DOI: http://dx.doi.org/10.17058/tecnolog.v20i2.7428

TECNO-LÓGICA
Revista do depto. de Química e Física, do depto. de Engenharia, Arquitetura e Ciências Agrárias e do Mestrado em Tecnologia Ambiental
http://online.unisc.br/seer/index.php/tecnologica

DESENVOLVIMENTO DE UMA APLICAÇÃO PARA RECONHECER SIGNIFICADOS DE PALAVRAS HOMÔNIMAS UTILIZANDO REDES NEURAIS ARTIFICIAIS

Eduardo Gabriel Côrtes[1,2], *Rodrigo Frantz*[1,2], *Rejane Frozza*[2,3*]

[1]*Curso de Ciência da Computação, Universidade de Santa Cruz do Sul (UNISC), CEP: 96815-900, Rio Grande do Sul Brasil.*

[2]*Departamento de Computação – Universidade de Santa Cruz do Sul (UNISC), CEP: 96815-900, Rio Grande do Sul Brasil.*

[3]*Programa de Pós-Graduação em Sistemas e Processos Industriais - Universidade de Santa Cruz do Sul (UNISC), Av. Independência, 2293, Santa Cruz do Sul, Rio Grande do Sul, Brasil.*

**E-mail: frozza@unisc.br*

Recebido em: 08/04/2016
Aceito em: 23/05/2016

RESUMO

Este trabalho propõe desenvolver uma aplicação para auxiliar no entendimento do significado de palavras ambíguas da língua portuguesa, que dependem necessariamente do contexto para serem identificadas. A aplicação foi implementada na linguagem de programação JAVA, utilizando a IDE NetBeans, juntamente com a ferramenta ADReNA que permitiu abstrair a implementação de uma Rede Neural Artificial (RNA) e possibilitou modelar, treinar e parametrizar uma RNA do tipo *backpropagation*. Foram desenvolvidas duas versões da aplicação, que se diferenciam pela forma que representam a camada de entrada da rede neural artificial. Testes realizados mostraram que a forma que a rede neural é representada gera influência em seus resultados. Após a realização dos testes, a aplicação se mostrou promissora no reconhecimento de significados de palavras homônimas.

Palavras-chave: Redes Neurais Artificiais. Palavras Ambíguas. Ferramenta ADReNA.

1 Introdução

Um ser humano, quando deseja se comunicar com outros humanos, faz uso da linguagem natural comum, na qual sua comunicação depende de um entendimento básico da linguagem e da semântica por todos os envolvidos para que faça sentido. Já, se um ser humano deseja se comunicar com um computador usando a mesma linguagem que é utilizada para se comunicar com outros humanos, é preciso que o computador tenha conhecimento sobre esta linguagem.

Segundo [8], a área de processamento de linguagem natural contém diversos desafios, entre eles destaca-se a questão da ambiguidade da língua portuguesa. Logo, este trabalho tem o objetivo de contribuir com o desenvolvimento de uma aplicação que busca resolver um dos problemas da ambiguidade da língua portuguesa, a homonímia de palavras.

A ambiguidade é quando algum aspecto da linguagem pode ter mais de um significado. Palavras homônimas são exemplos de ambiguidade, onde sua escrita ou pronúncia são as mesmas, mas podem assumir significados diferentes dependendo do contexto que as mesmas são aplicadas. Logo, este trabalho busca reconhecer o significado de palavras homônimas dentro de uma frase por meio do uso da técnica de redes neurais artificiais, advinda dos estudos da área de inteligência artificial. As Redes Neurais Artificiais (RNAs) são modelos computacionais baseados na aprendizagem e reconhecimento da estrutura neural de organismos inteligentes [4]. As RNAs são compostas por várias unidades de processamento, conhecidas como neurônios, que são conectados por canais de comunicação, as conexões, que estão associadas a valores, os pesos. São estruturadas em camadas de neurônios, com a camada de entrada, camadas intermediárias e a camada de saída.

Os pesos das conexões são ajustados de acordo com os exemplos apresentados para a RNA, o que caracteriza a sua propriedade mais importante que é a habilidade de aprender. Esse processo de ajuste de pesos é conhecido como treinamento, processo este que fará com que a rede neural tenha um aprendizado [3].

Os testes realizados no trabalho utilizaram duas versões do *software* desenvolvido para comparar qual obteve melhor resultado, variando a quantidade de camadas intermediárias e a quantidade de neurônios de cada camada.

Existem diversas ferramentas que possibilitam especificar modelos de redes neurais artificiais que possam ser utilizadas para ensino deste tema e para criação de aplicações específicas. A modelagem pode ser feita por meio de interface gráfica ou pela disponibilização de *Application Programming Interfaces* (APIs) ou bibliotecas de códigos-fonte.

DOI: http://dx.doi.org/10.17058/tecnolog.v20i2.7428
TECNO-LÓGICA
Revista do depto. de Química e Física, do depto. de Engenharia, Arquitetura e Ciências Agrárias e do Mestrado em Tecnologia Ambiental
http://online.unisc.br/seer/index.php/tecnologica

Em [14] é apresentada uma pesquisa e avaliação de ferramentas de RNAs, inclusive da ADReNA, ferramenta utilizada para desenvolvimento da aplicação neste trabalho. Os critérios utilizados na análise foram: Licença de uso (uso gratuito ou não); modelos e algoritmos de RNA disponíveis; disponibilidade de interface gráfica; linguagens de programação disponibilizadas para uso; propósito da ferramenta. Entre as ferramentas pesquisadas, a ADReNA foi uma das que atendem aos requisitos estabelecidos e pode ser utilizada para ensino de RNAs, dispondo de interface gráfica para modelagem das aplicações, sendo gratuita e possibilitando o uso de APIs.

Além da seção de introdução, este artigo apresenta na seção dois os aspectos teóricos dos principais assuntos abordados neste trabalho; na seção três é apresentada a ferramenta ADReNa utilizada para implementação; na seção quatro é explicado como a aplicação foi desenvolvida e como o problema foi representado em uma rede neural artificial; na seção cinco é feita uma descrição de como foram realizados os testes; a seção seis descreve os resultados obtidos nos teste e na seção sete é apresentada a conclusão do trabalho.

2 Domínio da Aplicação

O Processamento de Linguagem Natural é uma área de pesquisa e aplicação que explora como computadores podem ser usados para entender e manipular textos e discursos em linguagem natural para fazer coisas úteis [2]. O processamento da linguagem pode ser dividido em estágios teóricos da linguística, tais como, a sintaxe, a semântica e a pragmática. A primeira refere-se à ordem e à estrutura da linguagem, já a semântica trata do significado e a pragmática reflete o significado contextualizado. Um dos problemas de processamento encontrados na análise pragmática é a ambiguidade de palavras, que podem assumir diferentes significados dependendo do seu contexto.

Foram destacados alguns trabalhos relacionados ao tratamento de linguagem natural como os realizados por [1] e [9]. Em [1], os autores utilizam redes neurais para detectar erros com relação ao uso da crase. Já [9] busca sumarizar sentenças do texto fonte consideradas importantes onde a classificação das sentenças em graus de importância é feita pela rede neural com base em características extraídas durante o processo de sumarização.

Em [6], destaca-se a utilização de redes neurais artificias para realizar o processamento de linguagem natural, neste caso, foi implementada uma rede neural que tinha como objetivo aprender frases da língua portuguesa. A representação do problema em uma rede neural consistia em tratar cada palavra como um conjunto de micro características semânticas. Ou seja, cada palavra é representada por um vetor de *bits* (0,1), onde cada subconjunto de *bits* tem um significado associado.

2.1 Palavras Homônimas

Quando uma palavra é classificada como homônima, significa que sua escrita ou sua pronúncia são a mesma, porém, com significado diferente. Já as parônimas possuem escrita e pronúncia parecidas, mas com significados diferentes [11].

Os homônimos são classificados em três tipos, segundo [7]:

- Homógrafos: São homônimos com a mesma forma de escrita, mas com pronúncia diferente. Esse tipo de homônimo pode ter seu sentido identificado pela pronúncia, mas não é possível identificá-lo pela forma em que a palavra é estruturada. Exemplo: “colher” de comer e “colher” uma planta.

- Homófonas: São homônimos com a forma de escrita diferente, mas com a mesma pronúncia. Esse tipo de homônimo pode ter seu significado identificado pela forma em que a palavra é estruturada. Exemplo: “cela” de presídio e “sela” de cavalo.

- Homônimo perfeito: São homônimos que contêm a mesma forma de escrita e são pronunciados da mesma forma. Esse tipo de homônimo só pode ser identificado pelo contexto que o mesmo se aplica. Exemplo: “banco” de sentar e “banco” de dados.

Como a entrada de dados da aplicação é em formato de texto, e não em formato de áudio, é apenas relevante o reconhecimento de Homógrafos e Homónimos perfeito, já que o significado de uma Homófona poderia ser identificado apenas pela forma que sua escrita é estruturada.

2.2 Redes Neurais Artificiais

As Redes Neurais Artificiais são modelos computacionais baseados na aprendizagem e reconhecimento da estrutura neural de organismos inteligentes. Essa estrutura, conforme a Figura 1, é composta por unidades de processamentos interligadas, conhecidas como neurônios, que recebem e processam sinais de outros neurônios através de suas conexões [4].

Composta por várias unidades de processamento, as RNAs fazem operações apenas sobre seus dados locais, que são entradas recebidas pelas suas conexões. Essas unidades geralmente são conectadas por canais de comunicação que estão associados a valores peso, conforme ilustra a figura 1. A maioria dos modelos de redes neurais possui alguma regra de treinamento, onde os pesos de suas conexões são ajustados de acordo com os padrões de entrada apresentados, ou seja, aprendem através de exemplos.

DOI: http://dx.doi.org/10.17058/tecnolog.v20i2.7428
TECNO-LÓGICA
Revista do depto. de Química e Física, do depto. de Engenharia, Arquitetura e Ciências Agrárias e do Mestrado em Tecnologia Ambiental
http://online.unisc.br/seer/index.php/tecnologica

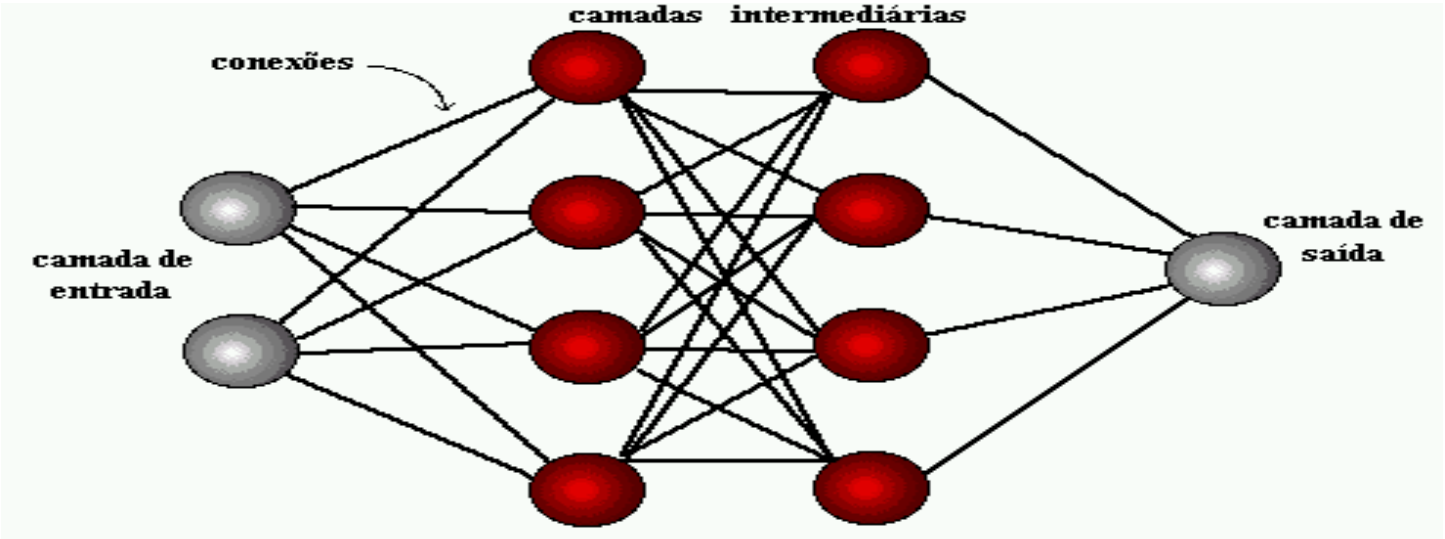

Figura 1 – Exemplo de uma rede neural artificial [3].

A propriedade mais importante das redes neurais é a habilidade de aprender de seu ambiente e, com isso, melhorar seu desempenho. O aprendizado em RNAs está normalmente associado à capacidade de as mesmas adaptarem os seus parâmetros como consequência da sua interação com o meio externo. O processo de aprendizado é interativo e por meio dele a RNA deve melhorar o seu desempenho gradativamente à medida que interage com o meio externo [10].

Pode-se denominar o algoritmo de aprendizado como um conjunto de regras bem definidas para a solução de um problema de aprendizado [3]. Existem muitos tipos de algoritmos de aprendizado específicos para determinados modelos de redes neurais. Estes algoritmos diferem entre si, principalmente, pelo modo como os pesos são modificados. Neste trabalho, foi utilizado o treinamento supervisionado com o algoritmo *backpropagation*.

2.3 Algoritmo Backpropagation

O algoritmo *backpropagation* utiliza treinamento supervisionado, que consiste em fornecer um conjunto de dados de entrada juntamente com a resposta desejada (saída). Os pesos da RNA são ajustados através de um cálculo de correção em função dos erros obtidos e dos pesos atuais das conexões entre os neurônios. Além disso, o algoritmo *backpropagation* permite ser usado em problemas que não são linearmente separáveis, pois permite a existência de camadas intermediárias entre as camadas de entrada e de saída da RNA [12].

O treinamento supervisionado da rede utilizando *backpropagation* consiste em dois passos. No primeiro, um padrão é apresentado às unidades da camada de entrada e, a partir desta camada, as unidades calculam sua resposta que é produzida na camada de saída, o erro é calculado. No segundo passo, o erro é propagado a partir da camada de saída até a camada de entrada, e os pesos das conexões das unidades das camadas intermediárias vão sendo modificados [3].

O trabalho desenvolvido em [13] utiliza o algoritmo *backpropagation* para predição da soroprevalência da Hepatite A. A rede neural artificial projetada, depois de treinada por meio da informação extraída em um modelo de regressão logístico, forneceu a classificação de cada indivíduo (soropositivo ou soronegativo) e a probabilidade do mesmo ser soropositivo. Os resultados do trabalho mostraram que o modelo neural utilizado proporcionou um bom desempenho, alcançando uma eficácia de classificação geral acima de 88%.

3 Ferramenta ADReNA

A ferramenta ADReNA foi desenvolvida por [5]. Tem como objetivo fornecer um ambiente de modelagem e teste de Redes Neurais Artificiais, além de fornecer API (*Application Programming Interface*) para as linguagens de programação Java e .NET que contempla os algoritmos *Backpropagation* e *Kohonen*. Para esta aplicação é utilizado o algoritmo *Backpropagation*, já que o treinamento da rede é supervisionado. O *Kohonen* possui treinamento não supervisionado.

A biblioteca do ADReNA contempla o algoritmo *Backpropagation*, permitindo assim que o usuário se preocupe apenas em definir parâmetros do algoritmo e representar o problema na rede neural artificial. Assim, a utilização da API se mostrou simples e eficiente para a modelagem e testes da aplicação proposta neste trabalho.

A aplicação desenvolvida neste trabalho acoplou a API Java do ADReNA por meio da importação de bibliotecas do

DOI: http://dx.doi.org/10.17058/tecnolog.v20i2.7428
TECNO-LÓGICA
Revista do depto. de Química e Física, do depto. de Engenharia, Arquitetura e Ciências Agrárias e do Mestrado em Tecnologia Ambiental
http://online.unisc.br/seer/index.php/tecnologica

NetBeans, o que abstraiu a implementação da Rede Neural Artificial e do algoritmo *Backpropagation* no desenvolvimento da aplicação, visto que estas funções já estavam implementadas na biblioteca. Assim, foi possível acioná-las sempre que necessárias durante o desenvolvimento.

4 Aplicação Desenvolvida

A aplicação foi desenvolvida utilizando a linguagem de programação JAVA, com a IDE (*Integrated development environment*) *NetBeans*, na qual é importada a API Java da ferramenta ADReNA, o que permite utilizar os recursos do algoritmo *Backpropagation*.

A estrutura da aplicação é composta por classes com funções de leitura do arquivo de entrada de dados da aplicação, transformação dos dados lidos dos arquivos em sequências binárias para treinar a RNA, transformação da sentença informada pelo usuário em uma sequência binária para o reconhecimento e interpretação da saída binária da RNA.

Para utilizar a rede neural artificial *Backpropagation* do ADReNA, é necessário instanciar a classe *Backpropagation*, informando nos parâmetros a quantidade de *bits* da camada de entrada, quantidade de *bits* da camada de saída, quantidade de camadas intermediárias com a quantidade de *bits* de cada camada, taxa de aprendizado e taxa de erro. Em seguida, é necessário informar o conjunto de treinamento, que é uma lista de um conjunto de *bits*, onde cada conjunto desta lista, no caso deste trabalho, representa um homônimo, uma frase de contexto e o significado do homônimo. A quantidade de *bits* representa a quantidade de neurônios necessária para o processamento da RNA para a aplicação definida.

Após definir os parâmetros, é possível treinar a rede neural artificial e reconhecer as sentenças informadas pelo usuário. Sabe-se que uma característica em redes neurais artificiais é a quantidade de tempo alta necessária para treinar um conjunto grandes de dados. Por outro lado, depois de treinada a rede, o tempo de reconhecimento torna-se menor.

A aplicação desenvolvida tem como entrada um conjunto de homônimos que possui dois ou mais significados, e cada significado possui um conjunto de exemplos que será utilizado pela ferramenta para realizar o treinamento da rede neural artificial.

Para mais agilidade na utilização da ferramenta, optou-se por utilizar um arquivo de texto para representar a entrada de dados. Conforma a Figura 2, a estrutura do arquivo é composta por palavra homônima seguida de seus significados, em que cada significado é seguido por seus exemplos. Para diferenciar cada tipo de dado, a palavra homônima não possui caracteres específicos em seu início, os significados de cada homônimo possuem o caractere '#' e os exemplos de cada significado possuem o caractere '*'. O caractere '$' é utilizado para representar palavras que não foram citadas nos exemplos de significados.

```
manga
#manga fruta
*a manga está madura
*vou comer uma manga
#manga de camiseta
*a manga da camiseta está curta
*molhou a manga da camisa
colher
#colher de comer
*vou comer a sopa com colher
*a colher está suja
#colher uma fruta
*vou colher os vegetais
*quero colher algumas frutas
$pegar
$trazer
```

Figura 2 – Exemplo de estrutura do arquivo de entrada de dados.

lo a entrada de dados através do botão "Abrir", conforme a Figura 3 que demonstra a interface da aplicação. À esquerda do botão, existe um campo de texto que informa o caminho do arquivo aberto.

Após abrir o arquivo, são habilitados os botões "Treinar" e "Reconhecer". Também, é carregada a lista de palavras no lado esquerdo da interface. No lado direito da lista é carregada uma tabela contendo os homônimos que estão no arquivo, juntamente com seus significados e exemplos. Na parte inferior, é carregada em um campo de seleção, a lista de homônimos do arquivo.

Depois de aberto o arquivo, é possível pressionar o botão "Treinar", o qual irá fazer o treinamento da rede neural artificial, utilizando os exemplos e significados dos homônimos carregados do arquivo.

Treinada a rede, é possível fazer o reconhecimento. Para isso, é preciso especificar na parte inferior da interface, o homônimo, na lista de seleção, e a frase no campo de texto. Após, é preciso pressionar o botão "Reconhecer". Os resultados são apresentados abaixo, na parte "Saída". São mostrados os valores da camada de saída da rede neural artificial e na parte "Significado" é mostrado o significado reconhecido pela RNA, de acordo com os significados carregados do arquivo de entrada, que foi aprendido pela RNA.

É importante salientar que a aplicação não irá permitir reconhecer palavras que não estejam na lista de palavras carregadas. A lista de palavras é obtida através das palavras utilizadas nos exemplos e das palavras que contêm o caractere '$' em seu início, carregadas do arquivo.

DOI: http://dx.doi.org/10.17058/tecnolog.v20i2.7428

TECNO-LÓGICA
Revista do depto. de Química e Física, do depto. de Engenharia, Arquitetura e Ciências Agrárias e do Mestrado em Tecnologia Ambiental
http://online.unisc.br/seer/index.php/tecnologica

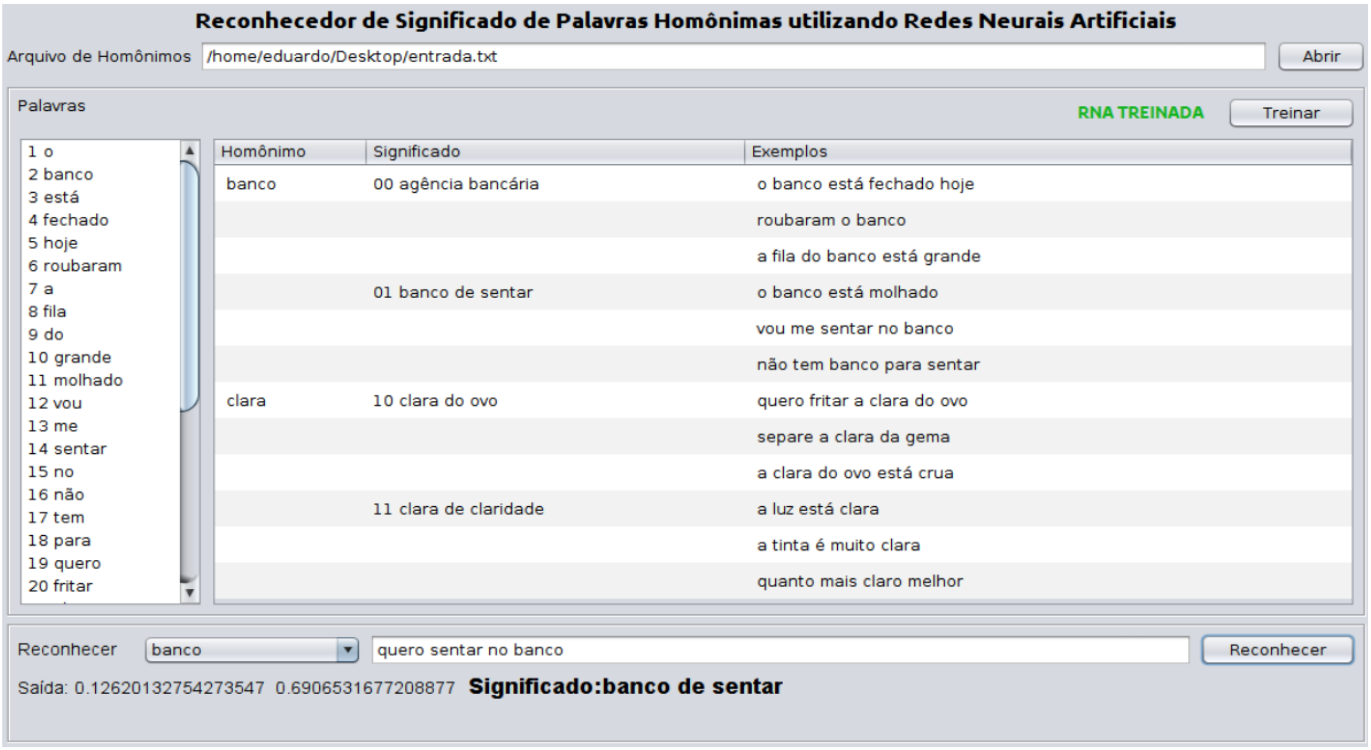

Figura 3 – Interface Gráfica da Ferramenta

4.1 Representação do Problema em uma Rede Neural Artificial

Para representar a camada de entrada da RNA, foram desenvolvidas duas versões da aplicação. Cada versão se diferencia pela maneira como são representados os *bits* da camada de entrada da RNA.

A primeira versão atribui um código binário a cada palavra da lista, assim, a quantidade de *bits* para representar uma palavra varia de acordo com a quantidade de palavras da lista. Por exemplo, na Figura 4 é utilizado uma lista de 32 palavras, logo, serão necessários 5 *bits* para representar cada palavra, já que 2 elevado na potência 5 é igual a 32.

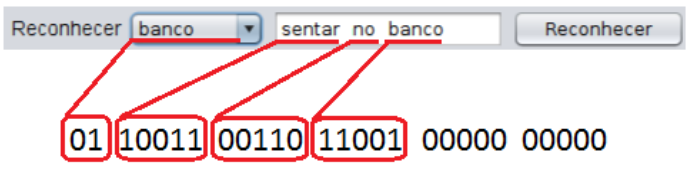

Figura 4 – Aplicação 1: Exemplo de representação do problema na RNA

Também, para a primeira versão, é necessário limitar o número de palavras na camada de entrada da RNA, já que a mesma precisa ter um tamanho fixo. Quando são utilizadas menos palavras do que o limite, a sequência de bits é preenchida com o código da palavra vazia (sequência de zeros). O homônimo que será identificado é apresentado na entrada da RNA. Para representá-lo, é utilizada uma quantidade de *bits* de acordo com a quantidade de homônimos. Logo, a entrada da RNA irá ser formada pelo código em *bits* do homônimo, mais os códigos em *bits* de cada palavra da frase.

Na segunda versão, é utilizado um *bit* para representar cada palavra, sendo assim, se a palavra está contida na frase, o *bit* que a representa recebe o valor 1 (um), caso contrário, recebe o valor 0 (zero), conforme o exemplo da Figura 5. Logo, a entrada da RNA irá ser formada pelo código em *bits* do homônimo, como a representação da primeira versão, e o conjunto de *bits* que representa cada palavra da lista de palavras.

DOI: http://dx.doi.org/10.17058/tecnolog.v20i2.7428
TECNO-LÓGICA
Revista do depto. de Química e Física, do depto. de Engenharia, Arquitetura e Ciências Agrárias e do Mestrado em Tecnologia Ambiental
http://online.unisc.br/seer/index.php/tecnologica

A saída de ambas as versões é o significado do homônimo

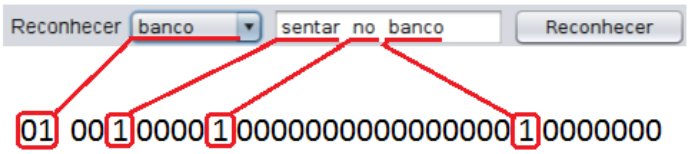

Figura 5 – Aplicação 2: Exemplo de representação do problema na RNA

da frase, apresentados na camada de entrada da RNA. Para representar o significado é atribuído um código binário a ele, logo, se o tamanho da lista de significados for igual a 8, são necessários 3 *bits* para representar a saída da RNA, já que 2 elevado na potência 3 é igual a 8.

5 Testes

Para o primeiro teste, foram utilizados como dados de entrada dois homônimos e atribuídos dois significados para cada um, onde cada significado contém quatro frases para seu treinamento. Logo, para o treinamento da RNA, foram utilizados 16 exemplos. O arquivo para o primeiro teste contém 2 homônimos, sendo necessário, neste caso, apenas um *bit* para representá-los na camada de entrada da RNA. Há 41 palavras diferentes nas 16 frases de exemplos conforme a Figura 6.

O segundo teste utilizou um conjunto de dados semelhante ao primeiro, mas com uma quantidade maior de dados. Para o segundo teste, foram utilizados como dados de entrada quatro homônimos e atribuídos dois significados para cada um, onde cada significado contém oito frases para o seu treinamento. Portanto, o arquivo para o segundo teste contém 4 homônimos, sendo necessário, neste caso, dois *bits* para representá-los na camada de entrada da RNA. Contém 107 palavras diferentes nas 64 frases de exemplos.

A taxa de aprendizado do algoritmo *backpropagation* foi definida em 0,9, já que uma taxa de aprendizagem muito baixa consome um tempo muito grande para o treinamento. Esta taxa pode variar de 0,1 a 0,9 e a decisão da melhor taxa dependerá das características de cada aplicação em específico. A taxa de erro, quando muita baixa, consome um maior tempo no treinamento, entretanto o valor de 0,001 mostrou-se aceitável, não consumindo uma grande quantidade de tempo para o treinamento desta RNA. Uma taxa de erro aceitável, de forma geral, seria de 0,005.

```
banco
#agência bancária
*o banco está fechado hoje
*roubaram o banco
*a fila do banco esta grande
*quero ir ao banco hoje
#banco de sentar
*o banco está molhado
*vou me sentar no banco
*não tem banco para sentar
*quero me sentar no banco
clara
#clara do ovo
*quero fritar a clara do ovo
*separe a clara da gema
*a clara do ovo está crua
*vou cozinhar uma clara de ovo
#clara de claridade
*a luz está clara
*a tinta é muito clara
*quanto mais clara melhor
*não quer uma luz clara
$gelado
$sem
```

Figura 6 – Dados de entrada para o primeiro teste.

Para ambas as versões da aplicação desenvolvida, o arquivo do primeiro teste contém 4 significados, dois para cada homônimo, logo, são necessários 2 *bits* para representar a camada de saída. Para o arquivo do segundo teste, são necessários 3 *bits* para representar os 8 significados, dois para cada homônimo.

Foram realizadas seis configurações diferentes para cada versão da aplicação, sendo 8 reconhecimentos no primeiro teste e 16 reconhecimentos no segundo. O primeiro teste totalizou 48 reconhecimentos para cada versão da aplicação, enquanto que o segundo teste totalizou 96 reconhecimentos para cada versão. Os resultados armazenados foram o número de erros e acertos de cada versão, além da média aritmética da diferença absoluta dos *bits* de saída da RNA com os *bits* de saída esperados. As configurações variam entre o número de camadas intermediárias e a quantidade de neurônios por camada, sendo que as configurações usadas em ambas as versões e testes foram as mesmas.

5.1 Teste da Primeira Versão

A primeira versão da aplicação se diferencia da segunda por atribuir a cada palavra um código diferente. A entrada da RNA é formada pelo código em *bits* do homônimo, mais os códigos de cada palavra da frase. Exemplo: a palavra “sentar” é representada pelos bits “010001” enquanto que a palavra “no” é representada pelos bits “010010”.

Como a lista de palavras do primeiro arquivo de teste é igual a 41, é necessário no mínimo 6 *bits* (2^6 = 64) para

DOI: http://dx.doi.org/10.17058/tecnolog.v20i2.7428
TECNO-LÓGICA
Revista do depto. de Química e Física, do depto. de Engenharia, Arquitetura e Ciências Agrárias e do Mestrado em Tecnologia Ambiental
http://online.unisc.br/seer/index.php/tecnologica

representar cada palavra. Já para o segundo arquivo de teste, é necessário no mínimo 7 *bits* (2^7 = 128) para representar as 107 palavras. Nesta versão, foi definido que o número máximo de palavras por frase no teste seria igual a 10. Logo, a camada de entrada da RNA é composta pelos *bits* que representam o homônimo mais os *bits* que representam a frase. Assim, a camada de entrada da RNA do primeiro teste contém 61 *bits* e do segundo contém 72 *bits*.

Descreve-se um exemplo de entrada na RNA: A frase "sentar no banco" é representada por "0 010001 010010 000010 000000 00000 000000 00000 000000 000000 00000" no primeiro teste, na qual o primeiro *bit* representa o homônimo "banco", seguido da primeira sequência de 6 *bits* que representa a palavra "sentar", a sequência seguinte representa a palavra "no" e a próxima a palavra "banco". A última sequência de "000000" representa palavras vazias, já que o número de palavras na entrada da RNA é 10.

5.2 Teste da Segunda Versão

A segunda versão da aplicação se diferencia da primeira por utilizar apenas um *bit* para representar se a palavra está na frase ou não. A entrada da RNA é formada pelo código em *bits* do homônimo e o conjunto de *bits* que representa cada palavra da lista de palavras.

Como a lista de palavras do primeiro arquivo de teste é igual a 41, são necessários 41 *bits* para representar todas as palavras e 107 *bits* para representar as 107 palavras do segundo arquivo de teste. Logo, a camada de entrada da RNA conterá os *bits* que representam o homônimo mais os *bits* que representam a frase. Assim, a camada de entrada do primeiro teste contém 42 *bits* e do segundo 109 *bits*.

Descreve-se um exemplo de entrada na RNA: A frase "sentar no banco" é representada por "0 0100000000000001100000000000000000000000" no primeiro teste, na qual o primeiro *bit* representa o homônimo "banco" e o restante dos *bits* representam cada palavra. Assim, a palavra "sentar" é representada pelo 18º *bit*, a palavra "no" pelo 19º *bit* e a palavra "banco" pelo 3º *bit*.

Esta versão da aplicação tem a vantagem de não definir um número limite de palavras para treinamento ou reconhecimento, porém, não diferencia na camada de entrada da RNA a ordem que as palavras estão e nem se uma mesma palavra repete na frase.

6 Resultados Obtidos

Ao realizar o teste da primeira versão com o primeiro arquivo de teste, obtiveram-se os resultados apresentados na Tabela 1.

Tabela 1 - Resultados da versão 1 da aplicação com o primeiro arquivo de teste.

Número de neurônios das camadas da RNA: entrada-intermediárias-saída	Acertos de Saída	Erros de Saída	Média de Erro por bit de saída da RNA
61-25-2	4	4	0,2673
61-40-2	5	3	0,2707
61-20-40-2	4	4	0,2920
61-40-20-2	6	2	0,2611
61-50-30-20-2	6	2	0,2530
61-70-50-30-10-2	3	5	0,3547
Totais	**28**	**20**	**0,2831**

Ao realizar o teste da primeira versão com o segundo arquivo de teste, obtiveram-se os resultados apresentados na Tabela 2.

Tabela 2 - Resultados da versão 1 da aplicação com o segundo arquivo de teste.

Número de neurônios das camadas da RNA: entrada-intermediárias-saída	Acertos de Saída	Erros de Saída	Média de Erro por bit de saída da RNA
72-25-3	14	2	0,1059
72-40-3	11	5	0,1427
72-20-40-3	13	3	0,1171
72-40-20-3	12	4	0,1651
72-50-30-20-3	12	4	0,1208
72-70-50-30-10-2	10	6	0,1327
Totais	**72**	**24**	**0,1307**

Conforme as Tabelas 1 e 2, os resultados mostraram que das 48 frases reconhecidas no primeiro teste, 28 tiveram seu significado identificado corretamente e 20 incorretamente, e das 96 frases reconhecidas no segundo teste, 72 tiveram seu significado reconhecido corretamente e 24 incorretamente. A média aritmética da diferença absoluta dos *bits* de saída da RNA com os *bits* de saída esperados do primeiro teste foi igual a 0,2831 e do segundo teste foi igual a 0,1307.

Ao realizar o teste da segunda versão com o primeiro arquivo de teste, obtiveram-se os resultados apresentados na Tabela 3.

Tabela 3 - Resultados da versão 2 da aplicação com o primeiro arquivo de teste.

Número de neurônios das camadas da RNA: entrada-intermediárias-saída	Acertos de Saída	Erros de Saída	Média de Erro por bit de saída da RNA
42-25-2	8	0	0,1184
42-40-2	7	1	0,1283
42-20-40-2	7	1	0,1069
42-40-20-2	7	1	0,1008
42-50-30-20-2	7	1	0,1043
42-70-50-30-10-2	7	1	0,0998
Totais	**43**	**5**	**0,1098**

Ao realizar o teste da segunda versão com o segundo arquivo de teste, obtiveram-se os resultados apresentados na Tabela 4.

DOI: http://dx.doi.org/10.17058/tecnolog.v20i2.7428
TECNO-LÓGICA
Revista do depto. de Química e Física, do depto. de Engenharia, Arquitetura e Ciências Agrárias e do Mestrado em Tecnologia Ambiental
http://online.unisc.br/seer/index.php/tecnologica

Tabela 4 - Resultados da versão 2 da aplicação com o segundo arquivo de teste.

Número de neurônios das camadas da RNA: entrada-intermediárias-saída	Acertos de Saída	Erros de Saída	Média de Erro por bit de saída da RNA
109-25-3	14	2	0,0698
109-40-3	15	1	0,0533
109-20-40-3	15	1	0,0535
109-40-20-3	15	1	0,0542
109-50-30-20-3	14	2	0,0547
109-70-50-30-10-3	12	4	0,0852
Totais	**85**	**11**	**0,0618**

Conforme as Tabelas 3 e 4, os resultados mostraram que das 48 frases reconhecidas, 43 tiveram seu significado identificado corretamente e 5 incorretamente, e das 96 frases reconhecidas no segundo teste, 85 tiveram seu significado reconhecido corretamente e 11 incorretamente. A média aritmética da diferença absoluta dos *bits* de saída da RNA com os *bits* de saída esperados do primeiro teste foi igual a 0,1098 e do segundo foi igual a 0,0618.

Fica evidente que a segunda versão da aplicação obteve resultados superiores, comparado com os resultados da primeira versão. A média de erro por *bit* de saída da RNA na primeira versão foi maior que o dobro do valor médio de erros que a segunda versão obteve, o que refletiu na quantidade de acertos e erros de reconhecimento da RNA.

As configurações de camadas intermediárias mostraram que para a primeira versão da aplicação, as melhores configurações são de três camadas intermediárias com 50, 30 e 20 neurônios respectivamente em cada e uma camada intermediária com 25 neurônios. Já a segunda versão mostrou que as melhores configurações foi a de quatro camadas intermediárias com 70, 50, 30, e 10 neurônios respectivamente em cada e uma camada intermediária com 40 neurônios.

7 Conclusão

Neste trabalho foram desenvolvidas duas aplicações que utilizam redes neurais artificiais para auxiliar no entendimento do significado de palavras ambíguas da língua portuguesa, que dependem necessariamente do contexto para serem identificadas.

Ao desenvolver duas versões e submetê-las a testes, obtiveram-se resultados que levaram a concluir que a forma que os dados são representados na rede neural artificial causa influência em seus resultados, já que uma das versões obteve resultados claramente superiores, pois a média de erro por *bit* de saída da RNA da primeira versão foi maior que o dobro do valor médio de erros que a segunda versão obteve Isso resultou em uma quantidade maior de erros da versão 1 e uma quantidade maior de acertos da versão 2.

O objetivo dos testes é verificar o desempenho das duas aplicações desenvolvidas no reconhecimento do significado de palavras homônimas em uma frase. Assim, os resultados mostram que a primeira versão da aplicação tem a vantagem de levar em consideração a ordem que as palavras estão estruturadas na frase, o que muitas vezes identifica com maior facilidade o significado de um homônimo. Porém, esta versão, quando utiliza um conjunto de dados pequeno para o treinamento, apresenta um resultado ruim quanto ao reconhecimento, pois é necessário que o conjunto de treinamento tenha uma variação na ordem das palavras na frase.

Os resultados da segunda versão permitem concluir que, mesmo com um conjunto pequeno de treinamento para a RNA, os resultados de reconhecimento são satisfatórios, já que a maneira de representar os dados de entrada na RNA é mais simples. Porém, quando utiliza um conjunto de treinamento maior, a camada de entrada irá aumentar em relação a palavras novas do conjunto.

Ao realizar os testes com dois conjuntos de dados de entrada, é possível concluir que a primeira versão da aplicação depende de um conjunto maior no seu treinamento para alcançar um resultado satisfatório, enquanto que a segunda versão apresenta resultados superiores independentemente da quantidade de dados para o treinamento da RNA. Logo, a segunda versão se mostrou promissora no reconhecimento de significados de palavras homônimas.

Em relação aos trabalhos relacionados com a utilização de RNA em processamento de linguagem natural [1], [6] e [9], citados na seção 2 deste artigo, destaca-se que este trabalho contribui com uma aplicação diferente das demais, o reconhecimento de homônimos, e o uso da ferramenta ADReNA, como alternativa a outras ferramentas de RNA, pelas suas características de usabilidade, facilitando a tarefa de modelagem e implementação de uma RNA.

DEVELOPMENT OF AN APPLICATION TO RECOGNIZE WORD MEANINGS HOMONYMOUS USING ARTIFICIAL NEURAL NETWORKS

ABSTRACT: This work presents the development of an application that assist in understanding of the meaning of ambiguous words of the Portuguese language, which necessarily depend on the context to be identified. The application was implemented in Java programming language using the NetBeans IDE, along with the ADReNA tool that allowed abstracting the implementation of an Artificial Neural Network (ANN) and enabled model, train and parameterizes a backpropagation ANN. Two versions of the tool have been developed, which differ by the way they represent the input layer of the neural network. Tests have shown that the way the neural network is represented brings great influence on its results. Although the tests have a less significant volume of information, the application showed very promising for the recognition meanings of homonyms.

DOI: http://dx.doi.org/10.17058/tecnolog.v20i2.7428

Keywords: Artificial Neural Networks. Ambiguous Words. ADReNA Tool.

Referências

[1] BONFANTE, Andréia Gentil; NUNES, Maria das Graças Volpe. Correção Gramatical da Crase usando Redes Neurais. III Encontro para o Processamento Computacional do Português Escrito e Falado, Porto Alegre, RS. Pág. 10-16. 1998.

[2] CHOWDHURY, Gobinda G. Natural Language Processing. Annual Review of Information Science and Technology, Vol. 37, Pág. 51-89, 2003.

[3] ICMP-USP. Instituto de ciências matemáticas e computação. Redes neurais artificiais. Disponível em: http://www.icmc.usp.br/~andre/research/neural/#cara. Acessado em: setembro de 2015.

[4] KARTALOPAULOS, Stamatios. Understandind Neural Networks and Fuzzy Logic: basic concepts and applications. New York: IEEE Press, 1996.

[5] KIST, Matheus Henrique; FROZZA, Rejane. ADReNA – Ambiente de Desenvolvimento de Aplicações em Redes Neurais Artificiais. Santa Cruz do Sul: UNISC. 2013. (Trabalho de Conclusão de Curso).

[6] MENDES, E. R. G.; ROSA J. L. G. O Estudo de Redes Neurais para o Processamento da Língua Natural. Anais do XIII Encontro de Iniciação Científica da PUC-Campinas. 2008.

[7] NORMA CULTA. Palavras Homônimas. Norma Culta, Gramática Online da Língua Portuguesa. Disponível em http://www.normaculta.com.br/palavras-homonimas/. Acessado em: novembro de 2015.

[8] OLIVEIRA, João Mendes; TONIN, Sávio Duarte; PRIETCH, Soraia Silva. Processamento de Linguagem Natural e suas Aplicações Computacionais. Escola Regional de Informática, ERIN. 2010.

[9] PARDO, Thiago A. S.; RINO, Lucia H. M.; NUNES, Maria das Graças Volpe. NeuralSumm: Uma Abordagem Conexionista para a Sumarização Automática de Textos. Anais do IV Encontro Nacional de Inteligência Artificial, Campinas-SP. 2003.

[10] REZENDE, Solange Oliveira. Sistemas Inteligentes Fundamento e Aplicações. Barueri: Manole, 2003. 525 p.

[11] SOUSA, Roberta. Info Escola: Parônimos e Homônimos. Disponível em: http://www.infoescola.com/portugues/paronimos-e-homonimos/. Acessado em: novembro de 2015.

[12] BARCA, Maria C. S.; SILVEIRA, Tiago R. S.; MAGINI, Marcio; Treinamento de Redes Neurais Artificiais: O Algoritmo Backpropagation. IX Encontro Latino Americano de Iniciação Cientifica. 2005. Pág. 46-49.

[13] SANTOS, Alcione M.; SEIXAS, José M.; PEREIRA, Brasílio B.; MEDRONHO, Roberto A. Usando Redes Neurais Artificias e Regressão Logística na Predição da Hepatite A. Revista Brasileira de Epidemiologia. 2005. Pág. 117-26.

[14] SUPTITZ, Ivan Luis; FROZZA, Rejane; MOLZ, Rolf Fredi. Análise Comparativa de Ferramentas de Redes Neurais Artificiais. XXXV Encontro Nacional de Engenharia de Produção (ENEGEP). 2015. Pág. 1-12.