171.7	171.1
25	4	253.9	**196.2**	224.0
26	2	459.6	313.1	**256.4**
27	5	282.8	**130.0**	150.7
28	2	294.7	216.3	**144.5**
29	2	422.5	**175.5**	198.7
30	3	**68.5**	213.3	85.2
31	2	**111.7**	206.0	126.7
32	2	248.3	233.7	**167.0**
33	30	53.0	**39.7**	49.7
34	15	67.6	**44.6**	99.0
35	4	**187.8**	271.6	219.6
36	2	**58.2**	153.2	84.1
37	3	45.7	137.6	44.8
38	3	66.6	159.3	**52.1**
39	2	101.2	226.0	106.9
40	2	90.4	144.3	**75.5**
41	5	67.6	68.7	62.9
42	3	206.6	159.1	**144.8**
43	4	43.4	153.4	**36.7**
44	14	72.7	103.8	76.6
45	3	**86.5**	179.7	123.9
46	12	3.2	68.6	6.6
47	2	45.9	148.5	51.1
48	3	101.9	160.8	**76.1**
49	35	39.4	36.4	72.9
50	3	219.0	175.6	**108.7**
51	3	482.2	**146.1**	183.5
52	2	302.7	258.8	**196.5**
average		159.10	143.58	107.16

criteria, except for farthest-first. The mean number of known classes after 600 iterations equals 52 for confidence and d-Confidence, meaning these criteria have achieved full coverage of the class labels to learn in all the cross validation folds. For farthest-first the average number of known classes, 50.3, is below 52 which means that farthest-first was not able to identify all class labels in all cross validation folds after 600 iterations. Farthest-first missed, in several folds, six classes with frequency of two, two classes with frequency of three and one class with a frequency of four. In such cases we have assigned the most favorable first-hit value for the unidentified classes – a value from 601 on. For instance, in a given fold where farthest-first misses two classes their first-hit values are assumed to be 601 and 602 – the very first queries after halting the learning process at 600 iterations. First-hit means were computed on this assumption – the most favorable assumption for farthest-first.

Finding under-represented classes There is no clear dominance, neither from d-Confidence nor from farthest-first, when finding unknown classes in the NG dataset (Figure 6.10a). However, both these criteria outperform confidence at this dataset. The difference between mean first-hit of d-Confidence and farthest-first in Table 6.14 – 20.45 for farthest-first and 21.57 for d-Confidence – is not statistically significant ($\alpha = 5\%$).

In R52, farthest-first starts by identifying unknown classes a little faster than d-Confidence (Figure 6.10a). However, after the initial learning stage, d-Confidence outperforms and dominates farthest-first. This behavior had already been observed, with tabular datasets, in the previous experimental phase. When identifying unknown classes, farthest-first leads, up to the 45th query, on average, taking a maximum advantage of two classes after 37 queries. After 45 queries, with 13.2 classes identified on average, d-Confidence clearly dominates farthest-first.

It is interesting to notice that farthest-first beats d-Confidence on the majority classes (Table 6.15) but, once all majority classes have been found and only minority classes are left unexposed, d-Confidence reveals its ability to find rare instances. The mean frequency of the classes that are first found in R52 by d-Confidence is 3.2, while it is 12.5 for confidence and 33.8 for farthest-first.

This aspect might be further investigated from a coarser point of view by analyzing how fast each AL criterion retrieves exemplary instances from a batch of classes instead of analyzing single classes one at a time. To analyze this particular aspect we provide a benchmark based on random query selection – averaged over 10 random samples.

We have recorded the number of queries required to identify bunches of distinct classes in multiples of 10 for R52 and multiples of 4 in NG. These bunches are constituted by classes sorted by increasing order of their first-hit. Figures 6.11 and 6.12 give an overview of the number of queries that are required in each setting to first hit a given number of distinct classes.

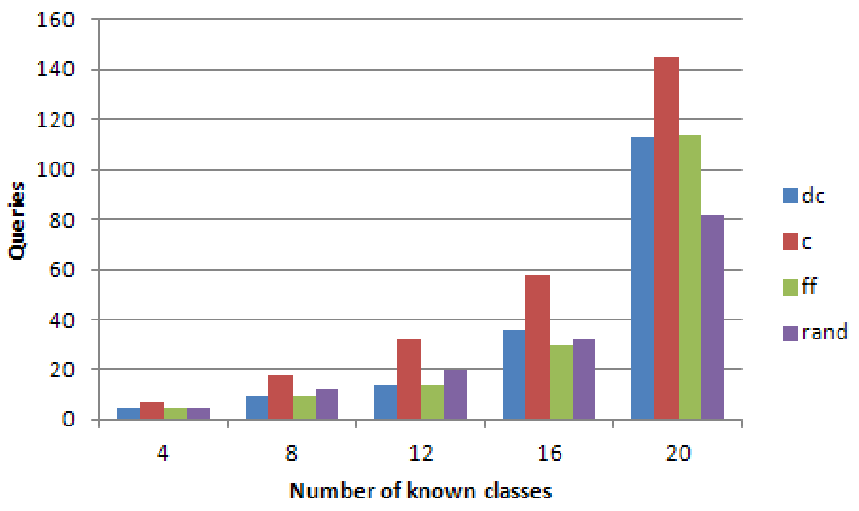


Figure 6.11: Queries required to identify bunches of distinct classes in NG dataset

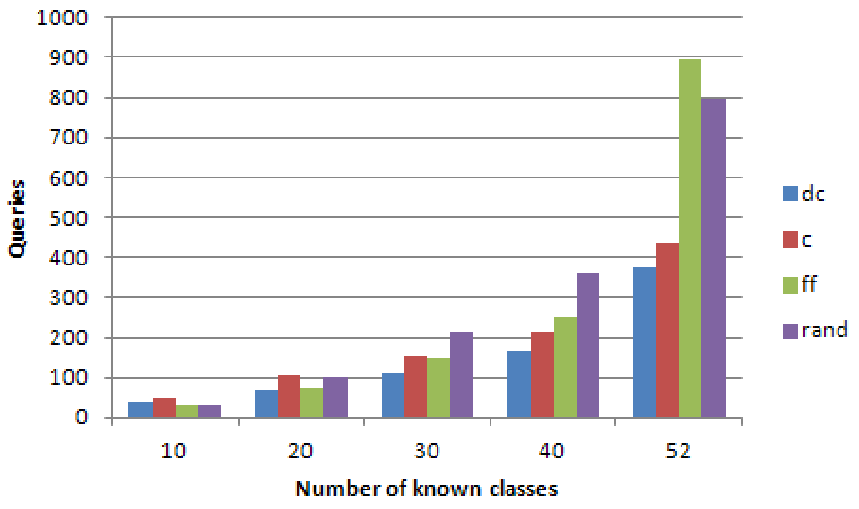
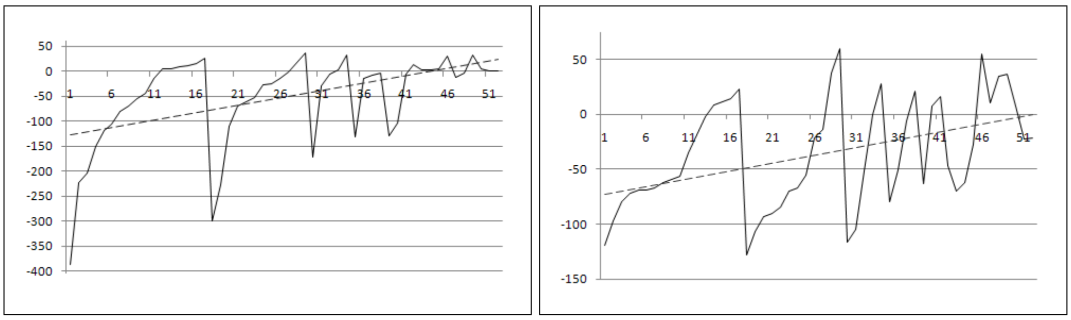


Figure 6.12: Queries required to identify bunches of distinct classes in R52 dataset

In the case of the R52 dataset, d-Confidence always finds new classes faster, that is with fewer queries, than confidence. The first bunch of 10 distinct classes – the first classes being identified are generally majority classes – is found as fast with random sampling and farthest-first as with d-Confidence but, from there on, when rare classes come by, d-Confidence takes the lead.

The outcome is quite different in the NG dataset. In this dataset d-Confidence still outperforms confidence but it is beaten by random selection of instances after identifying 13.3 classes on average – after 22 queries on average. The ability to retrieve exemplary instances from unknown classes of d-Confidence is comparable to that of farthest-first on NG. When in presence of balanced datasets, as NG, d-Confidence identifies new classes faster than random selection at the initial phase of the learning process but selecting instances by chance is better to identify instances in the latest stage of the learning process when few classes remain undetected.

Figures 6.13 provide additional evidence on the ability of d-Confidence to find rare instances.



(a) D-Confidence vs farthest-first

(b) D-Confidence vs confidence

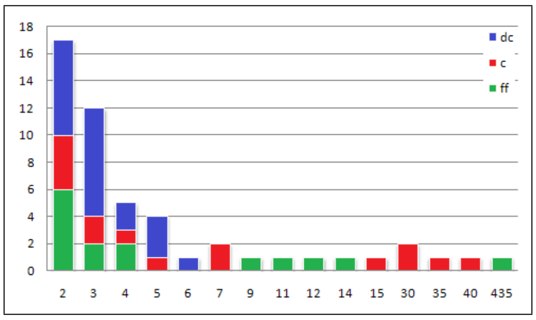
Figure 6.13: Average gain of d-Confidence over its baseline criteria to first hit classes on R52. Classes are sorted by increasing frequency

These charts represent the difference in d-Confidence first-hit compared to their baseline criteria. Negative differences mean that d-Confidence performed better, i.e., found representative instances of the class with fewer queries than its baseline criteria. In these charts, classes are sorted by increasing frequency, in first place, and then by decreasing gain. This sorting scheme is responsible for the patterns that are observed in both charts of Figure 6.13. For instance, all the minority classes that are represented in the horizontal axis of Figures 6.13a and 6.13b by the coordinates 1 to 17 have a frequency of two. For this group, the gain is sorted by increasing order thus generating the pattern that we may observe in both charts for each group of classes with the same frequency.

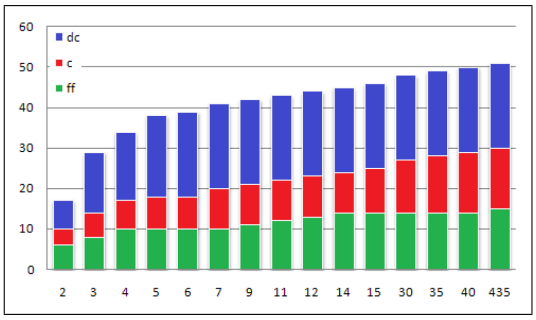
The dashed trend lines, represented in both charts (Figures 6.13), with a positive slope clearly show that the gain in d-Confidence first-hit, when compared to its baseline criteria, decreases when the class frequency increases. D-Confidence assures a significant reduction in the mean number of queries that are required to first hit classes in R52. This reduction is more important in minority classes, i.e., in the first classes appearing in the horizontal axis.

Another perspective of these results may clarify our point of view. Figure 6.14a presents the number of classes that were first found by each criteria for each different class frequency in the horizontal axis. Figure 6.14b represents the accumulated number of first found classes.

As detailed below, both these charts show evidence on the improved ability of d-Confidence to find exemplary instances of under-represented classes.



(a) First found classes



(b) Accumulated first found

Figure 6.14: Number of classes of a given frequency first found by each criteria on R52

When comparing d-Confidence against farthest-first we can observe that from the 17 classes in R52 that have a frequency of two, d-Confidence finds 11 before farthest-first. From the 12 classes with a frequency of three, d-Confidence finds 10 before farthest-first. From the 13 classes with frequency between four and nine, d-Confidence finds 10 with fewer queries than farthest-first. From the remaining 10 classes, with a frequency between 11 and 435, d-Confidence finds only two before farthest-first.

A similar comparison against confidence shows similar results. From the 17 classes in R52 that have a frequency of two, d-Confidence finds 13 before confidence. From the 12 classes with a frequency of three, d-Confidence finds 10 before confidence. From the 13 classes with frequency between four and nine, d-Confidence finds 10 with fewer queries than confidence. From the remaining 10 classes, with a frequency between 11 and 435, d-Confidence finds five before confidence.

Error D-Confidence accuracy dominates that of farthest-first (Figure 6.10b). The mean accuracy of d-Confidence over all iterations is 2% better that of farthest-first. This result is significant at 5% significance. The NG dataset has a fairly balanced class distribution. On the R52 dataset, which has an highly imbalanced class distribution, we can observe very distinct performance (Figure 6.10b).

At the R52 dataset the difference of mean error is significant in favor of confidence. D-Confidence reduces the labeling effort that is required to identify instances in R52, exhibiting better representativeness capabilities in this corpus. However, the error rate increases. Apparently, d-Confidence gets to know more classes from the target concept earlier although less sharply. In the R52 dataset d-Confidence is exchanging accuracy for representativeness.

Effect of data dimensionality on the distance factor of d-Confidence If we take a step back to analyze d-Confidence first-hit against farthest-first on the highly imbalanced Poker dataset we may find some unexpected outcome. In this case, farthest-first generally outperforms d-Confidence in finding rare instances, contrary to what happens in text corpora. This is probably a sign that distance might be a better discriminator in low-dimensional input spaces than it is in high-dimensional input spaces.

Distance functions might lose their usefulness in high-dimensional spaces where the distance to the nearest and farthest neighbors come very similar – the curse-of-dimensionality (Bellman, 1957). This effect is most noticeable when using $L_k - norm$ distances with a high value of k ($k \geq 3$). Euclidean distance, a $L_2 - norm$ metric, is not much affected (Aggarwal et al., 2001). To assess this effect on our datasets we have computed the *relative contrast* – measuring the relative distance of the nearest and farthest neighbors of a given query – for all instances in each dataset (Equation 6.4). In Table 6.16[4] we can observe that the discrimination between the nearest and farthest neighbors is not too sensitive to the data dimensionality. Despite the fact that the minimum relative contrast exhibits a negative correlation of 64% to the data dimensionality, the maximum relative contrast is not correlated and there is no evidence that high-dimensional data is

[4]Notation: *min.rc* and *max.rc* stand for the minimum and maximum relative contrast observed in each dataset; *global.rc* is a global contrast measure for each dataset computed by Equation 6.4 but using the maximum and minimum distances between all the instances in the dataset.

affecting the distance metric in use. The lack of correlation between the global contrast measure and data dimensionality supports this conclusion.

$$\frac{D_{Max} - D_{min}}{D_{min}} \tag{6.4}$$

Table 6.16: Relative contrast using Euclidean distance

Dataset	Dim.	Instances	min.rc	max.rc	global.rc
Iris	4	150	4.477	63.985	69.852
Vowels	10	330	2.424	51.371	65.253
Poker	10	500	2.117	7.426	9.630
Cleveland	13	298	1.085	59.944	68.921
Satlog	36	500	1.682	23.760	26.258
R52	6019	1000	0.315	52.292	67.812
NG	10333	500	0.428	23.034	32.197

6.4.4 Independent assessment of other active learning strategies

The work performed by our colleagues Motta et al. (Motta et al., 2012) – using network models and complex network characterization measures to select queries – includes an independent evaluation of several state-of-the-art query selection strategies for AL including d-Confidence. Given the relevance of this work to our own investigation we discuss their results related to error and known classes.

In their work, the authors evaluate, among others, the evolution of error and number of known classes as the learning process iterates. Their evaluation was performed over several UCI datasets. Error and known classes are estimated with 10-fold cross validation. SVM base classifiers where used with RBF kernels. The required parameters were tuned by the $tune.svm()$ function of the $e1071$ R package.

Motta et al. (Motta et al., 2012) have compared the performance w.r.t. error and known classes of four query selection criteria besides d-Confidence, including Simple, Kernel Farthest First, Hierarchical Clustering and BalancedEE.

Simple Tong et al. (Tong and Koller, 2002), working with SVM classifiers applied to text, propose *Simple*, an AL criterion that selects the query lying in the SVM margin closest to the dividing hyperplane. This will be the query that maximizes the reduction in version space – the set of hypotheses that are consistent with all labeled instances – thus contributing to faster convergence. This choice is equivalent to selecting the most uncertain instance, that is, the one the current classifier is less confident about.

Kernel Farthest First The *Kernel Farthest First* (KFF) (Baram et al., 2004) algorithm selects the unlabeled instance further away from all labeled instances in the feature space induced by the kernel function used by the classifier. A similar criterion to our version of farthest-first except that we compute distance in input feature space.

Hierarchical Sampling *Hierarchical Sampling* (HS) (Dasgupta and Hsu, 2008) is a cluster-based method that consistently improves label complexity – the number of queries that is sufficient to learn a concept – over supervised learning by detecting and exploiting clusters that are loosely aligned with class labels. It starts with a hierarchical clustering of the unlabeled pool. Then, random samples from each node of a partition of the data, given by a pruning of the tree, are queried and their labels are used to compute the purity of each node in the partition. Nodes with low levels of purity are replaced by their child nodes. This process can be halted, for instance, when a given purity level is achieved at each node. At each iteration, the sampling strategy favors less pure nodes.

BalancedEE *BalancedEE* (Osugi et al., 2005) is an AL strategy that decides at each iteration whether to explore or exploit. This decision is based on a binary random process assigning a probability p to explore (and $1-p$ to exploit). If the decision is to explore, the KFF (Baram et al., 2004) algorithm is applied to select the next query. Otherwise, the next query is selected with Simple (Tong and Koller, 2002). To determine how successful an exploration step was, the authors compute $d(h_{i-1}, h_i) \in [-1, +1]$, a measure of the change induced from one iteration to the next (Equation 3.7). If $d(h_{i-1}, h_i)$ is positive, implying that the learned hypothesis significantly changed at the current iteration, the probability p is kept high encouraging further exploration. If $d(h_{i-1}, h_i)$ is negative, p is reduced.

Assuring comparability of results In order to convey to the initial conditions that we have assumed in our experimental analysis, we focus our attention on the datasets used by Motta et al. (Motta et al., 2012) that have more than two target classes. These include 10 numerical datasets in total (Table 6.17). Eight of them come from the UCI repository (Frank and Asuncion, 2010). The other two are image datasets referred by *Img-corel* (Li and Wang, 2003) – photographs on ten different themes represented by 150 descriptors – and *Img-med* – images obtained by magnetic resonance imaging, each represented by 28 features.

Table 6.17: Datasets used by Motta et al. to evaluate active learning criteria

Dataset	Attributes	#Classes	#W	Class distribution
Balance	4	3	625	fair
Cleveland	13	5	298	imbalanced
Ecoli	7	8	336	imbalanced
Img-corel	150	10	1000	balanced
Img-med	28	12	540	imbalanced
Iris	4	3	150	balanced
Satimg	36	6	500	fair
Vehicle	18	4	846	balanced
Vowels	10	11	990	balanced
Wine	13	3	178	balanced

The learning process deployed by Motta et al. (Motta et al., 2012) starts with no initialization training set. The first training instance is selected at random from the working set and the learning process iterates from there. In our experimental setup, however, we have assumed that the learning process is to be initialized by a training set containing two labeled instances from two distinct classes. To assure compliance with this setup, our estimates of the number of known classes (Table 6.18) and error (Table 6.19) ignore the first two iterations/queries considered by Motta et al. being computed from the third iteration on. In Tables 6.18 and 6.19 statistically significant differences, at a significance level of 5%, are boldfaced.

Discussion These results show a clear dominance of d-Confidence regarding the early identification of target classes when compared to state-of-the-art AL strategies. D-Confidence identifies all target classes before the other criteria under evaluation in all datasets except Iris and Wine. On these two, however, there is no statical significance supporting the null hypothesis of different means of the number of known classes throughout the learning process.

Table 6.18: Mean number of know classes throughout the learning process (Motta et al.)

Dataset	simple	kff	dConf	hs	balee
Balance	2.924	2.933	**2.990**	2.921	2.942
Cleveland	4.877	4.819	**4.959**	4.712	4.837
Ecoli	6.398	7.285	**7.830**	6.178	7.062
Img-corel	8.520	9.479	**9.695**	9.292	9.433
Img-med	7.618	10.871	**11.531**	10.936	10.381
Iris	2.957	2.993	2.995	2.945	2.992
Satimg	3.370	5.840	**5.917**	5.706	5.180
Vehicle	3.937	3.944	**3.990**	3.901	3.951
Vowels	6.990	9.882	**10.632**	10.020	9.205
Wine	2.715	2.993	2.993	2.957	2.993

Table 6.19: Mean error throughout the learning process (Motta et al.)

Dataset	simple	kff	dConf	hs	balee
Balance	0.115	0.160	**0.105**	0.175	0.133
Cleveland	0.502	0.452	0.575	0.463	**0.435**
Ecoli	0.306	**0.207**	0.240	0.324	0.222
Img-corel	0.618	**0.342**	0.411	0.409	0.351
Img-med	0.658	0.352	**0.345**	0.387	0.373
Iris	0.062	0.060	**0.042**	0.08	0.054
Satimg	0.697	0.281	0.346	**0.262**	0.446
Vehicle	0.419	**0.370**	0.388	0.397	0.380
Vowels	0.762	0.51	0.511	0.543	0.573
Wine	0.246	0.044	**0.034**	0.050	0.040

D-Confidence also exhibits statistically significant improvements on accuracy throughout the learning process on four out of 10 datasets – Balance, Img-med, Iris and Wine. KFF reports the lowest mean error on three datasets – Ecoli, Img-corel and Vehicle. *hs* outperforms the other criteria on Satimg while $balee$ is the best performer regarding error on Cleveland. The improvements of d-Confidence regarding error are not as clear as those regarding the early identification of target classes. Nevertheless, d-Confidence outperforms the other criteria on more datasets than the rest also regarding error.

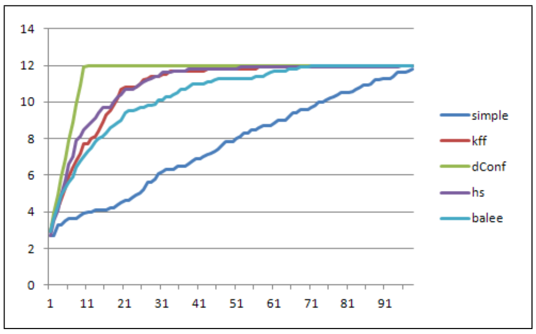
The evolution of the number of known classes and error during the learning cycle on typical cases – Img-med, Vowels and Ecoli – is represented in Figure 6.15. At the Img-med dataset d-Confidence outperforms the other AL criteria at both known classes (Figure 6.15a) and error (Figure 6.15b). At Vowels, d-Confidence outperforms the other AL criteria at known classes (Figure 6.15c) but there is no single top performer on error (Figure 6.15d). At Ecoli, d-Confidence is the top performer regarding known classes (Figure 6.15e) and KFF is the top performer regarding error (Figure 6.15f).

At both Img-med and Vowels, d-Confidence finds exemplary instances of all target classes during the very first iterations. Concerning the identification of known classes, d-Confidence performs very close to the optimal. D-Confidence LDC for Img-med – a dataset with 12 classes – is 13 and for Vowels – with 11 classes – is 12. This performance has no negative impact on error.

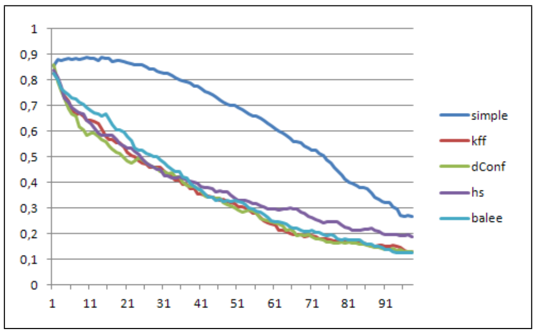
Main outcomes The results obtained by our colleagues (Motta et al., 2012) bring additional evidence on the improvements of d-Confidence over state-of-the-art AL strategies.

Their results show that d-Confidence outperforms Hierarchical Sampling (Dasgupta and Hsu, 2008), *hs*, a clustering based approach to AL, and BalancedEE (Osugi et al., 2005), $balee$, an approach explicitly focused on the balance between exploration and exploitation. D-Confidence also outperforms Simple (Tong and Koller, 2002) and KFF (Baram et al., 2004). Simple is based on maximum reduction of version space – which, in case of SVM, is equivalent to low confidence. KFF is based on distance. These last results reinforce our previous observations based on confidence and farthest-first.

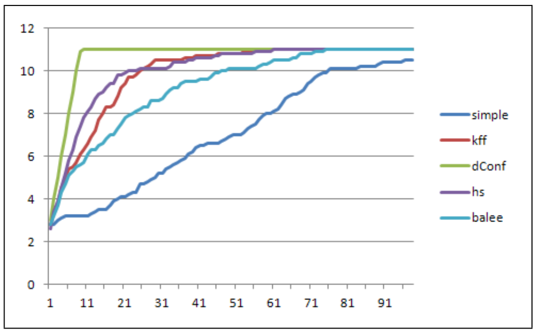
The observed improvements are very noticeable w.r.t. known classes, with a clear dominance of d-Confidence. This dominance is not as clear w.r.t. error. Nevertheless, d-Confidence still



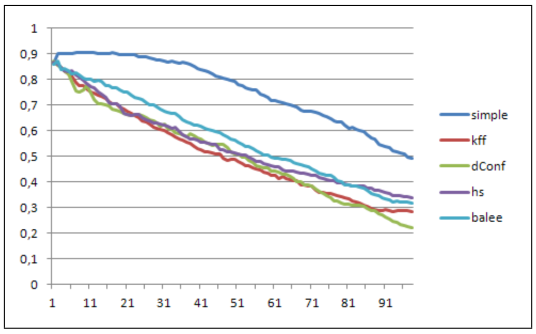
(a) Img-med known classes



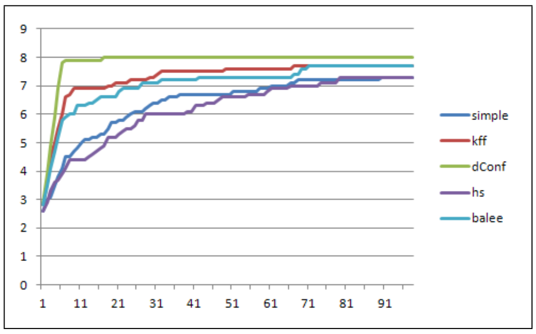
(b) Img-med error



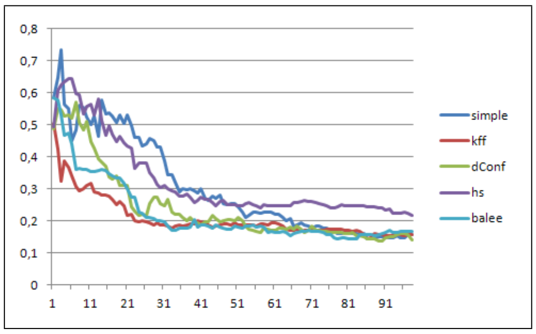
(c) Vowels known classes



(d) Vowels error



(e) Ecoli known classes



(f) Ecoli error

Figure 6.15: Evolution of known classes and error (results from Motta et al.)

presents the best performance on average when compared to the other criteria regarding error.

6.5 Exploration-exploitation trade-off

Balancing the trade-off between exploration and exploitation is an important issue in AL (Cebron and Berthold, 2009) contributing to maximize the marginal utility of queries along the learning process. The nature of d-Confidence – aggregating distance, that favors an exploration bias, and confidence, favoring an exploitation bias – grants dynamic shifting between exploration and exploitation as the learning process iterates. The shift between exploration and exploitation occurs in an unsupervised manner guided by the structure of the input space.

In a simplistic way, d-Confidence is formulated by $min\left(\frac{confidence}{distance}\right)$. When the confidence is similar for all unlabeled instances, the numerator is nearly constant, say C, and the driving force of d-Confidence is distance. In such circumstances d-Confidence selects queries with maximum distance to what is already known in order to minimize $\frac{C}{distance}$ thus, favoring exploration. On the other hand, when the distance between known classes and unlabeled instances is similar for all unlabeled instances, the denominator is nearly constant, D, and d-Confidence is mastered by confidence to minimize $\frac{confidence}{D}$ thus, favoring exploitation.

To investigate this characteristic of d-Confidence we have analyzed the variance of confidence, $V(c)$, and the variance of distance to known classes, $V(d)$, computed for all unlabeled instances, $x_j \in U_i$, during the learning process. Lower variance means that individual observations are closer to the mean and to each other. Low $V(d)$ means that the distance of unlabeled instances to known classes is barely constant. Likewise, low $V(c)$ means that the confidence on the labels of unlabeled instances is barely constant.

When $V(d)$ is lower than $V(c)$, d-Confidence is mainly driven by confidence and we say it is operating in *exploitation mode*, *MI*. When the opposite occurs, $V(c)$ is lower than $V(d)$, d-Confidence is mainly driven by distance and we say it is operating in *exploration mode*, *MR*.

Figures 6.16 represent for each dataset the average variance of confidence, $V(c)$, and the average variance of distance, $V(d)$, computed over all unlabeled instances $x_j \in U_i$ on the 10 cross validation folds as the learning process iterates. $V(c)$ and $V(d)$ are normalized to the maximum on each dataset for convenience of representation. There is no loss of generality arising from this normalization since the impact of either confidence or distance on d-Confidence at each iteration is conditioned by variability rather than absolute values. The evolution of the number of known classes throughout the learning process, *kc*, is also represented in these charts – also normalized to maximum. The operating mode regarding exploration versus exploitation is highlighted in these charts by vertical lines indicating the iterations when a shift of mode occurs. Exploration mode is referred by *MR* while exploitation mode is represented by *MI*.

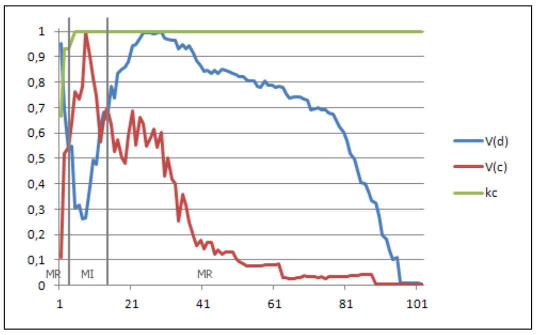
Dynamic shifting between exploration and exploitation modes is clear in most datasets. The Poker dataset (Figure 6.16e) is an exception where the compromise between exploration and exploitation is not evident.

At Iris (Figure 6.16a) after a brief initial period running on exploratory mode, d-Confidence shifts to exploitation for a few more iterations and then, around iteration 15, shifts again to exploration mode in which it remains for the rest of the learning cycle.

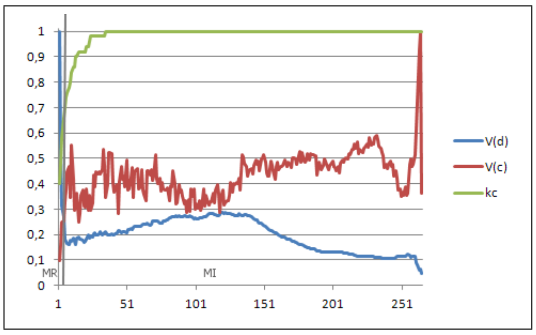
Vowels (Figure 6.16c) and the NG corpus (Figure 6.16a) are those exhibiting the more clear and effective trade-off. At the Vowels dataset we may observe an initial exploration bias until all the classes are identified followed by a period running in exploitation mode while the classifier is learning the target classes and confidence has high variability. Once the classification model is accurate enough, confidence variance decreases and d-Confidence shifts again to exploration mode. At the NG corpus the learning process starts running in exploratory mode until all target classes are known. Then, it shifts to exploitation mode where it remains until the end of the learning process while the classifier is being trained on the previously identified classes.

Satlog (Figure 6.16d) shifts its operating mode three times during the learning process. It starts in exploratory mode until all classes are identified. Then is shifts to exploitation while training the classifier and while confidence variance is high. Roughly half in the learning process, the variance of confidence has dropped and stabilized at a low level. This is an indication that the classifier is well trained and d-Confidence shifts to exploratory mode. At the last stage of the learning cycle, when input space is covered by the labeled instances, the normalized distance variance decreases below confidence and we shift to exploitation one more time.

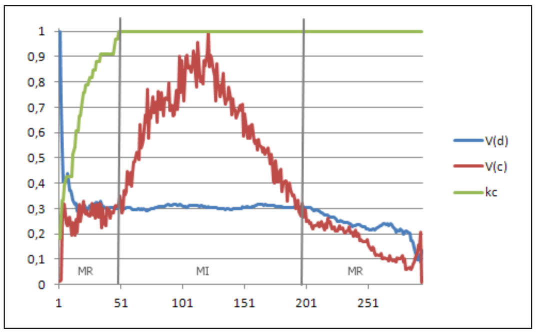
At the Cleveland dataset (Figure 6.16b) we are operating mostly in exploitation mode. Although there is an initial, very short, period running in exploratory mode, we shift very fast to the exploitation mode and remain operating at that mode for all the learning cycle. However, the



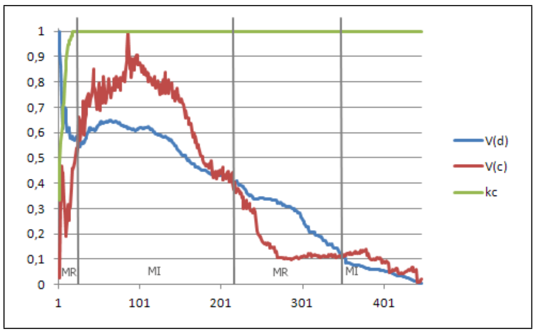
(a) Iris



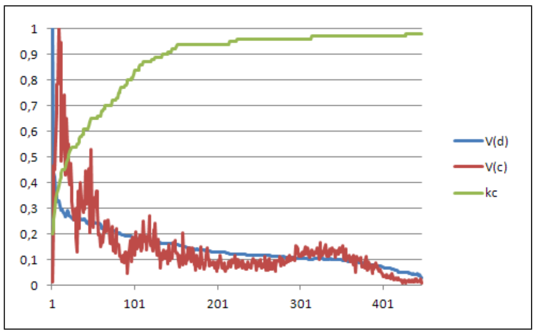
(b) Cleveland



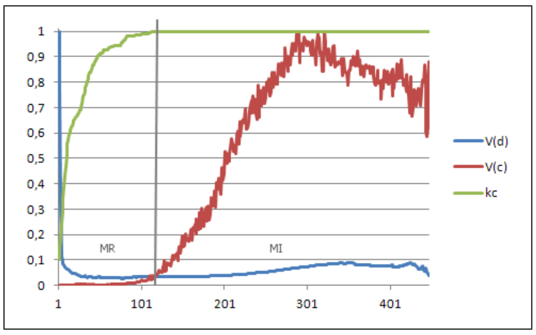
(c) Vowels



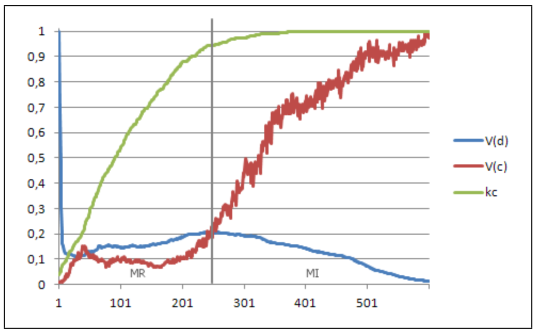
(d) Satlog



(e) Poker



(f) NG



(g) R52

Figure 6.16: Exploration-Exploitation trade-off

prevalence of confidence over distance is more clear during the second half of the learning cycle when the variance of confidence increases and that of distance decreases.

At the Reuters corpora (Figure 6.16a) there is a shift from exploration to exploitation around iteration 250 when more than 90% of the target classes are already identified. The distance variance, $V(d)$, starts decreasing a few iterations before identifying all target classes.

6.6 Prevailing outcomes

Experimental results provide evidence on the performance of d-Confidence towards our objectives.

Instance space coverage D-Confidence achieves a faster coverage of input space than both confidence and farthest-first irrespectively of the geometric properties of the working set. This improvement in instance space coverage is not made at the cost of accuracy. In fact, in general d-Confidence improves accuracy over confidence and farthest-first .

Base classifier The performance of d-Confidence seems to be slightly affected by the base classifier, mainly w.r.t. error. When referring to known classes, d-Confidence generally improves over its baseline criteria irrespectively of the base classifier. D-Confidence is suited for SVM classifiers where it generally improves over its baseline criteria. When using other base classifiers the performance is affected but improvements are still observable in general.

Confidence vs d-Confidence If we focus on SVM, we can observe that d-Confidence outperforms confidence, both at labeling effort and accuracy, over tabular datasets as well as over text corpora. D-Confidence dominates confidence w.r.t known classes throughout all the learning process. D-Confidence also outperforms confidence first-hit performance, in general. This dominance is also evident w.r.t. error except on highly imbalanced datasets where confidence outperforms d-Confidence.

Farthest-first vs d-Confidence D-Confidence clearly dominates farthest-first w.r.t. error when using SVM classifiers. The relative performance of these two criteria when it comes to known classes depends on the class distribution at the working set. On balanced datasets, d-Confidence clearly outperforms farthest-first. On imbalanced datasets d-Confidence still outperforms farthest-first on average; however farthest-first generally finds majority classes before d-Confidence.

State-of-the-art criteria vs d-Confidence In general d-Confidence requires less queries than state-of-the-art criteria to find representative instances from all target classes. This reduction of the labeling effort is expressive. The gain in minority classes is noteworthy. In general, this reduction on the labeling effort is not made at the cost of accuracy. Although in some scenarios d-Confidence is outperformed by other criteria w.r.t. accuracy, in general it is the most consistent approach also regarding predictive ability.

Data dimensionality The dimensionality of the input feature space does not compromise d-Confidence in general. Performance improvements over its baseline and other state-of-the-art criteria are observed irrespectively of the number of descriptive features. D-Confidence overcomes the weak discriminative power of Euclidean distance in high-dimensional spaces. Merging distance and confidence softens undesirable effects from the curse-of-dimensionality (Berchtold et al., 1997; Weber et al., 1998; Aggarwal et al., 2001) .

Balanced vs imbalanced class distributions In general, d-Confidence outperforms state-of-the-art criteria in finding exemplary instances from all the target classes. The gain is particularly relevant when finding under-represented classes in presence of highly imbalanced data. Under such circumstances, this gain however is achieved at the cost of accuracy. When in presence of imbalanced data, the exploratory bias of d-Confidence promotes exchanging accuracy for representativeness. This characteristic might be convenient to address the cold start problem (Attenberg and Provost, 2011) in an early stage of the learning process.

Exploration-exploitation trade-off The nature of d-Confidence boosts its exploration potential when the variance of the distances between unlabeled instances and known classes is higher than that of confidence. Exploitation behaviors become more evident when the opposite occurs. This is a main characteristic of d-Confidence that meets our goals. D-Confidence, formulated by

$min\left(\frac{confidence}{distance}\right)$, dynamically shifts between exploration and exploitation according to the current properties of both labeled and unlabeled sets. This is an automatic unsupervised process not requiring any tuning effort or additional cost.

Label disclosure complexity and error D-Confidence is an AL criterion combining confidence and distance that improves over existing strategies in terms of LDC – the number of queries needed to cover all target classes – without compromising predictive ability. There is a clear dominance of d-Confidence over other state-of-the-art criteria w.r.t. LDC. The effect on error is not as evident. Nevertheless, in the majority of the scenarios that we have essayed, d-Confidence presents improved or similar accuracy in comparison with its baseline or other state-of-the-art criteria.

6.7 Applications

D-Confidence has been applied to several real-world applications requiring text classification according to specific needs.

MailMaid is an application to assist e-mail users organizing their mailbox. MailMaid was developed in 2010 for the Portuguese Association for Artificial Intelligence[5] (APPIA). It is a tool to assist users to organize the APPIA mailbox according to each one's specific interests. D-Confidence supports the learning stage of the tool. MailMaid was developed by two Belgian students, Matthias Vermeiren and Joachim Seminck, during their BSc capstone project on top of the previous MSc thesis performed by Luiza Gabriel who studied the potential of d-Confidence to organize mail corpora. This MSc thesis, Automatic Email Organization (Gabriel, 2009), was awarded an honorable mention at the national TLeIA contest in 2009[6].

TIENA Tecnologias Inovadoras em mineração de textos para a Espacialização de Notícias Agrícolas: piloto cana-de-açúcar is a research project, coordinated by Embrapa[7] – Empresa Brasileira

[5]http://www.appia.pt, accessed on October 2012
[6]http://www.appia.pt/index.php?option=com_content&task=view&id=456&Itemid=166, accessed on October 2012
[7]http://www.embrapa.br, accessed on October 2012

de Pesquisa Agropecuária, that aims to organize text corpora on an holistic approach contemplating time, geographical and topic views. D-Confidence is being applied to support the topic view.

Free text assessment is another application of d-Confidence developed by a BSc student, Augusto Cruz, for his capstone project – Correcção semi-automática de testes. The aim of this application is to support teachers in the assessment of free text answers to open questions. D-Confidence is wrapped in the assessment process and reorders the students answers being presented to the teacher for assessment, categorizing them in accordance to the marks that the teacher is assigning as the process runs. The rationale is that by organizing answers in homogeneous groups, aligned with the marks recently assigned, the assessment process is less dependent on judgment drift and more reliable. This work was granted the best paper award of the Portuguese/Spanish track at the 9th European Conference on e-Learning (Escudeiro et al., 2010).

Automatic clipping is one more application of d-Confidence, that builds on top of Automatic Harvesting of Academic Events, a former capstone project developed by Els Bockaerts in 2009. The aim of this application is to assist a web site editor in keeping the *Activities* web page of the Erasmus website of the Computer Engineering Department at ISEP updated, helping incoming Erasmus students to know Porto and to find social activities in the Porto region. The application retrieves news on relevant events from a set of web sites previously specified. D-Confidence is used to build the classifier that will organize these news according to the structure of the Activities web page.

Chapter 7

Stopping Criteria

Active learning (AL) aims at selecting the most informative instances to query taking the goal of the learning task into account. When the goal is to reduce the cost of the learning process, as in our case, it is important to go a little further and analyze whether the most informative instance is still valuable enough. In our case, the utility of the queries should compensate their cost. Therefore, querying the oracle should stop once the cost of querying overcomes the utility of the unlabeled instances still remaining in the working set. When to stop querying is one important open issue in AL. This concern led us to investigate suitable stopping criteria for d-Confidence subject to the goals governing our research problem.

7.1 D-Confidence stopping criteria

Our preliminary reflections on stopping criteria drove us to propose three base criteria – classification gradient, steady entropy mean and steady entropy distribution – plus two hybrid criteria aggregating the former in a specific way as to improve over its foundational criteria – ensemble of classification gradient and steady entropy mean, ensemble of classification gradient and steady entropy distribution.

7.1.1 Base stopping criteria

All the base stopping criteria proposed in this work evaluate stopping conditions from variations of certain indicators observed between two consecutive iterations during the learning process.

Classification gradient at iteration i, cgr_i, is the percentage of instances $x_j \in U_i$ whose label, $\hat{y}^i_j$, as predicted by h_i is distinct from the label predicted at the previous iteration $i-1$, $\hat{y}^{i-1}_j$ (Equation 7.1).

$$cgr_i = \frac{\#\left\{x_j \in U_i : \hat{y}^i_j \neq \hat{y}^{i-1}_j\right\}}{\#U_i} \tag{7.1}$$

This criterion provides evidence on the ability of q_i – the query at the current iteration – to change the current hypothesis. When the change in the classifier predictions being induced by querying and adding the most informative unlabeled instance to the training set is residual then we assume that the set of unlabeled instances does not contain further evidence on the target concept that is worth considering. The utility of the remaining instances in U_i is low for the learning problem being studied and we may discard them.

Applying *cgr* requires tuning of the parameter that triggers the stop sign. This parameter, $\Delta\hat{Y}$, sets the minimum classification gradient below which querying should stop. This threshold sets the minimum percentage of predictions that have changed from one iteration to the next below which it is assumed that the cost of querying overcomes the potential utility of the unlabeled instances which, in this case, relates to the potential to induce changes in predictions.

Steady entropy mean at iteration i, sew_i, is based on the entropy of $\hat{Y}^i_j$ (Equation 7.2). $\hat{Y}^i_j$ is a discrete random variable of the predictions made by h_i given an instance $x_j \in U_i$.

$$H(\hat{Y}^i_j) = -\sum_{c_k \in C_i} p_i\left(c_k|x_j\right) ln\left(p_i\left(c_k|x_j\right)\right) \tag{7.2}$$

In Equation 7.2, C_i is the set of classes known at iteration i and $p_i\left(c_k|x_j\right)$ is the posterior probability of c_k given x_j at iteration i.

To compute sew_i we compare the median of $H(\hat{Y}^i) = \left\{H(\hat{Y}^i_1), ..., H(\hat{Y}^i_j), ..., H(\hat{Y}^i_{N_i})\right\}$ with the median of $H(\hat{Y}^{i-1}) = \left\{H(\hat{Y}^{i-1}_1), ..., H(\hat{Y}^{i-1}_j), ..., H(\hat{Y}^{i-1}_{N_{i-1}})\right\}$. N_i is the cardinality of U_i. This is done using a Wilcoxon test. *sew* stands for Steady Entropy Wilcoxon.

H_0: the median of $H(\hat{Y}^i)$ is equal to that of $H(\hat{Y}^{i-1})$

H_1: the median of $H(\hat{Y}^i)$ is not equal to that of $H(\hat{Y}^{i-1})$

At each iteration we record the p-value of the test to decide whether to reject the null hypothesis or not. The sew_i value is in fact an integer counting the number of consecutive iterations, including i, where the null hypothesis of the Wilcoxon test was not rejected. sew_i is reset to 0 at any iteration rejecting the null hypothesis.

This criterion is motivated by the assumption that if the unpredictability of $H(\hat{Y}^i)$ is centered at the same level as that of $H(\hat{Y}^{i-1})$ then we may assume that U_i cannot provide any more evidence on the target concept.

This criterion requires two parameters. We must set the significance of the Wilcoxon test, α, and the number of consecutive iterations not rejecting H_0 that must occur to stop the learning process, l.

Steady entropy distribution sek_i, is similar to sew_i but using the Kolmogorov-Smirnov test for equal probability distributions instead. *sek* stands for Steady Entropy Kolmogorov-smirnov.

H_0: $H(\hat{Y}^i)$ and $H(\hat{Y}^{i-1})$ come from the same distribution

H_1: $H(\hat{Y}^i)$ and $H(\hat{Y}^{i-1})$ do not come from the same distribution

This criterion is motivated by the assumption that if $H(\hat{Y}^i)$ and $H(\hat{Y}^{i-1})$ both follow the same probability distribution, then we may presume that U_i cannot provide any more evidence on the target concept.

Like *sew*, this criterion also requires two parameters: the significance of the Kolmogorov-Smirnov test, α, and the number of consecutive iterations not rejecting H_0 that must occur to stop the learning process, l.

7.1.2 Hybrid stopping criteria

One drawback of the criteria based on the entropy of $\hat{Y}_j$ alone is that we may have the same value for $H(\hat{Y}_j)$ generated for different predictions. Entropy is computed from the probabilities only with no association to the predicted labels. For instance, the entropy of the predictions $\hat{Y}^a$ and $\hat{Y}^b$ for the hypothetic case presented in Table 7.1 is $H(\hat{Y}^a) = H(\hat{Y}^b) = 0.5$ despite the fact that they correspond to different predictions, with $\hat{Y}_1^a = 1$ and $\hat{Y}_1^b = 2$.

Each of *sew* and *sek* are necessary but not sufficient conditions for stopping. Our hybrid stopping criteria attempt to overcome this drawback by complementing *sew* and *sek* with *cgr*. The

Table 7.1: Drawback of entropy as a stopping criteria

c_k	$p^a(c_k\|x_0)$	$p^b(c_k\|x_0)$
1	0.8	0.2
2	0.2	0.8

rationale is that classification gradient, *cgr*, provides evidence on the predicted labels themselves while entropy based criteria – *sew* and *sek* – inform on the probability distribution of those predictions. Together they are expected to be sufficient stopping conditions making the hybrid criteria more robust than their baseline alone.

We have defined two hybrid criteria each one being triggered when both its baseline criteria have been triggered.

Ensemble of classification gradient and steady entropy mean *hcw*, triggers at the first iteration *i* that simultaneously verifies *cgr* and *sew*. *hcw* stands for Hybrid Classification gradient Wilcoxon test for steady entropy mean.

Ensemble of classification gradient and steady entropy distribution *hck*, triggers at the first iteration *i* simultaneously verifying *cgr* and *sek*. *hck* stands for Hybrid Classification gradient Kolmogorov-smirnov test for steady entropy distribution.

7.2 Other stopping criteria under evaluation

To evaluate the stopping criteria being proposed in this thesis we have assessed their performance in comparison with five other criteria.

Average confidence (Vlachos, 2008) *cdrp*, computes the average confidence on a pre-labeled validation set and stops the learning process when this average remains stable or drops for a number of consecutive rounds. This criteria has one main disadvantage when compared to our proposals given our concrete research problem. It requires, according to the authors, a large pre-labeled validation set that adds a significant overhead to the cost of the learning process. This criterion requires tuning two parameters: the maximum change in average confidence from one iteration to the next allowing to state that confidence remains stable or drops, ΔP, and the number of consecutive drops that are necessary to trigger the stopping sign, l.

Max-confidence (Zhu and Hovy, 2007) *maxc*, based on class posteriors, stops the learning process when the entropy, $H(\hat{Y}_j^i)$, of the predictions of a query (Equation 7.2) is less than a very small predefined threshold, H, close to zero – the authors suggest $H = 0.001$. This method relies on the entropy of one single unlabeled instance, the one where the current classifier is less confident about. *Maxc* requires tuning of one parameter – the entropy threshold H.

Overall-uncertainty (Zhu et al., 2008) *ovru*, is based on max-confidence with the difference that while max-confidence only assesses the most informative instance at each iteration, i, overall-uncertainty assesses all unlabeled instances $x_j \in U_i$. At each iteration, overall-uncertainty computes the average entropy on the unlabeled set, $\overline{H}$, and checks whether it is below a very small threshold halting the learning process when this is verified. This method requires tuning the average entropy threshold, $\overline{H}$.

Min-error (Zhu and Hovy, 2007) *mine*, is based on the oracle's feedback when asked for the true label of a query. It considers whether the current classifier can correctly predict the label of the selected query – or achieve a high accuracy performance of predictions in batch mode AL – as the learning process iterates. Stopping is triggered when the average accuracy of the predicted labels is larger than a predefined accuracy threshold, AT – the authors use $AT = 0.9$. This method requires tuning the accuracy threshold.

Variance model (Ghayoomi, 2010), *cvar*, computes the variance of the confidence on the t top-confidence predicted labels and decides to stop when this variance decreases in l sequential iterations. This criterion is based on the assumption that the variance of confidence scores of the top-confidence predicted labels increases during an initial stage of the learning process, while the classifier is untrained, to become stable at a second stage, when the classifier has improved its performance and training is becoming more efficient, followed by a final stage, when the classifier is well trained and variance decreases. From among the stopping criteria being evaluated this is the one that requires most parameters. The variance model requires to set three parameters: the length of the batch required to estimate variance, t, the minimum number of consecutive iterations having variance decreasing, l, and the minimum change in variance, ΔV, between iterations to assume that the variance has decreased – the authors use $t = 5$, $l = 2$ and $\Delta V = 0.5$.

7.3 Experimental setup

The evaluation of stopping criteria was performed over 20 datasets and 10 stopping criteria – five of which are proposed by us. This evaluation aims to assess these stopping criteria with the purpose of selecting the most appropriate to use in d-Confidence AL. The criteria proposed by us – *cgr*, *sew*, *sek*, *hcw* and *hck* – are highlighted in bold in the tables presenting evaluation results.

7.3.1 Datasets used to evaluate stopping criteria

The evaluation of the proposed stopping criteria was performed over 20 datasets, the majority of them retrieved from UCI (Frank and Asuncion, 2010), including three text corpora. These datasets (Table 7.2) were selected to assure a broad coverage of their meta-properties, namely number of attributes – ranging from four to 10333, number of classes – from three to 52 – and class distribution – six balanced datasets, six imbalanced datasets and eight with a fairly balanced distribution. From the three text corpora used for evaluation, two – R52 and NG, previously used to evaluate d-Confidence – are modeled using TF×IDF and the other one – Amazon, retrieved from UCI – is modeled using TF.

From each of these datasets we have extracted a random sample to be used exclusively for *cdrp* validation as required by this criterion (Vlachos, 2008) (column *#Validation* in Table 7.2). The remaining instances at each dataset were partitioned into 10 random samples of equal dimension. One of these samples at a time is used at each of 10 validation folds for testing. The remaining constitute the working set at each validation fold for a given dataset, W – the AL pool. Estimates are generated by 10-fold cross validation.

7.3.2 Parameter tuning

Before proceeding to evaluation we have fine tuned the parameters of each stopping criteria. Fine tuning was based on the experimental results observed on three of our datasets. These

Table 7.2: Datasets used to evaluate stopping criteria

Dataset	Type	Attributes	Classes	#W	#Test	#Validation
Abalone	tabular	8	28	450	50	100
Amazon	text (TF)	10000	50	450	50	100
Balance	tabular	4	3	450	50	100
Car	tabular	6	4	450	50	100
Cleveland	tabular	13	5	210	30	58
Ctg	tabular	21	10	450	50	100
Digit	tabular	240	10	450	50	100
Iris	tabular	4	3	105	15	30
Isolet	tabular	617	26	450	50	100
Letter	tabular	16	26	450	50	100
Lrs	tabular	101	48	360	40	100
NG	text (TF$\times$IDF)	10333	20	350	50	100
Nursery	tabular	8	5	450	50	100
Optdigits	tabular	64	10	450	50	100
Poker	tabular	10	10	350	50	100
R52	text (TF$\times$IDF)	6019	52	800	100	100
Robot	tabular	24	4	450	50	100
Satlog	tabular	36	6	350	50	100
Segment	tabular	19	7	450	50	100
Vowels	tabular	10	11	227	33	70

three datasets were selected from their class distribution. We have ranked the datasets by increasing order of the absolute value of their class distribution skewness (Table 7.3) and then selected for parameter tuning one of the most balanced – Iris, the most imbalanced – Poker – and one fairly balanced dataset – Ctg. We have assumed as balanced one distribution having $-0.03 \leq skewness \leq 0.03$. A distribution with $-1 \leq skewness \leq -0.03 \vee 0.03 \leq skewness \leq 1$ is assumed to be fairly balanced and one with $skewness \leq -1 \vee skewness \geq 1$ is assumed to be imbalanced.

To tune the parameters we have evaluated the average penalty, computed over the three tuning datasets, incurred by each stopping criteria when the parameter value changes. At each dataset, the penalty is the absolute mean difference – computed on the 10 validation folds – between the iteration that triggers the stop sign and the ideal iteration to stop w.r.t. error. The iteration when the generalization error reaches its minimum for the first time is assumed to be the ideal iteration to stop w.r.t. error.

The value of each parameter starts at a hypothetical reasonable value and is changed to higher values and to lower values until a minimum penalty is observed. Initial values are those proposed

Table 7.3: Class distribution skewness

Dataset	Skewness	|Skewness|	Bias	Tuning
Iris	0.000	0.000	balanced	X
Vowels	0.000	0.000	balanced	
Amazon	0.0131	0.0131	balanced	
Isolet	0.0146	0.0146	balanced	
Optdigits	0.018	0.018	balanced	
Digit	-0.024	0.024	balanced	
Segment	0.036	0.036	fair	
NG	0.039	0.039	fair	
Letter	-0.052	0.052	fair	
Satlog	0.285	0.285	fair	
Ctg	0.394	0.394	fair	X
Nursery	0.488	0.488	fair	
Balance	-0.500	0.500	fair	
Robot	0.967	0.967	fair	
Cleveland	1.031	1.031	imbalanced	
Car	-1.032	1.032	imbalanced	
R52	1.234	1.234	imbalanced	
Abalone	1.491	1.491	imbalanced	
Lrs	1.961	1.961	imbalanced	
Poker	3.213	3.213	imbalanced	X

by the authors, when available, or commonly accepted benchmarks, like $\alpha = 5\%$. For those stopping criteria requiring more than one parameter, each one is tuned at a time while keeping the remaining constant. In first place we always fine tune the parameter evaluating the marginal progression from one iteration to the next – significance of the statistical test in both *sew* and *sek*, change in average confidence at $cdrp$ and change in confidence variance in $cvar$. Once these are fixed, we proceed fine tunning the number of individual stop signals required to assume that the stopping criterion is stable.

For instance, to tune the two parameters required by *sek* (Table 7.4) – significance of the Kolmogorov-Smirnov test, α, and the number of consecutive iterations accepting the null hypothesis, l – we have started by analyzing the effect of significance, starting with $\alpha = 5\%$ and changing this value up and down until a minimum was observed while l was kept constant at $l = 3$. The most common significance levels – 1%, 5% and 10% – were essayed. Then, the significance was set to the best performer, $\alpha = 5\%$, and l was changed until a minimum was found. Finally, the parameters were set to the best combination found through this process. In the case of *sek* the tuned parameters are $\alpha = 5\%$ and $n = 5$ as set at the fifth tuning run. This tuning process was performed to set the parameters of all stopping criteria under evaluation (see Table 7.5).

Table 7.4: Tuning process for *sek*

Run	α	l	Penalty
1	0.05	3	25
2	0.01	3	26
3	0.1	3	45
4	0.05	1	31
5	0.05	5	16
6	0.05	7	18

Table 7.5: Stopping criteria parameters

Criteria	$\Delta\hat{Y}$	α	ΔP	H	$\overline{H}$	AT	ΔV	l	t
cgr	0.01								
sew		0.05						3	
sek		0.05						5	
cdrp			0.01					3	
maxc				0.5					
mine							0.9		
ovru					0.5				
cvar							0.0001	2	5

$\Delta\hat{Y}$ is the minimum classification gradient triggering *cgr*. α is, for both *sew* and *sek*, the significance level of the corresponding statistical test. For both these criteria, *l* refers to the number of consecutive iterations accepting the null hypothesis that is required to stop. ΔP is the threshold change in average confidence required by *cdrp*. For *cdrp*, *l* is the number of consecutive drops in average confidence that is required to stop. H and $\overline{H}$ both refer to the threshold of the entropy of the predictions below which the corresponding criteria are triggered. H refers to the marginal entropy of the instance where the current classifier is least certain about while $\overline{H}$ refers to the average entropy of the predictions of all unlabeled instances. AT is the minimum accuracy of query predictions that is required to trigger *mine*. ΔV is the minimum change in the variance of confidence between two consecutive iterations required to assume that it is decreasing. Regarding *cvar*, *l* refers to the minimum number of consecutive iterations having variance decreasing that are required to stop while *t* refers to the length of the batch required to estimate the variance of confidence.

7.3.3 Evaluation plan

Having all the parameters tuned we have proceeded to evaluate the performance of the stopping criteria running d-Confidence on the previously described datasets (Section 7.3.1). For each dataset and for each iteration of the AL process we have recorded error, the number of known classes and the stopping criteria indicators.

The ideal iteration to stop was defined w.r.t. both error and known classes (Table 7.6). For each dataset, the ideal iteration to stop regarding error is the iteration where the generalization error achieves its minimum for the first time. The ideal iteration to stop regarding the number of known classes is the first iteration, *i*, during the learning process verifying $\forall c_k \in C, \exists < x_j, y_j > \in L_i : y_j = c_k$. The maximum number of iterations at each dataset is $N-2$. This limit corresponds to querying all instance in the working set, one at a time, except the two pre-labeled instances in L_1.

Table 7.6: Ideal iteration to stop querying

Dataset	Error	Known classes	Maximum
Iris	14	3	103
Cleveland	36	15	208
Vowels	221	27	225
Poker	80	99	348
Satlog	74	10	348
R52	353	257	798
NG	284	86	248
Abalone	416	442	448
Amazon	435	267	448
Balance	191	14	448
Car	440	392	448
Robot	358	33	448
Ctg	417	136	448
Digit	148	51	448
Isolet	323	165	448
Letter	362	138	448
Lrs	306	296	358
Nursery	229	326	448
Optdigits	447	63	448
Segment	436	59	448

For each dataset, the number of iterations required to trigger the stop signal is estimated by the average over the cross validation folds (Table 7.7). The entries in Table 7.7 marked with * refer to situations where stopping did not occur during the entire learning process. In such cases, a penalty of one iteration w.r.t. the worst criterion that effectively stopped is charged.

Table 7.7: Number of iterations to stop

Dataset	cdrp	ovru	maxc	mine	cvar	**cgr**	**sew**	**sek**	**hcw**	**hck**
Iris	24	21	32	13	5	24	11	19	24	24
Cleveland	6	207	208	116	5	104	52	35	104	104
Vowels	8	7*	7*	119	12	152	42	33	152	152
Poker	5	4*	4*	98	9	72	77	49	77	72
Satlog	8	169	148	55	13	56	38	43	56	56
R52	4	3*	3*	335	4	11	376	225	376	225
NG	4	3*	3*	324	324	4	179	162	179	162
Abalone	4	3*	3*	3*	29	445	124	138	445	445
Amazon	4	0*	1	433	7	434	97	72	434	434
Balance	9	93	126	91	7	89	11	92	89	92
Car	7	3	179	14	7	48	330	43	330	48
Robot	10	227	297	199	15	127	261	60	261	127
Ctg	4	3*	3*	429	437	47	406	430	406	430
Digit	4	249	283	3*	11	99	37	42	99	99
Isolet	4	3*	3*	198	4	225	66	46	225	225
Letter	4	3*	3*	272	11	293	119	64	293	293
Lrs	4	3*	3*	3*	8	317	93	64	317	317
Nursery	7	4	216	98	13	57	13	28	57	57
Optdigits	4	3*	3*	3*	3*	134	400	429	400	429
Segment	7	2*	2*	3	2*	4	380	416	380	416

7.4 Evaluation

The evaluation of the stopping criteria was performed with a main concern in cost reduction. The cost of querying and the opportunity cost of not querying are two opposite factors being considered. Cost analysis is guided by error and known classes, the relevant indicators of classifier quality given our objectives. Performance loss due to erratic stop is an alternative perspective of cost that we have also analyzed.

7.4.1 Stopping criteria performance

To have a clear view on the performance of each stopping criterion, we have computed from the results in Tables 7.6 and 7.7 the penalty incurred in each dataset w.r.t. error (Table 7.8) and known classes (Table 7.10). Penalty here is assumed to be the difference between the iteration when the first stop signal is triggered and the ideal stopping iteration regarding either error or the number of known classes. Negative differences mean that the stop signal was triggered before the ideal while positive differences mean it was triggered after the ideal. The majority of first stop signs w.r.t. error occurs before the ideal (Table 7.9).

Table 7.8: Penalty iterations w.r.t. error

Dataset	cdrp	ovru	maxc	mine	cvar	**cgr**	**sew**	**sek**	**hcw**	**hck**
Iris	10	7	18	-1	-9	10	-3	5	10	10
Cleveland	-30	171	172	80	-31	68	16	-1	68	68
Vowels	-213	-214	-214	-102	-209	-69	-179	-188	-69	-69
Poker	-75	-76	-76	18	-71	-8	-3	-31	-3	-8
Satlog	-66	95	74	-19	-61	-18	-36	-31	-18	-18
R52	-349	-350	-350	-18	-349	-342	23	-128	23	-128
NG	-280	-281	-281	40	40	-280	-105	-122	-105	-122
Abalone	-412	-413	-413	-413	-387	29	-292	-278	29	29
Amazon	-431	-435	-434	-2	-428	-1	-338	-363	-1	-1
Balance	-182	-98	-65	-100	-184	-102	-180	-99	-102	-99
Car	-433	-437	-261	-426	-433	-392	-110	-397	-110	-392
Robot	-348	-131	-61	-159	-343	-231	-97	-298	-97	-231
Ctg	-413	-414	-414	12	20	-370	-11	13	-11	13
Digit	-144	101	135	-145	-137	-49	-111	-106	-49	-49
Isolet	-319	-320	-320	-125	-319	-98	-257	-277	-98	-98
Letter	-358	-359	-359	-90	-351	-69	-243	-298	-69	-69
Lrs	-302	-303	-303	-303	-298	11	-213	-242	11	11
Nursery	-222	-225	-13	-131	-216	-172	-216	-201	-172	-172
Optdigits	-443	-444	-444	-444	-444	-313	-47	-18	-47	-18
Segment	-429	-434	-434	-433	-434	-432	-56	-20	-56	-20

The same analysis can be made w.r.t. known classes. Tables 7.10 and 7.11 present the results observed when comparing the iteration when the first stop signal is triggered and the ideal stopping iteration regarding the number of known classes.

Stop signs w.r.t. the number of known classes occur earlier than the ideal for *cdrp*, *cvar*, *ovru* and *maxc* and later than the ideal for *hcw*, *hck*, *cgr* and *mine* (Table 7.11). The clear trend to stop before the ideal w.r.t. error, irrespectively of the stopping criteria, is not observed regarding the number of known classes.

The behavior of *cdrp*, *cvar*, *ovru* and *maxc* from both points of view – error and known classes – is very similar exhibiting a clear trend to stop earlier than the ideal. The other criteria – *sew*, *sek*, *mine*, *cgr*, *hck* and *hcw* – have a tendency to stop earlier than the ideal when it comes to error and after the ideal regarding known classes.

Table 7.9: Number of datasets where stopping occurs before/after ideal w.r.t. error

Criteria	Before	After
hck	15	5
hcw	15	5
cgr	16	4
maxc	16	4
mine	16	4
ovru	16	4
cvar	18	2
sek	18	2
sew	18	2
cdrp	19	1

Table 7.10: Penalty iterations w.r.t. known classes

Dataset	cdrp	ovru	maxc	mine	cvar	**cgr**	**sew**	**sek**	**hcw**	**hck**
Iris	21	18	29	10	2	21	8	16	21	21
Cleveland	-9	192	193	101	-10	89	37	20	89	89
Vowels	-19	-20	-20	92	-15	125	15	6	125	125
Poker	-94	-95	-95	-1	-90	-27	-22	-50	-22	-27
Satlog	-2	159	138	45	3	46	28	33	46	46
R52	-253	-254	-254	78	-253	-246	119	-32	119	-32
NG	-82	-83	-83	238	238	-82	93	76	93	76
Abalone	-438	-439	-439	-439	-413	3	-318	-304	3	3
Amazon	-263	-267	-266	166	-260	167	-170	-195	167	167
Balance	-5	79	112	77	-7	75	-3	78	75	78
Car	-385	-389	-213	-378	-385	-344	-62	-349	-62	-344
Robot	-23	194	264	166	-18	94	228	27	228	94
Ctg	-132	-133	-133	293	301	-89	270	294	270	294
Digit	-47	198	232	-48	-40	48	-14	-9	48	48
Isolet	-161	-162	-162	33	-161	60	-99	-119	60	60
Letter	-134	-135	-135	134	-127	155	-19	-74	155	155
Lrs	-292	-293	-293	-293	-288	21	-203	-232	21	21
Nursery	-319	-322	-110	-228	-313	-269	-313	-298	-269	-269
Optdigits	-59	-60	-60	-60	-60	71	337	366	337	366
Segment	-52	-57	-57	-56	-57	-55	321	357	321	357

Table 7.11: Number of datasets where stopping occurs before/after ideal w.r.t. known classes

Criteria	Before	After
hcw	3	17
hck	4	16
cgr	7	13
mine	8	12
sek	10	10
sew	10	10
maxc	14	6
ovru	14	6
cvar	16	4
cdrp	19	1

7.4.2 Penalty cost

Given the aims of our research, the cost associated to stopping out of time is probably more relevant then the difference between real and ideal stopping iterations. To evaluate this we have assigned a marginal cost per iteration that corresponds to the cost of having classifiers which have not queried useful instances. This marginal cost, B, is the utility of unlabeled instances or, from another perspective, the opportunity cost of not querying. The cost of querying, A, is the other relevant cost associated to the stopping decision. The total cost associated to the penalty of stopping out of time, $A+B$, is the sum of the utility of the queries that were wasted for not being asked – when stopping before the ideal – and the querying cost of unnecessary queries – when stopping after the ideal.

When the utility of unlabeled instances is assumed to be equal to the cost of querying, both being equal to 1, $A = B = 1$, the total cost equals the absolute value of the difference between the iteration when the first stop signal is triggered and the ideal stopping iteration. Total penalty costs associated to each stopping criteria in each dataset, assuming $A = B = 1$, are presented in Table 7.12, regarding error, and in Table 7.13, regarding known classes.

The rank of the stopping criteria being evaluated by decreasing order of the penalty w.r.t. error (Table 7.14) shows that the new criteria being proposed perform well in general. Our five criteria are ranked in top six.

The average penalty cost of the hybrid stopping criteria combining classification gradient and steady entropy mean, *hcw*, outperforms all the other. The statistical significance of the observed

Table 7.12: Total cost w.r.t. error when A = B = 1

Dataset	cdrp	ovru	maxc	mine	cvar	**cgr**	**sew**	**sek**	**hcw**	**hck**
Iris	10	7	18	1	9	10	3	5	10	10
Cleveland	30	171	172	80	31	68	16	1	68	68
Vowels	213	214	214	102	209	69	179	188	69	69
Poker	75	76	76	18	71	8	3	31	3	8
Satlog	66	95	74	19	61	18	36	31	18	18
R52	349	350	350	18	349	342	23	128	23	128
NG	280	281	281	40	40	280	105	122	105	122
Abalone	412	413	413	413	387	29	292	278	29	29
Amazon	431	435	434	2	428	1	338	363	1	1
Balance	182	98	65	100	184	102	180	99	102	99
Car	433	437	261	426	433	392	110	397	110	392
Robot	348	131	61	159	343	231	97	298	97	231
Ctg	413	414	414	12	20	370	11	13	11	13
Digit	144	101	135	145	137	49	111	106	49	49
Isolet	319	320	320	125	319	98	257	277	98	98
Letter	358	359	359	90	351	69	243	298	69	69
Lrs	302	303	303	303	298	11	213	242	11	11
Nursery	222	225	13	131	216	172	216	201	172	172
Optdigits	443	444	444	444	444	313	47	18	47	18
Segment	429	434	434	433	434	432	56	20	56	20
$\overline{cost}$	273	265	242	153	238	153	127	156	57	81
σ_{cost}	146	145	154	158	159	149	106	131	46	96

Table 7.13: Total cost w.r.t. known classes when A = B = 1

Dataset	cdrp	ovru	maxc	mine	cvar	**cgr**	**sew**	**sek**	**hcw**	**hck**
Iris	21	18	29	10	2	21	8	16	21	21
Cleveland	9	192	193	101	10	89	37	20	89	89
Vowels	19	20	20	92	15	125	15	6	125	125
Poker	94	95	95	1	90	27	22	50	22	27
Satlog	2	159	138	45	3	46	28	33	46	46
R52	253	254	254	78	253	246	119	32	119	32
NG	82	83	83	238	238	82	93	76	93	76
Abalone	438	439	439	439	413	3	318	304	3	3
Amazon	263	267	266	166	260	167	170	195	167	167
Balance	5	79	112	77	7	75	3	78	75	78
Car	385	389	213	378	385	344	62	349	62	344
Robot	23	194	264	166	18	94	228	27	228	94
Ctg	132	133	133	293	301	89	270	294	270	294
Digit	47	198	232	48	40	48	14	9	48	48
Isolet	161	162	162	33	161	60	99	119	60	60
Letter	134	135	135	134	127	155	19	74	155	155
Lrs	292	293	293	293	288	21	203	232	21	21
Nursery	319	322	110	228	313	269	313	298	269	269
Optdigits	59	60	60	60	60	71	337	366	337	366
Segment	52	57	57	56	57	55	321	357	321	357
$\overline{cost}$	140	177	164	147	152	104	134	147	127	134
σ_{cost}	137	119	105	126	141	91	124	136	105	123

Table 7.14: Criteria rank of penalty cost w.r.t. error

Criteria	$\overline{Cost}$
hcw	57.4
hck	81.3
sew	126.8
mine	153.1
cgr	153.2
sek	155.8
cvar	238.2
maxc	242.1
ovru	265.4
cdrp	273.0

differences of mean penalty cost of *hcw* compared to each of the other stopping criteria was evaluated by t-tests on paired samples for equal means. The p-value of these tests (Table 7.15) confirms that *hcw* achieves lower penalty costs than all the other except *hck*, assuming a significance level of 5%.

Table 7.15: P-value for equal means of error penalty cost w.r.t. *hcw*

Criteria	p-value
cdrp	3.75×10^{-6}
ovru	7.78×10^{-6}
cvar	7.12×10^{-5}
maxc	1.35×10^{-4}
sek	0.002
cgr	0.007
sew	0.008
mine	0.014
hck	0.157

To investigate whether penalty costs are related to class distribution we have computed for *hcw* – the top performer – the correlation between penalty costs and distribution skewness. The correlation between the penalty cost regarding error and skewness is -0.267 while that of penalty cost regarding known classes is -0.390. Although these are not strong correlations they make us suspect of a light tendency to have lower *hcw* penalty costs associated to imbalanced class distributions than those associated to balanced data. This makes sense because achieving class-completeness – finding representative instances for all target classes – and learning classification models requires more queries for imbalanced datasets than for balanced ones. The *hcw* stopping criterion, on its own, depends on the stability of *cgr* and *sew* which also requires a sufficiently large number of queries. This, however, is not much dependent on the skewness of the underlying class distribution (Table 7.16). In Table 7.16[1], we have computed the average percentage of instances in W that are queried before stopping. These averages were computed over all balanced and imbalanced datasets as defined in Table 7.3. Class distribution has a high impact on the ideal iteration to stop w.r.t. known classes. The ideal iteration to stop w.r.t. known classes occurs after querying 56% of all instances in W, on average, for imbalanced data, and after 23% on balanced data. The same

[1]In Table 7.16, *Ideal e* and *Ideal kc* refer, respectively, to the ideal iteration to stop w.r.t. error and known classes. *hcw stop* refers to the iteration where *hcw* triggers for the first time. *Diff. to ideal e* and *Diff. to ideal kc* refer to the differences between *hcw stop* and, respectively, *Ideal e* and *Ideal kc*. These differences are proportional to penalty costs.

influence is not observed w.r.t. error. In fact, in balanced datasets, the ideal iteration to stop occurs on average after querying 69% of available queries while, for imbalanced datasets, this figure is 60%. The correlation between penalty cost and class distribution is explained by the differences between the ideal and the real stopping iterations. These differences are always lower for imbalanced than for balanced data (Table 7.16). As a consequence the corresponding penalty cost is also lower for imbalanced data.

Table 7.16: Percentage of instances in W that are queried before stopping

Distribution	Ideal e	Ideal kc	hcw stop	Diff. to ideal e	Diff. to ideal kc
Balanced	0.69	0.23	0.58	-0.11	0.35
Imbalanced	0.60	0.56	0.63	0.03	0.07

The dominance of *hcw* w.r.t. known classes penalty cost is not empirically supported by our results (Table 7.17). Despite the fact that *hcw* and *cgr* still present the lowest mean cost, we cannot reject the null hypothesis of equal means in none of the t-tests performed to evaluate the mean penalty cost of *hcw* in comparison with the other (Table 7.18).

Table 7.17: Criteria rank of penalty cost w.r.t. known classes

Criteria	$\overline{Cost}$
cgr	104.4
hcw	126.6
hck	133.6
sew	134.0
cdrp	139.5
sek	146.8
mine	146.8
cvar	152.1
maxc	164.4
ovru	177.5

7.4.3 Sensitivity analysis of penalty cost

For the cost analysis of the stopping criteria presented in Section 7.4.2 we have assumed that the cost of querying, A, and the opportunity cost of not querying, B – the utility of one unlabeled instance when stopping earlier than the ideal – were equal. However, while the cost of querying is constant as the learning process iterates – it only depends on the cost of service of the oracle

Table 7.18: P-value for equal medians of known classes penalty cost w.r.t. *hcw*

Criteria	p-value
ovru	0.212
maxc	0.336
cgr	0.422
sek	0.476
cvar	0.533
mine	0.595
hck	0.679
sew	0.734
cdrp	0.757

– the utility of unlabeled instances is not. The utility of unlabeled instances is higher before the ideal stopping iteration than after. Stopping at iteration i, before the ideal stopping point, s, means that we would still be able to improve the performance of h_i by querying $s-i$ more unlabeled instances. Then, those last $s-i$ unlabeled instances have some utility since they would add value to the current solution. Moreover, querying after the ideal iteration to stop, s, will not add any value to h_s, otherwise h_s would not be the ideal hypothesis and so s would not be the ideal iteration to stop.

Wasting the utility, B, of potentially useful unlabeled instances that are not queried has an impact on the penalty cost of extemporaneous stopping. We have analyzed the sensitivity of the penalty cost w.r.t. the utility of unlabeled instances by computing total penalty cost for different ratios of B, the opportunity cost of not querying, by A, the querying cost, ranging from 0.1 – the cost of querying is 10 times the utility of one unlabeled instance – to 10 – the utility is 10 times the cost of querying.

Both our hybrid criteria – *hcw* and *hck* – always outperform the other ones concerning the penalty cost w.r.t. error. Thus, we have focused on the analysis of penalty costs related to known classes – stopping before or after knowing all the target classes. We have computed total cost when the opportunity cost of not querying, B, varied in $\{1,2,4,10\}$ while the cost of querying was kept constant at $A=1$ – covering a range of B/A between 1 and 10 – and the inverse, $B=1$ and $A \in \{1,2,4,10\}$ – covering a range of B/A between 0.1 and 1. The evolution of total costs for each stopping criterion as B/A changes is presented in Figure 7.1 (cost unit is the cost of one query).

We may observe high sensitivity to changes in the relative costs of querying or not. When

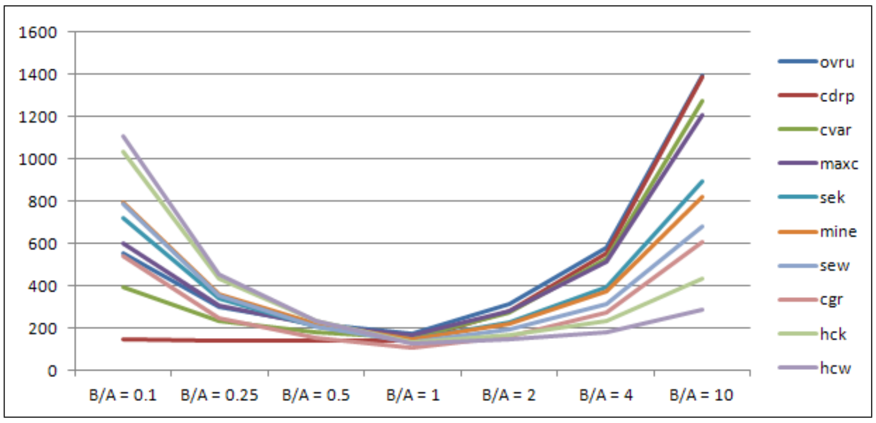


Figure 7.1: Total cost sensitivity to B/A, w.r.t. known classes

the opportunity cost of not querying is greater than or equal to the cost of querying, $B/A \geq 1$, the stopping criteria with best performance w.r.t. error are also top performers w.r.t. known classes. This costs balance is the common setting given our goals. We are focused in the first place on the early identification of classes which is coherent with the assumption that the query cost is lower than the opportunity cost of not querying. Stopping before the ideal w.r.t. known classes prevents the identification of all target classes, our main goal.

To evaluate the statistical significance of the changes in penalty costs we have performed t-tests for equal means of total penalty w.r.t. known classes for all stopping criteria in comparison to *hcw*. The corresponding p-values are presented in Table 7.19.

Table 7.19: P-value of t-test for equal means of penalty cost w.r.t. *hcw*

Criteria	B/A = 0.1	B/A = 0.25	B/A = 1	B/A = 4	B/A = 10
ovru	0.080	0.227	0.212	0.003	0.001
cdrp	0.001	0.009	0.757	0.006	0.001
cvar	0.015	0.061	0.533	0.009	0.003
maxc	0.114	0.251	0.336	0.011	0.006
sek	0.025	0.100	0.476	0.03	0.016
mine	0.209	0.373	0.595	0.121	0.081
sew	0.014	0.052	0.734	0.092	0.054
cgr	0.027	0.043	0.422	0.223	0.094
hck	0.450	0.628	0.679	0.347	0.299

Although *hcw* consistently outperforms *hck* regarding the penalty cost w.r.t. known classes, the difference is never statistically significant. As previously noticed, for a cost ratio of $B/A = 1$, none of the stopping criteria dominates all the other w.r.t. known classes. As expected, since *hcw* has a tendency to stop after having identified all target classes while other criteria stop before, when the cost ratio, B/A, decreases below 1 – when the cost of querying is assumed to be higher than the utility of one unlabeled instance – *hcw* gets dominated by its baseline criteria, *sew* and *cgr*, and also by *sek*, *cdrp* and *cvar*. At a cost ratio $B/A = 0.1$ there is still statistical evidence, at a significance level $\alpha = 10\%$, to conclude that the mean of total penalty costs incurred by *hcw*, *hck*, *mine* and *maxc* are similar.

The dominance of *hcw* is evident when the costs ratio is bigger than unity – when the utility of one unlabeled instance is higher than the cost of querying. In fact, at a costs ratio of $B/A = 4$ only *mine* and *cgr* incur in penalty costs that are comparable to those incurred by our hybrid approaches. At a costs ratio of $B/A = 10$ we have statistical evidence to reject H_0, with $\alpha = 10\%$, for all stopping criteria except *hck*.

7.4.4 Penalty cost space

For an holistic performance view we have represented the evaluated stopping criteria in what we call the *penalty cost space* (Figure 7.2). Our penalty cost space is a two-dimensional Cartesian space formed by penalty cost w.r.t. error and penalty cost w.r.t. known classes. The penalty cost space is divided in quadrants defined by halving the range of observed penalty costs in each dimension.

When the learning task aims mainly at error we should select a stopping criteria from quadrants II or III. When we are mainly concerned with known classes, we should use a stopping criteria in quadrants III or IV. Top performers – those achieving simultaneously lower penalty costs at both perspectives – are located in quadrant III while the worse performers will be located in quadrant IV.

In these charts we have used a square to represent the single stopping criteria we are proposing – *chr*, *sew* and *sek* – and a triangle to represent hybrid stopping criteria – *hcw* and *hck*. The stopping criteria from other authors are represented by circles. Colors are used to distinguish them. The full list of symbols used for representing stopping criteria is provided in Figure 7.3

The performance of the stopping criteria under evaluation for different B/A is depicted in Figures 7.4a to 7.4c. The chart in Figure 7.4d represents the evolution of penalty costs when the ratio B/A changes from 0.1 to 1 and to 10. The size of the symbols representing stopping criteria is

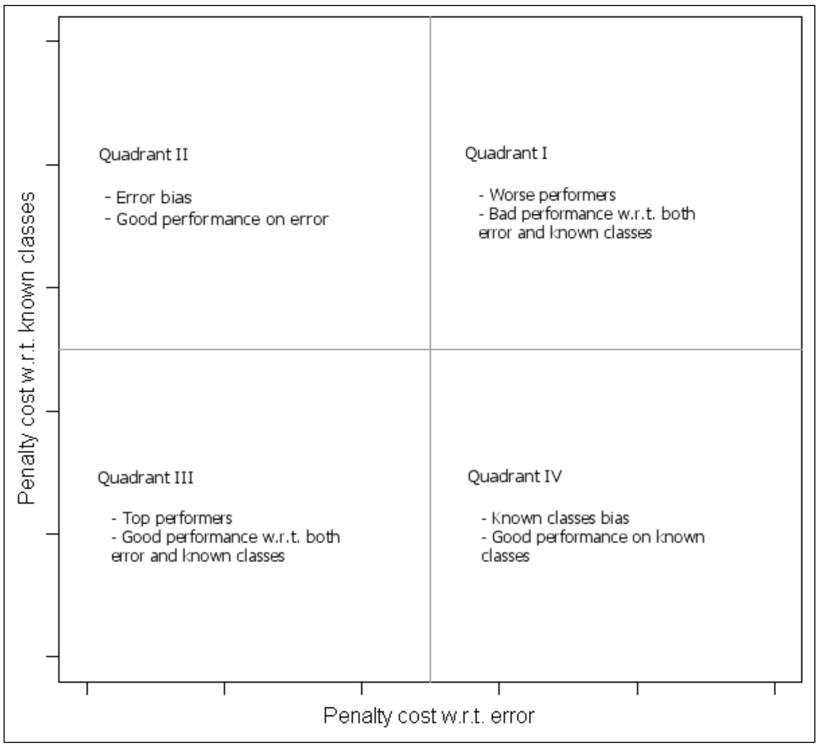

Quadrant II
- Error bias
- Good performance on error
Quadrant I
- Worse performers
- Bad performance w.r.t. both error and known classes
Quadrant III
- Top performers
- Good performance w.r.t. both error and known classes
Quadrant IV
- Known classes bias
- Good performance on known classes
Penalty cost w.r.t. known classes
Penalty cost w.r.t. error

Figure 7.2: Penalty cost space

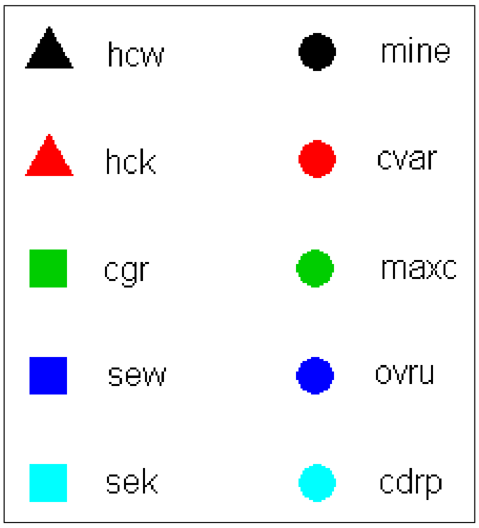
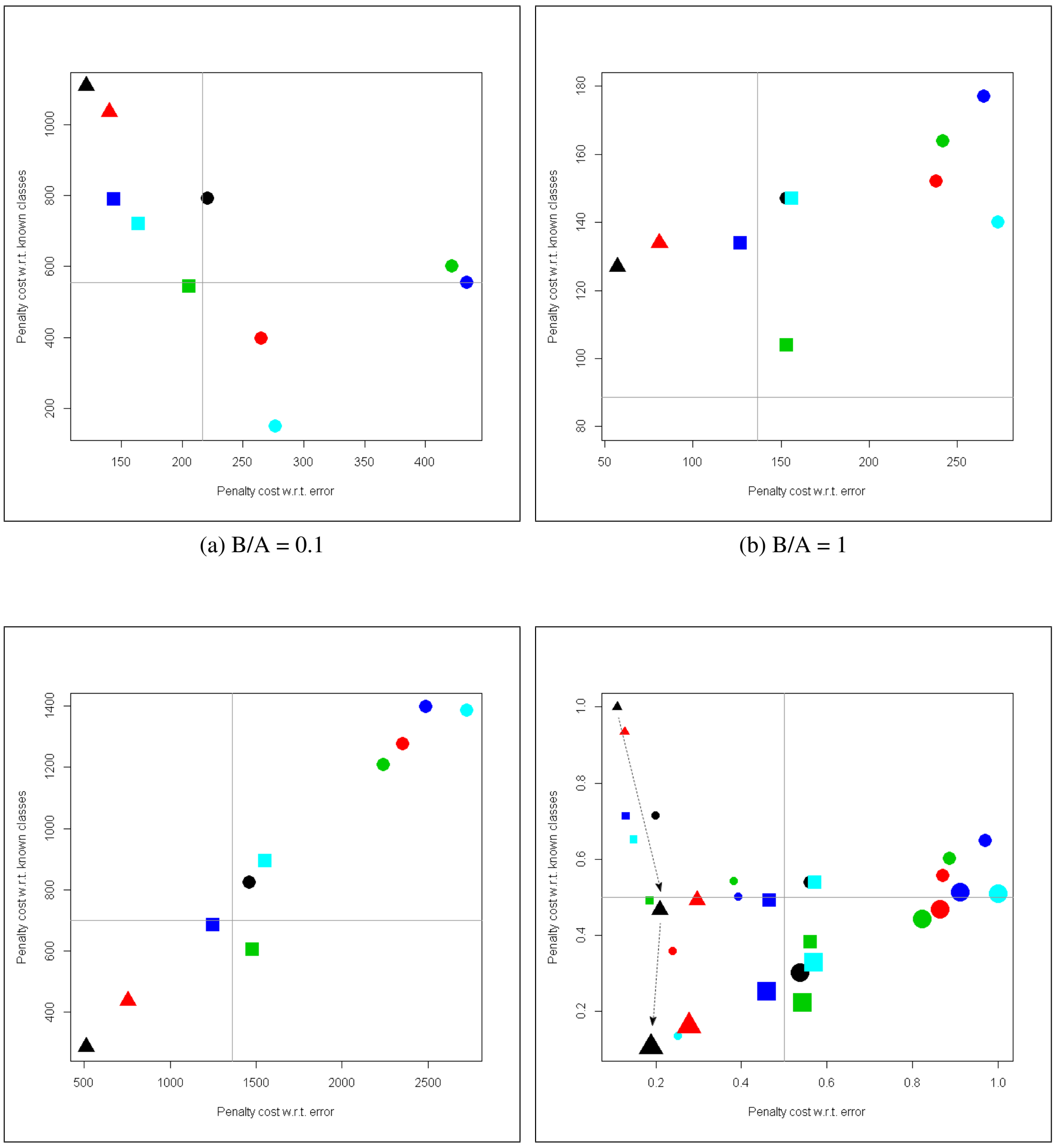


Figure 7.3: Symbols representing stopping criteria

indicative of the magnitude of B/A – the smaller symbols stands for penalty costs observed when $B/A = 0.1$. In this chart we have normalized the penalty costs for each different B/A – all the penalty costs observed at a given B/A were divided by the corresponding maximum. For a better understanding we have illustrated the path of *hcw* as B/A increases.



(a) B/A = 0.1
(b) B/A = 1
(c) B/A = 10
(d) Path as B/A changes, costs normalized to maximum

Figure 7.4: Cost sensitivity to B/A

In seven out of the nine scenarios in Table 7.20 *hcw* is always the top performer. In the remaining scenarios, that is when $B/A = 0.1$ and when considering total penalty cost or known classes penalty cost, *cgr* is the best performer. The best performer in all scenarios with no exception is always one of the stopping criteria proposed by us – either *hcw* or cgr. Only at the $B/A = 0.1$ scenario relating to known classes penalty cost there are two criteria not being proposed by us – $cdrp$ and $cvar$ – among top performers in quadrant I (QI).

Table 7.20: Stopping criteria top performers

Penalty ratio	Total cost	Error cost	Known classes cost
B/A = 0.1	**cgr**	**hcw**	**cgr**
		hck	cdrp
		sew	cvar
		sek	
		cgr	
B/A = 1	**hcw**	**hcw**	**hcw**
	hck	**sew**	**hck**
		hck	
		cgr	
B/A = 10	**hcw**	**hcw**	**hcw**
	hck	**hck**	**hck**
	sew	**sew**	**cgr**
			sew

As the opportunity cost of not querying – the utility of one unlabeled instance – comes more important when compared to the querying cost, the performance of both *hcw* and *hck* moves from QII to QIII (Figure 7.4d). Both these criteria present low cost w.r.t. error, irrespectively of the B/A ratio. Their performance is better when the utility of unlabeled instances is more important that the cost of querying.

7.4.5 An alternative perspective to cost

The impact that the nature of the problem, regarding utility and cost of querying, has on the cost incurred by the stopping criterion in use can be analyzed from a different point of view. Instead of considering the rate of the utility of one unlabeled instance by the cost of querying we may consider an alternative perspective and analyze the rate of negative – early stop – by positive – late stop – differences between the real stopping iteration and the ideal one.

The average of these differences (Table 7.21) computed over the 20 datasets used for evaluation, show that all stopping criteria have a tendency to stop earlier than the ideal w.r.t. error by a factor of two at least. When regarding the number of known classes, a similar trend is observed, although not as salient, for all stopping criteria except *hcw* and *sew*. In fact, both *hcw* and *sew* tend to stop querying only after having identified all the target classes. This is very relevant considering our concern with the identification of exemplary instances from all classes.

Also from this point of view, *hcw* demonstrates good performance. *hcw* has the smallest ratio of negative by positive differences to the ideal stopping point w.r.t. error. This means that although there is a tendency to stop earlier than the ideal, the bias is not as severe as with the other criteria. *hcw* also performs well w.r.t. known classes, being second only to *sew*. It is one of the only two criteria that, on average, stop after identifying all the classes.

Table 7.21: Ratio between average negative by average positive differences to ideal stop

Criteria	Error	Known classes
hcw	2.381	0.918
maxc	2.783	1.027
ovru	3.298	1.382
hck	3.802	1.344
mine	4.852	1.573
cgr	6.241	2.118
sew	7.114	0.840
cvar	8.711	1.148
sek	19.123	1.306
cdrp	28.679	6.940

7.4.6 Robustness

So far we have studied the performance of the stopping criteria based on the first time the stopping condition is satisfied. But this first signal may be spurious. It might happen that there is a big gap between the iteration when the first stop sign is triggered and the next. To evaluate the robustness of the stopping criteria we have computed how many iterations are required to get the third and the fifth stop signal after the first one was triggered. The number of iterations between the first stop sign and the third and fifth, in Figure 7.5, are the average of these indicators computed over the datasets where all the stopping criteria have triggered the stop sign at least five times – Iris, Satlog, Balance, Car, Robot and Nursery.

Stopping criteria *sew*, *cdrp*, *hcw*, *sek* and *hck* are robust, consistently triggering stop signs after the first. The first stop sign triggered by *ovru*, *cvar* and *mine* seems to be incidental since there is a big gap between the iterations when the stop sign is triggered for the first and third times. When using *maxc* or *cgr* the first three stop signs are close to each other but the fourth and the

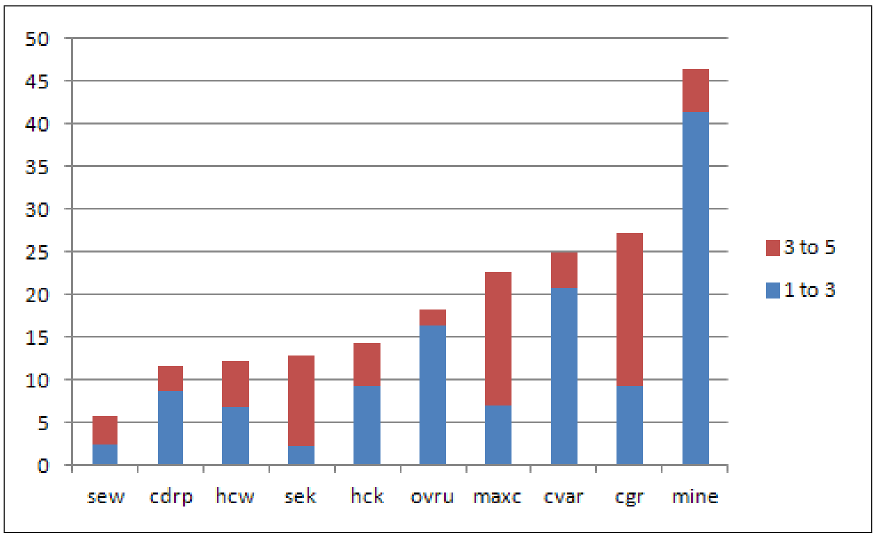


Figure 7.5: Mean number of iterations, after the first stop has been triggered, required to trigger the stop sign for the third and fifth time

fifth require a significant number of queries to be triggered. The stopping criteria with the worse robustness performance is *mine* taking much more iterations than the rest to trigger after the first time. Those stopping criteria that consistently trigger stop signs are more reliable than those that trigger occasionally.

7.4.7 Utility as performance loss

One last concern relates to the fact that we have assumed so far that the utility of one unlabeled instance – the opportunity cost of not querying, B – is constant during all the learning cycle, independently of the added value that it might bring to the current hypothesis. In fact, this utility may also be related to the potential improvement that additional queries may add to the hypothesis as more data is available to the classifier. The potential improvement that unlabeled instances in U_i may add is the performance loss of stopping at iteration i. From this point of view the utility is not constant.

As the learning process iterates, more labeled instances are available and each new hypothesis is built with more information about the target concept. Besides, AL is expected to add to the training set the most informative instances available at each iteration. As a consequence, it is expected that the utility of the remaining unlabeled instances decreases as the learning process iterates and the unlabeled pool gets reduced to a set of redundant instances – or at least not as informative.

The potential improvement – or performance loss – was computed for error and for known classes. The potential improvement w.r.t. error (Table 7.22) was estimated by the average of the differences between the error observed at the iteration triggering the stop sign and minimum error. Potential improvement regarding known classes (Table 7.23) is the average of the differences between the number of classes in the dataset and the number of classes known at the stopping iteration. These averages were computed over the six datasets where all the stopping criteria trigger before exhausting the working set.

Table 7.22: Query utility as potential error improvement

Dataset	Ideal error	cdrp	ovru	maxc	mine	cvar	**cgr**	**sew**	**sek**	**hcw**	**hck**
Iris	0.033	0.022	0.016	0.009	0.007	0.331	0.022	0.051	0.018	0.022	0.022
Satlog	0.141	0.543	0.007	0.011	0.02	0.421	0.021	0.059	0.043	0.021	0.021
Balance	0.065	0.353	0.041	0.011	0.049	0.567	0.054	0.301	0.046	0.054	0.046
Car	0.08	0.247	0.316	0.047	0.211	0.247	0.157	0.013	0.173	0.013	0.157
Robot	0.195	0.360	0.025	0.016	0.039	0.352	0.071	0.253	0.163	0.017	0.071
Nursery	0.107	0.532	0.567	0.007	0.019	0.383	0.179	0.383	0.217	0.179	0.179
average		0.343	0.162	0.017	0.057	0.384	0.084	0.177	0.11	0.051	0.083

Table 7.23: Query utility as potential known classes improvement

Dataset	#classes	cdrp	ovru	maxc	mine	cvar	**cgr**	**sew**	**sek**	**hcw**	**hck**
Iris	3	0	0	0	0	0	0	0	0	0	0
Satlog	6	1.6	0	0	0	0.8	0	0	0	0	0
Balance	3	0.1	0	0	0	0.2	0	0.1	0	0	0
Car	4	1.9	2	0.3	1.9	1.9	1	0.2	1	0.2	1
Robot	4	0.7	0	0	0	0.5	0	0	0	0	0
Nursery	5	2.2	2.8	0.9	1	1.9	1.5	1.9	1.8	1.5	1.5
average		1.08	0.80	0.20	0.48	0.88	0.42	0.37	0.47	0.28	0.42

Apparently *maxc* is the stopping criterion that takes better advantage of the utility of unlabeled instances since it stops when the utility of the remaining instances is the lowest among all – regarding error as well as known classes. However, this utility indicator must not be analyzed on its own. A trivial optimal solution, strictly from the performance loss point of view, is to stop when only one instance in the working set remains unlabeled. This instance will have the lowest utility thus, top performance from this point of view. However, such a stopping criterion is not adequate for its potentially high penalty cost. Utility, defined this way, must be complemented with the gap between real and ideal stop iterations. From this perspective *maxc* is a poor performer being the second last one w.r.t. known classes (Table 7.17) and the last but two w.r.t. error (Table 7.14).

From all the other, *hcw* exhibits the best performance also from the perspective of this utility indicator – stopping when the utility of the remaining unlabeled instances is low. The difference to *maxc* is that this evidence on the merits of *hcw* is supported by the distance between real and ideal stop iterations – *hcw* is first for error (Table 7.14) and second for known classes (Table 7.17).

7.5 Prevailing outcomes

In general the stopping criteria proposed in this thesis outperform state-of-the-art proposals. Our hybrid criteria, *hcw* and *hck*, are the top performers on the generality of the evaluation scenarios with advantage to *hcw*.

Stopping before or after the ideal Although in general the stopping criteria under evaluation have a tendency to stop querying before the ideal, mainly regarding error, *hcw* and *hck* are those that stop after the ideal iteration more consistently over the datasets used for evaluation. This behavior is particularly significant regarding known classes. Stopping only after having retrieved exemplary instances from all the target classes is a core characteristic given our goals that is clearly present in *hcw* and *hck* but not in the rest.

Penalty cost of stopping out of time From the point of view of penalty costs, *hcw* also outperforms the other stopping criteria. *hcw* dominates all the other criteria – except *hck* – when analyzing penalty costs w.r.t. error. The same dominance is observed regarding penalty costs related to known classes except when the cost of querying is assumed to be higher than the opportunity cost of not querying. However, such a relation of costs is contradictory to our goals – identifying exemplary instances from all target classes.

Robustness A group of stopping criteria, including *hcw*, *hck*, *sew*, *sec* and *cdrp*, trigger stopping signs consistently being more reliable than the rest. These stopping criteria trigger consecutive stop signs after the first while the rest trigger occasionally which indicates that the first stop signs triggered by those stopping criteria are casual. The robustness and reliability of *hcw*, *hck*, *sew*, *sec* and *cdrp* clearly outperforms the other.

Potential improvement utility *hcw* also outperforms the other criteria when we analyze the utility of unlabeled instances as the potential improvement they may add to the current hypothesis. Although *maxc* takes better advantage of this utility, this is a trivial solution that does not take into consideration the cost of querying, one of our main concerns.

For all these results, *hcw*, exhibiting the most favorable characteristics regarding our research goals, seems to be the most appropriate choice. The combination of low classification gradient and label distribution steadiness provides efficient stopping criteria.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

In general the results from our research support the hypotheses under investigation. An active learning (AL) strategy guided by d-Confidence – an aggregation of distance and confidence – is able to build accurate classifiers covering all the target classes with a reduced cost when compared to state-of-the-art approaches. The *hcw* stopping criterion – an hybrid criterion based on the classification gradient and the stability of the distribution of predictions – provides precise and robust indications to stop querying, outperforming current stopping criteria.

Building accurate, class-complete classifiers at reduced labeling effort D-Confidence is an effective query selection criterion for AL that retrieves exemplary instances from all the target classes in an early stage of the learning process outperforming common AL approaches in use (Figure 6.15). The stopping criterion based on the ensemble of classification gradient and steady entropy mean, *hcw*, triggers robust indications to halt the AL process at appropriate stages of the leaning process taking into account both the cost of querying and the utility of the remaining unlabeled instances (Table 7.12). D-Confidence and *hcw* together constitute an efficient and efficacious AL strategy to build accurate classifiers with a comprehensive coverage of the target classes at low cost, that is, requiring fewer queries than state-of-the-art approaches.

Although d-Confidence is tailored to use SVM base classifiers, it is also effective with other base classifiers – particularly neural networks and decision trees – assuring significant reduction in the labeling effort in general (Table 6.9). In such cases, however, the accuracy is not as good as when using SVM base classifiers (Table 6.12 and Figure 6.8). The base classifier used in the learning process has some influence on accuracy but apparently not on the labeling effort required to identify instances covering all target classes.

The reduction in the labeling effort achieved by d-Confidence, when compared to other state-of-the-art AL strategies, is particularly noticeable in retrieving instances from under-represented classes in highly imbalanced data. However, under such circumstances, this improvement is in some cases achieved at the cost of a marginal reduction of accuracy. This pattern is observed in high-dimensional unstructured data as well as on low-dimensional tabular data (Figures 6.8, 6.10 and 6.15).

The exploratory bias that d-Confidence generally exhibits at an early stage of the learning process may cause exchanging accuracy for representativeness when in presence of imbalanced data. Nevertheless, a marginal reduction in accuracy may be acceptable, mainly in those scenarios where the early identification of minority classes is critical – such as, clinical and financial data. D-Confidence provides significant improvements over state-of-the-art AL strategies. mainly in such scenarios.

Shifting between exploration and exploitation modes D-Confidence dynamically shifts its operative mode between exploration and exploitation. This shift is conducted by the geometric properties and by the class distribution in the labeled set used to train, L, and also by the geometric properties of the unlabeled set available to search for queries, U. The exploratory potential of d-Confidence is boosted when the variance of the distances between unlabeled instances in U and known classes in L is higher than that of confidence. Otherwise the exploitation behavior of d-Confidence is enhanced (Section 6.5). This shift occurs as a natural consequence of changes in the composition of U and L at each iteration of the learning process. Shifting between exploration and exploitation is an automatic unsupervised process that does not require any pre-tuning nor additional costs, contributing to improve the efficiency of d-Confidence.

Stop querying The hybrid stopping criterion aggregating classification gradient and steady entropy mean, *hcw*, consistently outperforms other common stopping criteria. It presents the lowest penalty cost w.r.t. both error and known classes except when the cost of querying is assumed to be much higher – one order of magnitude – than the opportunity cost of not querying (Tables 7.12 and 7.13, Figure 7.1). Such a cost value relationship, however, is not expected in our setting, mainly during the early stage of the learning process when still searching for exemplary instances to cover all target classes.

Both hybrid criteria proposed in this thesis, *hcw* and *hck*, have a tendency to halt the learning process only after all the target classes are identified, in contrast to the rest. This behavior significantly contributes to our goals when the cost of querying is assumed to be lower that the opportunity cost of not querying. Besides, *hcw* is a stable stopping criteria triggering even stop signs (Section 7.4.6).

The *hcw* criterion is also the one that takes most advantage of the potential improvement that a new query may add to the current classifier.

Early class exposure In general, d-Confidence exhibits significantly lower LDC – minimum number of queries required to retrieve exemplary instances from all target classes – than other state-of-the-art approaches. Moreover, d-Confidence is particularly fit to retrieve exemplary instances from minority classes (Tables 6.11 and 6.12, Figure 6.12). This is a relevant characteristic of d-Confidence that makes it particularly suited for domains with imbalanced class distributions. These outcomes naturally arise from the efficient coverage of the input space as provided by d-Confidence (Table 6.3).

Early class disclosure and fast instance space coverage are in part due to the inclusion of the distance factor in d-Confidence. This factor adds an exploratory bias, particularly influential when instance space still has unexplored clusters contributing to a high variance of the distance between labeled and unlabeled instances (Section 6.5).

Effectiveness assessment The evaluation of our proposal was in part supported by performance indicators that are typical in classification tasks, like error rate. However, these indicators do not provide enough information w.r.t. to our specific goals. Mainly class exposure is not addressed by performance measures focused on error. We propose three novel performance indicators for AL, known classes, first-hit and LDC, addressing class exposure (Section 6.3.1).

8.2 Summary of contributions

The main contributions of this work include:

(1) D-Confidence, a query selection criterion for AL, combining distance and confidence, that in general outperforms state-of-the-art approaches in building accurate class-complete classifiers (Tables 6.18 and 6.19). D-Confidence has top performance in the early identification of representative instances of all target classes, mainly on imbalanced data (Figures 6.12 and 6.14). In general, it reduces the labeling effort required to build classifiers with a particular benefit in domains where under-represented classes are critical.

(2) The hybrid stopping criterion, *hcw*, triggers halt signals when the utility of unlabeled instances is low (Tables 7.22 and 7.23). This stopping criterion takes best advantage of the utility of unlabeled instances in a robust fashion, consistently triggering stop signs at proper stages of the learning process (Section 7.4.6). *hcw* consistently outperforms other common stopping criteria w.r.t. the cost value relationship (Figure 7.4).

(3) A learning strategy supporting the full AL process. This strategy includes not only the query selection criterion, d-Confidence, but also the *hcw* stopping criterion that halts the learning process when no further improvements are expected. In the absence of a reliable stopping criteria, all the gains obtained from a proper selection criterion may be jeopardized since we will keep incurring in the cost of querying the oracle with ever lower return.

(4) Definition of a set of indicators – known classes, first-hit and LDC – to evaluate the performance of the AL process w.r.t. the coverage of target classes (Section 6.3.1).

(5) A comprehensive cost/benefit analysis of stopping criteria having into consideration the cost of querying and the opportunity cost of not querying (Section 7.4).

(6) A formal definition of a specific classification problem setup that is of particular relevance when the cost of querying is high, unlabeled data is easily available, class distribution is severely imbalanced and under-represented classes are critical. This formal problem statement (Section 5.4) provides a common background to support further investigation.

(7) The development of several applications of d-Confidence included in research projects, MSc theses and BSc capstone projects (Section 6.7).

8.3 Future work

There are several opportunities for further research on the d-Confidence strategy aimed to build accurate classifiers being aware of all target classes at low cost.

Semi-supervised d-Confidence In AL the training set is built iteratively by sequentially labeling and adding the most informative instances given the current hypothesis and available data. The least informative instances – those that are not expected to induce significant changes to the current hypothesis if labeled and added to the training set – are not deemed as relevant. However, these might reinforce the description of target classes that is available to train, contributing to improve the classifier accuracy. With semi-supervised techniques this added value might be obtained at no additional cost. Merging semi-supervised techniques in our AL strategy aiming to leverage the intrinsic value of highly confident unlabeled instances at no cost might improve accuracy without increasing labeling cost.

Representativeness and retrieval When the representativeness assumption fails to hold, the working set will not contain instances from all the target classes. In such circumstances, a potential source of instances from the target domain may be used to search and retrieve additional instances to complement the working set and fill the gap. This seems particularly relevant in text categorization given the amount of text information available in the Web which makes it a potential source for text instances on almost any topic. Developments in this matter may also contribute to reduce the computational cost of d-Confidence by deliberately using just a subset of the available data which is then updated with new instances on a need-to-know basis.

Distance computation Computing distances between all pairs of instances in the working set as required by d-Confidence demands for significant computational resources and is not scalable. This effort can be reduced by first pre-selecting a subset of instances being representative of the geometric structure of the working set and then extrapolating these results to compute d-Confidence.

Initialization D-Confidence is initialized by a set of two instances selected at random from the working set. When building this initialization set, our only concern was to initialize at the minimum cost. The potential utility of the instances selected to initialize was not considered. It might happen that the additional cost eventually required to improve the initialization set may payoff by creating conditions for a faster convergence and a lower LDC. This problem is being investigated by our colleagues at the University of São Paulo (Motta et al., 2012) who are analyzing the potential of complex network properties for this purpose.

References

Kjersti Aas and Line Eikvil. Text Categorisation: A Survey. Technical report, Norwegian Computing Center, Oslo, NO, 1999.

Naoki Abe and Hiroshi Mamitsuka. Query learning strategies using boosting and bagging. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 1–9, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc. ISBN 1-55860-556-8.

Giordano Adami, Paolo Avesani, and Diego Sona. Clustering documents into a web directory for bootstrapping a supervised classification. *Data & Knowledge Engineering*, 54:301–325, 2005.

Charu C. Aggarwal, Alexander Hinneburg, and Daniel A. Keim. On the surprising behavior of distance metrics in high dimensional spaces. In *Proceedings of the 8th International Conference on Database Theory*, ICDT '01, pages 420–434, London, UK, 2001. Springer-Verlag. ISBN 3-540-41456-8.

Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, pages 487–499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc. ISBN 1-55860-153-8.

Constantin F. Aliferis, Alexander Statnikov, Ioannis Tsamardinos, Subramani Mani, and Xenofon D. Koutsoukos. Local Causal and Markov Blanket Induction for Causal Discovery and Feature Selection for Classification Part I: Algorithms and Empirical Evaluation. *J. Mach. Learn. Res.*, 11:171–234, March 2010. ISSN 1532-4435.

Tanner K. Allen D. Infusing active learning into the large-enrollment biology class: seven strategies, from the simple to complex. *Cell Biology Education*, 4:262–268, 2005.

Dana Angluin. Queries and concept learning. *Machine Learning*, 2:319–342, 1988. ISSN 0885-6125.

Ron Artstein and Massimo Poesio. Inter-coder agreement for computational linguistics. *Comput. Linguist.*, 34(4):555–596, December 2008. ISSN 0891-2017.

Robert D. Atkinson and Daniel D. Castro. Digital quality of life. Technical report, Information Technology and Innovation Foundation, October 2008.

Josh Attenberg and Foster Provost. Inactive learning?: difficulties employing active learning in practice. *SIGKDD Explor. Newsl.*, 12(2):36–41, March 2011. ISSN 1931-0145.

Ricardo Baeza-Yates and Gonzalo Navarro. Integrating contents and structure in text retrieval. *SIGMOD Rec.*, 25:67–79, March 1996. ISSN 0163-5808.

Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval.* Addison Wesley, 1999.

Maria-Florina Balcan, Alina Beygelzimer, and John Langford. Agnostic active learning. In *ICML*, pages 65–72. ICML, 2006.

Yoram Baram, Ran El-Yaniv, and Kobi Luz. Online choice of active learning algorithms. *J. Mach. Learn. Res.*, 5:255–291, December 2004. ISSN 1532-4435.

Sugato Basu, Arindam Banerjee, and R. Mooney. Semi-supervised clustering by seeding. In *Proceedings of 19th International Conference on Machine Learning (ICML-2002*, 2002.

Sugato Basu, A. Banjeree, ER. Mooney, Arindam Banerjee, and Raymond J. Mooney. Active semi-supervision for pairwise constrained clustering. In *Proceedings of the 2004 SIAM International Conference on Data Mining (SDM-04)*, pages 333–344, 2004.

Eric Baum. Neural net algorithms that learn in polynomial time from examples and queries. *IEEE Transactions in Neural Networks*, 2:5–19, 1991.

Markus Becker and Miles Osborne. A two-stage method for active learning of statistical grammars. In *Proceedings of the 19th international joint conference on Artificial intelligence*, IJCAI'05, pages 991–996, San Francisco, CA, USA, 2005. Morgan Kaufmann Publishers Inc.

Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *J. Mach. Learn. Res.*, 7:2399–2434, December 2006. ISSN 1532-4435.

Richard Ernest Bellman. *Dynamic programming.* Princeton University Press, 1957.

Kristin P. Bennett and Ayhan Demiriz. Semi-supervised support vector machines. In *Advances in Neural Information Processing Systems*, pages 368–374. MIT Press, 1998.

Giacomo Berardi, Andrea Esuli, and Fabrizio Sebastiani. A utility-theoretic ranking method for semi-automated text classification. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '12, pages 961–970, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1472-5.

Stefan Berchtold, Christian Böhm, Daniel A. Keim, and Hans-Peter Kriegel. A cost model for nearest neighbor search in high-dimensional data space. In *Proceedings of the sixteenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, PODS '97, pages 78–86, New York, NY, USA, 1997. ACM. ISBN 0-89791-910-6.

James C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms.* Kluwer Academic Publishers, Norwell, MA, USA, 1981. ISBN 0306406713.

Michael Bloodgood and K. Vijay-Shanker. A method for stopping active learning based on stabilizing predictions and the need for user-adjustable stopping. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, CoNLL '09, pages 39–47, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-29-9.

Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, COLT' 98, pages 92–100, New York, NY, USA, 1998. ACM. ISBN 1-58113-057-0.

Igor A. Bolshakov and Alexander Gelbukh. Synonymous paraphrasing using wordnet and internet. In Farid Meziane and Elisabeth Métais, editors, *Natural Language Processing and Information Systems*, volume 3136 of *Lecture Notes in Computer Science*, pages 312–323. Springer Berlin Heidelberg, 2004. ISBN 978-3-540-22564-5.

Charles C. Bonwell and James A. Eison. *Active Learning: Creating Excitement in the Classroom.* Jossey-Bass, 1991.

M. R. Boutell, J. Luo, X. Shen, and C. M. Brown. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771, 2004.

Craig Boutilier, Richard S. Zemel, and Benjamin Marlin. Active collaborative filtering. In *Proceedings of the Nineteenth Annual Conference on Uncertainty in Artificial Intelligence*, pages 98–106, 2003.

Andrew P. Bradley. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recogn.*, 30(7):1145–1159, July 1997. ISSN 0031-3203.

Leo Breiman, Jerome Friedman, Charles J. Stone, and R. A. Olshen. *Classification and Regression Trees.* Chapman and Hall/CRC, 1 edition, January 1984. ISBN 0412048418.

Henry Brighton and Chris Mellish. Advances in instance selection for instance-based learning algorithms. *Data Min. Knowl. Discov.*, 6:153–172, April 2002. ISSN 1384-5810.

Klaus Brinker. Incorporating diversity in active learning with support vector machines. In *Proceedings of the Twentieth International Conference on Machine Learning*, 2003.

Klaus Brinker. On active learning in multi-label classification. In Myra Spiliopoulou, Rudolf Kruse, Christian Borgelt, Andreas Nürnberger, and Wolfgang Gaul, editors, *From Data and Information Analysis to Knowledge Engineering*, Studies in Classification, Data Analysis, and Knowledge Organization, pages 206–213. Springer Berlin Heidelberg, 2006. ISBN 978-3-540-31314-4. 10.1007/3-540-31314-1_24.

Nicola Cancedda, Eric Gaussier, Cyril Goutte, and Jean Michel Renders. Word sequence kernels. *The Journal of Machine Learning Research*, 3:1059–1082, March 2003. ISSN 1532-4435.

Rich Caruana. Multitask learning. *Mach. Learn.*, 28:41–75, July 1997. ISSN 0885-6125.

Rui M. Castro and Robert D. Nowak. Minimax bounds for active learning. In *Proceedings of the 20th annual conference on Learning theory*, COLT'07, pages 5–19, Berlin, Heidelberg, 2007. Springer-Verlag. ISBN 978-3-540-72925-9.

William B. Cavnar and John M. Trenkle. N-gram-based text categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175, Las Vegas, US, 1994.

Nicolas Cebron and Michael Berthold. Active learning for object classification: from exploration to exploitation. *Data Mining and Knowledge Discovery*, 18:283–299, 2009. ISSN 1384-5810. 10.1007/s10618-008-0115-0.

Nicolas Cebron, Michael R. Berthold, Universität Konstanz, Nicolas Cebron, and Michael R. Berthold. An adaptive multi objective selection strategy for active learning, 2007.

Nicolò Cesa-bianchi, Gábor Lugosi, and Gilles Stoltz. Minimizing regret with label efficient prediction. *IEEE Trans. Inform. Theory*, 51:77–92, 2005.

Nicolò Cesa-Bianchi, Claudio Gentile, and Luca Zaniboni. Worst-case analysis of selective sampling for linear classification. *J. Mach. Learn. Res.*, 7:1205–1230, December 2006. ISSN 1532-4435.

Soumen Chakrabarti. *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan Kaufmann, Amsterdam, 2003.

Soumen Chakrabarti, Byron Dom, and Piotr Indyk. Enhanced hypertext categorization using hyperlinks. *SIGMOD Rec.*, 27(2):307–318, 1998a. ISSN 0163-5808.

Soumen Chakrabarti, Byron Dom, Prabhakar Raghavan, Sridhar Rajagopalan, David Gibson, and Jon Kleinberg. Automatic resource compilation by analyzing hyperlink structure and associated text. In *Proceedings of the 7th International World Wide Web Conference*, 1998b.

O. Chapelle, B. Schoelkopf, and A. Zien, editors. *Semi-supervised Learning*. MIT Press, Cambridge, MA, 2006.

Olivier Chapelle. Active learning for parzen window classifier. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, 2005.

Gang Chen, Tian jiang Wang, Li yu Gong, and Perfecto Herrera. Multi-class support vector machine active learning for music annotation. *International Journal of Innovative Computing, Information and Control*, 6(3(A)):921 –930, MArch 2010.

Wei Chu, Martin Zinkevich, Lihong Li, Achint Thomas, and Belle Tseng. Unbiased online active learning in data streams. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 195–203, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0813-7.

Amanda Clare and Ross D. King. Knowledge discovery in multi-label phenotype data. In *Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery*, PKDD '01, pages 42–53, London, UK, 2001. Springer-Verlag. ISBN 3-540-42534-9.

William S. Cleveland, Susan J. Devlin, and Eric Grosse. Regression by local fitting: Methods, properties, and computational algorithms. *Journal of Econometrics*, 37(1):87–114, 1988. ISSN 0304-4076.

William G. Cochran. *Sampling Techniques*. John Wiley, 3 edition, 1977.

David Cohn, Les Atlas, and Richard Ladner. Training connectionist networks with queries and selective sampling. In *Advances in Neural Information Processing Systems*, 1990.

David Cohn, Les Atlas, and Richard Ladner. Improving generalization with active learning. *Machine Learning*, 15:201–221, May 1994. ISSN 0885-6125.

David Cohn, Zoubin Ghahramani, and Michael Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145, 1996.

T. Cover and P. Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21 –27, january 1967. ISSN 0018-9448.

J. Cowie and W. Lehnert. Information extraction. *Communications of the ACM*, 39(1):80–91, 1996.

Koby Crammer and Yoram Singer. A family of additive online algorithms for category ranking. *The Journal of Machine Learning Research*, 3:1025–1058, March 2003. ISSN 1532-4435.

W. Bruce Croft. Effective text retrieval based on combining evidence from the corpus and users. *IEEE Expert: Intelligent Systems and Their Applications*, 10:59–63, December 1995. ISSN 0885-9000.

Aron Culotta and Andrew McCallum. Reducing labeling effort for structured prediction tasks. In *Proceedings of the 20th national conference on Artificial intelligence - Volume 2*, pages 746–751. AAAI Press, 2005. ISBN 1-57735-236-x.

Cord M Delany S J Cunningham P. Supervised learning. *Learning*, pages 21–49, 2008.

Charlie K. Dagli. Combining diversity-based active learning with discriminant analysis in image retrieval. In *Proceedings of the Third International Conference on Information Technology and Applications (ICITA'05) Volume 2 - Volume 02*, ICITA '05, pages 173–178, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2316-1.

Wenyuan Dai, Qiang Yang, Gui R. Xue, and Yong Yu. Boosting for transfer learning. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 193–200, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-793-3.

Stephen D'Alessio, Keitha Murray, Robert Schiaffino, and Aaron Kershenbaum. The effect of using hierarchical classifiers in text categorization. In *6th International Conference Recherche d'Information Assistee par Ordinateur (RIAO-00)*, pages 302–313, 2000.

Andrew Moore Dan Pelleg. Active learning for anomaly and rare-category detection. In *Advances in Neural Information Processing Systems 18*, December 2004.

Tuong Dao, Ron Sacks-Davis, and James A. Thom. An indexing scheme for structured documents and its implementation. In *Proceedings of the Fifth International Conference on Database Systems for Advanced Applications (DASFAA)*, pages 125–134. World Scientific Press, 1997. ISBN 981-02-3107-5.

Sanjoy Dasgupta. Coarse sample complexity bonds for active learning. In *Advances in Neural Information Processing Systems 18*. 2005.

Sanjoy Dasgupta and Daniel Hsu. Hierarchical sampling for active learning. In *Proceedings of the 25th International Conference on Machine Learning*, 2008.

Sanjoy Dasgupta and Philip M. Long. Performance guarantees for hierarchical clustering. *J. Comput. Syst. Sci.*, 70:555–569, June 2005. ISSN 0022-0000.

Sanjoy Dasgupta, Adam Tauman Kalai, and Claire Monteleoni. Analysis of perceptron-based active learning. *J. Mach. Learn. Res.*, 10:281–299, June 2009. ISSN 1532-4435.

Michael Davy and Saturnino Luz. Active learning with history-based query selection for text categorisation. In Giambattista Amati, Claudio Carpineto, and Giovanni Romano, editors, *Advances in Information Retrieval*, volume 4425 of *Lecture Notes in Computer Science*, pages 695–698. Springer Berlin / Heidelberg, 2007. 10.1007/978-3-540-71496-5_71.

Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, 1996.

M. Dewey. A classification and subject index for cataloguing and arranging the books and pamphlets of a library. Project Gutenberg EBook, 2004.

Thomas G. Dietterich and Ghulum Bakiri. Solving multiclass learning problems via error-correcting output codes. *J. Artif. Int. Res.*, 2(1):263–286, January 1995. ISSN 1076-9757.

C. Dima, M. Hebert, and A. Stentz. Enabling learning from large datasets: applying active learning to mobile robotics. In *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, volume 1, pages 108 – 114 Vol.1, april-1 may 2004.

Cristian Dima and Martial Hebert. Active learning for outdoor obstacle detection. In Elsevier, editor, *Proc. Science and Systems I*, August 2005.

Sotiris Diplaris, Grigorios Tsoumakas, Pericles Mitkas, and Ioannis Vlahavas. Protein classification with multiple algorithms. In *Advances in Informatics*, pages 448–456. Spring-Verlag, 2005.

Pinar Donmez and Jaime G. Carbonell. Proactive learning: cost-sensitive active learning with multiple imperfect oracles. In *Proceeding of the 17th ACM conference on Information and knowledge management*, CIKM '08, pages 619–628, New York, NY, USA, 2008. ACM. ISBN 978-1-59593-991-3.

Pinar Donmez, Jaime G. Carbonell, and Paul N. Bennett. Dual strategy active learning. In *Proceedings of the 18th European conference on Machine Learning*, ECML '07, pages 116–127, Berlin, Heidelberg, 2007. Springer-Verlag. ISBN 978-3-540-74957-8.

Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons Inc, 1973. ISBN 0471223611.

Susan Dumais, John Platt, David Heckerman, and Mehran Sahami. Inductive learning algorithms and representations for text categorization. In *Proceedings of the seventh international conference on Information and knowledge management*, CIKM '98, pages 148–155, New York, NY, USA, 1998. ACM. ISBN 1-58113-061-9.

Bronwyn Bonnie Eisenberg. On the sample complexity of pac-learning using random and chosen examples. In *Proceedings of the 1990 Workshop on Computational Learning Theory*, pages 154–162. Morgan Kaufmann, 1991.

Seyda Ertekin, Jian Huang, Leon Bottou, and Lee Giles. Learning on the border: active learning in imbalanced data classification. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, CIKM '07, pages 127–136, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-803-9.

Nuno Escudeiro. Automatic web resource compilation using data mining. Master's thesis, Faculdade de Economia, Universidade do Porto, 2004.

Nuno Escudeiro and Paula Escudeiro. Correcção semi-automática de respostas em texto livre. *TICAI2010: TICs para a Aprendizagem da Engenharia ©IEEE, Sociedad de Educación: Capítulos Español y Portugués*, V:137–141, 2011.

Nuno Escudeiro and Alípio Jorge. *Semantics, Web and Mining*, volume 4289 of *LNCS*, chapter Semi-automatic Creation and Maintenance of Web Resources with webTopic, pages 82–102. Springer, Heidelberg, 2006.

Nuno Escudeiro and Alípio Jorge. Learning partially specified concepts with d-confidence. In *Brazilian Simposium on Artificial Intelligence, Web and Text Intelligence Workshop*, 2008.

Nuno Escudeiro and Alípio Jorge. D-confidence: an active learning strategy to reduce label disclosure complexity in the presence of imbalanced class distributions. *Journal of the Brazilian Computer Society*, 18:311–330, 2012. ISSN 0104-6500. 10.1007/s13173-012-0069-3.

Nuno Escudeiro, Paula Escudeiro, and Augusto Cruz. Correcção semi-automática de respostas em texto livre. In *The 9th European Conference on e-Learning*, 2010.

Andrea Esuli and Fabrizio Sebastiani. Active learning strategies for multi-label text classification. In *Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval*, ECIR '09, pages 102–113, Berlin, Heidelberg, 2009. Springer-Verlag. ISBN 978-3-642-00957-0.

Y. C. Fang, S. Parthasarathy, and F. Schwartz. Using clustering to boost text classification. In *Workshop on Text Mining (TextDM'2001)*, 2001.

Marin Ferecatu, Michel Crucianu, and Nozha Boujemaa. Sample selection strategies for relevance feedback in region-based image retrieval. In *Proceedings of the 5th Pacific Rim Conference on Advances in Multimedia Information Processing - Volume Part II*, PCM'04, pages 497–504, Berlin, Heidelberg, 2004. Springer-Verlag. ISBN 3-540-23977-4, 978-3-540-23977-2.

A. Frank and A. Asuncion. UCI machine learning repository, 2010.

Yoav Freund, H. Sebastian Seung, Eli Shamir, and Naftali Tishby. Information, prediction, and query by committee. In *Advances in Neural Information Processing Systems 5, [NIPS Conference]*, pages 483–490, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc. ISBN 1-55860-274-7.

Yoav Freund, H. Sebastian Seung, Eli Shamir, and Naftali Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28:133–168, September 1997. ISSN 0885-6125.

Jerome H. Friedman. On bias, variance, 0/1-loss, and the curse-of-dimensionality. *Data Min. Knowl. Discov.*, 1:55–77, January 1997. ISSN 1384-5810.

Ludimila Luiza Gabriel. Automatic email organization. Master's thesis, Instituto Superior de Engenharia do Porto, 2009.

Luigi Galavotti, Via J. Nardi, Fabrizio Sebastiani, and Maria Simi. Feature selection and negative evidence in automated text categorization. In *Proceedings of the 4 th European Conference on Research and Advanced Technology for Digital Libraries, ECDL-00*, 2000.

A. Gammerman, V. Vovk, and V. Vapnik. Learning by Transduction. In *Uncertainty in Artificial Intelligence*, pages 148–155. Morgan Kaufmann, 1998.

Zoubin Ghahramani. Unsupervised learning. *Advanced Lectures on Machine Learning*, LNAI 3176:1–41, 2004.

Rayid Ghani. Combining labeled and unlabeled data for text classification with a large number of categories. In *Proceedings of the 2001 IEEE International Conference on Data Mining*, ICDM '01, pages 597–598, Washington, DC, USA, 2001. IEEE Computer Society. ISBN 0-7695-1119-8.

Rayid Ghani, Rosie Jones, Dunja Mladenic, Kamal Nigam, and Sean Slattery. Data mining on symbolic knowledge extracted from the web. In *Proceedings of the Sixth International Conference on Knowledge Discovery and Data Mining (KDD-2000), Workshop on Text Mining*, 2000.

Masood Ghayoomi. Using variance as a stopping criterion for active learning of frame assignment. In *Proceedings of the NAACL HLT 2010 Workshop on Active Learning for Natural Language Processing*, ALNLP '10, pages 1–9, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

Bhaskar Kumar Ghosh. *Handbook of Sequential Analysis*. Marcel Dekker, New York, 1991.

Ran Gilad-bachrach, Amir Navot, and Naftali Tishby. Query by committee made real. In *Advances in Neural Information Processing Systems 18*, 2005.

Shantanu Godbole and Sunita Sarawagi. Discriminative methods for multi-labeled classification. In Honghua Dai, Ramakrishnan Srikant, and Chengqi Zhang, editors, *Advances in Knowledge Discovery and Data Mining*, volume 3056 of *Lecture Notes in Computer Science*, pages 22–30. Springer Berlin / Heidelberg, 2004. ISBN 978-3-540-22064-0.

King-Shy Goh, Edward Y. Chang, and Wei-Cheng Lai. Multimodal concept-dependent active learning for image retrieval. In *Proceedings of the 12th annual ACM international conference on Multimedia*, MULTIMEDIA '04, pages 564–571, New York, NY, USA, 2004. ACM. ISBN 1-58113-893-8.

Teresa Gonçalves, Cassiana Silva, Paulo Quaresma, and Renata Vieira. Analysing part-of-speech for portuguese text classification. In *Proceedings of the 7th international conference on Computational Linguistics and Intelligent Text Processing*, CICLing'06, pages 551–562, Berlin, Heidelberg, 2006. Springer-Verlag. ISBN 3-540-32205-1, 978-3-540-32205-4.

Harry Halpin, Valentin Robu, and Hana Shepherd. The complex dynamics of collaborative tagging. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 211–220, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-654-7.

Guang Han and Chunxia Zhao. AUC maximization linear classifier based on active learning and its application. *Neurocomputing*, 73(7-9):1272–1280, 2010. ISSN 0925-2312. Advances in Computational Intelligence and Learning - 17th European Symposium on Artificial Neural Networks 2009, 17th European Symposium on Artificial Neural Networks 2009.

Steve Hanneke. A bound on the label complexity of agnostic active learning. In *Proceedings of the 24th International Conference on Machine Learning*, 2007.

Abhay S. Harpale and Yiming Yang. Personalized active learning for collaborative filtering. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 91–98, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-164-4.

Zellig Harris. Distributional structure. *Word*, 10:146–162, 1954.

T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning*. Springer, corrected edition, July 2003. ISBN 0387952845.

Philip J. Hayes. *Intelligent high-volume text processing using shallow, domain-specific techniques*, chapter Text-based Intelligent Systems: Current Research in Text Analysis, Information Extraction and Retrieval, pages 227–241. L. Erlbaum Associates Inc., Hillsdale, NJ, USA, 1992. ISBN 0-8058-1189-3.

Xiaofei He. Laplacian regularized d-optimal design for active learning and its application to image retrieval. *Trans. Img. Proc.*, 19:254–263, January 2010. ISSN 1057-7149.

David Helmbold and Sandra Panizza. Some label efficient learning results. In *Proceedings of the tenth annual conference on Computational learning theory*, COLT '97, pages 218–230, New York, NY, USA, 1997. ACM. ISBN 0-89791-891-6.

D. Hochbaum and D. Shmoys. A best possible heuristic for the k-center problem. *Mathematics of Operations Research*, 10(2):180–184, May 1985.

Steven Hoi, Rong Jin, and Michael Lyu. Large-scale text categorization by batch mode active learning. In *Proceedings of the World Wide Web Conference*, 2006.

Steven C. H. Hoi, Rong Jin, Jianke Zhu, and Michael R. Lyu. Semisupervised SVM batch mode active learning with applications to image retrieval. *ACM Trans. Inf. Syst.*, 27(3):1–29, 2009. ISSN 1046-8188.

W. Hu. World wide web search technologies. Idea Group Publishing, 2002. edited by Shi Nansi.

Weiming Hu, Wei Hu, Nianhua Xie, and Steve Maybank. Unsupervised active learning based on hierarchical graph-theoretic clustering. *Trans. Sys. Man Cyber. Part B*, 39(5):1147–1161, 2009. ISSN 1083-4419.

Anna Huang, David Milne, Eibe Frank, and Ian H. Witten. Clustering documents with active learning using wikipedia. In *ICDM '08: Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, pages 839–844, Washington, DC, USA, 2008. IEEE Computer Society. ISBN 978-0-7695-3502-9.

Rebecca Hwa. Sample selection for statistical parsing. *COMPUTATIONAL LINGUISTICS*, 30: 253–276, 2004.

Vijay S. Iyengar, Chidanand Apte, and Tong Zhang. Active learning using adaptive resampling. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '00, pages 91–98, New York, NY, USA, 2000. ACM. ISBN 1-58113-233-6.

Nathalie Japkowicz. Learning from imbalanced data sets: A comparison of various strategies. In *Proceedings of Learning from Imbalanced Data Sets, AAAI Workshop*, pages 10–15. AAAI Press, 2000.

Jun Jiang and Horace H. Ip. Dynamic Distance-Based Active Learning with SVM. In *Proceedings of the 5th international conference on Machine Learning and Data Mining in Pattern Recognition*, MLDM '07, pages 296–309, Berlin, Heidelberg, 2007. Springer-Verlag. ISBN 978-3-540-73498-7.

Yuqian Jiang, Huaizhong Lin, Xuesong Wang, and Dongming Lu. A technique for improving the performance of naive bayes text classification. In *Proceedings of the 2011 international conference on Web information systems and mining - Volume Part II*, WISM'11, pages 196–203, Berlin, Heidelberg, 2011. Springer-Verlag. ISBN 978-3-642-23981-6.

Rong Jin and Luo Si. A bayesian approach toward active learning for collaborative filtering. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, UAI '04, pages 278–285, Arlington, Virginia, United States, 2004. AUAI Press. ISBN 0-9749039-0-6.

Thorsten Joachims. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. In Douglas H. Fisher, editor, *Proceedings of ICML-97, 14th International Conference on Machine Learning*, pages 143–151, Nashville, US, 1997. Morgan Kaufmann Publishers, San Francisco, US.

Thorsten Joachims. Text categorization with support vector machines: learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, number 1398, pages 137–142, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE.

Thorsten Joachims. Transductive inference for text classification using support vector machines. In *Proceedings of the Sixteenth International Conference on Machine Learning*, ICML '99, pages 200–209, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc. ISBN 1-55860-612-2.

Rosie Jones, Andrew McCallum, Kamal Nigam, and Ellen Riloff. Bootstrapping for text learning tasks. In *IJCAI Workshop on Text Mining: Foundation, Techniques and Applications*, 1999.

Matti Kääriäinen. *Algorithmic Learning Theory*, chapter Active learning in the non-realizable case, pages 63–77. Springer Berlin / Heidelberg, 2006.

Jaeho Kang, Kwang Ryel Ryu, and Hyuk-Chul Kwon. Using cluster-based sampling to select initial training set for active learning in text classification. In Honghua Dai, Ramakrishnan Srikant, and Chengqi Zhang, editors, *Advances in Knowledge Discovery and Data Mining*, volume 3056 of *Lecture Notes in Computer Science*, pages 384–388. Springer Berlin / Heidelberg, 2004. 10.1007/978-3-540-24775-3_46.

Ashish Kapoor, Kristen Grauman, Raquel Urtasun, and Trevor Darrell. Gaussian processes for object categorization. *International Journal of Computer Vision*, 88:169–188, 2010a. ISSN 0920-5691. 10.1007/s11263-009-0268-3.

Ashish Kapoor, Kristen Grauman, Raquel Urtasun, and Trevor Darrell. Gaussian processes for object categorization. *Int. J. Comput. Vision*, 88:169–188, June 2010b. ISSN 0920-5691.

D. Karakos, S. Khudanpur, J. Eisner, and C.E. Priebe. Unsupervised classification via decision trees: an information-theoretic perspective. In *Acoustics, Speech, and Signal Processing, 2005. Proceedings. (ICASSP '05). IEEE International Conference on*, volume 5, pages v/1081 – v/1084 Vol. 5, march 2005.

G. V. Kass. An exploratory technique for investigating large quantities of categorical data. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 29(2):119–127, 1980. ISSN 00359254.

Leonard Kaufman and Peter Rousseeuw. *Finding groups in data: An introduction to cluster analysis.* John Wiley & Sons Inc., 1990.

Michael Kearns. Efficient noise-tolerant learning from statistical queries. *J. ACM*, 45(6):983–1006, November 1998. ISSN 0004-5411.

Ashraf Kibriya, Eibe Frank, Bernhard Pfahringer, and Geoffrey Holmes. Multinomial naive bayes for text categorization revisited. In Geoffrey Webb and Xinghuo Yu, editors, *AI 2004: Advances in Artificial Intelligence*, volume 3339 of *Lecture Notes in Computer Science*, pages 235–252. Springer Berlin / Heidelberg, 2005. ISBN 978-3-540-24059-4.

Sang-Bum Kim, Kyoung-Soo Han, Hae-Chang Rim, and Sung H. Myaeng. Some Effective Techniques for Naive Bayes Text Classification. *IEEE Transactions on Knowledge and Data Engineering*, 18(11):1457–1466, November 2006. ISSN 1041-4347.

W. Kinzel and P. Ruján. Improving a network generalization ability by selecting examples. *EPL (Europhysics Letters)*, 13(5):473, 1990.

Svetlana Kiritchenko, Stan Matwin, Richard Nock, and A. Fazel Famili. Learning and evaluation in the presence of class hierarchies: Application to text categorization. In *Canadian Conference on AI*, pages 395–406, 2006.

Daphne Koller and Mehran Sahami. Toward optimal feature selection. In *International Conference on Machine Learning*, pages 284–292, 1996.

Daphne Koller and Mehran Sahami. Hierarchically classifying documents using very few words. In Douglas H. Fisher, editor, *Proceedings of the Fourteenth International Conference on Machine Learning (ICML 1997), Nashville, Tennessee, USA, July 8-12, 1997*, pages 170–178. Morgan Kaufmann, 1997. ISBN 1-55860-486-3.

Robert Krovetz. Viewing morphology as an inference process. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '93, pages 191–202, New York, NY, USA, 1993. ACM. ISBN 0-89791-605-0.

Miroslav Kubat and Stan Matwin. Addressing the curse of imbalanced training sets: One-sided selection. In *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 179–186. Morgan Kaufmann, 1997.

M. Arun Kumar and M. Gopal. Reduced one-against-all method for multiclass svm classification. *Expert Systems with Applications*, 38(11):14238–14248, 2011. ISSN 0957-4174.

Steve Lawrence, Kurt Bollacker, and C. Lee Giles. Indexing and retrieval of scientific literature. In *CIKM 99: Proceedings of the eighth international conference on Information and knowledge management*, pages 139–146, New York, NY, USA, 1999. ACM. ISBN 1-58113-146-1.

Florian Laws and Hinrich Schätze. Stopping criteria for active learning of named entity recognition. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 465–472, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. ISBN 978-1-905593-44-6.

David D. Lewis. Evaluating and optimizing autonomous text classification systems. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '95, pages 246–254, New York, NY, USA, 1995. ACM. ISBN 0-89791-714-6.

David D. Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 148–156. Morgan Kaufmann, 1994.

David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 3–12, New York, NY, USA, 1994. Springer-Verlag New York, Inc. ISBN 0-387-19889-X.

David D. Lewis and Marc Ringuette. A comparison of two learning algorithms for text categorization. In *Third Annual Symposium on Document Analysis and Information Retrieval*, pages 81–93, 1994.

Huan Li, Yuan Shi, Mingyu Chen, Alexander Hauptmann, and Zhang Xiong. Joint-al: Joint discriminative and generative active learning for cross-domain semantic concept classification. In *Proceedings of the 2010 IEEE Fourth International Conference on Semantic Computing*, ICSC '10, pages 60–66, Washington, DC, USA, 2010. IEEE Computer Society. ISBN 978-0-7695-4154-9.

J. Li and J. Z. Wang. Automatic linguistic indexing of pictures by a statistical modeling approach. *IEEE Trans.Pattern Anal. Mach. Intell.*, 25(9):1075—-1088, 2003.

Mingkun Li and Ishwar Sethi. Confidence-based active learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1251–1261, 2006.

Tao Li and Sarabjot Singh Anand. Diva: a variance-based clustering approach for multi-type relational data. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, CIKM '07, pages 147–156, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-803-9.

Xuchun Li, Lei Wang, and Eric Sung. Multi-label SVM Active Learning for Image Classification. In *IEEE 2004 International Conference on Image Processing (ICIP '04)*, pages 2207–2210, 2004.

Ray Liere and Prasad Tadepalli. Active learning with committees for text categorization. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pages 591–596, 1997.

Michael Lindenbaum, Shaul Markovitch, and Dmitry Rusakov. Selective sampling for nearest neighbor classifiers. *Mach. Learn.*, 54:125–152, February 2004. ISSN 0885-6125.

Nick Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Mach. Learn.*, 2:285–318, April 1988. ISSN 0885-6125.

Huan Liu and Hiroshi Motoda. *Instance Selection and Construction for Data Mining*. Kluver Academic Publishers, 2001.

Tantan Liu and Gagan Agrawal. Active learning based frequent itemset mining over the deep web. In *Proceedings of the 2011 IEEE 27th International Conference on Data Engineering*, ICDE '11, pages 219–230, Washington, DC, USA, 2011. IEEE Computer Society. ISBN 978-1-4244-8959-6.

Tao Liu, Shengping Liu, and Zheng Chen. An evaluation on feature selection for text clustering. In *International Conference on Machine Learning*, pages 488–495, 2003.

Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins. Text classification using string kernels. *J. Mach. Learn. Res.*, 2:419–444, March 2002. ISSN 1532-4435.

R. Lomasky, C. E. Brodley, M. Aernecke, D. Walt, and M. Friedl. Active class selection. In *Proceedings of the 18th European conference on Machine Learning*, ECML '07, pages 640–647, Berlin, Heidelberg, 2007. Springer-Verlag. ISBN 978-3-540-74957-8.

Julie Beth Lovins. Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics*, 11:22–31, 1968.

Zhenyu Lu, Anand I. Rughani, Bruce I. Tranmer, and Josh Bongard. Informative sampling for large unbalanced data sets. In *Proceedings of the 2008 GECCO conference companion on Genetic and evolutionary computation*, GECCO '08, pages 2047–2054, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-131-6.

Zhenyu Lu, Xindong Wu, and Josh Bongard. Adaptive informative sampling for active learning. In *SDM*, pages 894–905, 2010.

Luciano, Francisco A. Rodrigues, Gonzalo Travieso, and Villas P. R. Boas. Characterization of complex networks: A survey of measurements. *Advances in Physics*, 56(1):167–242, January 2007.

Tom M. and Mitchell. Generalization as search. *Artificial Intelligence*, 18(2):203–226, 1982. ISSN 0004-3702.

David J. C. MacKay. Information-based objective functions for active data selection. *Neural Comput.*, 4:590–604, July 1992. ISSN 0899-7667.

Sofus A. Macskassy, Arunava Banerjee, Brian D. Davison, and Haym Hirsh. Human performance on clustering web pages: A preliminary study. In *Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98)*, New York, NY, 1998.

Michael Mandel, Graham Poliner, and Daniel Ellis. Support vector machine active learning for music retrieval. *Multimedia Systems*, 12:3–13, 2006. ISSN 0942-4962. 10.1007/s00530-006-0032-2.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

Takeshi Masuyama and Hiroshi Nakagawa. Two step POS selection for SVM based text categorization. *IEICE Transactions*, 87-D(2):373–379, 2004.

Dominic Mazzoni, Kiri Wagstaff, and Michael Burl. Active learning with irrelevant examples. In Johannes Fürnkranz, Tobias Scheffer, and Myra Spiliopoulou, editors, *Machine Learning: ECML 2006*, volume 4212 of *Lecture Notes in Computer Science*, pages 695–702. Springer Berlin - Heidelberg, 2006. ISBN 978-3-540-45375-8. 10.1007/11871842_69.

Andrew McCallum and Kamal Nigam. Employing em and pool-based active learning for text classification. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 350–358, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc. ISBN 1-55860-556-8.

Prem Melville and Raymond J. Mooney. Constructing diverse classifier ensembles using artificial training examples. In *Proceedings of the 18th international joint conference on Artificial intelligence*, pages 505–510, San Francisco, CA, USA, 2003. Morgan Kaufmann Publishers Inc.

Prem Melville and Raymond J. Mooney. Diverse ensembles for active learning. In *Proceedings of the twenty-first international conference on Machine learning*, ICML '04, pages 74–, New York, NY, USA, 2004. ACM. ISBN 1-58113-838-5.

Prem Melville, Stewart M. Yang, Maytal Saar-tsechansky, and Raymond Mooney. Active learning for probability estimation using jensen-shannon divergence. In *Proceedings of the European Conference on Machine Learning (ECML-05)*, pages 268–279. Springer-Verlag, 2005.

Jurij Mihelič and Borut Robič. Solving the k-center problem efficiently with a dominating set algorithm. *Journal of Computing and Information Technology*, 13, 3:225—-233, 2005.

Tom M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

P. Mitra, C. A. Murthy, and S. K. Pal. A probabilistic active support vector learning algorithm. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(3):413–418, 2004.

C. N. Mooers. The theory of digital handling of non-numerical information and its implications to machine economics. In *Technical Bulletin No. 48. Cambridge, MA: Association of computing machinery meeting*. Zator Co., 1950.

R. Moskovitch, S. Cohenkashi, U. Dror, I. Levy, A. Maimon, and Y. Shahar. Multiple hierarchical classification of free-text clinical guidelines. *Artificial Intelligence in Medicine*, 37(3):177–190, July 2006. ISSN 09333657.

Robson Motta, Alneu Andrade Lopes, and Maria Cristina Oliveira. Centrality measures from complex networks in active learning. In *Proceedings of the 12th International Conference on Discovery Science*, DS '09, pages 184–196, Berlin, Heidelberg, 2009. Springer-Verlag. ISBN 978-3-642-04746-6.

Robson Motta, Alipio Mario Jorge, Alneu de Andrade Lopes, Nuno Escudeiro, and Maria Cristina Oliveira. Combining network and confidence based approaches for active learning (submitted). In *12th IEEE International Conference on Data Mining*, 2012.

Ion Muslea, Steven Minton, and Craig A. Knoblock. Active learning with multiple views. *Journal of Artificial Intelligence Research*, 27:203–233, 2006.

Gonzalo Navarro and Ricardo Baeza-Yates. A language for queries on structure and contents of textual databases. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '95, pages 93–101, New York, NY, USA, 1995. ACM. ISBN 0-89791-714-6.

M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003.

Hieu T. Nguyen and Arnold Smeulders. Active learning using pre-clustering. In *Proceedings of the 21st International Conference on Machine Learning*, pages 623–630. ACM Press, 2004.

Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. Text Classification from Labeled and Unlabeled Documents using EM. *Mach. Learn.*, 39:103–134, May 2000. ISSN 0885-6125.

Bruno M. Nogueira, Alípio M. Jorge, and Solange O. Rezende. Hierarchical confidence-based active clustering. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, SAC '12, pages 216–219, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-0857-1.

Blaž Novak, Dunja Mladenič, and Marko Grobelnik. Text classification with active learning. In Myra Spiliopoulou, Rudolf Kruse, Christian Borgelt, Andreas Nürnberger, Wolfgang Gaul, H.-H. Bock, W. Gaul, and M. Vichi, editors, *From Data and Information Analysis to Knowledge Engineering*, Studies in Classification, Data Analysis, and Knowledge Organization, pages 398–405. Springer Berlin Heidelberg, 2006. ISBN 978-3-540-31314-4.

Jana Novovičová, Antonín Malík, and Pavel Pudil. Feature selection using improved mutual information for text classification. In Ana Fred, Terry Caelli, Robert Duin, Aurélio Campilho, and Dick de Ridder, editors, *Structural, Syntactic, and Statistical Pattern Recognition*, volume 3138 of *Lecture Notes in Computer Science*, pages 1010–1017. Springer Berlin / Heidelberg, 2004. ISBN 978-3-540-22570-6.

Fredrik Olsson. *Bootstrapping Named Entity Annotation by Means of Active Machine Learning: A Method for Creating Corpora*. PhD thesis, University of Gothenburg, 2008.

Fredrik Olsson. A literature survey of active machine learning in the context of natural language processing. Technical report, Swedish Institute of Computer Science, 2009.

Fredrik Olsson and Katrin Tomanek. An intrinsic stopping criterion for committee-based active learning. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, CoNLL '09, pages 138–146, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-29-9.

Vivian Orengo and C. Huyck. A stemming algorithm for the portuguese language. In *Proceedings of the 8th International Symposium on String Processing and Information Retrieval(SPIRE)*, pages 186–193, 2001.

Thomas Osugi, Deng Kun, and Stephen Scott. Balancing exploration and exploitation: A new algorithm for active machine learning. In *Proceedings of the Fifth IEEE International Conference on Data Mining*, ICDM '05, pages 330–337, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2278-5.

V. Papathanasiou. Some characteristic properties of the fisher information matrix via cacoullos-type inequalities. *J. Multivar. Anal.*, 44(2):256–265, February 1993. ISSN 0047-259X.

H. Peng, F. Long, and C. Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, August 2005. ISSN 0162-8828.

Fernando Pereira, Naftali Tishby, and Lillian Lee. Distributional clustering of english words. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, ACL '93, pages 183–190, Stroudsburg, PA, USA, 1993. Association for Computational Linguistics.

Leonard Pitt and Leslie G. Valiant. Computational limitations on learning from examples. *Journal of the ACM*, 35:965–984, 1988.

M. Plutowski and H. White. Selecting concise training sets from clean data. *Neural Networks, IEEE Transactions on*, 4(2):305 –318, mar 1993. ISSN 1045-9227.

M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

Luc Pronzato. Survey paper: Optimal experimental design and some related control problems. *Automatica*, 44:303–325, February 2008. ISSN 0005-1098.

Foster Provost, David Jensen, and Tim Oates. Efficient progressive sampling. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '99, pages 23–32, New York, NY, USA, 1999. ACM. ISBN 1-58113-143-7.

R. Prudêncio and T.B. Ludermir. Active meta-learning with uncertainty sampling and outlier detection. In *Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, pages 346 –351, june 2008.

Guo-Jun Qi, Xian-Sheng Hua, Yong Rui, Jinhui Tang, and Hong-Jiang Zhang. Two-dimensional multilabel active learning with an efficient online adaptation model for image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(10):1880–1897, October 2009. ISSN 0162-8828.

C. Y. Quek. Classification of world wide web documents. Senior Honors Thesis School of Computer Science Carnegie Mellon University, 1998.

J. R. Quinlan. Induction of decision trees. *Mach. Learn.*, 1(1):81–106, March 1986. ISSN 0885-6125.

J. Ross Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993. ISBN 1-55860-238-0.

Abdellatif Rahmoun and Zakaria Elberrichi. Experimenting n-grams in text categorization. *Int. Arab J. Inf. Technol.*, 4(4):377–385, 2007.

Pedro Ribeiro and Nuno Escudeiro. On-line news "à la carte". In *Proceedings of the European Conference on the Use of Modern Information and Communication Technologies*, 2008.

Valentin Robu, Harry Halpin, and Hana Shepherd. Emergence of consensus and shared vocabularies in collaborative tagging systems. *ACM Trans. Web*, 3(4):14:1–14:34, September 2009. ISSN 1559-1131.

Nicholas Roy and Andrew McCallum. Toward optimal active learning through sampling estimation of error reduction. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 441–448, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1-55860-778-1.

Maytal Saar-Tsechansky and Foster Provost. Active sampling for class probability estimation and ranking. *Mach. Learn.*, 54:153–178, February 2004. ISSN 0885-6125.

Sujan Kumar Saha, Sudeshna Sarkar, and Pabitra Mitra. Feature selection techniques for maximum entropy based biomedical named entity recognition. *Journal of Biomedical Informatics*, 42(5):905 – 911, 2009. ISSN 1532-0464. <ce:title>Biomedical Natural Language Processing</ce:title>.

Airi Salminen and Frank W. Tompa. Pat expressions: an algebra for text search. *Acta Linguistica Hungarica*, 41:1–4, 1994.

G. Salton and C. Buckley. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41(4):288–297, 1990.

G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. In Karen Sparck Jones and Peter Willett, editors, *Readings in information retrieval*, chapter A vector space model for automatic indexing, pages 273–280. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997. ISBN 1-55860-454-5.

Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. In Karen Sparck Jones and Peter Willett, editors, *Readings in information retrieval*, chapter Term-weighting approaches in automatic text retrieval, pages 323–328. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997. ISBN 1-55860-454-5.

Claude Sammut and Ranan B. Banerji. Learning concepts by asking questions. In R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach: Volume II*, pages 167–192, Los Altos, CA, 1986. Morgan Kaufmann.

Tobias Scheffer, Christian Decomain, and Stefan Wrobel. Active hidden markov models for information extraction. In *Proceedings of the 4th International Conference on Advances in Intelligent Data Analysis*, IDA '01, pages 309–318, London, UK, UK, 2001. Springer-Verlag. ISBN 3-540-42581-0.

Greg Schohn and David Cohn. Less is more: Active learning with support vector machines. In *Proceedings of the International Conference on Machine Learning*, 2000.

Hinrich Schütze, Emre Velipasaoglu, and Jan O. Pedersen. Performance thresholding in practical text classification. In *Proceedings of the 15th ACM international conference on Information and knowledge management*, CIKM '06, pages 662–671, New York, NY, USA, 2006. ACM. ISBN 1-59593-433-2.

D. Sculley. Online active learning methods for fast Label-Efficient spam filtering. In *Fourth Conference on Email and Anti-Spam*, August 2007.

Fabrizio Sebastiani and Consiglio Nazionale Delle Ricerche. Machine learning in automated text categorization. *ACM Computing Surveys*, 34:1–47, 2002.

Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 1070–1079, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

H. Seung, M. Opper, and H. Sompolinsky. Query by committee. In *Proceedings of the 5th Annual Workshop on Computational Learning Theory*, 1992.

Claude E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, July 1948.

E. Shapiro. A general incremental algorithm that infers theories from facts. In *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, pages 446–451. Morgan Kaufman, 1981.

Xiaoxiao Shi, Wei Fan, and Jiangtao Ren. Actively transfer domain knowledge. In *Proceedings of the European conference on Machine Learning and Knowledge Discovery in Databases - Part II*, ECML PKDD '08, pages 342–357, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-87480-5.

Lawrence Shih, Jason D. M. Rennie, Yu han Chang, and David R. Karger. Text bundling: Statistics-based data reduction. In *Twentieth International Conference on Machine Learning*, pages 696–703, 2003.

A. Sigogne and M. Constant. Real-time unsupervised classification of web documents. In *Computer Science and Information Technology, 2009. IMCSIT '09. International Multiconference on*, pages 281–286, oct. 2009.

Carlos N. Silla, Jr. and Alex A. Freitas. A survey of hierarchical classification across different application domains. *Data Min. Knowl. Discov.*, 22:31–72, January 2011. ISSN 1384-5810.

Diego Sona, Sriharsha Veeramachaneni, Nicola Polettini, and Paolo Avesani. P.: Regularization for unsupervised classification on taxonomies. In *International Symposium on Methodologies for Intelligent Systems*, volume 4203, of *LNCS*, pages 691—696, 2006.

Xiaoyuan Su and Taghi M. Khoshgoftaar. A survey of collaborative filtering techniques, January 2009. ISSN 1687-7470.

Aixin Sun and Ee P. Lim. Hierarchical text classification and evaluation. In *ICDM*, pages 521–528, 2001.

Shiliang Sun and David R. Hardoon. Active learning with extremely sparse labeled examples. *Neurocomputing*, 73(16-18):2980 – 2988, 2010. ISSN 0925-2312. 10th Brazilian Symposium on Neural Networks (SBRN2008).

Sebastian Thrun. The handbook of brain theory and neural networks, 1998.

Robert Tibshirani and Geoffrey Hinton. Coaching variables for regression and classification. *Statistics and Computing*, 8(1):25–33, January 1998. ISSN 0960-3174.

D.M. Titterington, U.E. Makov, and A.F. Smith. *Statistical Analysis of Finite Mixture Distributions*. John Wiley & Sons, 1985.

K. Tomanek and U. Hahn. Approximating learning curves for active-learning-driven annotation. In *Proceedings of LREC'08*, 2008.

Katrin Tomanek, Joachim Wermter, and Udo Hahn. An approach to text corpus construction which cuts annotation costs and maintains reusability of annotated data. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 486–495, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

Simon Tong and Edward Chang. Support vector machine active learning for image retrieval. In *Proceedings of the nineth ACM international conference on Multimedia*, MULTIMEDIA '01, pages 107–118, New York, NY, USA, 2001. ACM. ISBN 1-58113-394-4.

Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *J. Mach. Learn. Res.*, 2:45–66, March 2002. ISSN 1532-4435.

Christoph Trattner, Christian Körner, and Denis Helic. Enhancing the navigability of social tagging systems with tag taxonomies. In *Proceedings of the 11th International Conference on Knowledge Management and Knowledge Technologies*, i-KNOW '11, pages 18:1–18:8, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0732-1.

G. Tsoumakas and I. Katakis. Multi label classification: An overview. *International Journal of Data Warehouse and Mining*, 3(3):1–13, 2007.

L. G. Valiant. A theory of the learnable. *Commun. ACM*, 27:1134–1142, November 1984. ISSN 0001-0782.

C.J. van Rijsbergen. *Information Retrieval*. Butterworths, 1979.

Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1 edition, September 1998. ISBN 0471030031.

Andreas Vlachos. A stopping criterion for active learning. *Computer Speech & Language*, 22(3): 295 – 312, 2008. ISSN 0885-2308.

Meng Wang and Xian-Sheng Hua. Active learning in multimedia annotation and retrieval: A survey. *ACM Trans. Intell. Syst. Technol.*, 2:10:1–10:21, February 2011. ISSN 2157-6904.

Tian-Jiang Wang, Gang Chen, and Perfecto Herrera. Music retrieval based on a multi-samples selection strategy for support vector machine active learning. In *Proceedings of the 2009 ACM symposium on Applied Computing*, SAC '09, pages 1750–1751, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-166-8.

M. K. Warmuth, G. Ratsch, M. Mathieson, J. Liao, and C. Lemmen. Active learning in the drug discovery process. *Advances in Neural Information Processing Systems*, 2:1449–1456, 2002.

Manfred K. Warmuth, Jun Liao, Gunnar Rätsch, Michael Mathieson, Santosh Putta, and Christian Lemmen. Active learning with support vector machines in the drug discovery process. *Journal of Chemical Information and Computer Sciences*, 43:667–673, 2003.

Roger Weber, Hans-Jörg Schek, and Stephen Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *Proceedings of the 24rd International Conference on Very Large Data Bases*, VLDB '98, pages 194–205, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc. ISBN 1-55860-566-5.

Sholom Weiss, Nitin Indurkhya, Tong Zhang, and Fred Damerau. *Text Mining: Predictive Methods for Analyzing Unstructured Information*. SpringerVerlag, 2004. ISBN 0387954333.

Robert Wetzker, Carsten Zimmermann, Christian Bauckhage, and Sahin Albayrak. I tag, you tag: translating tags for advanced user models. In *Proceedings of the third ACM international conference on Web search and data mining*, WSDM '10, pages 71–80, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-889-6.

W. John Wilbur and Karl Sirotkin. The automatic identification of stop words. *Journal of Information Science*, 18:45–55, January 1992. ISSN 0165-5515.

Ian H. Witten and Eibe Frank. *Data mining: practical machine learning tools and techniques with Java implementations*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2000. ISBN 1-55860-552-5.

E. Wong and B. Hajek. *Stochastic Processes in Engineering Systems*. Springer-Verlag, 1985.

Yan Xu, Bin Wang, JinTao Li, and Hongfang Jing. An extended document frequency metric for feature selection in text categorization. In Hang Li, Ting Liu, Wei-Ying Ma, Tetsuya Sakai, Kam-Fai Wong, and Guodong Zhou, editors, *Information Retrieval Technology*, volume 4993 of *Lecture Notes in Computer Science*, pages 71–82. Springer Berlin / Heidelberg, 2008. ISBN 978-3-540-68633-0.

Zhao Xu, Kai Yu, Volker Tresp, Xiaowei Xu, and Jizhi Wang. Representative sampling for text classification using support vector machines. In *Proceedings of the 25th European conference on IR research*, ECIR'03, pages 393–407, Berlin, Heidelberg, 2003. Springer-Verlag. ISBN 3-540-01274-5.

Zhao Xu, Kai Yu, Volker Tresp, Xiaowei Xu, and Jizhi Wang. Representative sampling for text classification using support vector machines. In *European Conference on Information Retrieval Research 2003*, volume 2633, pages ,393—-407. Springer Berlin / Heidelberg, 2004.

Bishan Yang, Jian-Tao Sun, Tengjiao Wang, and Zheng Chen. Effective multi-label active learning for text classification. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 917–926, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-495-9.

Yiming Yang and Christopher G. Chute. An example-based mapping method for text categorization and retrieval. *ACM Trans. Inf. Syst.*, 12(3):252–277, July 1994. ISSN 1046-8188.

Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 412–420, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc. ISBN 1-55860-486-3.

Yiming Yang, Seán Slattery, and Rayid Ghani. A study of approaches to hypertext categorization. *J. Intell. Inf. Syst.*, 18(2-3):219–241, 2002.

Kai Yu, Jinbo Bi, and Volker Tresp. Active learning via transductive experimental design. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 1081–1088, New York, NY, USA, 2006. ACM. ISBN 1-59593-383-2.

Weiwei Yuan, Yongkoo Han, Donghai Guan, Sungyoung Lee, and Young-Koo Lee. Initial training data selection for active learning. In *Proceedings of the 5th International Conference on Ubiquitous Information Management and Communication*, ICUIMC '11, pages 5:1–5:7, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0571-6.

Cha Zhang and Tsuhan Chen. An active learning framework for content-based information retrieval. *IEEE Transactions on Multimedia*, 4(2):260–268, 2002.

Min-Ling Zhang and Zhi-Hua Zhou. A k-nearest neighbor based algorithm for multi-label classification. In *IEEE International Conference on Granular Computing*, volume 2, pages 718–721 Vol. 2. The IEEE Computational Intelligence Society, 2005.

Xian Zhang and Xiaoyan Zhu. A new type of feature — loose n-gram feature in text categorization. In *Proceedings of the 3rd Iberian conference on Pattern Recognition and Image Analysis, Part I*, IbPRIA '07, pages 378–385, Berlin, Heidelberg, 2007. Springer-Verlag. ISBN 978-3-540-72846-7.

Zhaohui Zheng, Xiaoyun Wu, and Rohini Srihari. Feature selection for text categorization on imbalanced data. *SIGKDD Explor. Newsl.*, 6:80–89, June 2004. ISSN 1931-0145.

J. Zhu, H. Wang, B.K. Tsou, and M. Ma. Active learning with sampling by uncertainty and density for data annotations. *Audio, Speech, and Language Processing, IEEE Transactions on*, 18(6): 1323 –1331, 2010a. ISSN 1558-7916.

Jingbo Zhu and Eduard Hovy. Active learning for word sense disambiguation with methods for addressing the class imbalance problem. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, June 2007.

Jingbo Zhu and Huizhen Wang. Learning a stopping criterion for active learning for word sense disambiguation and text classification. In *3rd International Joint Conference on Natural Language Processing*, 2008.

Jingbo Zhu, Huizhen Wang, and Eduard Hovy. Multi-criteria-based strategy to stop active learning for data annotation. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 1129–1136, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. ISBN 978-1-905593-44-6.

Jingbo Zhu, Huizhen Wang, Eduard Hovy, and Matthew Ma. Confidence-based stopping criteria for active learning for data annotation. *ACM Trans. Speech Lang. Process.*, 6:3:1–3:24, April 2010b. ISSN 1550-4875.

Xiaojin Zhu. Semi-supervised learning literature survey. *SciencesNew York*, Tech. Rep.(1530), 2008.

Xiaojin Zhu and Andrew B Goldberg. Introduction to semi-supervised learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 3(1):1–130, 2009.

Xingquan Zhu, Peng Zhang, Xiaodong Lin, and Yong Shi. Active learning from data streams. In *Proceedings of the 2007 Seventh IEEE International Conference on Data Mining*, pages 757–762, Washington, DC, USA, 2007. IEEE Computer Society. ISBN 0-7695-3018-4.

Xingquan Zhu, Peng Zhang, Xiaodong Lin, and Yong Shi. Active learning from stream data using optimal weight classifier ensemble. *Trans. Sys. Man Cyber. Part B*, 40:1607–1621, December 2010c. ISSN 1083-4419.