UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

INDUÇÃO DE MODELOS DE ÁRVORE DE DECISÃO PARA RECONHECIMENTO DE ENTIDADES NOMEADAS NA LITERATURA BIOMÉDICA

TRABALHO DE GRADUAÇÃO

Matheus Miller de Campos Viana

Santa Maria, RS, Brasil

2013

INDUÇÃO DE MODELOS DE ÁRVORE DE DECISÃO PARA RECONHECIMENTO DE ENTIDADES NOMEADAS NA LITERATURA BIOMÉDICA

Matheus Miller de Campos Viana

Trabalho de Graduação apresentado ao Curso de Ciência da Computação da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para a obtenção do grau de

Bacharel em Ciência da Computação

Orientadora: Profª. Drª. Ana Trindade Winck

Trabalho de Graduação N. 356
Santa Maria, RS, Brasil

2013

Universidade Federal de Santa Maria
Centro de Tecnologia
Curso de Ciência da Computação

A Comissão Examinadora, abaixo assinada,
aprova o Trabalho de Graduação

INDUÇÃO DE MODELOS DE ÁRVORE DE DECISÃO PARA RECONHECIMENTO DE ENTIDADES NOMEADAS NA LITERATURA BIOMÉDICA

elaborado por
Matheus Miller de Campos Viana

como requisito parcial para obtenção do grau de
Bacharel em Ciência da Computação

COMISSÃO EXAMINADORA:

Ana Trindade Winck, Drª.
(Presidente/Orientadora)

Giovani Rubert Librelotto, Dr. (UFSM)

Luis Alvaro de Lima Silva, Dr. (UFSM)

Santa Maria, 20 de Fevereiro de 2013.

AGRADECIMENTOS

Em primeiro lugar, eu quero agradecer à minha família que sempre esteve ao meu lado. Ao meu pai Miller, à minha mãe Luciane e ao meu irmão Thiago, que nos momentos de incerteza sempre me encorajaram e confiaram na minha capacidade. Também aos meus tios Mauren e Everton, que foram meus companheiros desde o início dessa caminhada, ajudando-me a passar por cada um dos obstáculos que, por diversas vezes, surgiam. Não posso esquecer da minha vó Olívia, que é o meu maior exemplo de força e de superação, que sempre passou por todas as dificuldades que a vida colocou em seu caminho, e me ensinou que não se deve desistir daquilo que queremos. Aos meus primos Alisson, Gustavo e Guilherme, que foram meus companheiros de futebol, música e festa nos momentos críticos de final de semestre, porque, às vezes, a mente cansa e é preciso se desligar de tudo.

Quero agradecer também aos colegas de faculdade, Kevin, Lucas, Bruno, Davi, Hugo, Leonardo, Vitor, Matheus e Felipe; e de estágio, Cristiano, Daiane, Kérolen, Natália, Ariane, Sarah e Vanessa; que acompanharam esta caminhada e hoje os considero grandes amigos. Quero agradecer em especial à professora Ana pelo acompanhamento e pelo apoio quando as dificuldades surgiram durante o desenvolvimento deste trabalho. Enfim, quero agradecer a todos os demais familiares, amigos, colegas, professores, conhecidos que de alguma forma contribuíram para que eu alcançasse o meu objetivo. Muito obrigado a todos!

“Você nem sempre consegue aquilo que quer, mas se correr atrás, você pode alcançar o que você precisa.”

— MICK JAGGER E KEITH RICHARDS

RESUMO

Trabalho de Graduação
Curso de Ciência da Computação
Universidade Federal de Santa Maria

INDUÇÃO DE MODELOS DE ÁRVORE DE DECISÃO PARA RECONHECIMENTO DE ENTIDADES NOMEADAS NA LITERATURA BIOMÉDICA

AUTOR: MATHEUS MILLER DE CAMPOS VIANA
ORIENTADORA: ANA TRINDADE WINCK
Local da Defesa e Data: Santa Maria, 20 de Fevereiro de 2013.

O aumento da quantidade de documentos textuais, especialmente os relacionados à literatura biomédica, tem encorajado muitas pesquisas em Mineração de Textos. Um importante campo de pesquisa diz respeito ao Reconhecimento de Entidades Nomeadas (REN), onde Entidades Nomeadas (EN) são termos ou objetos em um dado contexto. No domínio biomédico, doenças e tratamentos são exemplos de Entidades Nomeadas. A identificação dessas Entidades Nomeadas se tornou um desafio, uma vez que *corpora* biomédicos possuem características particulares, principalmente porque um objeto biomédico pode ser, muitas vezes, representado de diferentes maneiras. Dentre os diferentes métodos de REN, destaca-se o reconhecimento através do contexto. Neste trabalho, é proposta uma metodologia baseada em Árvores de Decisão para o REN na literatura biomédica.

Palavras-chave: Mineração de Textos. Árvore de Decisão. Reconhecimento de Entidades Nomeadas.

ABSTRACT

Undergraduate Final Work
Undergraduate Program in Computer Science
Federal University of Santa Maria

A DECISION-TREE MODEL-BASED APPROACH FOR NAMED ENTITIES RECOGNITION IN BIOMEDICAL LITERATURE

AUTHOR: MATHEUS MILLER DE CAMPOS VIANA
ADVISOR: ANA TRINDADE WINCK
Defense Place and Date: Santa Maria, February 20th, 2013.

The increasing amount of textual documents, especially those related to biomedical literature, has encouraged many researches in Text Mining. One important field of investigation relates to Named Entities Recognition (NER), where Named Entities (NE) are referred terms or objects in a given context. In the biomedical domain, diseases and treatments can be cited as examples of NE. The recognition of biomedical NE has become a challenge, since biomedical *corpora* have particular characteristics, mainly because a given biological object can be often represented in different terminological ways. Among the different methods of NER, one of them is the recognition through the context. In this work is proposed a Decision-Tree Model-based approach for NER in biomedical literature.

Keywords: Text Mining. Decision Tree. Named Entities Recognition.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE APÊNDICES

LISTA DE ABREVIATURAS E SIGLAS

ARFF	*Attribute-Relation File Format*
CSV	*Comma Separated Values*
EN	Entidade Nomeada
KDD	*Knowledge Discovery from Data*
KDT	*Knowledge Discovery in Texts*
MD	Mineração de Dados
MT	Mineração de Textos
PLN	Processamento de Linguagens Naturais
REN	Reconhecimento de Entidades Nomeadas
TF-IDF	*Term Frequency - Inverse Document Frequency*
TR	*Text Retrieval*
WEKA	*Waikato Environment for Knowledge Analysis*
XML	*eXtensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

Com os avanços tecnológicos da era digital, o custo para armazenamento de dados diminuiu consideravelmente, fazendo com que a quantidade de dados em meios digitais crescesse exponencialmente. Isso contribiu para o aumento da complexidade para encontrar informações relevantes nesses conjuntos de dados (*datasets*). Assim, faz-se necessário o desenvolvimento de metodologias que possibilitem a análise desses grandes conjuntos de dados. Mineração de Dados é uma área de pesquisa em Inteligência Artificial que compreende um conjunto de técnicas e algoritmos que auxilia no desenvolvimento dessas metodologias.

Especialmente na área biomédica, os dados estão, muitas vezes, disponíveis de maneira não-estruturada, isto é, armazenados em textos. Dessa forma, muitas pesquisas para a área biomédica concentram-se em uma das ramificações da Mineração de Dados: a Mineração de Textos. Para a descoberta de conhecimento nos textos biomédicos, alia-se a Mineração de Textos ao Processamento de Linguagens Naturais. Isso é necessário pois a maioria desses documentos textuais de interesse da área biomédica dizem respeito a artigos científicos. Dessa forma, por se tratar de documentos escritos em linguagem natural, carregam em seu conteúdo particularidades da língua, bem como termos e objetos com características particulares da área. Muitos desses termos e objetos podem ser tratados como Entidades Nomeadas, ou seja, termos referenciados em um determinado contexto.

1.1 Motivação

No domínio biomédico, doenças e tratamentos são exemplos de Entidades Nomeadas. Para o caso de doenças, tem-se em textos científicos algumas maneiras diferentes de representar um mesmo termo. Por exemplo, o termo *gm1-gangliosidosis*, que diz respeito a uma doença lisossômica, também pode ser chamado de *lysosomal storage disorder* e *gangliosidoses* (JIMENO et al., 2008).

No parágrafo anterior, os termos destacados em itálico são exemplos de como um mesmo objeto (no caso, uma determinada doença) pode ser representado de diferentes modos. Nesse sentido, o reconhecimento automatizado de sentenças de artigos científicos que carregam estes termos em seu conteúdo pode ajudar especialistas da área a identificar mais facilmente a ocorrência de determinadas doenças e seus tratamentos, bem como a existência de novos termos para um mesmo assunto.

1.2 Objetivos

Este trabalho propõe uma metodologia para processamento de um *corpus* (coleção de documentos) da área biomédica para a identificação de Entidades Nomeadas a partir de modelos de Árvore de Decisão.

A base de análise deste trabalho é composta por um *corpus* estruturado de maneira similar à organização de documentos XML, que contém sentenças de artigos científicos da área biomédica, descritos em (JIMENO et al., 2008). Este *corpus* reúne sentenças pré-processadas e anotadas do PubMed – um indexador de citações e alguns artigos completos da literatura biomédica.

1.3 Estrutura do Texto

Este trabalho está organizado em seis capítulos conforme expresso a seguir. O capítulo 2 traz uma fundamentação teórica acerca dos assuntos abordados no trabalho, tratando da Mineração de Dados e do Processamento de Linguagens Naturais. O capítulo 3 fala sobre o desenvolvimento do trabalho, descrevendo passo a passo a metodologia proposta. O capítulo 4 apresenta os resultados obtidos através de testes utilizando a metodologia proposta. O capítulo 5 expõe alguns trabalhos relacionados ao presente trabalho. Por fim, o capítulo 6 conclui o trabalho, fazendo as considerações finais e indicando possíveis trabalhos futuros, e o apêndice A apresenta os pseudocódigos das rotinas utilizadas para o desenvolvimento da metodologia.

2 FUNDAMENTAÇÃO

O processo de Reconhecimento de Entidades Nomeadas é realizado a partir da combinação de técnicas de Mineração de Dados e Processamento de Linguagens Naturais. Neste capítulo, são apresentados os conceitos que servem de base para o entendimento do que é Mineração de Dados e Processamento de Linguagens Naturais.

2.1 Mineração de Dados

Mineração de Dados (MD) é o processo de extrair conhecimento a partir de grandes conjuntos de dados. Outro termo popular para descrever este processo é Descoberta de Conhecimento a partir de Bases de Dados (de *Knowledge Discovery in Databases*, ou KDD) (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996) (HAN; KAMBER, 2005).

O processo consiste em uma sequência de sete passos, conforme ilustrado na Figura 2.1.

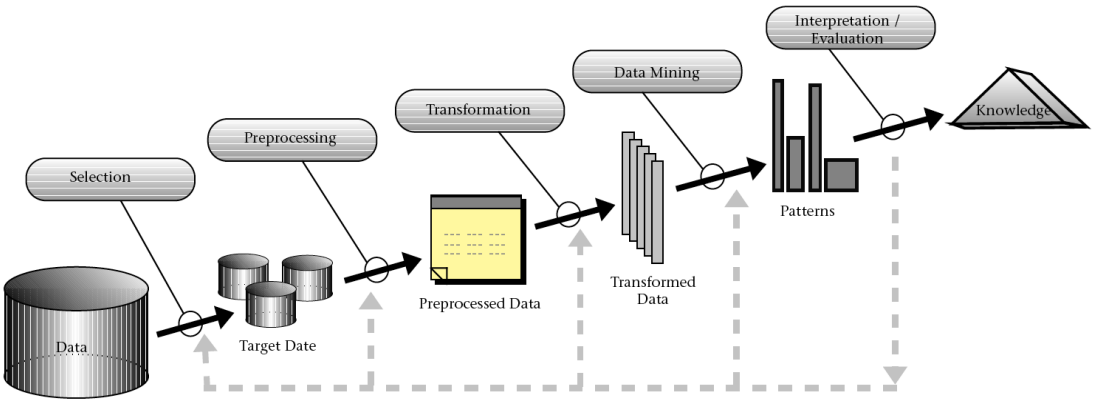

Figura 2.1 – Descrição do processo de KDD. Adaptado de (HAN; KAMBER, 2005)

Esses passos podem ser sumarizados da seguinte maneira:

- Limpeza: remoção de ruídos e dados inconsistentes;
- Integração: combinação de diferentes fontes de dados;
- Transformação: transformação dos dados para um formato apropriado para a mineração;
- Seleção: seleção de dados relevantes para o objeto da análise;

- Mineração: aplicação de algoritmos de Inteligência Artificial para extrair padrões da base de dados;
- Avaliação dos padrões: identificação dos padrões relevantes para o objeto da análise com base em métricas de confiança e suporte, entre outras;
- Apresentação do conhecimento.

Os quatro primeiros passos correspondem ao preprocessamento de dados, fase em que o conjunto de dados é preparado para a mineração. A mineração de dados propriamente dita é apenas um passo dentro do processo inteiro, mas é durante esta etapa do processo que os algoritmos de aprendizagem de máquina são aplicados para que padrões não-triviais essenciais para a análise sejam descobertos.

2.1.1 Preprocessamento

As bases de dados existentes atualmente estão sujeitas a ruídos, a falta ou inconsistência de dados e a outros vários problemas. Isso devido à grande quantidade de dados que são armazenados e às diferentes fontes de onde esses dados são extraídos. Dados de baixa qualidade levam a resultados de baixa qualidade. Assim, preprocessar os dados passa a ser uma parte de suma importância na tarefa de minerar dados (HAN; KAMBER, 2005).

Existem diversas etapas de preprocessamento de dados. Essas podem ser aplicadas individualmente ou de forma combinada, dependendo das características da base de dados sendo utilizada, bem como das reais necessidades em relação à tarefa de mineração. Essas etapas são descritas a seguir.

- Limpeza dos dados: Limpeza dos dados é um tratamento realizado sobre os dados para assegurar a qualidade dos fatos que eles representam. Esta etapa do preprocessamento de dados é realizada a partir da identificação de ruídos (ou faltas, ou inconsistências) nos dados e, se existentes, da remoção ou, se possível, da correção desses dados.
- Transformação e integração dos dados: Muitas vezes, uma mesma entidade do mundo real pode ter valores de atributo diferentes em bases de dados diferentes. Nesta etapa, os dados são combinados para possuírem um valor único e coerente e, assim, facilitar o processo de mineração.

- Seleção dos atributos: Dependendo do objetivo da análise, alguns dos dados não possuem relevância para a indução dos modelos. Assim, é necessária a aplicação de medidas para determinar quais são os dados relevantes para se alcançar o objetivo da análise. Seleção dos atributos é a etapa do preprocessamento de dados em que essas medidas são calculadas.

2.1.2 Tarefas de Mineração de Dados

Tarefas de MD são utilizadas para extrair conhecimento de grandes volumes de dados (TAN; STEINBACH; KUMAR, 2006). Para que sejam minerados, esses dados são tipicamente organizados em um *dataset* composto por diferentes atributos que representam o domínio do conjunto de dados sendo minerados, sendo cada instância um exemplo para o *dataset*. A relação atributo-valor diz respeito à característica de um determinado atributo para um dado exemplo.

Construir um *dataset* apropriado pode, também, ser um dos desafios em minerar dados. Isso porque a disposição adequada dos atributos para descrever o problema é essencial para que os algoritmos de mineração cumpram seu objetivo. Tendo um *dataset* bem definido, parte-se para a escolha dos algoritmos de MD a serem aplicados. As principais técnicas de MD são classificação, regressão, associação e agrupamento. Neste trabalho, é utilizada a técnica de classificação.

A técnica de classificação consiste, basicamente, em prever a classe à qual uma instância pertence, onde esse processo de predição pode ser feito localizando as regras que particionam o conjunto de dados em grupos disjuntos. Para tanto, tem-se um *dataset* composto por diferentes atributos preditivos, os quais descrevem o conjunto de dados; e por um atributo de interesse, denominado atributo-alvo ou atributo-classe.

Dentre as diferentes técnicas de classificação existentes, pode-se destacar a classificação com Árvores de Decisão. Trata-se de uma técnica cujo modelo resultante é apresentado na forma de uma árvore, sendo de fácil interpretação, se comparado às técnicas baseadas em modelos caixa-preta, como *Support Vector Machine*, Redes Neurais, entre outras (FREITAS; WIESER; APWEILER, 2010).

2.1.2.1 Árvores de Decisão

Os classificadores de Árvore de Decisão (QUINLAN, 1986) constroem uma árvore, sendo que cada folha possui uma classe associada e cada nó, um predicado associado a ele. Para

classificar uma instância, devem-se realizar testes nos valores dos atributos nos nós da árvore, começando na raiz e terminando em uma folha, que representa a classe prevista para a instância (TAN; STEINBACH; KUMAR, 2006). A Figura 2.2 ilustra esse conceito. No exemplo, tem-se um conjunto de dados com seis exemplos e cinco atributos, onde os quatro primeiros são atributos preditivos que indicam sintomas de um dado paciente, e o último é o atributo-classe, que indica se o paciente está doente ou saudável. Para esse conjunto de dados, é induzida uma árvore de decisão que faz testes em relação aos sintomas para predizer a condição de saúde do paciente.

Exemplo	Febre	Enjôo	Manchas	Dor	Diagnóstico
T1	sim	sim	pequenas	sim	doente
T2	não	não	grandes	não	saudável
T3	sim	sim	pequenas	não	saudável
T4	sim	não	grandes	sim	doente
T5	sim	não	pequenas	sim	saúdavel
T6	não	não	grandes	sim	doente

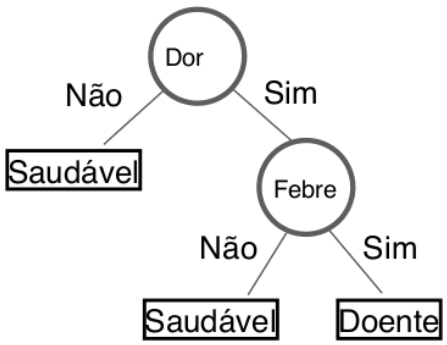

Figura 2.2 – Exemplo de árvore de decisão

O algoritmo básico para indução de árvore de decisão, é o algoritmo de Hunt. Esse algoritmo serviu de base para a construção de algoritmos mais bem elaborados, como é o caso do ID3 (QUINLAN, 1986) e do C4.5 (QUINLAN, 1993). Os passos básicos para indução de árvore, segundo o algoritmo de Hunt, são os seguintes:

1. Escolha um atributo;
2. Estenda a árvore adicionando um ramo para cada valor do atributo;
3. Considerando o atributo escolhido, passe os exemplos para as folhas;
4. Para cada folha:
   (a) Se todos os exemplos pertencerem ao mesmo atributo alvo, associe este atributo à folha;
   (b) Senão, repita os passos de 1 a 4.

2.1.2.2 Métricas de avaliação

A qualidade dos modelos induzidos por algoritmos de árvore de decisão pode ser avaliada a partir de diferentes métricas, dentre as quais pode-se destacar:

- Acurácia
- Curva ROC
- Tamanho da árvore

A acurácia de um modelo representa o quão satisfatória foi a classificação. Uma das maneiras de avaliar a acurácia é a partir de uma medida chamada validação cruzada (WITTEN; FRANK; HALL, 2011), a qual divide o conjunto de dados em n partes iguais, utilizando $n - 1$ para treino e 1 para teste, n vezes. Com esse método, tem-se as instâncias que foram preditas corretamente, onde se obtém:

- VP - Instâncias classificadas como Verdadeiro Positivo;
- VN - Instâncias classificadas como Verdadeiro Negativo;
- FP - Instâncias classificadas como Falso-Positivo;
- FN - Instâncias classificadas como Falso-Negativo.

Com essas métricas, é possível calcular a acurácia, conforme ilustrado na fórmula 2.1:

$$\text{Acurácia} = \frac{\text{Predições Corretas}}{\text{Total de Predições}} = \frac{VP + VN}{VP + VN + FP + FN} \tag{2.1}$$

A Curva ROC (*Receiver Operation Characteristics Curve*) é um enfoque que representa um *trade-off* entre as taxas de VP (VPR) e FP (FPR) de um classificador (Fórmulas 2.2 e 2.3), de modo que o ideial é que VPR seja próximo de 1 e FPR próximo de 0.

$$VPR = \frac{VP}{VP + FN} \tag{2.2}$$

$$FPR = \frac{FP}{VP + FP} \tag{2.3}$$

A curva ROC é utilizada para medir a performance relativa de diferentes classificadores, em especial quando há classes desbalanceadas. Assim, a área abaixo da curva ROC, denominada de AUC (*Area Under Curve*), fornece a medida que compara as performances do classificador. Quanto maior o valor de AUC (mais próximo de 1), melhor a performance global do classificador.

Por fim, o tamanho da árvore é uma métrica importante a ser avaliada, pois quanto menor a árvore induzida, mais fácil de interpretar e de aplicar é o modelo.

2.2 Processamento de Linguagens Naturais

Processamento de Linguagens Naturais (PLN) é uma área da Ciência da Computação que estuda o desenvolvimento de técnicas e algoritmos que analisam, reconhecem ou geram textos em linguagens humanas (LOPES; VIEIRA, 2010). Como as linguagens naturais possuem muitas ambiguidades, o PLN se torna diferente do processamento de linguagens de programação, que são definidas para evitar essas ambiguidades. Na literatura biomédica, exemplos de ambiguidade são *melanoma superficial* e *melanoma nodular*. Mesmo que esses termos sejam diferentes, eles representam a mesma doença, câncer de pele.

O Reconhecimento de Entidades Nomeadas (REN) se tornou um dos objetivos em PLN. Entidades Nomeadas (EN) são termos especializados, e identificar tais termos é uma tarefa difícil. Uma das alternativas para a identificação de ENs é a utilização do contexto (palavras adjacentes a uma palavra ou conjunto de palavras) para diferenciar termos que são EN de um determinado domínio de termos que não o são (GOULART; LIMA, 2009). Na literatura biomédica, exemplos de ENs são doenças e tratamentos.

2.2.1 Reconhecimento de Entidades Nomeadas e Mineração de Textos

Atualmente, grande parte das informações existentes em meios digitais estão contidas em documentos textuais. Dessa forma, a Mineração de Textos surge como área importante em Mineração de Dados para descobrir padrões em textos.

Mineração de Textos (MT) é um conjunto de métodos usados para navegar, organizar, achar e descobrir informações em bases de textos. Assim como Mineração de Dados e KDD, Descoberta de Conhecimento em Textos (de *Knowledge Discovery in Text*, ou KDT) é outro termo popular para a Mineração de Textos (BARION; LAGO, 2008). O KDT consiste em

realizar o tratamento do texto para convertê-lo para uma forma estruturada (preprocessamento) e em aplicar algoritmos para descobrir o conhecimento (mineração).

No preprocessamento do texto, é que entram as técnicas de Processamento de Linguagens Naturais. Algumas técnicas de PLN utilizadas em REN são a remoção de *stopwords*, o *stemming* e o cálculo do TF-IDF.

Stopwords são palavras que aparecem com muita frequência no texto e que não possuem importância significativa para o contexto que está sendo analisado. Em função disso, o processo de remoção dessas palavras torna-se necessária para diminuir o tamanho das estruturas de dados que serão mineradas posteriormente, facilitando o trabalho dos algoritmos de classificação. As *stopwords* são geralmente organizadas em um arquivo denominado *stoplist*.

Em linguagens naturais, as palavras são classificadas gramaticalmente de diversas maneiras. As classes gramaticais possuem variações como gênero, número, tempo, entre outras. Assim sendo, o processo de *stemming* remove essas variações para que seja mantida a raiz da palavra, que indica o conceito a que tal palavra se relaciona dentro do texto.

A última etapa do preprocessamento do texto corresponde ao cálculo do TF-IDF (de *Term Frequency - Inverse Document Frequency*), que é uma medida numérica que expressa a relevância de um termo para um documento textual dentro de uma coleção. A importância aumenta proporcionalmente ao número de vezes que um termo aparece dentro de um documento, sendo compensada pela frequência do termo dentro da coleção (HAN; KAMBER, 2005).

A expressão matemática que determina o TF-IDF é dada pela fórmula 2.6. O TF-IDF de um termo *t* em um documento *d* é calculado a partir da multiplicação da frequência de *t* em *d* (tf, fórmula 2.4) pelo resultado da medida do idf de *t* (fórmula 2.5).

$$TF(d,t) = \begin{cases} 0 & \textbf{se } freq(d,t) = 0 \\ 1 + log(1 + log(freq(d,t))) & \textbf{se } freq(d,t) \neq 0 \end{cases} \tag{2.4}$$

$$IDF(t) = log\frac{1 + |d|}{|d_t|} \tag{2.5}$$

$$TF\text{-}IDF(d,t) = TF(d,t) \times IDF(t) \tag{2.6}$$

2.2.2 Métricas de avaliação

Após a mineração do texto preprocessado, é preciso fazer uma análise dos resultados obtidos. No caso deste trabalho, por modelos de Árvore de Decisão. A análise dos resultados é

feita a partir de medidas de Recuperação de Texto (de *Text Retrieval*, ou TR). Recuperação de Texto é uma técnica relacionada à organização e recuperação de informação em textos. Algumas medidas para a realização de TR são precisão, *recall* e *F-score* (HAN; KAMBER, 2005).

Precisão é a percentagem de documentos recuperados que são, de fato, relevantes para a análise, ao passo que *recall* (ou revocação) é a percentagem de documentos que são relevantes para a análise e que são, de fato, recuperados. As definições formais da precisão e do *recall* são dadas pelas fórmulas 2.7 e 2.8, respectivamente.

$$Precisão = \frac{|Relevantes \cap Recuperadas|}{|Recuperadas|} \tag{2.7}$$

$$Recall = \frac{|Relevantes \cap Recuperadas|}{|Relevantes|} \tag{2.8}$$

Nas fórmulas acima, entende-se por:

- $|Relevantes \cap Recuperadas|$: as instâncias que foram recuperadas corretamente;
- $|Recuperadas|$: total de instâncias recuperadas, sejam elas corretas ou não;
- $|Relevantes|$: todas as instâncias que foram recuperadas corretamente, mais as instâncias que não foram recuperadas mas que deveriam ter sido.

O cálculo do *F-score* (ou medida-F) nada mais é que a relação entre a precisão e o *recall*, dada pela média harmônica entre essas duas medidas, conforme ilustra a fórmula 2.9.

$$F\text{-}score = \frac{Precisão \times Recall}{(Precisão + Recall)/2} \tag{2.9}$$

A metodologia proposta está dividida entre várias etapas que utilizam as técnicas de Mineração de Dados e Processamento de Linguagens Naturais apresentadas neste capítulo. O próximo capítulo apresentará cada uma das etapas da metodologia proposta.

2.3 Considerações do Capítulo

Neste capítulo, foi apresentado o referencial teórico a respeito de mineração de dados e processamento de linguagens naturais. Enfatizou-se a importância do preprocessamento para um bom resultado do processo de mineração de dados, em especial sobre a tarefa de classificação.

Em relação à tarefa de classificação, destacou-se os algoritmos de árvore de decisão, relatando o processo de indução dos modelos e suas principais medidas de avaliação de qualidade. Em relação ao processamento de linguagens naturais, foi dado destaque ao reconhecimento de entidades nomeadas e o processo de mineração de textos onde, da mesma forma, foram apresentadas as suas principais medidas de avaliação. Os conceitos apresentados nesse capítulo são importantes para a descrição da metodologia proposta neste trabalho.

3 DESENVOLVIMENTO

Este capítulo apresenta a metodologia proposta neste trabalho para o Reconhecimento de Entidades Nomeadas na literatura biomédica, com base em modelos induzidos de árvore de decisão. A Figura 3.1 representa a metodologia proposta, em uma sequência de quatro principais etapas e um total de 21 passos, que serão expandidos no decorrer deste capítulo.

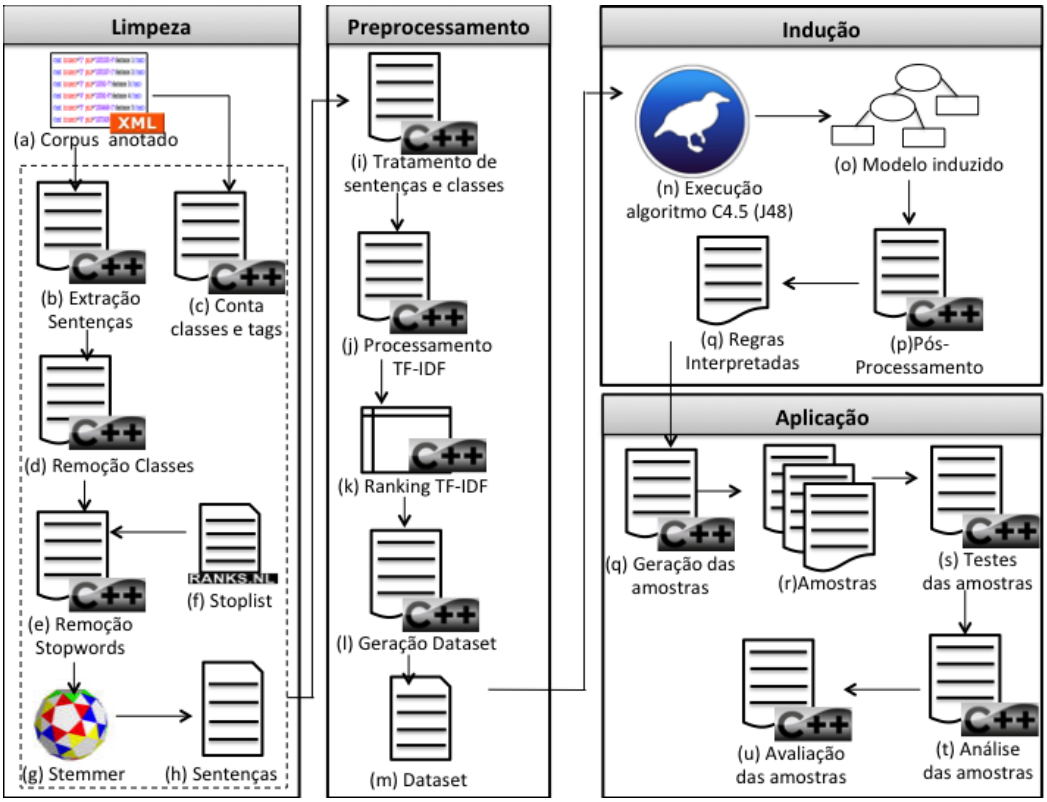

Figura 3.1 – Esquema da metodologia

O *corpus* utilizado é um documento textual anotado que contém sentenças de artigos científicos do domínio e que está organizado de maneira similar a um documento XML. As sentenças que compõem o *corpus* descrito em (BIOTEXT, 2013) foram retiradas do PubMed. As *tags* indicam os termos, ou conjuntos de termos, que são as ENs que devem ser identificadas e o último termo de cada linha indica o nome da classe correspondente à sentença, conforme ilustra a Figura 3.2.

Dessa forma, a metodologia proposta é dividida em quatro etapas: limpeza, preprocessamento, indução e aplicação. Na etapa de limpeza, são aplicadas técnicas de PLN ao conjunto de sentenças para que o *corpus* seja preprocessado. Os textos biomédicos, da mesma forma que textos em geral, possuem várias palavras em seu conteúdo que não são importantes para a aná-

```
Reduction of <DIS> vasoreactivity </DIS> and <DIS> thrombogenicity </DIS> with <TREAT> laserthermal angioplasty </TREAT> comparison wi
Effects of <TREAT_VAG> ultrasound energy </TREAT_VAG> on <DIS_VAG> total peripheral artery occlusions </DIS_VAG> initial angiographic
<TREAT> Highdose chemotherapy with autologous stemcell support </TREAT> for <DIS> epithelial ovarian cancer </DIS> ||TREAT_FOR_DIS
<TREAT> Pelvic floor stimulation </TREAT> in the treatment of <DIS> adult urinary incontinence </DIS> ||TREAT_FOR_DIS
<TREAT> Chronic vagus nerve stimulation </TREAT> for treatment of <DIS> seizures </DIS> ||TREAT_FOR_DIS
<TREAT> Tandem highdose chemoradiotherapy with autologous stemcell support </TREAT> in the treatment of newly diagnosed or responsive
Special report comparative efficacy of different types of <TREAT> pneumatic compression pumps </TREAT> for the treatment of <DIS> lymp
Special report <TREAT> pressurereducing support surfaces </TREAT> in the prevention and treatment of <DIS> pressure ulcers </DIS> grou
<TREAT> Intravenous immune globulin </TREAT> for <DIS> recurrent spontaneous abortion </DIS> ||TREAT_FOR_DIS
<TREAT> External counterpulsation </TREAT> for treatment of <DIS> chronic stable angina pectoris </DIS> ||TREAT_FOR_DIS
<TREAT> Intraarticular hyaluronan injections </TREAT> for treatment of <DIS> osteoarthritis </DIS> of the knee ||TREAT_FOR_DIS
<DIS_PREV> Pneumococcal </DIS_PREV> <TREAT_PREV> vaccine </TREAT_PREV> a second look ||PREVENT
```

Figura 3.2 – Estrutura do *corpus* utilizado para a indução dos modelos

lise, as *stopwords*. Além disso, diversas palavras estão relacionadas entre si porque se referem a um mesmo conceito. Assim, após a extração das sentenças do *corpus*, é realizada a remoção das *stopwords* e o *stemming*, para se obter um *corpus* com os radicais das palavras que possuem importância para o objetivo de análise.

Na etapa de preprocessamento, é feito o tratamento de sentenças e classes do *corpus*. Como o *corpus* possui diversas classes semelhantes entre si, é possível agrupá-las para a obtenção de modelos com maior acurácia. Em seguida, são escolhidos os termos com maior importância através de um *ranking* TF-IDF para a geração dos *datasets* que serão utilizados para a indução dos modelos de Árvore de Decisão.

Na etapa de indução, os *datasets* gerados são processados pelo *Weka* e os modelos de Árvore de Decisão são induzidos utilizando os algoritmos ID3 e C4.5, com várias combinações de parâmetros para que se escolham os modelos dos quais serão geradas as regras que realizam o REN.

Na etapa de aplicação, é gerado conjunto de amostras com sentenças escolhidas aleatoriamente a partir do *corpus* original para o teste das regras induzidas a partir dos modelos de Árvore de Decisão. Com base nos resultados desses testes é que são feitas as análises das medidas de precisão, *recall* e *F-score*.

3.1 Limpeza

Nesta etapa, são aplicadas as rotinas que realizam a extração de sentenças, a remoção de *stopwords* e o *stemming* do *corpus*.

Conforme ilustrado na Figura 3.2, o *corpus* (Figura 3.1-a) possui várias sentenças marcadas com *tags* que identificam as ENs presentes nelas. O primeiro passo na etapa de limpeza é a extração das sentenças que possuem importância para a geração dos modelos que realizarão o REN (Figura 3.1-b). Para tanto, foi desenvolvido um algoritmo que percorre todas as sentenças do *corpus* e alimenta termos e classes em uma matriz de sentenças. O pseudocódigo que realiza

essa rotina está descrito em A.1.

O *corpus* contém sentenças que tratam dos mais diversos assuntos da literatura biomédica. Utilizando a rotina descrita no pseudocódigo A.2, são identificadas as classes distintas e as *tags* distintas, e calculados os números de sentenças por classe e por *tag* (Figura 3.1-c). Essa rotina foi desenvolvida para que pudesse ser realizado um mapeamento das características do *corpus* sendo utilizado. A partir dessa rotina, obtêm-se os valores demonstrados nas Tabelas 3.1 e 3.2. A Tabela 3.1 apresenta todas as classes presentes no *corpus* original, e a frequência com que essas classes ocorrem. A Tabela 3.2 faz o mesmo mapeamento para as *tags*, onde cada *tag* representa uma EN no *corpus*.

Tabela 3.1 – Classes originais

Classe	#Sentenças
NONE	1690
TO_SEE	72
DISONLY	554
TREATONLY	157
PREVENT	60
VAGUE	37
TREAT_FOR_DIS	778
SIDE_EFF	28
TREAT_NO_FOR_DIS	4

Tabela 3.2 – Tags originais

Tag	#Sentenças
DISONLY	630
TREATONLY	169
DIS_PREV	63
TREAT_PREV	63
TREAT_VAG	37
DIS_VAG	37
TREAT	830
DIS	830
TREAT_SIDE_EFF	30
DIS_SIDE_EFF	30
TREAT_NO	4
DIS_NO	4

A partir da Tabela 3.1, pode-se verificar que o *corpus* contém 9 classes distintas e encontra-se muito desbalanceado. Isso é, existem 1690 sentenças que não possuem nenhuma classe (definidas como "NONE"). Como essas sentenças podem implicar em um viés na indução do modelo (próximos passos), optou-se por removê-las. Também optou-se por remover as

sentenças definidas como "TO_SEE". Assim, aplicando-se a rotina descrita pelo pseudocódigo A.3, sentenças que possuem as classes "NONE"ou "TO_SEE"são removidas do *corpus* porque não possuem ENs em seu conteúdo e, portanto, não influenciam no REN (Figura 3.1-d). Depois da remoção dessas sentenças, o *corpus* passa a ter em seu conteúdo um total de 1762 sentenças.

Logo após a extração de sentenças e o processamento das classes, é feita a remoção das *stopwords* (Figura 3.1-e). A lista de *stopwords* (*stoplist*) utilizada nessa etapa da limpeza do *corpus* foi obtida em (STOPWORDS, 2013) (Figura 3.1-f). A rotina que remove as *stopwords* faz uso dessa *stoplist* e está descrita pelo pseudocódigo A.4. Além das *stopwords* contidas na *stoplist*, também são considerados como *stopwords* os sinais de pontuação e os números que estão contidos nas sentenças do *corpus*. A cada iteração da rotina, são removidos os sinais de pontuação e os números que compõem o termo e, se após isso, o termo não for uma *string* vazia, é verificado se o termo está contido na lista de *stopwords*. Caso positivo, remove-se o termo da sentença.

O último passo da etapa de limpeza do *corpus* é o *stemming* (Figura 3.1-g). Este passo é realizado com o auxílio do *software Snowball* (SNOWBALL, 2013). Para cada sentença do *corpus*, todos os termos que não são *tags* nem classes são analisados pelo *stemmer* (os afixos são removidos dos termos, restando apenas seus radicais) e substituídos pelo respectivo *stem*, atualizando, assim, o *corpus* (Figura 3.1-h). A rotina que realiza este passo está descrita no pseudocódigo A.5.

3.2 Preprocessamento

Preprocessamento é a etapa em que o *corpus* é analisado com o intuito de encontrar termos relevantes para o REN. Isso é feito a partir do cálculo do TF-IDF para cada um dos termos existentes no *corpus*. Com os termos considerados relevantes a partir do cálculo do TF-IDF, os *datasets* são gerados para a indução dos modelos.

3.2.1 Tratamento de sentenças e classes

O *corpus* possui sentenças cujas classes são muito semelhantes e que podem ser agrupadas como TREATONLY, TREAT_FOR_DIS e TREAT_NO_FOR_DIS. Assim, é aplicada uma rotina, descrita pelo pseudocódigo A.6, que transforma o *corpus* atual em um novo *corpus* com apenas 3 classes (Figura 3.1-i). Optou-se pelo agrupamento das classes porque classificação

com muitas classes tende a um erro maior de classificação. Dessa forma, as classes são agrupadas de maneira empírica, considerando a semântica de cada uma delas. A Tabela 3.3 mostra a relação entre as classes originais e as novas classes assinaladas após a modificação.

Tabela 3.3 – Classes transformadas

Original	Transformada
DISONLY	DISEASE
TREATONLY	TREATMENT
PREVENT	OTHER
VAGUE	OTHER
TREAT_FOR_DIS	TREATMENT
SIDE_EFF	OTHER
TREAT_NO_FOR_DIS	TREATMENT

3.2.2 Processamento do TF-IDF

Neste passo da etapa de preprocessamento do texto, são executadas as rotinas que identificam os termos distintos do *corpus* e que calculam seus respectivos valores de TF-IDF (Figura 3.1-j). O pseudocódigo A.7 descreve a rotina que identifica os termos distintos e, a partir do resultado obtido são calculados os valores de TF-IDF (pseudocódigo A.8). Como o *corpus* é formado por um único documento, o valor de d na equação 2.6 é igual a 1.

Após o processamento do TF-IDF, foram encontrados 4364 termos distintos. A Tabela 3.4 expõe os maiores valores calculados e as medidas estatísticas para os 4364 termos processados.

Tabela 3.4 – Maiores valores e medidas estatísticas do TF-IDF

Maiores valores		Medidas estatísticas	
patient	0,470447	Menor valor	0,30103
treatment	0,462098	Maior valor	0,470447
cancer	0,45733	Média	0,3311089
therapi	0,453152	Mediana	0,30103
studi	0,4527	Desvio Padrão	0,0361191
diseas	0,450083		
effect	0,449531		

O cálculo do TF-IDF nessa etapa da metodologia se fez necessário para identificar aqueles termos presentes no *corpus* que tem mais relevância. Assim, uma seleção adequada de termos pode ser útil para a construção do *dataset* e, assim, produzir modelos mais interessantes.

3.2.3 *Ranking* TF-IDF

Como o número de termos distintos presentes no *corpus* é muito alto, faz-se necessário a redução do conjunto de termos que serão considerados para a geração dos *datasets* que serão minerados. Este é o passo da etapa de preprocessamento em que é definido o ponto de corte para reduzir o tamanho dos *datasets* a serem minerados.

São propostas três maneiras de calcular o ponto de corte. As fórmulas 3.1, 3.2 e 3.3 representam as expressões que calculam os diferentes pontos de corte propostos.

$$PontoCorte = Média + DesvioPadrão \tag{3.1}$$

$$PontoCorte = Média + 2 \times DesvioPadrão \tag{3.2}$$

$$PontoCorte = Média + 3 \times DesvioPadrão \tag{3.3}$$

Aplicando-se a rotina descrita pelo pseudocódigo A.9 (Figura 3.1-k) para cada uma das fórmulas de cálculo de ponto de corte descritas, obtém-se as quantidades de termos descritas pela Tabela 3.5.

Tabela 3.5 – *Ranking* TF-IDF

Peso da média	Peso do desvio padrão	# Termos
1	1	819
1	2	188
1	3	14

3.2.4 Geração de *datasets*

Com os termos obtidos no ranking de TF-IDF, são gerados os *datasets* que serão minerados (Figura 3.1-l). Os *datasets* são populados da seguinte maneira:

- Cada atributo preditivo é um termo relevante;
- O atributo-classe é um tipo de EN (DISEASE, TREATMENT ou OTHER);
- Cada registro é uma sentença do *corpus*;
- A relação atributo-valor corresponde à presença ou ausência de cada termo na sentença.

A rotina que gera a matriz de valores que será convertida em *dataset* é descrita pelo pseudocódigo A.10. Esse algoritmo faz uso de cada termo selecionado no A.9, onde cada um desses termos passa a ser um atributo do *dataset*. A rotina verifica, para cada sentença do *corpus*, se existem os termos (ou atributos) selecionados. Quando existem, a relação atributo-valor passa a ser 1, caso contrário é 0. Ao final de cada sentença é capturada a sua classe e inserida no *dataset* (Figura 3.1-m).

O *software* utilizado para a indução dos modelos de Árvore de Decisão é o *Weka* (WEKA, 2013) (WITTEN; FRANK; HALL, 2011). Os tipos de arquivos reconhecidos pelo *Weka* são o ARFF e o CSV, entre outros. Um arquivo do tipo ARFF (*Attribute-Relation File Format*) é o formato padrão do *Weka*. Esse arquivo, conforme a Figura 3.3, é composto por três principais seções:

- *Relation*, contendo um nome para o arquivo;
- *Attribute*, onde todos os atributos são sequencialmente declarados, incluindo seu tipo;
- *Data*, contendo os registros do *dataset* separados por vírgula, obedecendo a ordem dos atributos declarados.

Como arquivos do tipo ARFF são compostos por uma estrutura mais complexa que as do tipo CSV, as matrizes de valores geradas são gravadas em arquivos do tipo CSV. Porém, os arquivos do tipo ARFF é que são processados pelo *Weka*. Assim, os arquivos CSV gerados são convertidos para arquivos ARFF utilizando o conversor de CSV para ARFF existente no próprio *Weka* para a geração dos *datasets*. A Figura 3.4 ilustra a estrutura de um dos *datasets* gerados.

```
@relation dataset_3_classes_avg_plus_3_st_dev-weka.filters.unsupervised.attribute.NumericToNominal-Rfirst-last

@attribute patient {0,1}
@attribute treatment {0,1}
@attribute cancer {0,1}
@attribute therapi {0,1}
@attribute studi {0,1}
@attribute diseas {0,1}
@attribute effect {0,1}
@attribute lung {0,1}
@attribute clinic {0,1}
@attribute cell {0,1}
@attribute treat {0,1}
@attribute chemotherapi {0,1}
@attribute case {0,1}
@attribute CLASS {DISEASE,TREATMENT,OTHER}

@data
0,0,0,0,1,0,0,0,1,0,0,0,0,DISEASE
1,0,0,0,0,0,0,0,0,0,0,0,0,TREATMENT
0,0,0,0,0,0,0,0,0,0,0,0,1,OTHER
0,0,0,0,1,0,0,0,0,0,0,0,0,DISEASE
0,0,0,0,0,0,0,0,0,0,0,0,0,DISEASE
```

Figura 3.3 – Estrutura de um arquivo do tipo ARFF

patient Nominal	treatment Nominal	cancer Nominal	therapi Nominal	studi Nominal	diseas Nominal	effect Nominal	lung Nominal	clinic Nominal	cell Nominal	treat Nominal	chemotherapi Nominal	case Nominal	**CLASS** Nominal
0	0	0	0	0	0	0	0	0	0	0	0	0	DISEASE
0	0	0	0	0	0	0	0	0	0	0	0	0	DISEASE
1	0	0	0	0	0	0	0	0	0	0	0	1	TREATMENT
0	0	0	0	0	0	0	0	0	0	0	0	0	DISEASE
0	0	0	0	0	0	0	0	0	0	0	0	0	TREATMENT
0	0	0	0	0	0	0	0	0	0	0	0	1	DISEASE
0	0	0	0	0	0	0	0	0	0	0	0	0	DISEASE
1	0	0	0	1	0	0	0	0	0	0	0	0	TREATMENT
0	0	0	1	0	0	0	0	0	0	0	0	0	TREATMENT
0	0	0	0	0	1	0	0	0	0	0	0	0	DISEASE
1	0	0	0	0	0	0	0	0	0	0	0	0	TREATMENT
0	0	1	0	0	0	0	0	0	0	0	0	0	OTHER
0	0	0	0	0	0	0	0	0	0	0	0	0	OTHER
0	1	0	0	0	0	0	0	0	0	0	0	0	TREATMENT
0	0	0	0	0	0	0	0	0	0	0	0	0	OTHER

Figura 3.4 – Estrutura de um *dataset* gerado

3.3 Indução

Nesta etapa, é realizada a indução dos modelos de Árvore de Decisão para cada um dos *datasets* gerados, bem como o posprocessamento dos modelos induzidos e a geração das regras que possibilitam o REN.

3.3.1 Modelos

A indução dos modelos de Árvore de Decisão (Figura 3.1-n) é feita a partir da mineração dos *datasets* gerados utilizando os algoritmos ID3 e J48 do *Weka*. O algoritmo J48 é baseado no algoritmo C4.5 desenvolvido por Quinlan (QUINLAN, 1993).

Além dos diferentes algoritmos, também foram configurados diferentes parâmetros de execução em relação à confiança das regras, ao número mínimo de objetos presentes nas folhas e ao número de *folds*, para a indução dos modelos (Figura 3.1-o). A Tabela 3.6 mostra os resultados obtidos após a indução dos modelos. A primeira coluna indica um identificador do número da execução. A segunda coluna mostra o peso do desvio padrão utilizado para o cálculo do ponto de corte para a seleção dos termos que fazem parte do *dataset* minerado. As quatro colunas seguintes indicam o algoritmo e os parâmetros utilizados. É feita uma comparação entre o algoritmo ID3 e o algoritmo J48 (os parâmetros utilizados para o algoritmo ID3 são os parâmetros-padrão do *Weka*). As colunas restantes dizem respeito às métricas da árvore gerada. São expostos o número de folhas, o tamanho, a acurácia e os valores da área abaixo curva ROC (AUC) para a classe DISEASE, para a classe TREATMENT e para a árvore como um todo.

A partir dos dados da Tabela 3.6, foram escolhidos os modelos #7 e #11, que apresentaram as melhores combinações de tamanho, de acurácia e de área abaixo da curva ROC.

Tabela 3.6 – Resultados obtidos com o *Weka* para os *datasets* gerados

#	△DP	Alg	Conf	Obj	Folds	Folhas	Tam	Acc	Auc-D	Auc-T	Auc-G
1	1	ID3	d	d	d	548	1095	73.66%	0.786	0.769	0.767
2	2	ID3	d	d	d	563	1125	69.64%	0.773	0.758	0.757
3	3	ID3	d	d	d	120	239	65.61%	0.76	0.749	0.742
4	1	J48	0.25	2	3	85	169	76.45%	0.821	0.826	0.819
5	2	J48	0.25	2	3	62	123	76.56%	0.84	0.827	0.825
6	3	J48	0.25	2	3	10	19	67.03%	0.747	0.745	0.735
7	1	J48	0.25	3	4	73	145	77.35%	0.838	0.831	0.826
8	2	J48	0.25	3	4	57	113	76.62%	0.844	0.833	0.829
9	3	J48	0.25	3	4	9	17	67.08%	0.747	0.745	0.735
10	1	J48	0.25	5	5	55	109	76.90%	0.834	0.823	0.822
11	2	J48	0.25	5	5	46	91	77.01%	0.843	0.829	0.824
12	3	J48	0.25	5	5	9	17	67.08%	0.747	0.745	0.735
13	1	J48	0.5	2	3	189	377	73.38%	0.796	0.808	0.8
14	2	J48	0.5	2	3	116	231	76.39%	0.85	0.836	0.834
15	3	J48	0.5	2	3	16	31	65.83%	0.759	0.754	0.744
16	1	J48	0.5	3	4	146	291	75.54%	0.825	0.825	0.818
17	2	J48	0.5	3	4	91	181	76.45%	0.861	0.844	0.841
18	3	J48	0.5	3	4	15	29	66.12%	0.763	0.757	0.747
19	1	J48	0.5	5	5	86	171	75.14%	0.828	0.82	0.763
10	2	J48	0.5	5	5	82	163	76.62%	0.859	0.842	0.839
21	3	J48	0.5	5	5	15	29	66.63%	0.75	0.751	0.739
22	1	J48	0.75	2	3	199	397	72.64%	0.797	0.803	0.796
23	2	J48	0.75	2	3	125	249	75.71%	0.845	0.832	0.83
24	3	J48	0.75	2	3	21	41	66.00%	0.757	0.752	0.742
25	1	J48	0.75	3	4	146	291	74.80%	0.82	0.818	0.812
26	2	J48	0.75	3	4	97	193	76.39%	0.861	0.845	0.842
27	3	J48	0.75	3	4	17	33	66.23%	0.761	0.755	0.744
28	1	J48	0.75	5	5	86	171	74.97%	0.829	0.821	0.82
29	2	J48	0.75	5	5	82	163	76.27%	0.861	0.844	0.84
30	3	J48	0.75	5	5	15	29	66.69%	0.75	0.751	0.739

3.3.2 Posprocessamento

Logo após a indução dos modelos, é realizado o posprocessamento das árvores geradas para a extração das regras que realizarão o REN (Figura 3.1-p). As regras são constituídas por uma lista de predicados e uma classe. Cada predicado é formado por um termo que faz parte do *dataset* e um teste que indica a presença ou ausência desse termo em uma sentença do *corpus*.

O pseudocódigo A.11 descreve a rotina recursiva que gera o conjunto de regras para os modelos escolhidos. Algumas das regras geradas (Figura 3.1-q) por essa rotina estão ilustradas na Figura 3.5.

```
chemotherapi:0, therapi:0, treatment:0, effect:0, treat:0, lung:0, diseas:0, patient:1 -> TREATMENT(183.0/77.0)
chemotherapi:0, therapi:0, treatment:0, effect:0, treat:0, lung:0, diseas:1, patient:0 -> DISEASE(64.0/13.0)
chemotherapi:0, therapi:0, treatment:0, effect:0, treat:0, lung:0, diseas:1, patient:1 -> TREATMENT(10.0/4.0)
chemotherapi:0, therapi:0, treatment:0, effect:0, treat:0, lung:1, patient:0, cancer:0 -> TREATMENT(10.0/2.0)
chemotherapi:0, therapi:0, treatment:0, effect:0, treat:0, lung:1, patient:0, cancer:1 -> DISEASE(11.0/5.0)
chemotherapi:0, therapi:0, treatment:0, effect:0, treat:0, lung:1, patient:1 -> TREATMENT(32.0/1.0)
chemotherapi:0, therapi:0, treatment:0, effect:0, treat:1 -> TREATMENT(58.0/9.0)
chemotherapi:0, therapi:0, treatment:0, effect:1, diseas:0 -> TREATMENT(74.0/19.0)
chemotherapi:0, therapi:0, treatment:0, effect:1, diseas:1 -> OTHER(7.0/1.0)
chemotherapi:0, therapi:0, treatment:1 -> TREATMENT(220.0/27.0)
chemotherapi:0, therapi:1 -> TREATMENT(147.0/10.0)
chemotherapi:1 -> TREATMENT(79.0/3.0)
```

Figura 3.5 – Exemplo de regras geradas

3.4 Aplicação

A etapa de aplicação é a última etapa da metodologia proposta. Nela são gerados *corpora*[1] de testes que são processados utilizando as regras geradas no posprocessamento dos modelos de Árvore de Decisão induzidos pelo algoritmo J48 do *Weka* para reconhecer as ENs que possam existir em suas sentenças. Após o processamento dos *corpora* de testes são calculadas as medidas de precisão, *recall* e *F-score* para cada *corpus* de teste.

3.4.1 Geração das amostras

Neste passo da etapa de aplicação, são gerados os *corpora* que serão utilizados para o teste das regras geradas a partir dos modelos de árvore gerados (Figura 3.1-q). Cada *corpus* de teste possui algumas sentenças do *corpus* gerado pela rotina descrita em A.6, escolhidas aleatoriamente.

O pseudocódigo A.12 descreve a rotina que escolhe as sentenças que irão compor os *corpora* de testes. São gerados dois *corpora* para cada conjunto de sentenças, sendo um marcado e o outro sem marcação alguma, para que se possa calcular as medidas de precisão, *recall* e *F-score* para cada amostra. O número de sentenças por *corpus* é dado por uma porcentagem predefinida (5% ou 10%) do total de sentenças do *corpus* inicial (Figura 3.1-r).

3.4.2 Teste das amostras

Os testes das amostras (Figura 3.1-s) são realizados através das regras geradas no posprocessamento dos modelos. Cada sentença de cada amostra não-marcada é equiparada com cada uma das regras até que se estabeleça correspondência com uma delas. A rotina que realiza os testes para uma determinada amostra é descrita pelo pseudocódigo A.13.

[1] Enquanto o termo *corpus* significa o plural de textos, ou uma coleção de textos, o termo *corpora* significa o plural de *corpus*

3.4.3 Avaliação das amostras

Este é o passo da etapa de aplicação em que são calculadas as medidas de precisão, *recall* e *F-score* para cada uma das amostras geradas. As amostras não-marcadas são analisadas com base nas regras geradas na etapa de indução (Figura 3.1-t). Cada amostra é processada pela rotina descrita pelo pseudocódigo A.14, que retorna uma matriz de três colunas similar às matrizes de confusão geradas pelo *Weka* na etapa de indução.

Em seguida, são calculadas as medidas de avaliação (Figura 3.1-u) através da rotina descrita pelo pseudocódigo A.15, gerando matrizes cujas linhas representam os valores de precisão, *recall* e *F-score* para as classes DISEASE, TREATMENT e OTHER (primeira, segunda e terceira linhas, respectivamente). A Figura 3.6 ilustra a matriz com os valores das medidas de avaliação.

```
   p       r       f
0,684   0,929   0,788 | DISEASE
0,872   0,774   0,820 | TREATMENT
1,000   0,429   0,600 | OTHER
```

Figura 3.6 – Exemplo de matriz de valores das medidas de avaliação

3.5 Considerações do Capítulo

Neste capítulo, foi apresentada a metodologia desenvolvida para reconhecimento de entidades nomeadas a partir de modelos de árvore de decisão, bem como a sua aplicação no corpus utilizado. Para a completa aplicação, foram desenvolvidos 15 códigos, descritos como pseudocódigos no Apêndice A.

Para buscar o melhor modelo para os dados utilizados, foram gerados três diferentes datasets, submetidos a diferentes configurações dos algoritmos de indução de árvore, totalizando 30 modelos induzidos. Após a avaliação das métricas dos modelos, dois deles foram selecionados para serem testados na fase de aplicação e, assim, avaliar o efetivo reconhecimento de sentenças com entidades nomeadas.

4 RESULTADOS

Neste capítulo, são apresentados os resultados obtidos aplicando-se a metodologia proposta. A Tabela 4.1 expõe os valores das medidas de avaliação calculadas para as amostras geradas na etapa de aplicação da metodologia.

A primeira coluna indica o percentual utilizado para calcular o número de sentenças contidas em cada amostra. A segunda coluna identifica a amostra que está sendo avaliada. A terceira coluna indica o modelo que foi utilizado para a geração das regras que analisam a amostra. As colunas seguintes indicam os valores de precisão, *recall* e *F-score* para cada uma das classes DISEASE, TREATMENT e OTHER.

Pela Tabela 4.1 observa-se que, para a classe DISEASE, os valores de precisão estão entre 0.608 e 0.800, de *recall* entre 0.815 e 0.960 e *F-score* entre 0.719 e 0.863. Para a classe TREATMENT, os valores estão entre 0.841 e 0.942 para precisão, entre 0.737 e 0.891 para *recall* e entre 0.787 e 0.916 para *F-score*. Essas medidas indicam que o processo mostrou-se bastante satisfatório para o reconhecimento de sentenças com entidades nomeadas em relação a essas duas classes.

A Figura 4.1 apresenta um gráfico com o comportamento das medidas de avaliação apresentadas na Tabela 4.1, para cada uma das configurações: amostra de teste e modelo sendo utilizado. As medidas de avaliação estão separadas por classe (D, T e O para DISEASE, TREATMENT e OTHER, respectivamente). Esses gráficos foram gerados a partir dos dados da Tabela 4.1, conforme segue:

- As linhas 1 a 10 da tabela estão representadas no gráfico 5% 1DP;
- As linhas 11 a 20 da tabela estão representadas no gráfico 10% 1DP;
- As linhas 21 a 30 da tabela estão representadas no gráfico 5% 2DP;
- As linhas 31 a 40 da tabela estão representadas no gráfico 10% 2DP.

Por esses gráficos, é possível observar que o comportamento para as classes DISEASE e TREATMENT é semelhante, ratificando os bons valores de precisão, *recall* e *F-score* obtidos para cada classe.

A Tabela 4.2 mostra os valores médios de precisão, *recall* e *F-score* para cada configuração de teste. Dessa tabela, foi gerado um gráfico, apresentado na Figura 4.2. Neste gráfico,

Tabela 4.1 – Testes

			Disease			Treat			Other		
%	#	△DP	Prec	Recall	FS	Prec	Recall	FS	Prec	Recall	FS
5	1	1	0.684	0.929	0.788	0.872	0.774	0.820	1.000	0.429	0.600
5	2	1	0.706	0.923	0.800	0.918	0.818	0.865	0.600	0.429	0.500
5	3	1	0.690	0.906	0.784	0.929	0.780	0.848	0.500	0.333	0.400
5	4	1	0.674	0.829	0.744	0.900	0.766	0.828	0.400	0.333	0.364
5	5	1	0.743	0.929	0.825	0.922	0.855	0.887	1.000	0.400	0.571
5	6	1	0.718	0.875	0.789	0.867	0.813	0.839	0.500	0.250	0.333
5	7	1	0.763	0.906	0.829	0.867	0.848	0.857	0.800	0.400	0.533
5	8	1	0.756	0.969	0.849	0.932	0.788	0.854	0.667	0.500	0.571
5	9	1	0.688	0.815	0.746	0.882	0.789	0.833	0.600	0.750	0.667
5	10	1	0.765	0.963	0.852	0.942	0.891	0.916	1.000	0.333	0.500
10	1	1	0.736	0.869	0.797	0.910	0.850	0.879	0.500	0.250	0.333
10	2	1	0.800	0.938	0.863	0.935	0.861	0.897	0.875	0.636	0.737
10	3	1	0.671	0.859	0.753	0.851	0.740	0.791	0.429	0.250	0.316
10	4	1	0.736	0.941	0.826	0.915	0.765	0.833	0.857	0.600	0.706
10	5	1	0.714	0.902	0.797	0.912	0.790	0.847	0.875	0.700	0.778
10	6	1	0.737	0.889	0.806	0.883	0.822	0.851	0.833	0.417	0.556
10	7	1	0.622	0.902	0.736	0.896	0.789	0.839	0.667	0.250	0.364
10	8	1	0.787	0.922	0.849	0.924	0.850	0.885	0.556	0.417	0.476
10	9	1	0.711	0.894	0.792	0.871	0.796	0.831	0.750	0.353	0.480
10	10	1	0.643	0.865	0.738	0.889	0.800	0.842	1.000	0.500	0.667
5	1	2	0.684	0.929	0.788	0.872	0.774	0.820	1.000	0.429	0.600
5	2	2	0.639	0.885	0.742	0.894	0.764	0.824	0.600	0.429	0.500
5	3	2	0.698	0.938	0.800	0.929	0.780	0.848	0.667	0.333	0.444
5	4	2	0.689	0.886	0.775	0.897	0.745	0.814	0.500	0.333	0.400
5	5	2	0.676	0.821	0.742	0.865	0.818	0.841	1.000	0.400	0.571
5	6	2	0.700	0.875	0.778	0.886	0.813	0.848	0.500	0.250	0.333
5	7	2	0.744	0.906	0.817	0.844	0.826	0.835	1.000	0.400	0.571
5	8	2	0.756	0.969	0.849	0.932	0.788	0.854	0.667	0.500	0.571
5	9	2	0.622	0.852	0.719	0.894	0.737	0.808	0.750	0.750	0.750
5	10	2	0.758	0.926	0.833	0.906	0.873	0.889	1.000	0.333	0.500
10	1	2	0.679	0.869	0.763	0.906	0.813	0.857	0.500	0.125	0.200
10	2	2	0.763	0.906	0.829	0.896	0.851	0.873	1.000	0.364	0.533
10	3	2	0.679	0.891	0.770	0.841	0.740	0.787	0.250	0.083	0.125
10	4	2	0.729	0.912	0.810	0.882	0.765	0.820	0.833	0.500	0.625
10	5	2	0.705	0.902	0.791	0.911	0.781	0.841	0.875	0.700	0.778
10	6	2	0.728	0.937	0.819	0.910	0.802	0.853	0.833	0.417	0.556
10	7	2	0.608	0.882	0.720	0.876	0.780	0.825	0.800	0.250	0.381
10	8	2	0.776	0.922	0.843	0.913	0.840	0.875	0.625	0.417	0.500
10	9	2	0.690	0.879	0.773	0.849	0.785	0.816	1.000	0.353	0.522
10	10	2	0.610	0.904	0.729	0.903	0.764	0.828	1.000	0.429	0.600

o eixo x mostra os valores de precisão, *recall* e *F-score* para cada uma das classes avaliadas, separadas por D, T e O (DISEASE, TREATMENT e OTHER, respectivamente). O eixo y re-

presenta as configurações de amostra (5% ou 10%) e peso do desvio padrão dos testes (1 ou 2).

Tabela 4.2 – Valores médios de precisão, *recall* e *F-score* para as configurações de teste

%	DP	Prec-D	Rec-D	FSc-D	Prec-T	Rec-T	FSc-T	Prec-O	Rec-O	FSc-O
5	1	0,719	0,904	0,801	0,903	0,812	0,855	0,707	0,416	0,504
10	2	0,716	0,898	0,796	0,898	0,806	0,850	0,734	0,437	0,541
5	1	0,697	0,899	0,784	0,892	0,792	0,838	0,768	0,416	0,524
10	2	0,697	0,900	0,785	0,889	0,792	0,837	0,772	0,364	0,482

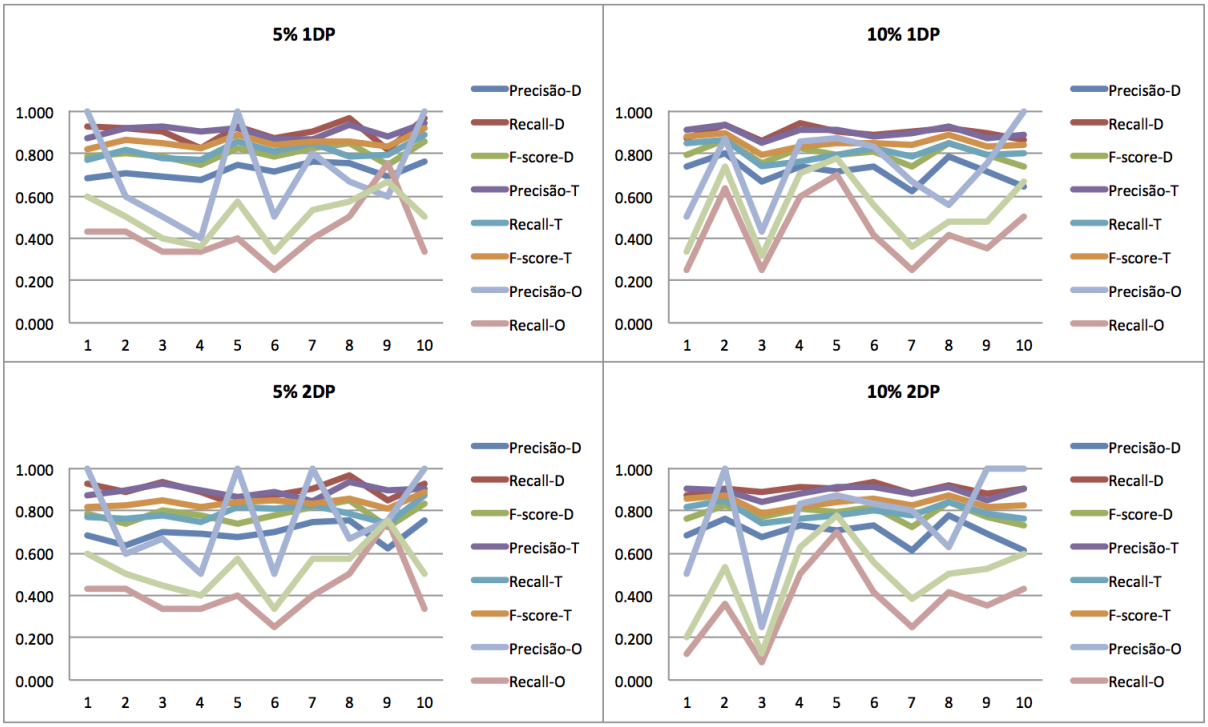

Figura 4.1 – Comportamento das medidas de avaliação

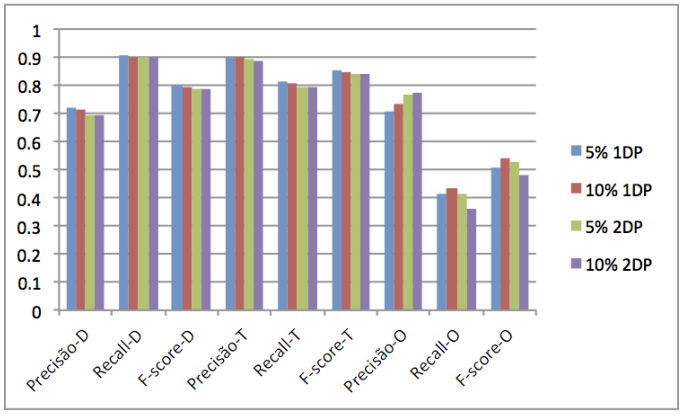

Figura 4.2 – Média das medidas de avaliação

A partir dos gráficos das Figuras 4.1 e 4.2 é possível notar que as medidas apresentam um comportamento semelhante para as classes DISEASE e TREATMENT, com valores satisfatórios para a análise. As medidas para a classe OTHER apresentam um comportamento semelhante entre as amostras, mas apresenta uma variação entre os modelos. Apesar dessa disparidade, esse comportamento não parece influenciar na aplicação, pois o objetivo concentra-se na identificação de EN dos tipos DISEASE e TREATMENT.

5 TRABALHOS RELACIONADOS

O trabalho realizado por (WINCK et al., 2010) processa documentos em nível semântico, tendo a bioinformática como área de interesse. Nesse trabalho, estuda-se sobre o planejamento racional de fármacos (RDD), auxiliando na identificação de proteínas (receptores) e compostos candidatos a fármacos (ligantes) em textos científicos. A abordagem proposta é a de tratar tais estruturas como entidades nomeadas (EN). O reconhecimento dessas EN é proposto através do contexto das mesmas. Para tanto, apresenta-se uma metodologia que, a partir de um *corpus* anotado no domínio de RDD, faz uso de regras de associação para produzir modelos que sejam capazes de indicar quais termos relevantes ao contexto indicam a presença de uma EN do tipo receptor ou ligante, nas sentenças dos documentos analisados.

No trabalho de (JU; WANG; ZHU, 2011) é apresentada uma abordagem para identificação de ENs no domínio biomédico, utilizando SVM (*Support Vector Machines*). Nesse trabalho, os autores fazem uso do *corpus* GENIA, uma coleção de *abstracts* extraídos do MedLine. Sua proposta é classificar uma EN como sendo biológica ou não-biológica. Para ENs biológicas, os resultados de precisão e *recall* foram de 94,33% e 71.67%, respectivamente.

O trabalho aqui proposto se difere dos trabalhos relacionados, por propor uma metodologia que faz uso de modelos de árvore de decisão para auxiliar no reconhecimento de entidades nomeadas. Em relação ao trabalho proposto por (WINCK et al., 2010), além da metodologia deste ser mais completa, os resultados obtidos em termos de precisão, *recall* e *F-score* também foram mais altos. Em relação ao trabalho de (JU; WANG; ZHU, 2011), pode-se dizer que não só a metodologia é diferente, como este trabalho propõe identificar tipos precisos de EN da literatura biomédica (DISEASE e TREATMENT), e não apenas classificá-las como sendo da área biomédica ou não.

6 CONCLUSÃO

O Reconhecimento de Entidades Nomeadas tem sido um desafio em pesquisas de Processamento de Linguagens Naturais e Mineração de Textos. O foco deste trabalho diz respeito ao Reconhecimento de Entidades Nomeadas na literatura biomédica. A abordagem é baseada em Árvores de Decisão, onde o objetivo é induzir modelos que indicam quais termos são relevantes em uma sentença para sugerir se uma dada Entidade Nomeada pode ser encontrada. Desse modo, foi planejada uma metodologia processada em quatro etapas: limpeza, preprocessamento, indução e aplicação sobre um *corpus* público sobre doenças e tratamentos, já marcado.

Essa metodologia contribui para acelerar o processo de identificação desses termos em sentenças na literatura biomédica e, assim, para auxiliar especialistas da área. Analisando os resultados da mineração dos *datasets* no *Weka*, foram escolhidos dois modelos de Árvore de Decisão. Estes modelos foram testados com um conjunto de amostras de tamanhos variáveis e sentenças escolhidas aleatoriamente no *corpus*. Os resultados obtidos foram satisfatórios e isso pode acarretar no desenvolvimento de trabalhos futuros utilizando essa metodologia.

Além do presente trabalho, obteve-se a publicação de um resumo que explica a metodologia proposta no *X-Meeting 2012* (*8th International Conference of the Brazilian Association for Bioinformatics and Computational Biology*), intitulado *A Decision-Tree Model-based Approach for Named Entities Recognition on Biomedical Literature* (VIANA; WINCK, 2012).

Algumas sugestões de trabalhos futuros são:

- Testar em uma base diferente da anotada, ou seja, em sentenças de artigos extraídos diretamente do PubMed. Para este trabalho, será necessário o auxílio de um especialista da área biomédica para se chegar a uma avaliação correta dos resultados.

- Mudar o *dataset* para, em vez de indicar apenas a presença ou ausência de um termo em uma sentença, indicar também a distância de um termo em relação a uma Entidade Nomeada da sentença. Dessa forma, será possível testar se os modelos conseguem identificar a Entidade Nomeada propriamente dita, não apenas se ela pertence a alguma sentença.

- Aplicar a metodologia em outros tipos de *corpora*, com sentenças que possuam outras Entidades Nomeadas além de doenças e tratamentos.

REFERÊNCIAS

BARION, E. C. N.; LAGO, D. Mineração de Textos. **Revista de Ciências Exatas e Tecnologia**, [S.l.], v.3, 2008.

BIOTEXT. **Sentences with roles and relations**. Acessado em 08 de Janeiro de 2013, http://biotext.berkeley.edu/data/dis_treat_data/sentences_with_roles_and_relations.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. The KDD process for extracting useful knowledge from volumes of data. **Communications of the ACM**, [S.l.], v.39, 1996.

FREITAS, A. A.; WIESER, D. C.; APWEILER, R. On the importance of comprehensible classification models for protein function prediction. **IEEE/ACM Transactions on Computational Biology and Bioinformatics**, [S.l.], v.99, 2010.

GOULART, R. R. V.; LIMA, V. L. S. O Contexto no Reconhecimento de Entidades Nomeadas em Textos de Biomedicina. **Simpósio de Tecnologias da Informação e da Língua (STIL)**, [S.l.], 2009.

HAN, J.; KAMBER, M. **Data Mining**: concepts and techniques. 2nd.ed. [S.l.]: Elsevier, 2005.

JIMENO, A. et al. Assessment of disease named entity recognition on a corpus of annotated sentences. **BMC Bioinformatics**, [S.l.], v.9, 2008.

JU, Z.; WANG, J.; ZHU, F. Named Entity Recognition From Biomedical Text Using SVM. **IEEE 2011 International Conference on Computer and Management (CAMAN)**, [S.l.], 2011.

LOPES, L.; VIEIRA, R. Processamento de Linguagem Natural e o Tratamento Computacional de Linguagens Científicas. In: LINGUAGENS ESPECIALIZADAS EM CORPORA: MODOS DE DIZER E INTERFACES DE PESQUISA. **Anais...** EDIPUCRS, 2010.

QUINLAN, J. R. Induction of Decision Trees. **Machine Learning**, [S.l.], v.1, 1986.

QUINLAN, J. R. **C4.5**: programs for machine learning. [S.l.]: Morgan Kaufmann Publishers, 1993.

SNOWBALL. **Snowball stemmer**. Acessado em 08 de Janeiro de 2013, http://snowball.tartarus.org/.

STOPWORDS. **English Stopwords**. Acessado em 08 de Janeiro de 2013, http://www.ranks.nl/resources/stopwords.html.

TAN, P. N.; STEINBACH, M.; KUMAR, V. **Introduction to Data Mining**. [S.l.]: Pearson Addison Wesley, 2006.

VIANA, M. M. C.; WINCK, A. T. A Decision-Tree Model-Based Approach for Named Entities Recognition on Biomedical Literature. **X-Meeting 2012 (8th International Conference of the Brazilian Association for Bioinformatics and Computational Biology)**, [S.l.], 2012.

WEKA. **Weka 3 - Data Mining with Open Source Machine Learning Software in Java**. Acessado em 08 de Janeiro de 2013, http://www.cs.waikato.ac.nz/ml/weka/.

WINCK, A. T. et al. Association Rules to Identify Receptor and Ligand Structures through Named Entities Recognition. **The Twenty Third International Conference on Industrial, Engineering & Other Applications of Applied Intelligent Systems**, [S.l.], 2010.

WITTEN, I. H.; FRANK, E.; HALL, M. A. **Data Mining**: practical machine learning tools and techniques. 3rd.ed. Burlington, MA: Morgan Kaufmann, 2011.

APÊNDICES

APÊNDICE A – Pseudocódigos

Neste apêndice, estão os pseudocódigos que descrevem as rotinas que foram implementadas neste trabalho.

Algoritmo A.1: Pseudocódigo da rotina que extrai as sentenças do *corpus*

Seja A um *corpus* XML anotado
Seja t um termo de A
Seja c um termo identificado como classe
Seja s uma sentença de A contendo distintos t
Seja qs a quantidade de sentenças em A
Seja S uma matriz de sentenças com qs linhas preenchidas e duas colunas, armazenando os termos extraídos de uma sentença s e sua respectiva classe
para cada s em A **faça**
    $qs++$
    para cada t em s **faça**
        se t não for identificado como c **então**
            $S[qs][1] \leftarrow t$
        senão
            $S[qs][0] \leftarrow S[qs][0] + t$
        fim se
    fim para
fim para

Algoritmo A.2: Pseudocódigo da rotina que conta classes e marcadores distintos presentes no *corpus*

Seja A um *corpus* XML anotado
Seja s uma sentença de A
Seja t um termo de A
Seja m um termo identificado como um marcador
Seja qm a quantidade de m distintos em A
Seja M uma matriz contendo qm linhas preenchidas e duas colunas, armazenando um m distinto e a quantidade de m em A, respectivamente
Seja c um termo identificado como classe
Seja qc a quantidade de c distintos em A
Seja C uma matriz contendo qc linhas preenchidas e duas colunas, armazenando um c distinto e a quantidade de c em A, respectivamente
para cada s em A **faça**
    para cada t em s **faça**
        se t for identificado como m **então**
            se t não estiver contido em M **então**
                $qm++$
                $M[qm][0] \leftarrow t$
                $M[qm][1]++$
            fim se
        fim se
        se t for identificado como c **então**
            se c não estiver contido em C **então**
                $qc++$
                $C[qc][0] \leftarrow t$
                $C[qc][1]++$
            fim se
        fim se
    fim para
fim para

Algoritmo A.3: Pseudocódigo da rotina que remove as sentenças do *corpus* que não possuem ENs em seu conteúdo

Seja S uma matriz de sentenças s gerada no Algoritmo A.1,
com duas colunas armazenando termos de s e sua respectiva classe
Seja t um termo de S
para cada s em S **faça**
    se classe de s == "NONE" **OU** classe de s == "TO_SEE" **então**
        Remove s de S
    fim se
fim para

Algoritmo A.4: Pseudocódigo que remove as *stopwords*, os números e os sinais de pontuação do *corpus*

Seja S uma matriz de sentenças s gerada no Algoritmo A.6, com duas colunas armazenando termos de s e sua respectiva classe
Seja t um termo de S
Seja c um caractere de t
Seja W uma stoplist
para cada s em S **faça**
    para cada t em s **faça**
        para cada c em t **faça**
            se c for um número **OU** c for um sinal de pontuação **então**
                Remove c de t
            fim se
        fim para
        se t não for uma string vazia **E** t estiver contido em W **então**
            Remove t de s
        fim se
    fim para
fim para

Algoritmo A.5: Pseudocódigo que realiza o processo de *stemming* do *corpus*

Seja S uma matriz de sentenças s atualizada no Algoritmo 3
Seja t um termo de S
Seja qt a quantidade de distintos t em S
Seja s uma sentença de S contendo distintos t
Seja B um conjunto de qt termos t e seu respectivo radical
para cada s em S **faça**
    para cada t em s **faça**
        se t estiver contido em B **então**
            Atualiza t com seu respectivo radical em B
        fim se
    fim para
fim para

Algoritmo A.6: Pseudocódigo da rotina que agrupa as classes do *corpus*

Seja S uma matriz de sentenças s gerada no Algoritmo A.1, com duas colunas armazenando termos de s e sua respectiva classe
para cada s em S **faça**
  se classe de s == "DISONLY" **então**
    classe de $s \leftarrow$ DISEASE
  fim se
  se classe de s == "TREATONLY" **OU** classe de s == "TREAT_FOR_DIS" **OU** classe de s == "TREAT_NO_FOR_DIS" **então**
    classe de $s \leftarrow$ TREATMENT
  fim se
  se classe de s == "VAGUE" **OU** classe de s == "PREVENT" **OU** classe de s == "SIDE_EFF" **então**
    classe de $s \leftarrow$ OTHER
  fim se
fim para

Algoritmo A.7: Pseudocódigo que identifica os termos distintos existentes no *corpus*

Seja S uma matriz de sentenças s atualizada no Algoritmo A.6
Seja t um termo de S
Seja m um termo t identificado como um marcador
Seja c um termo c identificado como classe
Seja qt a quantidade de distintos t em S
Seja T um vetor com qt termos preenchidos
para cada s em S **faça**
  para cada t em S **faça**
    se t não for identificado como m **E** t não for identificado como c **então**
      se t não estiver contido em T **então**
        $qt++$
        $T[qt] \leftarrow t$
      fim se
    fim se
  fim para
fim para

Algoritmo A.8: Pseudocódigo que calcula o valor de TF-IDF para cada um dos termos distintos do *corpus*

```
Seja T uma matriz com qt linhas preenchidas e três colunas, armazenando
os termos distintos de T gerado no Algoritmo A.7, sua frequência seu respectivo
valor de TF-IDF
Seja S uma matriz de sentenças s atualizada no Algoritmo A.5
Seja t um termo de S
Seja tf o valor de TF calculado para o termo, conforme equação 2.4
Seja idf o valor de IDF calculado para o termo, conforme equação 2.5
Seja tf_idf o valor de TF-IDF calculado para o termo, conforme equação 2.6
para cada s em S faça
    para cada t em s faça
        se t não for identificado como m E t não for identificado como c então
            se t estiver contido em T então
                i ← posição de T contendo t
                T[i][1] + +
            fim se
        fim se
    fim para
fim para
para cada i de T faça
    Computa tf
    Computa idf
    Computa tf_idf
    T[i][2] ← tf_idf
fim para
```

Algoritmo A.9: Pseudocódigo que seleciona os termos que serão utilizados para a geração dos *datasets*

Seja T uma matriz de termos e suas respectivas frequências e TF-IDF gerada no Algoritmo A.8
Seja l o número de linhas de T
Seja p um peso definido como parâmetro para calculo da escolha do desvio padrão
Seja $media$ a média dos valores de TF-IDF dos termos de T
Seja dp o desvio padrão dos valores de TF-IDF dos termos T
Seja pc o ponto de corte para selecionar termos de T
Seja ts o número de termos selecionados
Seja TS um vetor contendo ts termos selecionados
para todos t em T **faça**
    Computa $media$
    Computa dp
    $pc \leftarrow media + (p \times dp)$
fim para
$i \leftarrow 0$
enquanto $i \neq l$ **E** $T[i][2] \geq pc$ **faça**
    Inclui $T[i][2]$ em TS
    $i++$
fim enquanto

Algoritmo A.10: Pseudocódigo que gera um *dataset* utilizando o conjunto de termos selecionados

Seja S uma matriz de sentenças atualizada no Algoritmo A.6
Seja s uma sentença de S
Seja qs o total de sentenças em S
Seja t um termo de S
Seja $classe$ a classe de s
Seja TS um vetor de termos selecionados no Algoritmo A.9
Seja ts o total de termos em TS
Seja ns o número da sentença em S e nt o número do termo em TS
Seja $Dataset$ uma matriz contendo qs linhas e $ts + 1$ atributos
para todas as células de $Dataset$ **faça**
    Inicializa célula com 0
fim para
para cada s em S **faça**
    $ns++$
    para cada t em s **faça**
        se t estiver contido em TS **então**
            $nt \leftarrow$ posição de t em TS
            $Dataset[ns][nt] \leftarrow 1$
        fim se
    fim para
    Obtém $classe$ de s
    $Dataset[ns][ts + 1] \leftarrow classe$
fim para

Algoritmo A.11: Pseudocódigo que gera o conjunto de regras

Seja A o modelo de árvore retornado pelo *Weka*
Seja n a raiz da árvore
Seja n_e o nó filho à esquerda de n
Seja n_d o nó filho à direita de n
Seja f o nó identificado como folha de A
Seja t o termo contido em n
Seja k_e o valor da aresta esquerda de n
Seja k_d o valor da aresta direita de n
Seja c a classe contida em f
Seja R um conjunto de regras
Seja r uma regra, representada por uma tupla que armazena uma lista de predicados e uma classe
Seja P uma lista de predicados
Seja p um predicado, representado por uma tupla que armazena um termo e um valor de aresta
Procedure $geraRegra(R, P, n)$
se n for uma folha **então**
    $f \leftarrow n$
    $c \leftarrow$ classe de f
    Insere P em r
    Insere c em r
    Insere r em R
    Limpa r
    retorna R
senão
    $t \leftarrow$ termo de n
    $k\_e \leftarrow$ valor da aresta esquerda de n
    $k\_d \leftarrow$ valor da aresta direita de n
    Insere t em p
    Insere k_e em p
    Insere p em P
    $R \leftarrow geraRegra(R, P, n\_e)$
    Remove p de P
    Remove k_e de p
    Insere k_d em p
    Insere p em P
    $R \leftarrow geraRegra(R, P, n\_d)$
    Remove p de P
    Remove k_d de p
    Remove t de p
fim se
retorna R

Algoritmo A.12: Pseudocódigo que gera as amostras para a aplicação das regras geradas em A.11

Seja M uma matriz de sentenças atualizada no Algoritmo A.6
Seja m o número de sentenças em M
Seja AM uma matriz de sentenças como M, que representa uma amostra marcada
Seja am o número de sentenças em AM
Seja A um vetor de sentenças, que representa uma amostra não-marcada
Seja i um índice de uma sentença
Seja V um vetor de índices de sentenças
Seja v o número de índices em V
Seja C um conjunto de vetores de índices de sentenças
Seja c o número de vetores em C
Seja ns o número de sentenças por amostra
Seja na o número de amostras serem geradas
Seja p a porcentagem de M que deve ser utilizada para gerar as amostras
Seja CM um conjunto de amostras marcadas
Seja ag o número de amostras geradas
Seja CA um conjunto de amostras não-marcadas
$ns \leftarrow (n \times p)/100$
$c \leftarrow 0$
enquanto $c \neq na$ **faça**
    $v \leftarrow 0$
    enquanto $v \neq ns$ **faça**
        $i \leftarrow$ valor aleatório entre 1 e m
        se i não estiver contido em V **então**
            $V[v] \leftarrow i$
            $v++$
        fim se
    fim enquanto
    se V não estiver contido em C **então**
        $C[c] \leftarrow V$
        $c++$
    fim se
fim enquanto
$ag \leftarrow 0$
enquanto $ag \neq na$ **faça**
    $am \leftarrow 0$
    enquanto $am \neq ns$ **faça**
        $i \leftarrow C[ag][am]$
        $AM[am][0] \leftarrow M[i][0]$ {inserção da sentença na amostra marcada}
        $AM[am][1] \leftarrow M[i][1]$ {inserção da classe na amostra marcada}
        $A[am] \leftarrow M[i][0]$
        $am++$
    fim enquanto
    $CM[ag] \leftarrow AM$
    $CA[ag] \leftarrow A$
    $ag++$
fim enquanto

Algoritmo A.13: Pseudocódigo que estabelece a correspondência entre cada sentença de uma amostra com uma regra do conjunto de regras gerado

```
Seja S o conjunto de sentenças não-marcadas gerada pelo Algoritmo A.12
Seja s uma sentença de S
Seja R o conjunto de regras gerada pelo Algoritmo A.11
Seja r uma regra de R
Seja P a lista de predicados de r
Seja c a classe de r
Seja p um predicado de P
Seja t_p o termo de p
Seja k_p o valor de teste de p
Seja CR o conjunto de classes que resultam dos testes
para cada s em S faça
   para cada r em R faça
      P ← lista de predicados de R
      para cada p em P faça
         t_p ← termo de p
         k_p ← valor de teste de p
         se k_p == 0 então
            se t_p estiver presente em s então
               Pula para a próxima r em R
            fim se
         senão
            se t_p não estiver presente em s então
               Pula para a próxima r em R
            fim se
         fim se
      fim para
      c ← classe de r
      Insere c em CR
      Sai do laço
   fim para
fim para
```

Algoritmo A.14: Pseudocódigo que gera a matriz de confusão para os resultados dos testes com uma amostra

```
1: Seja AM uma amostra marcada gerada no Algoritmo A.12
2: Seja C um vetor de classes de AM
3: Seja CR o vetor de classes obtido pelos testes
4: Seja sm uma sentença marcada de AM
5: Seja c o número de classes em C
6: Seja M uma matriz de confusão
7: Seja lin uma linha da matriz de confusão
8: Seja col uma coluna da matriz de confusão
9: Seja i um contador
10: c ← 0
11: para cada sm en AM faça
12:     C[c] ← sm[1]
13:     c++
14: fim para
15: para cada célula de M faça
16:     Inicializa célula com 0
17: fim para
18: i ← 0
19: enquanto i ≠ c faça
20:     se C[i] == DISEASE então
21:         lin ← 0
22:     senão se C[i] == TREATMENT então
23:         lin ← 1
24:     senão
25:         lin ← 2
26:     fim se
27:     se CR[i] == DISEASE então
28:         col ← 0
29:     senão se CR[i] == TREATMENT então
30:         col ← 1
31:     senão
32:         col ← 2
33:     fim se
34:     M[lin][col]++
35:     i++
36: fim enquanto
```

Algoritmo A.15: Pseudocódigo que calcula os valores de precisão, *recall* e *F-score* para uma amostra

```
1: Seja M a matriz de confusão gerada pelo Algoritmo A.14
2: Seja V a matriz com os valores de precisão, recall e F-score para cada
   classe da amostra
3: Seja P o valor da precisão
4: Seja R o valor do recall
5: Seja F o valor do F-score
6: Seja rec a soma das instâncias recuperadas
7: Seja v_rec o vetor de somas das instâncias recuperadas
8: Seja rel a soma das instâncias relevantes
9: Seja v_rel o vetor de somas das instâncias relevantes
10: Seja n o número de colunas de M
11: Seja lin uma linha de M
12: Seja col uma coluna de M
13: para lin = 0 → n − 1 faça
14:     rec ← 0
15:     rel ← 0
16:     para col = 0 → n − 1 faça
17:         rec ← rec + M[col][lin]
18:         rel ← rel + M[lin][col]
19:     fim para
20:     Insere rec em v_rec
21:     Insere rel em v_rel
22: fim para
23: para lin = 0 → n − 1 faça
24:     para col = 0 → n − 1 faça
25:         se lin == col então
26:             P ← M[lin][col]/v_rec[col]
27:             R ← M[lin][col]/v_rel[col]
28:             F ← (P × R)/((P + R)/2)
29:             Insere P em v_P
30:             Insere R em v_R
31:             Insere F em v_F
32:         fim se
33:     fim para
34: fim para
35: para lin = 0 → n − 1 faça
36:     V[lin][0] ← v_P[lin]
37:     V[lin][1] ← v_R[lin]
38:     V[lin][2] ← v_F[lin]
39: fim para
```