Leonardo Hirokazu de Souza Hamada

Desambiguando Homógrafos-Heterófonos em Português Brasileiro com Algoritmos de Indução de Regras e Classificadores

Belém

2016

Leonardo Hirokazu de Souza Hamada

Desambiguando Homógrafos-Heterófonos em Português Brasileiro com Algoritmos de Indução de Regras e Classificadores

Dissertação apresentada à Universidade Federal do Pará, como parte dos requisitos do Programa de Pós-Graduação em Ciência da Computação, para a obtenção do Título de Mestre. Área de concentração: Sistemas Inteligentes

Universidade Federal do Pará – UFPA

Instituto de Ciências Exatas e Naturais – ICEN

Programa de Pós-Graduação em Ciência da Computação – PPGCC

Orientador: Prof. Dr. Nelson Cruz Sampaio Neto

Belém

2016

Dados Internacionais de Catalogação- na-Publicação
(CIP)
Biblioteca Central /UFPA

Hamada, Leonardo Hirozaku de Souza ,1978-
Desambiguando Homógrafos-Heterófonos em Português Brasileiro com Algoritmos de Indução de Regras e Classificadores, em Belém, Pará / Leonardo Hirozaku de Souza Hamada. - 2016.
93f. : il. ; 29 cm

Orientador: Prof. Dr. Nelson Cruz Sampaio Neto.
Dissertação (Mestrado) - Universidade Federal do Pará, Instituto de Ciências Exatas e Naturais – ICEN, Programa de Pós-Graduação em Ciência da Computação, Belém, 2016.

1. Sintese de voz . 2. Sistema de processamento da fala. 3. Homógrafos e Heterófonos - Desambiguação . 4. Análise morfossintática. 5. Analise semântica. I. Titulo.

CDD – 22 ed. 006.54

UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE CIÊNCIAS EXATAS E NATURAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

LEONARDO HIROZAKU DE SOUZA HAMADA

DESAMBIGUANDO HOMÓGRAFOS-HETERÓFONOS EM PORTUGUÊS BRASILEIRO COM ALGORITMOS DE INDUÇÃO DE REGRAS E CLASSIFICADORES

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal do Pará como requisito para obtenção do título de Mestre em Ciência da Computação, defendida e aprovada em 03/08/2016, pela banca examinadora constituída pelos seguintes membros:

Nelson Cruz Sampaio Neto
Prof. Dr. Nelson Cruz Sampaio Neto
Orientador – PPGCC/UFPA

Jefferson M. de Morais
Prof. Dr. Jefferson Magalhães de Morais
Membro Interno – PPGCC/UFPA

Fabíola Pantoja Oliveira Araújo
Profa. Dra. Fabíola Pantoja Oliveira Araújo
Membro Interno – FACOMP/UFPA/CASTANHAL

Visto: Jefferson M. de Morais
Prof. Dr. Jefferson Magalhães de Morais
Coordenador do PPGCC/UFPA

Prof. Dr. Jefferson Magalhães de Morais
Coordenador do PPGCC
Mat.: SIAPE: 2378314

Aos meus pais e irmãos, pela dedicação e apoio incondicional que recebi o tempo todo.

Agradecimentos

Ao **Prof. Dr. Nelson Neto**, pela oportunidade em permitir realizar este trabalho e apoio ao longo do curso.

Aos colegas e diretores, pela cooperação que possibilitou na conclusão deste trabalho.

"O primeiro passo em encontrar a solução de um problema, com frequência envolve descobrir outro problema com solução existente."

(anônimo)

Resumo

No Português falado e em outras línguas, existe um conjunto de palavras que se escreve exatamente da mesma forma e que no entanto ao pronunciá-las, possuem uma diferença de tonicidade e no caso do português, esta diferença ocorre na vogal central. Ao agrupar em pares, uma palavra soa com entoação aberta e uma outra com a mesma escrita, de forma mais fechada. Palavras com este comportamento linguístico se denominam homógrafos-hetorófonos e no discurso, podem assumir classes gramaticais distintas ou pertencer à mesma classe e também estar em uma outra forma verbal, por exemplo. A preocupação da pronúncia esperada destas palavras é de relevância nos sistemas de conversão texto-fala, tecnologia computacional capaz de automaticamente transformar texto comum em voz audível. Neste contexto, uma solução que possa satisfatoriamente prever de forma automática a pronúncia correta dos homógrafos-heterófonos durante o processo que gera a voz sintética, produz melhorias na inteligibilidade e naturalidade. Problemas de decisões em elementos ambíguos da linguagem natural, do ponto de vista computacional, são objetos de estudos centrais na área de processamento da linguagem natural. Para o caso dos homógrafos-heterófonos, a abordagem sensível é utilizar as técnicas estudadas para desambiguação do sentido da palavra, que incluem o emprego de algoritmos de aprendizado de máquina, regras construídas manualmente ou induzidas por algoritmos, repositório e representação de conhecimento, *corpus* linguísticos e outros. Este trabalho propõe e modela a solução de desambiguação automática de homógrafos-heterófonos utilizando estratégia de aprendizado de máquina, especificamente a supervisionada, utilizando como fonte de treino para os algoritmos de desambiguação, exemplos de frases coletados de *corpus* onde os homógrafos-heterófonos contidos foram manualmente desambiguados. Para verificar a empregabilidade desta solução proposta em um sistema de conversão texto-fala personalizado para o Português Brasileiro, é necessário realizar algumas análises de desempenho, principalmente em novos textos em que o desambiguador ainda não teve acesso. Verificou-se que o desempenho de acertos de desambiguação automática das pronúncias dos homógrafos-heterófonos é fundamentalmente dependente da natureza do texto utilizado durante o treino dos classificadores e algoritmos de indução de regras. As maiores dificuldades foram encontradas durante a coleta de amostras de frases para a base de treino, sendo que há poucos exemplos para o emprego de alguns homógrafos-heterófonos, que em geral ocorrem de forma desbalanceada em um dos pares de pronúncia em textos do Português Brasileiro.

Palavras-chave: desambiguação de homógrafos-heterófonos. aprendizado de máquina supervisionado. processamento de linguagem natural. desambiguação do sentido da palavra.

Abstract

In Portuguese and other languages, there exists a set of words that are written exactly the same, however when those words are pronounced, there is a difference in tone and for the Portuguese language, this difference is in the central vowel. When grouped in pairs, one word is heard with an open tone and another with the same written form, is heard with a closed tone. Words that possesses these behaviors are called heterophonic-homographs and in speech, can admit the same or different grammatical classes, or it can be in another verb form, for example. A concern in proper pronunciation of those words is encountered in text-to-speech systems, a technology that automatically transforms common text in audible voice. In this context, a satisfactory solution capable of automatically predicting the correct pronunciation of heterophonic-homographs during the processes that leads to the synthetic voice, could bring improvement in intelligibility and naturalness. Decision problems in ambiguous elements in natural language are central object of natural language processing field. For heterophonic-homographs case, a sensible approach is to employ word sense disambiguation methods, including machine learning algorithms, manually or automatically constructed rules, knowledge repository and representations, linguistic corpus and others. This work proposes and models a solution to automatically disambiguate heterophonic-homographs with supervised machine learning, using phrases collected from corpus where heterophonic-homographs were manually disambiguated. To verify the suitability of this proposed solution in a text-to-speech system customized for the Brazilian Portuguese, some performance analysis in new texts that where not used during training is needed. The conclusion reached is that the accuracy of automatic disambiguation of heterophonic-homographs is highly dependent in the text nature used to train the classifiers and the rule induction algorithms. There was difficulty in collecting phrase samples of some heterophonic-homographs because of the differences in occurrence and use of the pronunciation pair in Brazilian Portuguese texts.

Keywords: heterophonic-homographs disambiguation. supervised machine learning. natural language processing. word sense disambiguation.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ACC	*Accuracy* (acurácia)
AODE	*Aggregating One-Dependence Estimators*
ARFF	*Attribute-Relation File Format*
CV	*Cross-Validation*
DD	*Data-Driven*
FURIA	*Unordered Fuzzy Rule Induction*
HH	Homógrafo-Heterófono
IA	Inteligência Artificial
KNN	*k-Nearest Neighbor*
MCC	*Matthews Correlation Coefficient*
PB	Português Brasileiro
PE	Português Europeu
PLN	Processamento de Linguagem Natural
POS	*Part-Of-Speech* (parte do discurso)
RIPPER	*Repeated Incremental Pruning to Produce Error Reduction*
Ridor	Aprendiz de regras *RIpple-DOwn*
SO	Sistema Operacional
SQL	*Structured Query Language*
SVM	*Support Vector Machine*
TTS	*Text-To-Speech*
WEKA	*Waikato Environment for Knowledge Analysis*
WSD	*Word Sense Disambiguation*

Sumário

1 Introdução

Observe o uso da palavra “gosto” nas duas frases a seguir da obra “Dom Casmurro” de Machado de Assis:

(i) Sim, mas eu não **gosto** de imitações em casa.

(ii) O **gosto** que isto me deu foi enorme.

Percebe-se que a pronúncia dada à palavra “gosto” na primeira frase é diferente da segunda frase. Enquanto que na primeira frase a palavra “gosto” é pronunciada de forma mais aberta “g[O]sto” (“o” aberto), na segunda frase esta é pronunciada de forma fechada “g[o]sto” (“o” fechado).

No exemplo, a presença de ambiguidade na pronúncia da palavra “gosto” entre uma frase e outra, mesmo tendo a mesma escrita, é um fenômeno linguístico conhecido como homógrafos-heterófonos (HH) (SEARA et al., 2001). Em geral, a diferença está na tonicidade da vogal central entre os pares de HH, que podem ter significados, formas verbais ou classes gramaticais diferentes. Em algumas situações, a informação morfossintática é suficiente para resolver o problema da desambiguação, como no caso acima, onde em (i) a palavra “gosto” é um verbo e em (ii) é um substantivo.

Nos casos em que os HH pertencem à classes gramaticais distintas, foi proposto por Silva, Braga e Resende Jr (2012) que a análise morfossintática seria suficiente para a diferenciação, e para os HH de mesma classe gramatical, como no exemplo a seguir, uma análise semântica seria o mais adequado.

(i) Ele estava com uma **sede** insuportável.

(ii) A **sede** da empresa fica em Paris.

Na primeira frase do exemplo anterior, “sede” é um substantivo com o sentido de vontade de beber algo e tem pronúncia fechada, já na segunda frase, “sede” também é um substantivo, porém no sentido de escritório central de uma empresa e tem pronúncia aberta. Por conseguinte, a classe gramatical da palavra “sede” sozinha não fornece informação necessária para diferenciação da sua pronúncia sem ambiguidade. Assim, a desambiguação de pronúncia dos HH usando métodos automáticos requer o emprego de múltiplas estratégias.

1.1 Contextualização e terminologias

Segundo Cambria e White (2014), processamento de linguagem natural (PLN) é uma coleção de técnicas computacionais e teorias para análise automática e representação da linguagem humana. Os autores destacam que a vasta maioria das técnicas atuais ainda é baseada na representação sintática do texto e, nas aplicações PLN mais populares, tratam a análise textual como casamento de palavras e padrões, ou seja, para verificar o significado de um segmento de texto, o processamento é feito a nível de palavra.

A desambiguação do sentido da palavra (WSD, do inglês "*word sense disambiguation*") é uma subárea do PLN (MCCARTHY, 2009), onde sistemas computacionais são projetados para identificar o sentido ou significado correto de uma palavra dentro de um dado contexto (NAVIGLI, 2012). A WSD tem potencial de auxiliar nas aplicações onde a procura dos significados apropriados das palavras no contexto é de fundamental importância (MCCARTHY, 2009).

Desconsiderando os sinais de pontuação, um texto T pode ser visto como uma sequência de palavras $(w_1, w_2, ..., w_n)$, onde WSD é formalmente a tarefa de atribuir os sentidos apropriados para todas ou algumas das palavras em T, isto é, construir um mapa A a partir das palavras para os seus respectivos sentidos, tal que $A(i) \subseteq$ *Sentidos* $D(w_i)$, onde *Sentidos* $D(w_i)$ é o conjunto dos sentidos codificados no dicionário D para a palavra w_i, e $A(i)$ é o subconjunto dos sentidos de w_i que são apropriados no contexto T. O mapa A pode atribuir mais de um significado para cada palavra $w_i \in T$, porém é tipicamente selecionado o sentido mais apropriado, isto é, $|A(i)| = 1$ (HUSAIN; BEG, 2013).

Computacionalmente, considera-se a ambiguidade de pronúncia dos HH como sendo um problema tipicamente WSD, pois enquadra-se na definição da mesma, e em relação aos sistemas PLN, a ambiguidade entre palavras pode ser caracterizada da seguinte forma:

> Em um dado momento, um sistema PLN recebe um segmento de informação que pode ter várias interpretações e este precisa decidir qual interpretação é a mais apropriada para aquele contexto. A fim de resolver essa dificuldade, é necessário desambiguar semanticamente, sintaticamente ou estruturalmente duas ou mais formas distintas, com base nas propriedades que circundam o contexto (CARDIE, 1996).

A Figura 1 é uma reprodução do trabalho desenvolvido pelos pesquisadores Braga e Marques (2007) sobre um sistema de conversão texto-fala (TTS, do inglês "*text-to-speech*") para o Português Europeu (PE). Nessa pesquisa, os autores destacam que devido a natureza da transcrição ortográfica automática, os HH são casos em que se deve escolher entre duas pronúncias possíveis, ou seja, informar ao conversor grafema-fone se a vogal central é aberta ou fechada, e que a solução é complexa. Para determinar a pronúncia desejada, os autores desenvolveram um conjunto de regras acoplado ao analisador morfológico,

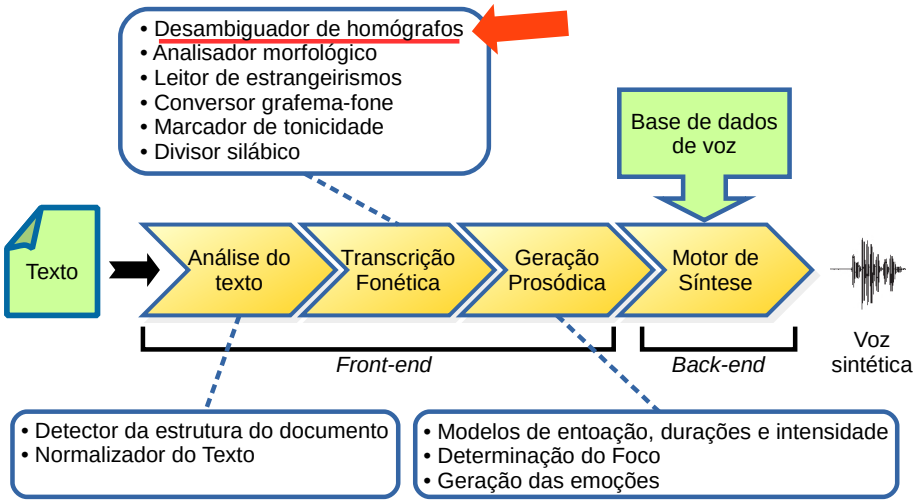

Figura 1 – Visão genérica da arquitetura de um TTS, com seus componentes funcionais constituintes separados nos módulos *front-end*, que é composto por ferramentas computacionais PLN, e *back-end*, que é responsável pelo processamento e geração de voz sintética, segundo Costa et al. (2012).

Fonte: A figura é uma reprodução de Braga e Marques (2007, p. 30)

atingindo uma taxa de acerto para pronúncias corretas dos HH na ordem de 96,9% em resultados experimentais no *corpus* Natura-Diário do Minho (BRAGA; MARQUES, 2007).

Uma das linhas de pesquisa bem sucedidas no PLN é a abordagem baseada no conceito *data-driven* (DD), que define uma arquitetura onde a computação é ditada pela dependência dos dados. Em conjunto com algoritmos de aprendizado de máquina, essa estratégia tem sido aplicada para construir modelos estatísticos e classificadores a partir das informações extraídas de grandes bases de texto (comumente chamadas na literatura de *corpus*) (MORAIS; VIOLARO, 2005). Neste trabalho, a abordagem DD será adotada para desambiguar a pronúncia dos HH.

O problema da WSD é considerado IA-completo, isto é, de dificuldade equivalente aos problemas centrais da Inteligencia Artificial (IA), devido a uma variedade de fatores (NAVIGLI, 2012). Nesse sentido, a questão da desambiguação de pronúncias dos HH, em paralelo a WSD, podem ser convertidas em problemas de classificação, onde as pronúncias são classes, e esta é a visão compartilhada por este trabalho.

1.2 Justificativa e Contribuição

A resolução do problema de ambiguidade entre palavras é um ponto central para aplicações de processamento de linguagem natural e suas tarefas associadas, incluindo entre outras, tradução automática (ou *machine translation*), recuperação de informação,

síntese e reconhecimento de voz e sumarização automática de texto (IDE; VÉRONIS, 1998), (RESNIK, 2006 *apud* MCCARTHY, 2009).

Mesmo que em geral, o número de HH existentes represente um percentual muito pequeno nos segmentos de texto no Português Brasileiro (PB) (a frequência é menor que 1% na pesquisa realizada por Silva, Braga e Resende Jr (2009)), no contexto do TTS, a transcrição fonética equivocada e a consequente reprodução da voz não soam de forma agradável a qualquer ouvinte, atraindo muita atenção para o erro corrente. Por isso, diminuir os erros de pronúncia entre HH melhora significativamente a qualidade da voz sintética no que tange a sua naturalidade e inteligibilidade.

Vale ressaltar que esta pesquisa aborda uma área e um idioma onde os recursos e as ferramentas PLN ainda são escassos e necessitam de contribuições e o compartilhamento da implementação em forma de código-aberto poderá contribuir neste aspecto. Por exemplo, apesar da reconhecida importância, não é possível encontrar um desambiguador de HH disponível para o PB. Diante do exposto, todas as bases de texto etiquetadas e as ferramentas aqui desenvolvidas serão livremente disponibilizados para a comunidade e o público geral interessado.

A abordagem apresentada neste trabalho, baseada em algoritmos inteligentes de classificação supervisionada e indução de regras, visa diminuir a lacuna existente nessa linha de pesquisa para o PB, que apesar de poucas diferenças entre o PE, soluções localizadas tendem a ter melhor desempenho. A ideia é a construção e disponibilização de um desambiguador automático de HH com mecanismos de rápida atualização por meio da adição de novas amostras na base de conhecimento.

Um dos principais desafios, portanto, é a coleta de amostras suficientes para cada palavra de interesse, pois, apesar de existirem textos em grande quantidade acessíveis na Internet, é necessária uma busca específica para obter as amostras, etiquetá-las e alimentar a base de treino. Os resultados iniciais comprovaram a viabilidade do método, ou seja, é possível desambiguar os HH em frases arbitrárias com uma boa taxa de acerto, desde que os classificadores e algoritmos de indução de regras sejam treinados adequadamente.

Este trabalho apontam algumas dificuldades com relação a portabilidade, ou seja, o treino em texto de um domínio diferente ao do texto a ser desambiguado tende a ser inadequado nos sistemas de desambiguação supervisionada (HAMADA; NETO, 2015), em outras palavras, ao treinarmos os classificadores e algoritmos de indução de regras com um conjunto de texto jornalistico, e utilizarmos estes para tentarmos desambiguar os HH contidos em textos de natureza literária, o desempenho tende a degradar.

1.3 Objetivos

O objetivo central deste trabalho é empregar técnicas de aprendizado de máquina para tratar a desambiguação de HH em PB. Diferentemente da maioria dos trabalhos nessa linha, a ideia não é elaborar (apenas) um conjunto de regras linguísticas, mas sim explorar o uso de classificadores inteligentes construídos a partir de treinamento supervisionado para resolver o problema de ambiguidade entre palavras.

Assim, o produto final deste estudo é um componente de *software* modular destinado inicialmente a um sistema real de sintetizador de voz, tendo como função exclusiva a desambiguação de HH identificados nas frases de entrada. Para chegar a este objetivo, foi necessário a exploração da portabilidade e afinação (ou *tuning*) de diferentes algoritmos de aprendizado de máquina através de treinamento e avaliação em textos de diferentes domínios.

Os objetivos específicos do presente trabalho são:

- Avaliar o uso de diferentes algoritmos de aprendizado de máquina na tarefa de desambiguação das várias categorias de HH;
- Estudar a importância de certas características da base de conhecimento na resposta do sistema, como seu tamanho, diferença de domínio, entre outras;
- Disponibilizar um desambiguador de HH funcional e de código-livre para a comunidade.

1.4 Metodologia de pesquisa

A metodologia em que este trabalho se apoia é a identificação, investigação e utilização de técnicas de aprendizado de máquina para solucionamento de problemas de natureza semelhantes, consultando a literatura e observando práticas adotadas em sistemas implementados.

As técnicas identificadas precisam, além de serem aplicáveis ao problema de desambiguação de HH, possuir características compatíveis com o ambiente operacional, ou seja, devem ser métodos que utilizam recursos computacionais disponíveis de forma eficiente com tempo de resposta aceitável.

Objetivando o livre acesso dos códigos desenvolvidos neste projeto, as ferramentas auxiliares e as linguagens de programação empregadas são igualmente de livre acesso ou gratuito com pouca restrição de uso.

Pesquisou-se também na literatura mais recente sobre o tema de desambiguação de HH no PB, servindo como referencia e assim, assegurar que os métodos aplicados neste trabalho ainda não foram empregados por outros autores neste mesmo tipo de problema.

Também houve a necessidade de combinar e adaptar conceitos dinamicamente durante o desenvolvimento, principalmente quanto à avaliações e métricas de desempenho dos algoritmos de aprendizado de máquina. Estas adaptações foram necessárias para atender a demanda de desempenho aceitável, e para encontrar soluções flexíveis concentrando na replicabilidade das operações. Ferramentas auxiliares foram criadas para dar apoio adicional no desenvolvimento e utilizou-se múltiplas linguagens de programação, considerando a adequação para cada caso.

1.5 Organização do trabalho

O restante do trabalho está estruturado da seguinte forma:

O Capítulo 2 (Revisão bibliográfica) apresenta o estado-da-arte sobre WSD na literatura, discute as abordagens disponíveis e as vantagens e desvantagens de cada método. Relata o desenvolvimento histórico e as aplicações que beneficiam com as técnicas baseadas em WSD. Conclui com uma revisão dos trabalhos mais relevantes sobre desambiguação de HH em PB e PE.

O Capítulo 3 (Materiais e métodos) explica a arquitetura do desambiguador de HH, lista a coleção de textos (*corpus*) utilizada, discute a função e importância do etiquetador morfossintático, lista os classificadores e os algoritmos de indução de regras empregados, descreve os atributos para os algoritmos de aprendizado de máquina, conceitua as métricas de avaliação adotadas e relata as ferramentas auxiliares usadas no trabalho.

O Capítulo 4 (Análise dos resultados) está subdividida em duas partes: a primeira demonstra como o domínio predominante do texto, ou seja, sua natureza, tem influência sobre o desambiguador de HH; e a segunda parte, apresenta as etapas para a construção do desambiguador de HH proposto neste trabalho e demonstra a capacidade de desambiguação de HH em um *corpus* de avaliação.

Por fim, o Capítulo 5 (Conclusão) lista as contribuições produzidas decorrente deste trabalho, discute as limitações do desambiguador de HH desenvolvido, principalmente das que são inerentes dos métodos supervisionados e destaca as possíveis atividades relacionadas em trabalhos futuros.

2 Revisão bibliográfica

As inovações atuais nas abordagens em WSD prometem abrir novos rumos na área da PLN (NAVIGLI, 2012). Em Resnik (2006 *apud* MCCARTHY, 2009) e Navigli (2009), são citadas aplicações que podem conter os métodos WSD para melhorar seu desempenho, entre elas:

- **Recuperação de informação:** onde o usuário entra com uma consulta de apenas umas poucas palavras e o sistema retorna os documentos relevantes dentro de uma coleção;
- **Extração de informação:** é o reconhecimento de instancias específicas de conceitos em domínios específicos. Por exemplo, na bioinformática tem-se o interesse em solucionar ambiguidades para dar nomes a genes e proteínas;
- **Pergunta-resposta:** onde o usuário faz uma pergunta ao sistema e este pesquisa em uma coleção de documentos retornando a resposta contendo a informação apropriadamente formada e combinada;
- **Tradução automática:** onde o sistema traduz um texto para uma outra língua designada. A WSD foi historicamente concebida como a tarefa principal a ser solucionada nesta aplicação;
- **Análise de conteúdo:** tarefas que analisam o conteúdo dos textos em termos de ideias, temas, etc. Por exemplo, a classificação de *blogs* tem ganho interesse ultimamente;
- **Processamento de texto:** é uma aplicação relevante na área de PLN. Aqui a WSD pode auxiliar na correção gramatical e em outras tarefas baseadas em evidência do contexto;
- **Lexicografia:** é a tarefa profissional de elaboração de dicionários. A WSD pode prover agrupamentos empíricos de acordo com o significado e indicadores de significância estatística do contexto para significados novos ou existentes;
- **A *Web* semântica:** todas as aplicações citadas acima podem se beneficiar da *Web* semântica. Tratam-se de tecnologias para tornar a *Web* mais legível, não pelo ser humano, mas pelas máquinas, facilitando, por exemplo, a criação de programas que coletam conteúdos da *Web* a partir de diferentes fontes, processando as informações e compartilhando os resultados com outros programas (BERNERS-LEE; HENDLER; LASSILA, 2001).

A categorização dos métodos empregados na WSD podem variar de acordo com a visão e interpretação de cada pesquisador e seguem um mesmo conjunto de ideias, apesar de diferir em pequenos detalhes. Generalizando, as técnicas WSD desenvolvidas na literatura recaem em duas categorias, segundo Husain e Beg (2013):

- **WSD supervisionado:** Emprega técnicas de aprendizagem de máquina e requer conjuntos de exemplos rotulados para formar o conjunto de treino para os classificadores. Estes exemplos são codificados como vetores cujo elementos representam atributos, acompanhado do elemento especial que representa o rótulo do sentido apropriado (ou classe). Em geral, produz melhores resultados que os métodos não-supervisionados;
- **WSD não-supervisionado:** São métodos baseados em *corpora* sem rótulos, ou seja, nesse tipo de técnica, a ideia é que o algoritmo identifique grupos sem que ninguém mostre exemplos de classes.

No método supervisionado, cada palavra é tratada como um problema de classificação diferente (HUSAIN; BEG, 2013), e a desambiguação de HH, como será mostrado na Seção 4.1, assim como na WSD, é extremamente dependente do domínio da aplicação, como afirma Màrquez et al. (2006). Em outras palavras, não parece razoável pensar que o material de treinamento é grande e representativo o suficiente para cobrir todos os tipos possíveis de amostras. Adicionalmente, é preciso estudar até que ponto um treinamento tendo como base um texto jornalístico pode ser portado (usabilidade em outros domínios) para um texto literário, por exemplo. Até onde se pesquisou, essa estratégia ainda não foi abordada para o PB.

Uma outra forma de categorizar os métodos WSD é segundo a utilização de dicionários (HUSAIN; BEG, 2013):

- **WSD baseado em conhecimento:** Depende de recursos de conhecimento léxico externo, como os dicionários legíveis por máquina, enciclopédias, ontologias, etc. São chamados de métodos "ricos em conhecimento";
- **WSD baseado em *corpus*:** Estes métodos não utilizam recursos externos para desambiguação. São chamados de métodos "pobres em conhecimento". A informação é unicamente obtida diretamente do *corpora* (ZAMPIERI, 2012).

Na abordagem supervisionada adotada neste trabalho de desambiguação de HH, o método é enquadrado na categoria WSD baseado em conhecimento, pois a base de treino é enriquecida com etiquetas de classes gramaticais provida pelo etiquetador morfossintático e é usada também no ato de desambiguação de novas frases.

Um outro recurso externo bem conhecido é o WordNet[1], uma ampla base de dados lexical da língua inglesa onde substantivos, verbos, adjetivos e os advérbios são agrupados em forma de conjuntos de sinônimos cognitivos. O projeto OpenWordNet-PT é uma das iniciativas para criar uma versão para o PB (PAIVA; RADEMAKER; MELO, 2012). Os melhores sistemas que empregam WSD baseado em conhecimento usam grafos semânticos para explorar a sua estrutura e selecionar o sentido da palavra (NAVIGLI, 2012).

As palavras podem ser interpretadas de múltiplas maneiras dependendo do contexto devido à ambiguidade da linguagem natural (HUSAIN; BEG, 2013; ZAMPIERI, 2012), tendo como consequência de que utilizar um único método de desambiguação torna-se limitado, assim, para melhorar a acurácia no problema WSD, tipicamente são usados variações que mesclam diferentes métodos, por exemplo, combinar métodos supervisionados com os não-supervisionados (MONTOYO; SUÁREZ; PALOMAR, 2002), combinar métodos que são baseadas em conhecimentos com as baseadas em *corpus* (MONTOYO et al., 2005) ou com os supervisionados (BASILE et al., 2008), outras combinações e estratégias são possíveis.

2.1 Sobre WSD e desambiguação de HH em outras línguas

De acordo com Navigli (2012), até o momento, a maior parte dos trabalhos em WSD ainda é voltada para a língua inglesa, devido a abundância de recursos disponíveis. Porém, há um crescente interesse por outros idiomas como o italiano, espanhol, chinês, japonês, turco, além de alguns trabalhos destinados à língua portuguesa (PE e PB).

A tarefa associada à WSD é considerada histórica na área de PLN, concebida como uma tarefa essencial na tradução automática já no final da década de 1940, segundo o levantamento de Navigli (2009). Já naquele período, os pesquisadores tinham em mente os ingredientes essenciais dos métodos WSD, como o contexto em que a palavra ocorre, informações estatísticas das palavras e seu sentido, recursos de conhecimentos e outros. Logo percebeu-se que a WSD era um problema bastante difícil e os recursos computacionais eram limitados para a época.

Na década de 1960, os pesquisadores reconheceram a dificuldade dos métodos WSD como um obstáculo no desenvolvimento de sistemas de tradução automática. Durante a década de 1970, a abordagem concentrou-se em IA para a compreensão da linguagem, mas tais técnicas não conseguiram gerar soluções que levassem à generalizações para todos os casos, devido a falta de dados.

O momento crítico no desenvolvimento de métodos WSD ocorreu na década de 1980, quando vasta quantidade de recursos lexicais foram disponibilizados, possibilitando a utilização de métodos automáticos de extração de conhecimento. Na década de 1990,

[1] Mais informações sobre WordNet em: <https://wordnet.princeton.edu/>

métodos estatísticos foram empregados em grande escala e surgiram eventos internacionais, como o SENSEVAL[2], para avaliação de sistemas WSD em diferentes línguas (MCCARTHY, 2009).

A partir do trabalho de Yarowsky (1997), mostrava-se o potencial do WSD em aplicações TTS. Nessa pesquisa, o autor apresenta uma tipologia de HH na língua inglesa e algumas técnicas tradicionalmente usadas na desambiguação, tais como N-*gram taggers*, classificadores bayesianos e árvores de decisão, bem como a proposta de um sistema híbrido ao combinar as técnicas descritas.

Yarowsky também desenvolveu um dos primeiros algoritmos semi-supervisionados conhecido como algoritmo de Yarowsky (ABNEY, 2004), o interesse em estas classes de algoritmos semi-supervisionados se deve a existência abundante de dados não etiquetados em contraste com os dados etiquetados, que são muito custosos de serem criados. Os algoritmos semi-supervisionados são considerados uma classe entre os métodos supervisionados e os não-supervisionados, requerendo apenas uma pequena fração de dados iniciais etiquetados no *corpus* original. No entanto, em um estudo mais recente sobre o algoritmo de Yarowsky (MADARIAGA; CASTILLO, 2009), verificou-se que a taxa de acertos de desambiguação em WSD não é favorável em *corpus* contendo domínio abrangente e gerais, apesar de ter um bom desempenho em *corpus* homogêneo, ou seja, de natureza única, o que também se traduz em um desfavorecimento da aplicação deste algoritmo para desambiguar HH em textos de múltipla natureza.

Em seu trabalho, Màrquez et al. (2006) faz uma compreensiva pesquisa do estado-da-arte sobre os métodos de WSD com aprendizado de máquina supervisionada. Nesse trabalho, os autores avaliam comparativamente a eficiência de cinco algoritmos de aprendizado de máquina quando submetidos ao problema WSD na língua inglesa, usando bases de texto de diferentes domínios para treino e teste. O autor conclui que para assegurar a portabilidade de domínios, um processo de afinação (ou *tuning*) para o novo domínio é necessária.

O trabalho de Teodorescu et al. (2011) aborda o tema da construção de um etiquetador baseado em aprendizado de máquina usando o modelo Naïve Bayes voltado aos sistemas de conversão texto-fala para a língua romena. O mesmo etiquetador foi utilizado para desambiguar os HH de acordo com a etiqueta da classe gramatical atribuída às palavras, por exemplo, entre verbos e substantivos.

2.2 Desambiguação de HH em PB e PE

Os HH manifestam-se de forma reduzida na língua portuguesa, aparecendo em cerca de 1% das palavras nas bases de texto da pesquisa feita for Silva, Braga e Resende Jr (2012). O problema reside no fato de que desconsiderar essas informações, no contexto

[2] Mais informações em: <http://www.senseval.org/>

da síntese de voz (foco deste trabalho), fatalmente irá conduzir a transcrição fonética a falhas que resultarão no prejuízo da inteligibilidade, de forma bastante desagradável ao ouvinte ou usuário, já que teremos apenas um fone de saída para cada grafema de entrada (SHULBY; MENDONÇA; MARQUIAFÁVEL, 2013).

Em Ribeiro, Oliveira e Trancoso (2003), um desambiguador que mescla regras linguísticas e modelos probabilísticos de Markov é descrito, e a influência das informações morfossintáticas na tarefa de desambiguação é analisada dentro de um sistema TTS para o PE. O trabalho mais relevante para o PE é creditado à Braga e Marques (2007), que desenvolveram um desambiguador de HH baseadas em regras linguísticas composto por 21 algoritmos que processam 82 pares de HH destinado à um sistema TTS. A vantagem do uso de regras linguísticas, uma vez que estas são elaboradas, é a boa velocidade de processamento dos dados contendo o HH a ser desambiguado, resultando em um baixo custo computacional. Nesse trabalho, os autores declaram ter obtido 100% de acerto para 14 destes algoritmos aplicados nos HH exemplos tratados por estas regras, alegando uma taxa de acerto geral de 96,9% no *corpus* Natura-Diário do Minho. Porém, os autores não realizaram comparações com métodos empregando algoritmos de aprendizado de máquina.

Uns dos primeiros trabalhos relacionados à desambiguação de HH no PB, Seara et al. (2001), Seara et al. (2002) apresentam um analisador morfossintático para solucionar o problema de alternância vocálica entre substantivos e verbos para aplicações em sistemas TTS, sem no entanto abordar a desambiguação semanticamente. Em trabalhos que utilizam em conjunto a abordagem morfossintática e também a semântica (FERRARI; BARBOSA; RESENDE JR, 2003; BARBOSA; FERRARI; RESENDE JR, 2003a; BARBOSA; FERRARI; RESENDE JR, 2003b), as gramáticas implementadas foram avaliadas apenas com um ou dois exemplos.

No trabalho de Shulby, Mendonça e Marquiafável (2013), faz-se um estudo consistindo em duas regras preparadas para desambiguar 226 pares de HH no PB. Essas regras conseguem prever as pronúncias das vogais <e> e <o>, dizendo que para a vogal <e> a transcrição fonética será [E] (aberta) quando o HH for classificado como verbo e [e] (fechada) quando o HH for classificado como substantivo e a vogal <e> estiver na sílaba tônica. Para a vogal <o>, a regra de transcrição fonética é similar.

O trabalho de Braga e Marques (2007) serviu de inspiração para o tratamento de HH em TTS para o PB feito por Silva, Braga e Resende Jr (2012), resultando no trabalho mais completo para o PB até onde se pesquisou, conseguindo desambiguar 111 pares de HH tratados com 23 algoritmos (regras) de desambiguação, uma para cada categoria, usando informações morfossintáticas e semânticas armazenadas em bibliotecas externas.

Para o português, as referências centrais são Braga e Marques (2007) e Silva, Braga e Resende Jr (2012) para o PE e o PB, respectivamente. Em ambos, a desambiguação é realizada através de regras morfossintáticas e semânticas para centenas de pares de HH

divididos em categorias. Para cada categoria de HH (p. e. pares que pertencem a mesma classe gramatical, classes gramaticais diferentes, etc.) foi implementado um algoritmo, que são descritos em pseudocódigo. Porém, existe uma dificuldade para implementar esses algoritmos, que é a ausência de detalhamento e não disponibilização das bibliotecas externas usadas pelos mesmos, citando: listas de lemas morfemas, verbos irregulares, expressões impessoais, entre outras.

3 Materiais e métodos

Com o uso de bases de texto, ou *corpus* linguístico como modelo representativo da língua portuguesa, foi construído um desambiguador híbrido de HH baseado em regras de decisão induzidas por algoritmos e em conjunto com classificadores binários, que serão induzidos/treinados com os exemplos anotados manualmente.

Diferentemente dos trabalhos de desambiguação de HH para o PB e PE existentes, onde emprega-se regras gramaticais elaboradas especificamente para esta tarefa de desambiguação, a abordagem adotada neste trabalho apresenta um nível de detalhamento em relação ao *corpus* de treino utilizado, cita os algoritmos selecionados, descreve e justifica todas as etapas para a construção do desambiguador.

Para validação da solução adotada, realizou-se a análise do desempenho geral em um *corpus* que não foi utilizado para a construção do desambiguador. Dessa forma é possível observar os casos de acertos ótimos e casos que são mais problemáticos e, assim, observar se o produto ainda requer ajustes mediante alimentação de mais amostras de treino ou já encontra-se em um estágio satisfatório.

As seções deste capítulo objetiva descrever os componentes estruturais que formam parte do desambiguador de HH construído neste trabalho. Iniciando com a seção descrevendo a arquitetura geral do desambiguador, onde é apresentado a localização dos módulos, e são descritos as suas respectivas funções. Em seguida, descreve-se o *corpus* coletado para formar a base de treino do desambiguador, onde estes foram categorizados em três grupos conforme a sua natureza predominante. A seção seguinte, trata-se do etiquetador morfossintático, uma ferramenta externa incorporado ao desambiguador de HH para fornecer as etiquetas gramaticais associado à cada palavra da frase que contém um HH reconhecido. Duas seções descrevem acerca dos classificadores e algoritmos de indução de regras adotadas para este trabalho e justifica-se suas escolhas. Uma seção é dedicada para apresentar as métricas de avaliação adotada no desenvolvimento do desambiguador de HH, sendo de grande importância para avaliar o desempenho. Na última seção deste capítulo, descreve-se as ferramentas auxiliares utilizadas para facilitar a construção da base de treino a partir do *corpus* coletado.

3.1 Descrição da arquitetura

Para atingir o objetivo de construir um desambiguador, foi necessário identificar métodos eficazes para tratar a desambiguação de cada par de HH, caso a caso. A arquitetura é composta basicamente por dois submódulos onde ocorre a execução dos algoritmos de

classificação. Um submódulo é constituído por desambiguadores que utilizam indução de regras automáticas e o outro é composto por classificadores treinados. Ambos os submódulos são construídos a partir dos dados de treinamento supervisionado obtidos de *corpus*.

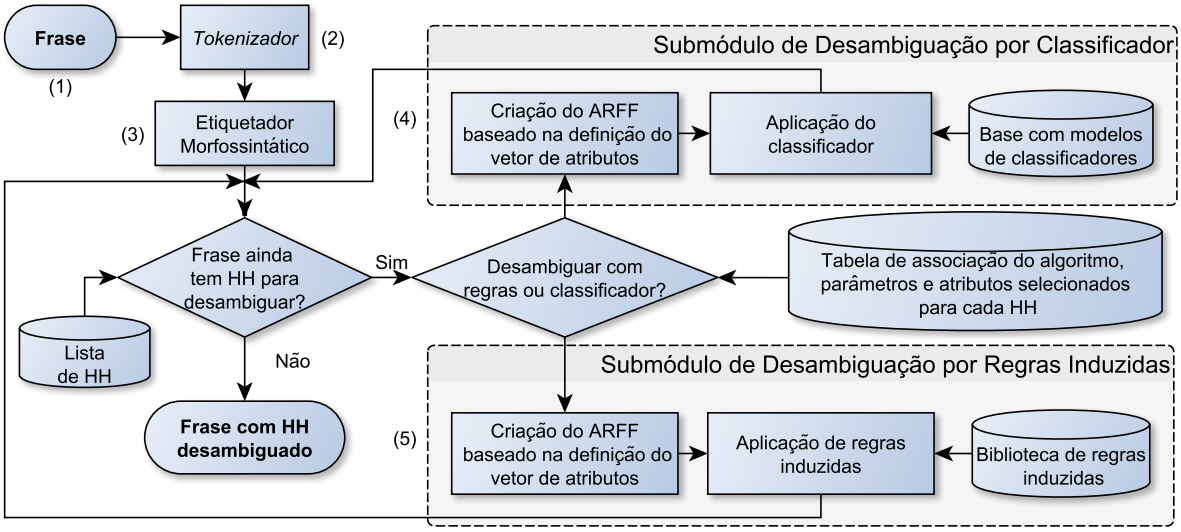

Figura 2 – Representação da arquitetura geral adotada no processo de desambiguação de HH. (1) Frase de entrada; (2) tokenização e limpeza de caracteres especiais; (3) etiquetagem de classes gramaticais; (4) desambiguação por classificador atribuído; (5) desambiguação por regra induzida.

Na visão da arquitetura apresentada na Figura 2, o processo de desambiguação inicia-se recebendo uma frase escrita em PB (1), a frase é tratada pelo tokenizador (2), que é um passo de normalização na qual quebra-se o texto em um conjunto de palavras, retirando pontuações e símbolos não relevantes.

Em seguida, o texto normalizado é tratado pelo etiquetador morfossintático (3), isto é, cada palavra presente na frase de entrada é atribuída uma classe gramatical, por exemplo: “o_ART bar_N estava_VAUX lotado_ADJ”, onde ART, N, VAUX e ADJ são etiquetas para artigo, substantivo, verbo auxiliar e adjetivo, respectivamente.

No passo seguinte, a lista de HH reconhecidos é consultada para determinar se entre as palavras originárias da frase de entrada existem unidades que precisam ser desambiguadas. Essa lista foi obtida a partir do trabalho de Silva, Braga e Resende Jr (2009) e posteriormente expandida. Caso seja identificado um HH, uma base de dados será consultada para indicar o algoritmo “ideal” para desambiguá-lo.

Para que o algoritmo desambiguador gere o resultado desejado, é necessário transformar a frase normalizada e etiquetada em um formato legível pela implementação do algoritmo desambiguador (4 e 5). Isso é feito gerando a representação da frase na forma

de vetor de atributos selecionados para cada HH. No passo final, o modelo do algoritmo pré-treinado é executado recebendo como entrada a instância do vetor de atributos.

A Figura 2 mostra então, o fluxo de processamento do desambiguador de HH de forma integrada, ou seja com todos os seus componentes devidamente configurados e em prontidão para executar suas respectivas tarefas designadas. Tendo como dado de entrada uma frase arbitrária, a saída do sistema é então esta própria frase com as pronúncias dos HH reconhecidos automaticamente desambiguados. Independentemente da frase de entrada contiver ou não um HH reconhecido, a frase de saída sempre estará etiquetada pelo etiquetador morfossintático para possível aproveitamento e processamento posterior pelos componentes de um TTS.

3.2 Tokenização

Normalmente os textos contém diversos tipos de símbolos e uma ação necessária foi a tokenização para remoção, substituição e separação destes símbolos especiais contidos nas frases. Essa operação foi realizada por meio do uma série de instruções dada ao programa utilitário para manipulação de sequencias textuais chamado *sed*[1], disponíveis para sistemas operacionais (SO) derivativos do UNIX e similares como LINUX, BSD e macOS. Existem também versões adaptadas [2] para SO Windows.

3.3 Estruturação do *corpus* de treino

Esta seção descreve as coleções de textos puros não anotados que serviu como ponto de partida para construção dos modelos de aprendizado linguístico. Estes textos são recursos desestruturados (NAVIGLI, 2009) que foram agrupados em dois conjuntos, conforme sua natureza ou domínio predominante: jornalístico e literário. Adicionalmente, formou-se um terceiro conjunto, objetivando adicionar amostras de texto cuja natureza é casual.

3.3.1 *Corpus* jornalístico

Este *corpus* é composto pelos seguintes conjuntos de textos predominantemente de natureza jornalística:

(i) Mac-Morpho, *corpus* anotado composto de textos jornalísticos extraídos de dez seções do jornal diário Folha de São Paulo do ano de 1994, contendo cerca de um milhão de palavras (ALUÍSIO et al., 2003). Utilizou-se a versão revisada por Fonseca e Rosa (2013);

[1] Disponível em <http://www.gnu.org/software/sed/sed.html>
[2] Disponível em:<http://gnuwin32.sourceforge.net/packages/sed.htm>

(ii) CETEN-Folha[3], composto de textos com cerca de 24 milhões de palavras extraídos do jornal Folha de S. Paulo e compilado pelo NILC/USP;

(iii) Texto integral da Constituição[4] da República Federativa do Brasil de 1988;

(iv) Aproximadamente 25% dos artigos em português da enciclopédia livre Wikipédia[5];

(v) LAPSNEWS, uma coletânea de textos jornalísticos retirados de 10 jornais brasileiros disponíveis na Internet, contendo aproximadamente 120 mil frases (NETO et al., 2011).

O Mac-Morpho e CETEN-Folha são dois *corpus* de referências que contém as etiquetagens gramaticais associadas à cada palavra. No entanto, as etiquetas originais não foram consideradas a fim de simplificar a construção do desambiguador de HH. A decisão para este procedimento se deve à incompatibilidade da notação e representação das etiquetas, principalmente do *corpus* CETEN-Folha, que é diferente do formato emitido pelo etiquetador adotado neste trabalho.

3.3.2 *Corpus* literário

Este *corpus* é composto pelas seguintes obras literárias:

(i) Total de 21 obras escritas por José de Alencar, romancista nascido em Mecejana, CE, em 1º de maio de 1829;

(ii) Total de 11 obras escritas por Machado de Assis, poeta nascido no Rio de Janeiro, RJ, em 21 de junho de 1839 (ABL, 2011);

(iii) Total de 149 contos escritos por Olavo Bilac, poeta nascido no Rio de Janeiro, RJ, em 16 de dezembro de 1865;

(iv) Total de 62 poemas e poesias escritas por Castro Alves, poeta nascido em Muritiba, BA, em 14 de março de 1847;

(v) Total de 5 obras escritas por Euclides Rodrigues da Cunha, escritor nascido em Cantagalo (Rio de Janeiro) em 20 de janeiro de 1866 (Portal S. F., 1998);

(vi) A Bíblia Sagrada, versão de João Ferreira de Almeida[6];

(vii) *Corpus* eletrônico anotado Tycho Brahe (GALVES; FARIA, 2010), composto de 66 textos escritos por autores nascidos entre 1380 e 1881.

[3] Disponível em <http://www.linguateca.pt/CETENFolha/>
[4] Disponível em <http://www.planalto.gov.br/ccivil_03/constituicao/constituicao.htm>
[5] Conteúdo obtido em 20 de fevereiro de 2015 via <http://dumps.wikimedia.org/ptwiki/>
[6] Disponível em <http://www.culturabrasil.pro.br/zip/biblia.rtf>

(viii) Total de 18 obras[7] por Aluísio Azevedo, caricaturista, jornalista, romancista e diplomata, nasceu em São Luís, MA, em 14 de abril de 1857;

(ix) A obra Noite na Taverna[8] por Álvares de Azevedo, poeta, contista e ensaísta, nasceu em São Paulo em 12 de setembro de 1831.

3.3.3 *Corpus* complementar

Após anotação de uma quantidade significativa de HH nos *corpus* anteriores, percebeu-se que os textos de domínios literários e jornalísticos continham o emprego relativamente escasso de HH em primeira pessoa incluindo variações contextuais restritas, e por este motivo, para fim de enriquecimento adicional, decidiu-se incluir um terceiro *corpus* para complementar a base de treino.

Este *corpus* é composto inteiramente de postagens na língua portuguesa pelos usuários do serviço Twitter, contendo HH reconhecidos neste trabalho, coletados entre 17 a 25 de março e 6 a 26 de outubro de 2015, utilizando *scripts* do autor Hale (2014), exclusivo para este propósito.

O desafio de anotar este *corpus* é a natureza casual das frases que, muitas vezes, contém erros ortográficos e abreviações típicas das redes sociais. As frases de interesse foram criteriosamente selecionadas para se tornar parte da base de treino. Foram anotados 4.713 ocorrências de HH neste *corpus* e incorporados à base de treino final.

3.4 Etiquetador morfossintático

No intuito de reconhecer um certo HH, o sistema de desambiguação compara cada palavra na frase de entrada com os HH registrados em uma base de dados. Para isso, antes, a frase precisa ser segmentada (ou tokenizada), ou seja, as palavras da frase devem ser separadas e os sinais de pontuações e símbolos especiais retirados. O texto tokenizado é então encaminhado ao etiquetador morfossintático (POS *tagger*, do inglês "*Part-of-Speech Tagger*") e depois ao algoritmo de desambiguação, representado na Figura 3, que responderá qual é a pronúncia adequada do HH, ou seja, se a sua vogal central deve ser aberta ou fechada.

Na Figura 3, percebe-se uma separação entre a análise gramatical e a resolução de ambiguidade. Essa divisão deve-se ao fato das línguas neolatinas, como o português, serem altamente flexionais (ou fusionais). Nesse sentido, a análise morfológica pode ser relevante e ajudar a resolver certos casos de ambiguidade, como veremos mais adiante.

[7] Disponível em <http://www.biblio.com.br/conteudo/AluizioAzevedo/casapensao.htm>
[8] Disponível em <http://www.biblio.com.br/defaultz.asp?link=http://www.biblio.com.br/conteudo/alvaresazevedo/noitenataverna.htm>

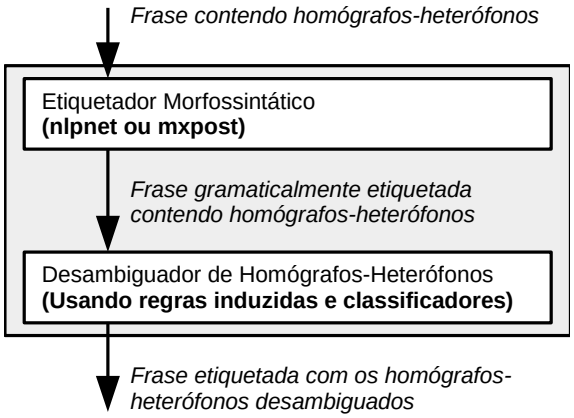

Figura 3 – Abstração do processo de desambiguação de HH. Em destaque as estratégias adotadas neste trabalho.

Fonte: Adaptado de Ribeiro, Oliveira e Trancoso (2003).

Sendo uma importante tarefa para o PLN, a etiquetagem morfossintática é um passo inicial para muitas aplicações, e para este trabalho de desambiguação de HH, adotou-se o nlpnet (FONSECA; ROSA, 2013), etiquetador disponibilizado de forma livre[9] inspirado no trabalho de Collobert et al. (2011) e baseado em redes neurais perceptron multicamadas e espaço vetorial. A Tabela 1 lista as etiquetas das classes gramaticais e a sua descrição, estas são atribuídas às palavras na frase após a mesma ser processada pelo etiquetador nlpnet.

Também fez-se uso de um outro etiquetador chamado de MXPOST, este é baseado na técnica de máxima entropia e foi inicialmente disponibilizado para a língua Inglesa, sendo adaptado para o PB por Aires et al. (2000), tendo o *corpus* Mac-Morpho como sua base de treino. Este etiquetador foi empregado na fase inicial de desenvolvimento do trabalho de desambiguação de HH, e alguns dos resultados encontra-se na Seção 4.1.

[9] Disponível em <https://github.com/erickrf/nlpnet>

Tabela 1 – Descrição das etiquetas gramaticais atribuídos pelos etiquetadores nlpnet e MXPOST.

Etiqueta	**Classe Gramatical**
ADJ	Adjetivo
ADV	Advérbio
ADV-KS	Advérbio conectivo
ADV-KS-REL	Advérbio relativo subordinativo
ART	Artigo (def. ou indef.)
CUR	Símbolo de moeda corrente
IN	Interjeição
KC	Conjunção coordenativa
KS	Conjunção subordinativa
N	Nome
NPROP	Nome próprio
NUM	Numeral
PCP	Particípio
PDEN	Palavra denotativa
PREP	Preposição
PREP+ADV	Contração preposição com advérbio
PREP+ART	Contração preposição com artigo
PREP+PROADJ	Contração preposição com pronome adjetivo
PREP+PRO-KS	Contração preposição com pronome conectivo Subordinativo
PREP+PRO-KS-REL	Contração preposição com pronome relativo conectivo
PREP+PROPESS	Contração preposição com pronome pessoal
PREP+PROSUB	Contração preposição com pronome substantivo
PROADJ	Pronome adjetivo
PRO-KS	Pronome conectivo Subordinativo
PRO-KS-REL	Pronome relativo conectivo Subordinativo
PROPESS	Pronome pessoal
PROSUB	Pronome substantivo
PU	Sinal de pontuação
V	Verbo
VAUX	Verbo auxiliar

Fonte: Adaptado de Fonseca e Rosa (2013).

3.5 Classificadores

A classificação é uma tarefa onde um modelo é construído para prever uma categoria, como, por exemplo, “sim” ou “não”, “aberta” ou “fechada”. Para que os algoritmos de classificação funcionem, é necessário separar o processo em duas partes: treino, onde o algoritmo analisa os dados de treinamento; e a classificação propriamente dita, onde dados de teste são usados para estimar a acurácia dos algoritmos (HAN; KAMBER; PEI, 2011, p. 328). Dessa forma, a desambiguação de palavras pode ser facilmente formulada como um problema de classificação supervisionada, ou seja, conhecimento extraído a partir de textos.

Os classificadores usados neste trabalho para desambiguação de HH foram selecionados explorando a biblioteca de implementações do ambiente WEKA (HALL et al.,

2009). Assim, os classificadores selecionados e os motivos foram:

(i) AdaBoostM1: Implementa o *boosting*, técnica que produz melhoramentos e redução de erros em cima de um algoritmo de aprendizagem "fraca", este que em princípio precisa ser apenas um pouco melhor que a adivinhação (FREUND; SCHAPIRE, 1996). Neste trabalho, empregou-se o AdaBoostM1 com o algoritmo J48, este último considerado um algoritmo "forte";

(ii) AODE: Tem natureza estatística, objetiva ser uma alternativa ao algoritmo tradicional NaiveBayes, procurando melhorar algumas caracteríticas ao relaxar as suposições de independência (BIELZA; LARRAÑAGA, 2014; WEBB; BOUGHTON; WANG, 2005);

(iii) NaiveBayes: Simples e de natureza estatística, é um algoritmo clássico para resolver ambiguidade em outras línguas. Utiliza o teorema de Bayes e pressupõe independência de atributos (BIELZA; LARRAÑAGA, 2014);

(iv) J48: É uma implementação do algoritmo C4.5 para construção de árvore de decisão no qual cada nó interno contém um teste. Os testes são avaliados por fórmulas baseado na teoria da informação (WITTEN; FRANK, 2005, p. 198);

(v) KStar: Algoritmo de aprendizado por instância, isto é, permite classificar por meio de uma base de exemplos pré-classificados, assume-se que instâncias semelhantes terão classificações semelhantes, e o grau de semelhança é calculado por uma função de distância baseada na entropia (CLEARY; TRIGG, 1995);

(vi) RandomForest: Utiliza várias árvores de decisão elegendo a resposta por voto majoritário e ameniza o problema de *overfitting* durante o treino (WITTEN; FRANK, 2005, p. 407) (BREIMAN, 2001).

O algoritmo AdaBoostM1 foi empregado devido às suas propriedades combinatórias que permitem superar as fraquezas de uma única abordagem supervisionada. Esse algoritmo tem sido aplicado em diversos problemas práticos com sucesso e a sua alta acurácia é comparável ao desempenho do algoritmo *Support Vector Machine* (SVM), que possui um dos melhores resultados em WSD, segundo Màrquez et al. (2006). Não foi selecinado o algoritmo SVM para este trabalho de desambiguação de HH devido que a implementação disponível no ambiente WEKA mostrou-se problemático durante o treino com os modelos de dados preparados, inviabilizando a escolha da mesma.

Os algoritmos AODE e NaiveBayes são classificadores de redes Bayesianas. Eles apresentam algumas vantagens, como acomodação para dados e cenário complexos, e também são usados com frequência em soluções bem sucedidas para problemas do mundo

real (BIELZA; LARRAÑAGA, 2014). Neste trabalho, comprovou-se que o algoritmo AODE possui bom desempenho de desambiguação, porém seu custo é elevado e apresenta limitação quanto ao tamanho máximo para treino na prática.

O algoritmo J48 gera árvores de decisões que na prática são consideradas um excelente método para descoberta de conhecimento e mineração de dados (BHARGAVA et al., 2013). Árvores pequenas são mais fáceis interpretar e trabalham bem com atributos redundantes (NORDMAN, 2016).

O KStar pertence ao grupo de algoritmos supervisionados que a medida que novos exemplos são classificados, estes são progressivamente adicionados ao modelo residente em memória. Outro algoritmo que pertence a esse grupo é o *k-Nearest Neighbor* (KNN), que, por sua vez, está entre os que possuem o mais alto desempenho em WSD (HUSAIN; BEG, 2013). Preferiu-se utilizar o KStar em vez do KNN devido a que nos testes preliminares de exploração de classificadores na interface do WEKA observou-se resultados iniciais mais promissores.

O algoritmo RandomForest, que tem desempenho comparável ao Adaboost, é um tipo de agrupamento de árvores preditoras geradas por seleção aleatória de atributos. Ele pode gerar estimativas internas de erro, robustez, correlação e importância de variável, além de se beneficiar do processamento em paralelo com relativa facilidade (BREIMAN, 2001).

Entre os algoritmos de aprendizado de máquina supervisionado, as redes neurais também mostram ter um dos melhores desempenhos, porém estes experimentos que foram conduzidos por outros autores com estes algoritmos foram em um número reduzido de palavras e ainda existe a dificuldade de obter grande quantidade de dados, sendo necessário otimizar os parâmetros como limiares, decaimentos, etc (HUSAIN; BEG, 2013). Para o presente trabalho, não foi incluso um algoritmo baseado em rede neural devido à dificuldade experimentado durante o treino no ambiente WEKA, o que inviabilizou a escolha da mesma.

3.5.1 Atributos bases para classificadores

O formato de arquivo ARFF para processamento pelo WEKA define a estruturação concisa do dados de entrada, a partir do qual as operações como seleção de atributos, classificação e indução de regras podem ser executados. A natureza dos atributos a serem processados pelos classificadores foi adotado de um modelo para vetor de atributos inspirado no trabalho de Màrquez (2000) definidos para contextos locais (NAVIGLI, 2009), ou seja, são os atributos de um número pequeno de palavras que estão em torno do HH de interesse, podem consistir em etiquetas gramaticais, palavras e formas derivadas, posições em relação ao HH alvo, etc.:

$$p_{-3},\ p_{-2},\ p_{-1},\ p_{+1},\ p_{+2},\ p_{+3},\ w_{-1},\ w_{+1},\ (w_{-2},\ w_{-1}),\ (w_{-1},\ w_{+1}),\ (w_{+1},\ w_{+2}),$$

$$(w_{-3},\ w_{-2},\ w_{-1}),\ (w_{-2},\ w_{-1},\ w_{+1}),\ (w_{-1},\ w_{+1},\ w_{+2}),\ (w_{+1},\ w_{+2},\ w_{+3})$$

Figura 4 – Representação dos atributos base que, após seleção, serão usado pelos classificadores.

onde $w_{\pm 3}$ é o contexto de palavras consecutivas ao redor da palavra w a ser desambiguada, e $p_{\pm 3}$ é a etiqueta fornecida pelo etiquetador morfossintático para a palavra $w_{\pm 3}$. Desse modo, a corretitude destes atributos é dependente do desempenho do etiquetador adotado.

Ao todo são 15 atributos. Os arquivos ARFFs foram gerados automaticamente através de um *script* elaborado na linguagem Python[10]. A justificativa de usar atributos contendo palavras vizinhas e outros elementos associados, conceitualmente denominados de N-gramas, deve-se ao seu uso predominante em aplicações PLN modernas, e representa o modelo espacial de vetores em que são aplicados os classificadores padrões (SIDOROV et al., 2014).

Dois exemplos (frase e vetor de atributos) são apresentados abaixo com base na definição do vetor dada anteriormente.

i) "O governador eleito almoçou uma **colher** de arroz e 40 gramas de carne":

```
1 ADJ,VERB,ART,PREP,N,CONJ,uma,de,almoçou_uma,uma_de,de_arroz,eleito_almoç
    ou_uma,almoçou_uma_de,uma_de_arroz,de_arroz_e,1
```

ii) "Os fiscais vão **colher** amostras para análise":

```
1 ART,N,VERB,N,PREP,N,vão,amostras,fiscais_vão,vão_amostras,amostras_para,
    Os_fiscais_vão,fiscais_vão_amostras,vão_amostras_para,amostras_para_aná
    lise,0
```

As definições dos atributos entre parênteses como $(w_{+1},\ w_{+2},\ w_{+3})$ são colocações de palavras consecutivas, o caractere de separação utilizado no ARFF foi o __ (*underscore*), conforme os exemplos. Além dos atributos, o vetor contém um campo binário, chamado classe, que marca a tonicidade da vogal diferencial do HH presente na frase ("1" para aberta e "0" para fechada). Essa marcação foi realizada manualmente com auxílio de uma interface *Web*, descrito na Seção 3.8.

Uma observação a ser feita sobre estas definições de vetor de atributos (para regras e classificadores) é de que cada atributo presente nas instâncias de treino devem ser declaradas no cabeçalho do ARFF para ser válido. Isto significa que, para classificar

[10] Documentações sobre o Python poderá ser consultado em: <http://www.python.org>

instâncias (frases) desconhecidas, deve-se utilizar os mesmos cabeçalhos de treino, e os atributos gerados pela frase desconhecida que não esteja contido no cabeçalho de treino do HH considerado não são levados em conta e são descartados. Uma estratégia adotada para mitigar o descarte de atributos foi a substituição, ou seja, substituir a palavra de uma classe gramatical por outra conhecida da mesma classe, caso esta nova palavra não exista no cabeçalho de treino.

3.6 Algoritmos de indução de regras

A indução de regras é uma técnica de classificação e uma subárea do aprendizado de máquina. Consiste em gerar conjuntos de testes determinísticos que são de fácil compreensão e entendimento, e tem correspondência de como as pessoas expressam conhecimentos (REZENDE, 2005). A indução de regras requer exemplos, tem desvantagem de sofrer mais *overfitting* em relação à outros algoritmos de classificação e a velocidade é menor do que as árvores de decisão (DOMINGOS, 2016).

A possibilidade de poder analisar as regras geradas para desambiguar HH é um atrativo apesar de sofrer algumas desvantagens. A seleção de algoritmos de indução de regras seguiram os mesmos critérios de seleção dos classificadores ou seja, por exploração dos algoritmos candidatos viáveis no ambiente visual WEKA e inspecionando resultados preliminares. São listados três algoritmos de indução de regras a seguir:

(i) FURIA: O *Unordered Fuzzy Rule Induction* é um método de classificação baseado em regras *fuzzy*, este aprende um conjunto de regras para cada classe e não possui uma regra padrão (HÜHN; HÜLLERMEIER, 2009);

(ii) Ridor: O algoritmo de aprendizado de regras *RIpple-DOwn* gera primeiramente a regra padrão, a partir daí gera-se sucessivas exceções que são um conjunto de regras que prediz a classe além do padrão (HALL et al., 2009);

(iii) JRip: Implementação disponível no WEKA para o algoritmo *Repeated Incremental Pruning to Produce Error Reduction* (RIPPER), para aprendizado de regras proposicionais eficientes em *datasets* ruidosos grandes (COHEN, 1995).

Estes algoritmos são capazes de emitir a representação textual das regras e o processo de produção é apresentada na Seção 4.2. Todas as regras produzidas estão no Apêndice B.

3.6.1 Atributos bases para indução de regras

Os arquivos ARFF necessários para induzir as regras de decisão foram derivados da definição do vetor de atributos a seguir:

$$w_{-3},\ w_{-2},\ w_{-1},\ w_{+1},\ w_{+2},\ w_{+3},\ p_{-3},\ p_{-2},\ p_{-1},\ p_{+1},\ p_{+2},\ p_{+3},$$

$$pre_{-3},\ pre_{-2},\ pre_{-1},\ pre_{+1},\ pre_{+2},\ pre_{+3},\ suf_{-3},\ suf_{-2},\ suf_{-1},\ suf_{+1},\ suf_{+2},\ suf_{+3}$$

Figura 5 – Representação dos atributos base que, após seleção, serão usado pelos algoritmos de indução de regras.

Na definição dada pela Figura 5, o símbolo $w_{\pm 3}$ representa as palavras vizinhas ao HH com distância até 3 para à esquerda ou direita do mesmo. O símbolo $p_{\pm 3}$ representa a etiqueta da classe gramatical atribuída pelo etiquetador nas palavras vizinhas com distância até 3 para à esquerda ou direita ao HH. Os símbolos $pre_{\pm 3}$ e $suf_{\pm 3}$ representam os prefixos e sufixos das palavras vizinhas ao HH com distâncias até 3 para à esquerda ou direita ao HH. Os prefixos e sufixos formadores de palavras do PB reconhecidos foram consultados na obra de Cunha e Cintra (2013). Como nem todas as palavras do PB tem sufixos e prefixos, neste caso, os atributos estarão ausentes.

A partir da definição simbólica dada acima, e para cada frase de entrada, as instâncias de treino ou avaliação podem ser geradas como nos exemplos a seguir, com duas frases contendo o HH **boto**:

i) "Uma subespécie de **boto** habita em exclusivo o sistema fluvial do Madeira":

```
1 uma,subespécie,de,habita,em,exclusivo,ART,N,PREP,V,PREP,ADJ,?,sub-,?,?,e-,
      ex-,-a,?,?,-ita,-m,-ivo,0
```

ii) "Todo dia eu **acordo** e **boto** uma cuca nova":

```
1 #Instância de atributos para o HH acordo:
2 todo,dia,eu,e,boto,uma,PROADJ,N,PROPESS,KC,V,ART,?,di-,e-,?,?,?,-o,-ia,-u
      ,?,-o,-a,1
3
4 #Instância de atributos para o HH boto:
5 eu,acordo,e,uma,cuca,nova,PROPESS,V,KC,ART,N,ADJ,e-,a-,?,?,?,?,-u,-o,?,-a,-
      a,-a,1
```

As instâncias apresentadas acima foram utilizadas para formação da base de treino a partir de onde os algoritmos de indução de regras obteve os exemplos de uso. Note que alguns atributos são inexistentes, e no ARFF é representado com o caractere **?** (interrogação).

3.7 Métricas de avaliação

Para possibilitar a avaliação do desempenho dos classificadores e das regras induzidas com suas diferentes configurações, utiliza-se testes estatísticos, ou adota-se medidas de

desempenho como indicadores, para fins de diferenciação. A métrica simples usada para medida de desempenho, a acurácia (ACC), sofre de várias deficiências, entre elas, a de possuir viés para a classe mais predominante e apresentar valores idênticos para arranjos diferentes na tabela de contingência (JURMAN; RICCADONNA; FURLANELLO, 2012). Por exemplo, mesmo que um dado valor para a acurácia resultante de um experimento seja bastante elevada, isto não quer dizer que o desempenho para todas as classes também sejam elevadas, o mais provável é que a classe mais abundante esteja dominando o resultado quando o classificador responde por padrão também a classe mais abundante, em detrimento das classes em menor número. Devido a isto, uma métrica adicional foi selecionada.

A tabela da Figura 6 contém os valores proporcionais tp (*true positive* – positivos verdadeiros), fp (*false positive* – falsos positivos), fn (*false negative* – falsos negativos) e tn (*true negative* – negativos verdadeiros). Os valores correspondentes A, B, C e D contém as seguintes quantidades:

A – Quantidade de classificação das pronúncias verdadeiramente abertas (tp);

B – Quantidade de classificação como pronúncias abertas porém incorretas (fp);

C – Quantidade de classificação como pronúncias fechadas porém incorretas (fn);

D – Quantidade de classificação das pronúncias verdadeiramente fechadas (tn).

Figura 6 – Uma tabela ou matriz de contingência binária. Cada previsão individual pode recair em uma das quatro possibilidades.

	+R	**−R**	
+P	**A (tp)**	**B (fp)**	**pp**
−P	**C (fn)**	**D (tn)**	**pn**
	rp	**rn**	**1**

Fonte: Adaptado de Powers (2011, p. 2).

A acurácia é simples de ser calculada a partir dos valores da tabela de contingência conforme a Equação 3.1.

$$ACC = \frac{A + D}{A + B + C + D} \tag{3.1}$$

Assim como a acurácia, o coeficiente de correlação de Matthews (MCC, do inglês Matthews *correlation coefficient*) (POWERS, 2011) é calculado a partir da tabela

de contingência pela Equação 3.2. Este coeficiente tem propriedades de simetria e seu emprego é adequado em situações de classes desbalanceadas, fenômeno onde existe grande predominância de algumas classes e poucas de outras.

$$MCC = \frac{A \times D - B \times C}{\sqrt{(A+C)(B+D)(A+B)(C+D)}} \tag{3.2}$$

O valor do MCC compreende o intervalo $[-1, 1]$, e quando o valor for igual a 1, o resultado da classificação na matriz de contingência e a correlação é perfeita (POWERS, 2011), quando o valor for -1, indica que o resultado da classificação na matriz é errônea ao extremo, e o MCC será 0 quando todas as amostras são classificadas em apenas uma classe ou então todos os valores na matriz de contingência são iguais. Em geral, a métrica MCC tem equilíbrio entre diversos tipos de comportamento que surgem na classificação (JURMAN; RICCADONNA; FURLANELLO, 2012).

Estas duas métricas foram utilizadas em conjunto durante a produção dos modelos pré-treinados (Seção 4.2), pois existe a necessidade de comparação das diferentes configurações de parâmetros dos algoritmos de classificação e indução, e também da necessidade da comparação entre algoritmos diferentes que indique qual o mais adequado para cada HH. Portanto, adotou-se a estratégia de que um algoritmo é superior a um outro ou se os parâmetros de um algoritmo são superiores a um outro conjunto de parâmetros, caso a avaliação 10–*fold cross-validation* (10–CV) resultasse no valor da acurácia maior ou igual sem que a métrica MCC degradasse, compensando-se assim o problema do viés da acurácia para a classe mais frequente.

A estratégia *k-fold cross-validation* é popularmente empregada para a seleção de modelos (algoritmos) (ARLOT; CELISSE, 2010), a ideia é dividir o conjunto de dados D em k subconjuntos mutualmente exclusivos de tamanhos iguais. O algoritmo é treinado k vezes, cada treino $t \in \{1, 2, ..., k\}$ é feito em $D \backslash D_t$ e testado em D_t. Métrica como a acurácia estimada pela CV pode ser obtida dividindo as classificações corretas pelo número de instâncias no conjunto de dados. Tipicamente utiliza-se $k = 10$ (KOHAVI, 1995).

3.8 Ferramentas auxiliares

Um dos maiores desafios que a abordagem supervisionada apresenta é a necessidade do especialista prover quantidade suficiente de amostras ao sistema de aprendizado de máquina, uma situação conhecida como "gargalo da aquisição de conhecimento" (GALE; CHURCH; YAROWSKY, *apud* NAVIGLI, 2009). Neste trabalho, essa dificuldade se traduz em ter que desambiguar manualmente um número elevado de HH nas frases para compor a base de treino do desambiguador. Com o intuito de flexibilizar a leitura e

anotação das frases de treino, elaborou-se uma ferramenta a fim de auxiliar esta tarefa manual.

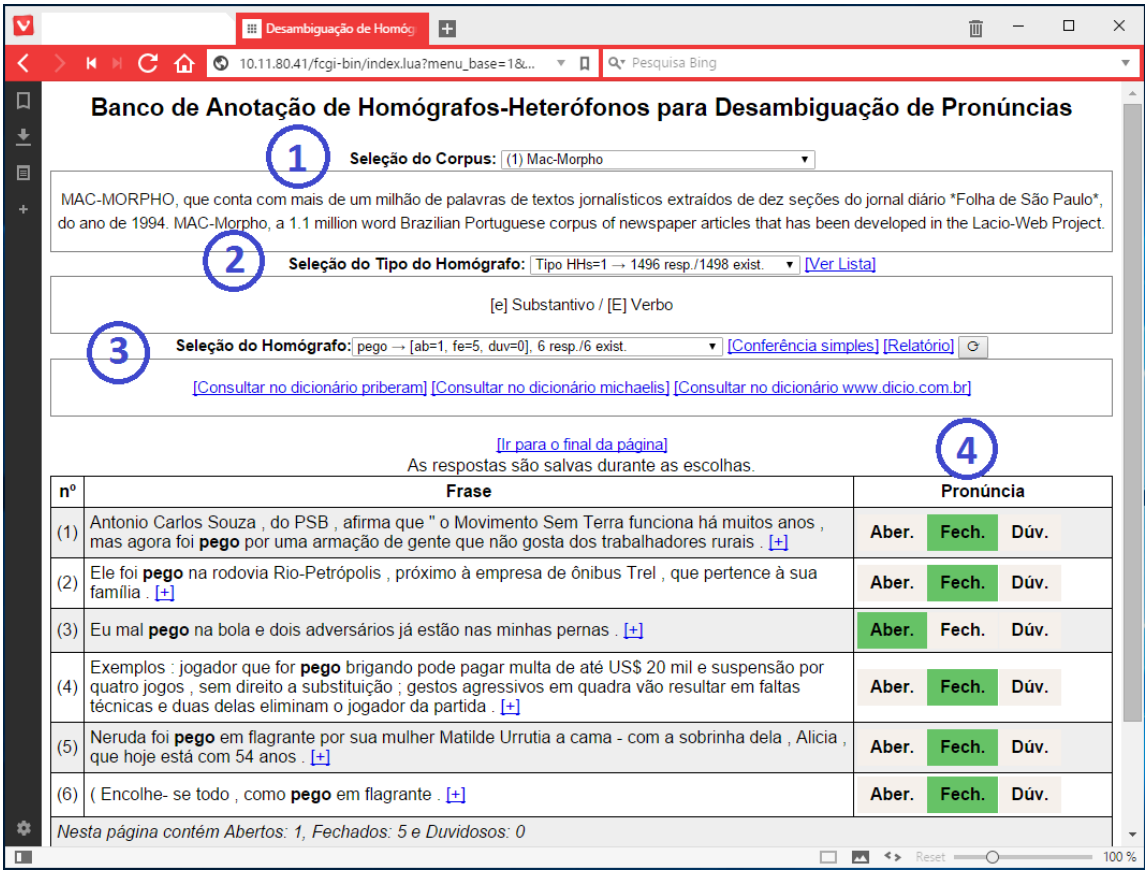

Figura 7 – Interface auxiliar para anotação manual dos HH no *corpus* de treino.

A motivação para produzir esta ferramenta, além da necessidade, é a facilidade de escalonamento para marcação por uma equipe de pessoas colaboradoras e também da importância de realizar revisões nestas marcações. Uma captura de tela é apresentada na Figura 7. Esta ferramenta possui interface *Web* de acesso local ou remoto no endereço <http://homografos.ddns.net:81/fcgi-bin-r/index.lua>, e empregou-se a linguagem de programação Lua[11] para construção. Os dados do *corpus* de treino, assim como as informações de marcação, foram armazenados em um banco de dados embarcado SQLite[12] para processamento posterior.

Por meio desta interface visual, o especialista terá mais conforto em concentrar na leitura da frase nos *corpus* e realizar o julgamento e a marcação da pronúncia do HH. A ferramenta permite a seleção do *corpus* por meio do menu (1); Permite a seleção de categorias de HH (SILVA; BRAGA; RESENDE JR, 2012) a ser trabalhada (2); As frases serão selecionadas de acordo com a escolha específica do HH contido na mesma (3); O

[11] Mais informação sobre a linguagem Lua em: <http://www.lua.org>
[12] Documentação da biblioteca SQLite disponível em: <http://www.sqlite.org>

julgamento da pronúncia do HH (em negrito) na frase será registrada ao clicar o botão correspondente (4).

O *corpus* para construção da base de treino e as marcações manuais das pronúncias dos HH nas frases são armazenadas no banco de dados (Figura 8) que posteriormente poderão ser extraídos via comandos SQL em *scripts* para realização de experimentos de aprendizado de máquina supervisionada e na efetiva construção do desambiguador de HH.

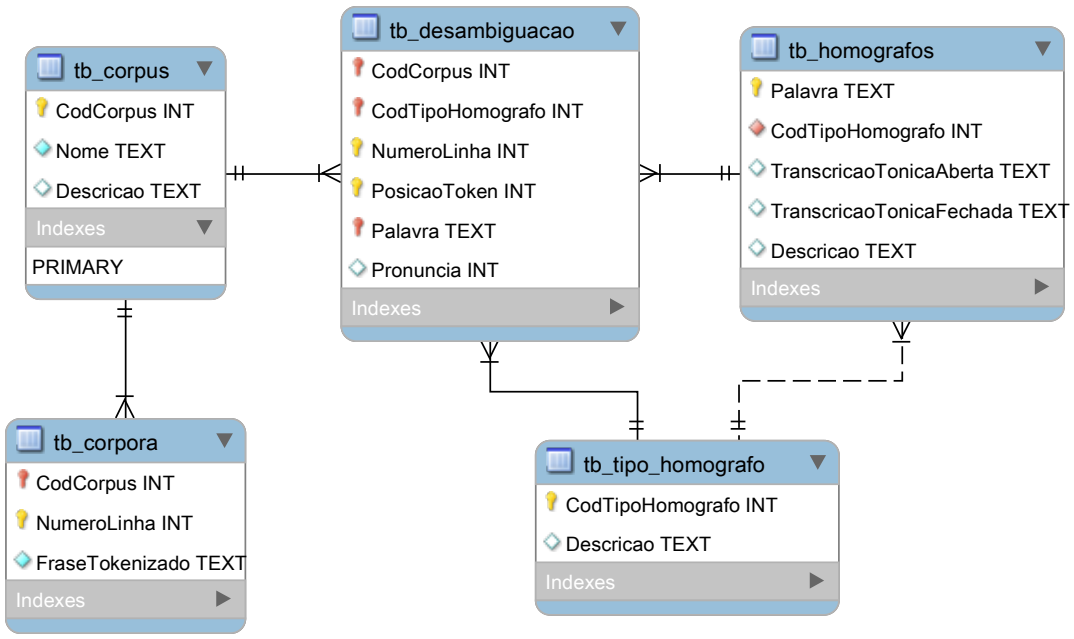

Figura 8 – Representação visual do esquema do banco de dados utilizado para armazenar as marcações das pronúncias realizadas pelo especialista.

Com intuito de prover uma interface mais acessível ao desambiguador de HH, uma ferramenta de exploração simples foi elaborada para observar o comportamento do desambiguador. Aqui é possível inserir frases arbitrárias em PB na região de digitação (1) e estas submetidas ao desambiguador de HH ao clicar o botão (2), conforme a captura de tela da Figura 9, em seguida o desambiguador tentará identificar e inferir as pronúncias dos HH encontrados (1), exibindo também o vetor de atributos (2) submetido ao seu algoritmo correspondente (3), assim como os parâmetros ótimos utilizados (4), conforme a captura de tela da Figura 10.

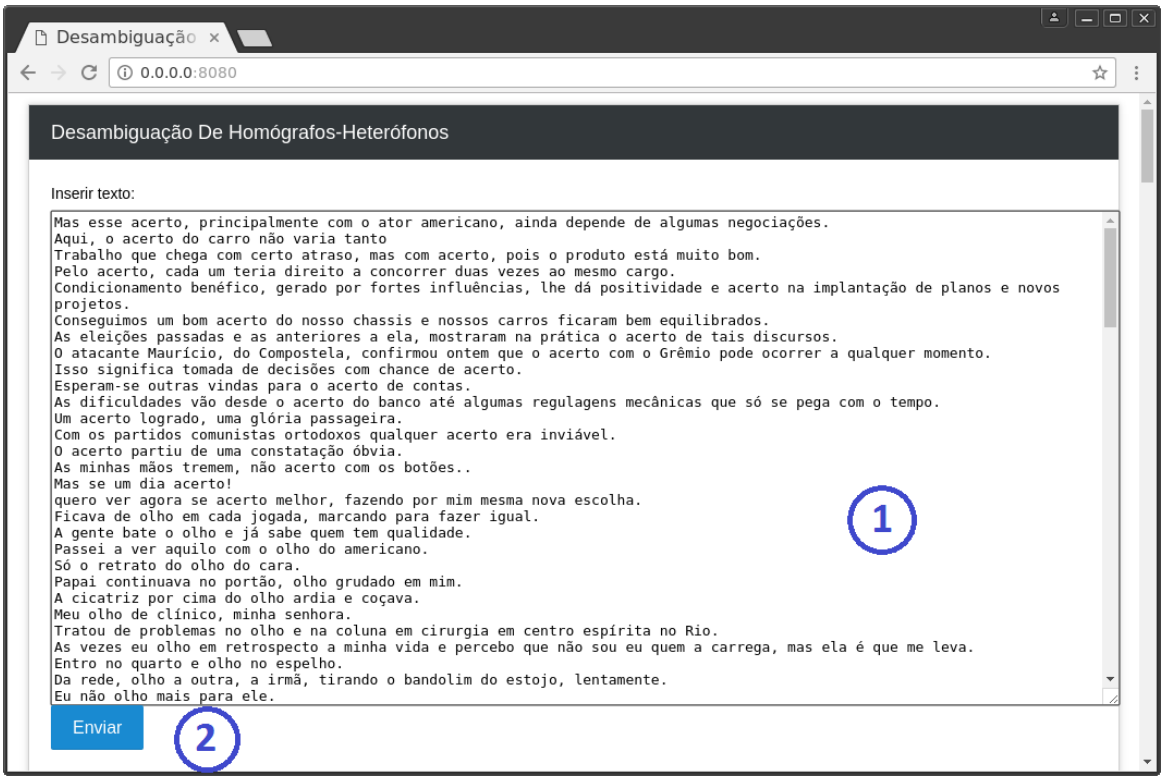

Figura 9 – Interface de entrada de frases arbitrárias para o desambiguador de HH desenvolvido.

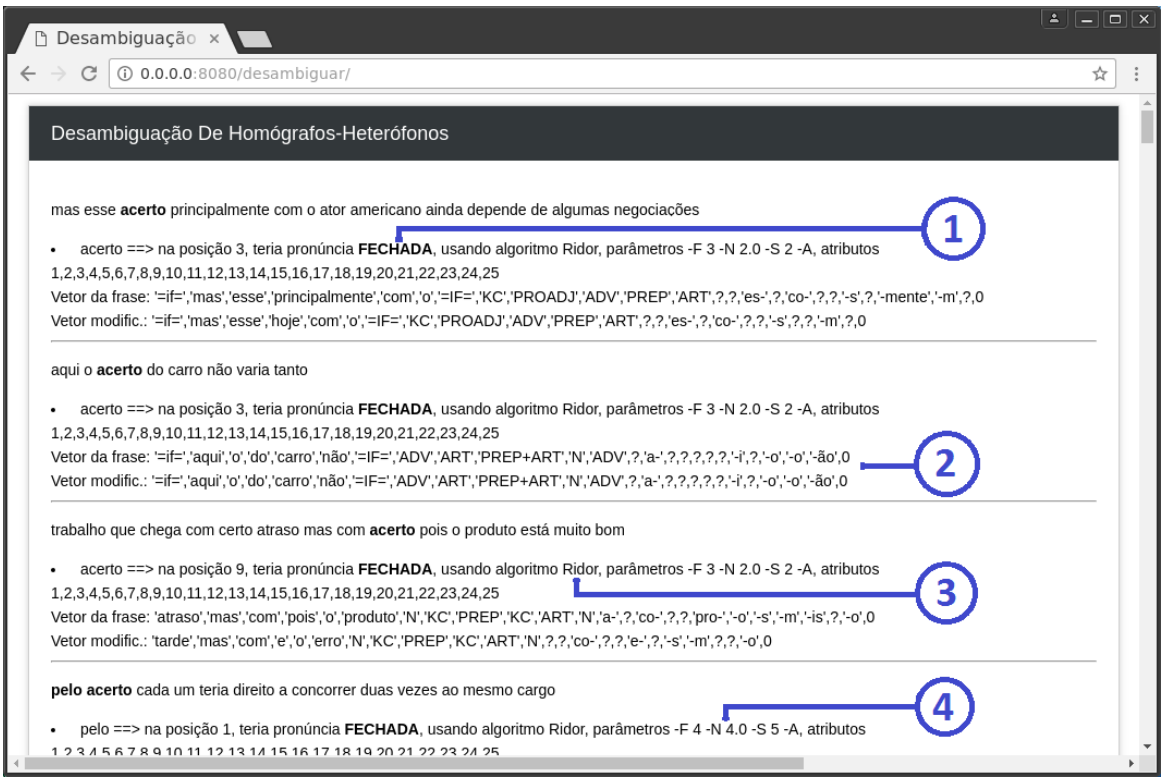

Figura 10 – Interface de saída mostrando as pronúncias estimadas dos HH encontradas nas frases de entrada pelo desambiguador.

4 Análise dos resultados

A apresentação dos resultados gerados é feita em duas partes. A Seção 4.1 visa demonstrar que o domínio do texto, isto é, a natureza do conteúdo textual predominante na base de treino, que neste caso são jornalísticos ou literários, possuem características que afetam a desambiguação de HH. Os resultados apresentados a partir destes experimentos nesta seção foram publicados em forma de artigo (HAMADA; NETO, 2015) e apresentado no simpósio STIL 2015 em Natal–RN.

Já a Seção 4.2, apresenta-se o processo de construção do desambiguador propriamente dito, discutindo-se as diversas etapas intermediárias e justificando estes processos, culminando com uma análise de desempenho de desambiguação em um *corpus* de avaliação que não foi utilizada na base de treino.

4.1 Experimentos de influência do domínio do texto

O desempenho do desambiguador é influenciado quando este é treinado em um domínio e então avaliado em outro. Para demonstrar esta característica, experimentos combinando algoritmos de classificação e base de treino–teste foram realizados. Não foi usado o etiquetador nlpnet, mas sim o MXPOST para reproduzir os experimentos nesta seção, pois aqui reflete o resultado do trabalho publicado em Hamada e Neto (2015).

4.1.1 *Corpus* de treino usados nos experimentos

Formulou-se dois *corpora*: um com textos jornalísticos (*corpus* A) e outro com textos literários (*corpus* B). A autoria dos textos que compõem os mesmos se encontra na Seção 3.3. A partir dessas bases de dados, localizou-se as frases que continham HH existentes no português falado no Brasil para formar os conjuntos de treino e avaliação. Observou-se, no entanto, que os pares de pronúncias ocorrem muitas vezes de forma desbalanceada.

***Corpus* A:** *Corpus* Mac-Morpho revisado[1], versão 1; *corpus* CETEN-Folha; Texto da Constituição; 25% dos artigos em português da enciclopédia Wikipédia e *Corpus* LAPSNEWS;

***Corpus* B:** Aluísio Azevedo; Álvares de Azevedo; José de Alencar; Machado de Assis; Olavo Bilac; Castro Alves; e Euclides da Cunha. Também utilizou-se o *corpus* eletrônico anotado Tycho Brahe.

[1] Disponível em <http://nilc.icmc.usp.br/macmorpho>

Normalmente, cada HH é tratado como um problema de classificação diferente. Portanto, a coletânea de um *corpus* representativo deve considerar as particularidades de cada palavra, a fim de decidir o número de exemplos necessários para aprendizagem, os atributos mais úteis, e assim por diante. Assim, optou-se pelos HH que apresentaram as menores diferenças entre suas ocorrências aberta e fechada, conforme a Tabela 2.

Tabela 2 – Distribuição dos HH selecionados a partir dos *corpus* A e B.

Palavra	*Corpus* A			*Corpus* B		
	Abertas	Fechadas	Ocorrências	Abertas	Fechadas	Ocorrências
“colher”	116	331	447	17	95	112
“começo”	40	436	476	34	113	147
“corte”	109	104	213	5	183	188
“fora”	675	69	744	208	77	285
“gosto”	140	108	248	86	520	606
“rola”	116	9	125	25	37	62
“sede”	604	21	625	16	89	105
Total	1800	1078	2878	391	1114	1505

Em seguida, o anotador morfossintático MXPOST foi usado para etiquetar as frases escolhidas. Não foi usado o nlpnet, que é um etiquetador mais recente, pois esta seção reflete o resultado do trabalho publicado em Hamada e Neto (2015).

4.1.2 Vetor de atributos usados nos experimentos

A definição do vetor de atributos usados para classificadores foi definida na Subseção 3.5.1. Após a etiquetagem, o vetor de atributos de cada frase (conteúdo do arquivo de entrada do WEKA) foi produzido a partir de um *script* elaborado em Python para este propósito, de acordo com o modelo do vetor de atributo, levando em conta a vogal diferencial do HH classificada manualmente como aberta ou fechada.

4.1.3 Classificadores usados nos experimentos

Um conjunto de quatro classificadores representativos e detalhados na Seção 3.5 foi utilizado neste experimento. São eles (os parâmetros padrões estão entre parênteses): NaiveBayes, AODE (`-F 10`), J48 (`-C 0.25 -M 2`) e RandomForest (`-I 10 -K 0 -S 1`). A versão do WEKA utilizado nos experimentos desta seção foi 3.6.11. Não foi incluso os outros algoritmos selecionados para o trabalho, já que este experimento foi um dos primeiros a ser realizado durante a pesquisa e que reflete o mesmo conjunto de algoritmos publicado no artigo (HAMADA; NETO, 2015).

4.1.4 Métodos usados nos experimentos

A comparação entre os algoritmos foi realizada através de uma série de experimentos controlados usando exatamente os mesmos conjuntos de treino e teste. Também visando avaliar a dependência da desambiguação de HH com relação ao domínio da aplicação, foram elaboradas sete combinações possíveis para os conjuntos treino-teste. Por exemplo, a notação A–B indica que o algoritmo foi treinado com o *corpus* A e avaliado com o *corpus* B, assim como a notação A+B–B diz que formou-se a base de treino com a união dos *corpora* A e B, e a base de teste apenas com *corpus* B. Utilizou-se o teste 10-CV, exceto nos casos A–B e B–A, já que os mesmos tem o conjunto de dados para treino e teste naturalmente disjuntos.

4.1.5 Resultados dos experimentos

A Tabela 3 apresenta a média de acertos dos quatro algoritmos para todas as combinações dos conjuntos treino-teste. O ZeroR representa um simples classificador de referência do WEKA em que a pronúncia mais frequente no conjunto de treino é usada para classificar todos os exemplos presentes no conjunto de teste, servindo dessa forma como um valor de desempenho base (ou *baseline*). O melhor resultado para cada caso é destacado em negrito.

Tabela 3 – Acurácia dos classificadores para todas as combinações de treino-teste.

Classificador	Acurácia média (%)						
	A+B–A+B	A+B–A	A+B–B	A–A	B–B	A–B	B–A
ZeroR	79,37	76,88	67,63	79,06	80,32	43,87	51,33
AODE	**93,61**	**94,38**	**88,10**	**95,55**	**89,75**	70,70	75,50
J48	91,02	92,93	78,47	94,40	82,34	67,32	63,52
NaiveBayes	88,82	90,26	85,64	87,12	79,03	**81,80**	**82,34**
RandomForest	86,52	86,38	75,60	90,04	84,49	66,81	60,17

Observou-se que o AODE superou os outros algoritmos, exceto nas combinações A–B e B–A, onde prevaleceu o algoritmo Naive Bayes simples. Nesses casos em específico, os conjuntos de treino e teste são totalmente disjuntos, já que a ideia é exatamente avaliar a portabilidade de textos de diferentes domínios: literário e jornalístico. Como já era esperado, os resultados obtidos nessas combinações foram inferiores em comparação às demais. Restringindo a análise aos resultados do classificador AODE, a queda foi de aproximadamente 25% e 20% para A–B e B–A, respectivamente.

Os resultados inferiores obtidos com relação a portabilidade de domínio (generalização) podem ser explicados, entre outros fatores, pela diferente distribuição de pronúncias entre os *corpora* A e B. Assim, este experimento foi repetido equilibrando artificialmente os exemplos de cada pronúncia entre as duas bases de texto, conforme a Tabela 4. Os

resultados são mostrados na Tabela 5. Percebe-se novamente queda de desempenho nas combinações A–B e B–A, ou seja, mesmo quando a mesma distribuição de pronúncias é conservada entre os exemplos de treino e teste, a portabilidade não é garantida. Esse fato mostra que os algoritmos adquirem diferentes (e não permutais) sugestões de classificação de ambas as fontes. Outro ponto relevante foi que o conjunto A+B–A (ou A+B–B) continuou com aproximadamente o mesmo desempenho do conjunto A–A (ou B–B) em todos os algoritmos. Isto é, o conhecimento obtido a partir de um único *corpus* quase abrange o conhecimento de combinar ambos os *corpora*.

Tabela 4 – Distribuição equilibrada dos HH selecionados a partir dos *corpus* A e B.

Palavra	*Corpus* A			*Corpus* B		
	Abertas	Fechadas	Ocorrências	Abertas	Fechadas	Ocorrências
“colher”	17	17	34	17	17	34
“começo”	34	34	68	34	34	68
“corte”	5	5	10	5	5	10
“fora”	69	69	138	69	69	138
“gosto”	86	86	172	86	86	172
“rola”	9	9	18	9	9	18
“sede”	16	16	32	16	16	32
Total	236	236	472	236	236	472

Tabela 5 – Experimento com a mesma distribuição dos HH entre os *corpora*.

Classificador	Acurácia média (%)						
	A+B–A+B	A+B–A	A+B–B	A–A	B–B	A–B	B–A
ZeroR	35,86	26,64	27,82	26,64	27,82	50,00	50,00
AODE	85,42	87,95	80,80	84,87	77,31	**82,72**	**81,86**
J48	75,00	76,09	74,44	81,48	61,71	71,47	70,56
NaiveBayes	**86,92**	**90,16**	**84,24**	**91,31**	**86,29**	77,44	80,56
RandomForest	69,23	72,63	59,00	69,52	62,87	77,65	66,11

O primeiro experimento mostrou que os classificadores treinados com o *corpus* A não funcionaram bem com o *corpus* B, e vice-versa. Então, esse segundo experimento explora o efeito do processo de incremento (ou *tuning*) na tentativa de tornar os sistemas supervisionados portáveis, em outras palavras, que o classificador consiga atingir desempenho satisfatório em múltiplos domínios. Esse processo consiste em adicionar ao conjunto de treino original uma quantidade relativamente pequena de amostras do novo domínio. O tamanho dessa porção supervisionada varia de 10% a 50%, em passos de 10%, sendo que os 50% restantes são reservados para os testes. Os conjuntos treino-teste desse experimento serão denotados por A+%B–B e B+%A–A.

Para analisar a real contribuição do conjunto de treino original na desambiguação de HH no novo domínio, os valores de acurácia para as combinações %B–B e %A–A

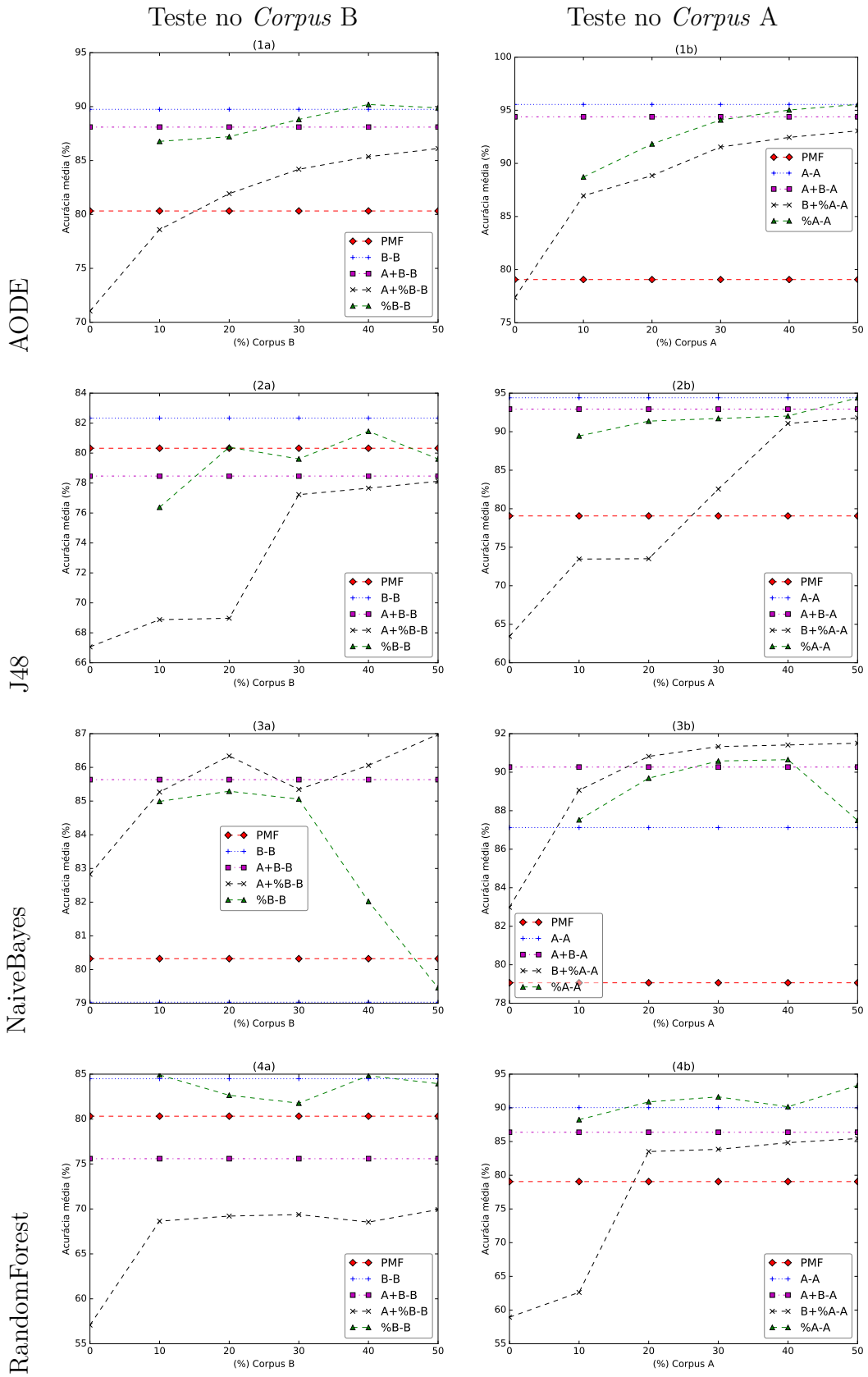

Figura 11 – Resultado do experimento de afinação do *corpus* de treino.

também foram calculados. Os resultados desse experimento são apresentados na Figura 11. Cada gráfico contém as curvas X+%Y–Y e %Y–Y, além de três linhas horizontais, que correspondem ao limite inferior, representado pelo rótulo PMF (pronúncia mais frequente dada pelo algoritmo ZeroR), e aos limites superiores X+Y–Y e Y–Y de cada algoritmo. Conforme esperado, a acurácia de todos os métodos aumenta (em direção ao limite superior) à medida que mais amostras do novo domínio são adicionadas ao conjunto de treino. Verificou-se também a degradação provocada pelo *corpus* de treino original na acurácia dos classificadores, com a curva %Y–Y superando a curva X+%Y–Y em praticamente todos os algoritmos. Resumindo, os gráficos mostraram que não é interessante manter os exemplos de treino originais. Ao contrário, uma melhor estratégia, embora desapontadora, é simplesmente usar o *corpus* incrementado.

4.2 Construção dos modelos pré-treinados

Para desenvolver a arquitetura apresentada na Figura 2, cada componente foi construído em ordem sequencial. A Figura 12 apresenta o fluxo de trabalho onde os componentes do sistema proposto são gerados. A etapa (1) consiste em anotar manualmente as pronúncias corretas dos HH no *corpus* de treino; em (2) são criados candidatos à vetores de atributos através do processo de seleção para cada HH, feito logo nesta etapa para possibilitar a exploração mais numerosa de opções nas etapas posteriores; em (3) são gerados os parâmetros candidatos para cada algoritmo em combinação com os vetores de atributos selecionados na etapa (2) anterior; em (4) para ampliação de base de treino de um dado HH, exemplos de outros HH são adicionados e testados no intuito de melhorar o desempenho da desambiguação do HH em questão; em (5) é realizada a avaliação para determinar qual é o algoritmo apropriado para cada HH (esta etapa é realizado simultaneamente com o passo anterior); e em (6) são gerados os modelos pré-treinados que podem ser aplicados na desambiguação de novas frases.

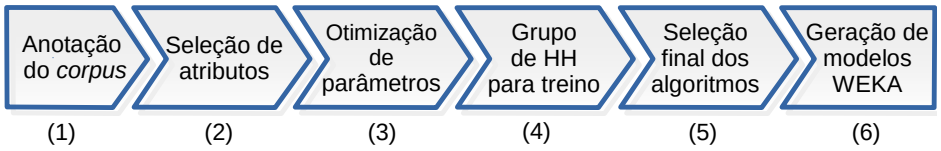

Figura 12 – *Workflow* das sucessivas etapas para construção dos componentes de desambiguação de HH.

Portanto, os objetivos desta seção são: (i) produzir um conjunto de regras simples de alta acurácia na desambiguação de um subconjunto de HH e (ii) construir uma base de treino eficiente para desambiguar HH através de classificadores. Essas técnicas serão estudadas por meio de bases de texto na língua portuguesa. Assim, busca-se aplicar o uso de técnicas de aprendizado de máquina e algoritmos de indução de regras na solução do problema de ambiguidade entre palavras.

A versão do WEKA utilizado para gerar os modelos foi a 3.7.12, e o etiquetador morfossintático adotado foi o nlpnet[2] versão 1.2.1. O restante desta seção discute os resultados gerados durante o processo de construção.

4.2.1 Anotação do *corpus*

A primeira etapa na construção do desambiguador é a anotação manual das pronúncias dos HH nas frases originárias da base de dados coletada para servir de modelo linguístico do PB, conforme apresentado na Seção 3.3. A Tabela 6 lista a quantidade de exemplos anotados em todo o *corpus* de treino.

[2] Disponível em <https://pypi.python.org/pypi/nlpnet/1.2.1>

Tabela 6 – Distribuição de 106 HH na base de treino final, totalizando 44.630 ocorrências anotadas; PA=Número de Pronúncias Abertas; PF=Número de Pronúncias Fechadas.

Palavra	PA	PF	Palavra	PA	PF	Palavra	PA	PF	Palavra	PA	PF
aborto	1	181	cor	16	434	interesse	16	453	rolha	0	42
acerto	177	687	coro	0	38	interesses	0	181	rolo	1	67
acordo	140	635	corte	74	203	jogo	26	473	rota	74	20
adorno	0	26	cortes	47	2	leste	227	46	rotas	84	19
almoço	27	302	desemprego	0	175	lobo	37	778	seca	15	251
apego	41	141	desespero	7	560	lobos	22	136	secas	0	213
apelo	31	1038	desgosto	1	47	medo	2	369	seco	4	223
aperto	36	300	desses	12	864	medos	12	26	sede	295	170
apoio	7	292	deste	107	1642	meta	1405	82	selo	1	664
apreço	0	138	destes	12	552	metas	59	2	sobre	1	2713
besta	4	196	emprego	1	247	modelo	1	266	soco	8	184
bestas	0	33	endosso	1	20	molho	5	332	sopro	0	86
bola	442	0	enredo	0	40	namoro	35	137	suborno	0	101
bolas	81	8	enterro	4	45	olho	234	2041	sufoco	0	55
boto	39	44	erro	17	604	pega	188	11	termos	0	1349
cerca	21	1122	esforço	0	137	pego	103	145	toco	48	17
cerro	0	67	esmero	0	24	pela	0	3404	tola	0	20
choro	37	148	este	1	1768	pelas	0	509	topo	4	53
colher	183	495	flagelo	1	114	pelo	0	2090	torno	23	202
colheres	47	0	fora	907	181	peso	5	312	torre	15	155
começo	99	307	forma	581	1	piloto	0	88	torres	0	89
concerto	2	515	formas	335	1	posto	17	74	travessa	81	47
conforto	1	163	força	4	503	reforço	0	46	troco	84	83
conserto	20	197	gelo	0	355	rego	0	105	vede	0	52
consolo	1	137	gosto	186	605	relevo	44	211	zelo	15	210
contorno	5	79	governo	0	1173	rogo	19	7			
controle	0	215	gozo	8	128	rola	250	68			

Nota-se que na Tabela 6, existem muitos HH que no *corpus* não possuem uma das pronúncias, ou seja, a pronúncia correspondente aberta ou fechada não foi localizada pelo especialista durante a etapa de anotação. Por exemplo, as palavras: “adorno”, “apreço”, “bestas”, “bola”, “cerro”, “colheres”, “controle”, “coro”, “desemprego”, “enredo”, “esforço”, “esmero”, “gelo”, “governo”, “interesses”, “pela”, “pelas”, “pelo”, “piloto”, “reforço”, “rego”, “rolha”, “secas”, “sopro”, “suborno”, “sufoco”, “termos”, “tola”, “torres” e “vede”, não teve a outra pronúncia encontrada no *corpus*. Isto se deve a que algumas destas simplesmente não está presente no *corpus*, de emprego infrequente e outras estão praticamente em desuso nos textos modernos em PB.

4.2.2 Seleção de atributos

O objetivo da seleção de atributos é prover um conjunto compacto e eficiente de atributos para o algoritmo de desambiguação, gerando um modelo de aproximação de acurácia relativamente alta. A seleção de atributos é um processo que seleciona um

subconjunto dos atributos relevantes a partir do conjunto de dados para a construção do modelo. Lembrando que o conceito fundamental do aprendizado de máquina é a aproximação de uma função $f()$ a partir da entrada $X = x_1, x_2, ..., x_m$ e uma saída Y baseado nos pontos de dados memorizados X_i, Y_i, onde $i = 1, ..., N$. O que se presume é que a saída Y nem sempre é determinada pelo conjunto completo de atributos de entrada $x_1, x_2, .., x_N$, ao invés disso, a saída é então baseada no subconjunto destes atributos $x_{(1)}, x_{(2)}, ..., x_{(n)}$ onde $n < N$ (NNAMOKO et al., 2014).

Como o objetivo é aproximar $f()$ entre X e Y, é razoável ignorar estes atributos que pouco influenciam na saída, e que para o presente trabalho é a pronúncia esperada. Os atributos resultantes oferecem o potencial de diminuir a confusão pelos classificadores e algoritmos indução de regras, descartando informações desnecessárias para prever a pronúncia dos HH a partir dos exemplos providos. Esta etapa intermediária gerou novas versões de vetores de atributos a partir das definições dos vetores iniciais. Partindo das informações contidas na base de treino anotada, utilizou-se três métodos de seleção de atributos oferecidos pelo ambiente WEKA listados a seguir:

- **CfsSubsetEval / GeneticSearch:** O CfsSubsetEval dá preferência aos subconjuntos de atributos mais correlacionados com as classes e que são simultaneamente pouco inter-correlacionados (HALL et al., 2009). A estratégia de busca utilizada em combinação foi GeneticSearch, que emprega o algoritmo genético simples.

- **WrapperSubsetEval / RankSearch:** Avalia o conjunto de atributos utilizando um algoritmo de aprendizado com validação cruzada, que neste caso foi o NaiveBayes, O método de busca foi o RankSearch, que utiliza avaliadores para ranquear todos os atributos.

- **WrapperSubsetEval / GeneticSearch:** Nesta segunda configuração de WrapperSubsetEval, utilizou-se o algoritmo AdaBoostM1 combinado com o classificador J48, Para a estratégia de busca, utilizou-se o método GeneticSearch.

Além das definições de vetores de atributos bases para os algoritmos de indução e classificadores, cada HH foi analisado em conjunto com sua correspondente base de treino por cada um dos algoritmos de seleção citados acima, gerando variações diferentes da definição do vetor base. Com isso foi possível avaliar até quatro vetores de atributos diferentes nas etapas seguintes, contando com o vetor base, para cada HH.

Os vetores de atributos gerados foram, então, processados de forma independente associados à cada HH no restante do processo descrito na Figura 12. Na Tabela 11 e Tabela 12 do Apêndice A, lista-se a seleção final para o vetor de atributos para cada HH, após o término de todas as etapas do *workflow* para geração dos modelos pré-treinados.

4.2.3 Seleção de modelo

Esta etapa consiste na otimização de parâmetros e tem a importância de produzir um conjunto ótimo de opções de configuração para cada algoritmo associado a cada HH, já que o mesmo tem seu desempenho ligado diretamente a estes parâmetros, que, por sua vez, necessitam ser ajustados e refinados. Durante o procedimento, para cada HH, e para cada um dos vetores de atributos selecionados na etapa anterior, cria-se os conjuntos de parâmetros otimizados para cada classificador e algoritmo de indução de regras estudado.

A procura dos parâmetros ótimos foi semelhante ao método GridSearch disponível no WEKA, porém implementada programaticamente. Neste método, varia-se um dos parâmetros incrementalmente e gradualmente mantendo os outros fixos, ao final de cada varredura da faixa de valor aceitável, incrementa-se o valor de um dos outros parâmetros, e o ciclo se repete. Entende-se por valores aceitáveis aqueles pertencentes ao intervalo estipulado pelo WEKA. A avaliação de cada um desses conjuntos de parâmetros foi feita com o teste 10–CV, permitindo que a própria base de treino de cada HH pudesse ser utilizada como meio de avaliação.

Assim, para cada HH, e para cada definição de vetor de atributos associado ao mesmo, gerou-se um (ou mais de um, em caso de empate) conjunto de parâmetros ótimos para cada classificador (AdaBoostM1, NaiveBayes, J48, KStar e RandomForest) e indução de regras (FURIA, Ridor e JRip) de acordo com o valor da métrica MCC. A acurácia não foi comparada aqui pois a mesma tende a acompanhar o MCC com valores favoráveis. A Tabela 13 do Apêndice A mostra o algoritmo de desambiguação escolhido para cada HH e seus respectivos parâmetros otimizados.

4.2.4 Grupos de HH para treino

Esta etapa foi introduzida após observar que alguns HH poderiam ter sua base de treino aumentada com exemplos de outros HH no intuito de melhorar seu desempenho de desambiguação. A ideia é combinar a base de treino com um ou mais HH distintos, tal que essa combinação gere uma base melhorada. Entre os diversos HH em PB, existem grupos que se podem empregar em frases com características semelhantes. A base de treino ampliada foi feito da seguinte forma. Para cada HH, buscou-se um outro HH parceiro tal que a base de treino conjunta teve um desempenho mais favorável segundo o teste 10–CV, isto é, tenha acurácia elevada sem perda na métrica MCC. O procedimento foi executado novamente para encontrar um terceiro HH cuja incorporação na base de treino atual também melhore o desempenho, e assim sucessivamente. Caso não ocorra mais ganho, a etapa é encerrada para o HH em questão.

A procura dos grupos de HH para formação de uma base de treino otimizada para determinado HH foi executada para cada uma das configurações (candidatos de

classificadores e algoritmos de indução combinados com respectivos vetores de atributos selecionados) geradas nas etapas anteriores. A Tabela 13 do Apêndice A mostra o grupo de HH que compõem a base de treino definitiva de cada HH.

4.2.5 Seleção final dos algoritmos

As etapas anteriores testaram diversos algoritmos e suas respectivas configurações na tarefa de desambiguação de palavras. Obviamente, o algoritmo que apresentou melhor desempenho, considerando cada HH individualmente, foi o escolhido. Quando o desempenho foi numericamente idêntico em relação às métricas adotadas, deu-se prioridade aos algoritmos de indução de regras, que proporcionam possibilidade de análise através da inspeção, já que a mesma é bastante legível pelo ser humano. A Tabela 13 do Apêndice A apresenta o algoritmo escolhido para desambiguar cada um dos HH. O tempo gasto para completar a etapa de marcação manual foi de um semestre e as etapas de processamento restante levou cerca de três dias em um servidor dedicado (Dell PowerEdge T300, processador Intel Xeon 2,83GHz, 2GB de RAM e SO Arch Linux 64-bit).

4.2.6 Geração dos modelos pré-treinados

Munido das informações geradas pelas etapas anteriores, procede-se a criação dos modelos pré-treinados. Esses modelos definitivos dos algoritmos escolhidos (classificadores e indução de regras), um para cada HH a ser desambiguado, são gerados utilizando integralmente a sua respectiva base de treino em conjunto com a base de treino do grupo de HH identificado para melhoria de desempenho.

4.3 Avaliação do desambiguador

A avaliação do desempenho de desambiguação foi feita nos HH contidos nas frases do *corpus* LAPSNEWS, que foi reservado exclusivamente para essa finalidade. As pronúncias dos pares de HH contidos na LAPSNEWS também foram manualmente anotadas. Como dito, a LAPSNEWS não pertence à base de treino originalmente usada para gerar os classificadores treinados e induzir as regras, assim, o desambiguador processou novas instancias em frases nunca vistas antes. O resultado mostra uma acurácia média de 98,32% para os HH que ocorrem na LAPSNEWS. Porém, existem alguns casos particulares em que a abordagem adotada teve menor sucesso, conforme a Tabela 7 e graficamente na Figura 13, em um total de 91 HH reconhecidos.

Observa-se pelo gráfico da Figura 13 que os HH “bolas”, “corte”, “lobo” e “sede” tiveram desempenhos inferiores em relação aos outros HH. Ao notar que estes HH assumem o papel de substantivo para as pronúncias tanto abertas como fechadas, o desambiguador tem dificuldade de tratar estes casos em que envolvem um nível mais elevado de análise

contextual, estando dentro das expectativas previstas. Por exemplo, no caso de "bolas", a forma fechada "b[o]las" é um substantivo de ocorrência infrequente (em PE[3]: alteração de bolo; feito de carne, em PB[4]: veneno para matar cães) no *corpus* de treino, e como também a forma aberta "b[O]las", de maior ocorrência, é um substantivo, o desambiguador tende a confundir-se entre as duas pronúncias.

A palavra "pega" admite a pronúncia fechada como na frase: "Sua cachorra, uma vira-lata chamada Kika, foi **pega** na rua", do *corpus* de treino CETEN-Folha. Assim, frases semelhantes foram anotadas com a pronúncia fechada na base de treino. O desambiguador teve um desempenho de 89,47% de acertos para o HH "pega" nas frases do *corpus* LAPSNEWS, ficando abaixo da média geral (98,23%). Nesse caso, a dificuldade para identificar a pronúncia correta encontra-se no tempo verbal[5].

[3] "bola", in Dicionário Priberam da Língua portuguesa [online], 2008-2013, <http://www.priberam.pt/dlpo/bola> [consultado em 26-04-2016]

[4] Consultado em <http://michaelis.uol.com.br/moderno/portugues/index.php?lingua=portugues-portugues&palavra=bola>

[5] Mais detalhes em <http://www.migalhas.com.br/Gramatigalhas/10,MI206216,31047-Pego+e+ou+pego+e>

Tabela 7 – Resultado da desambiguação de HH conhecidos no *corpus* de avaliação LAPS-NEWS, totalizando 10.411 ocorrências em 91 HH. Legendas: ACC = acurácia; A = abertos certos; D = fechados certos; B = abertos incorretos e C = fechados incorretos.

Palavra	ACC (%)	A	D	B	C	Palavra	ACC (%)	A	D	B	C
aborto	100,00	0	38	0	0	gozo	100,00	0	1	0	0
acerto	100,00	0	55	0	0	interesse	100,00	0	286	0	0
acordo	98,39	0	793	13	0	interesses	100,00	0	131	0	0
adorno	100,00	0	1	0	0	jogo	100,00	0	147	0	0
almoço	98,65	0	73	1	0	leste	99,63	270	0	0	1
apelo	100,00	0	18	0	0	lobo	90,48	0	19	2	0
aperto	100,00	0	88	0	0	lobos	100,00	0	10	0	0
apoio	99,10	0	331	2	1	medo	99,46	0	183	1	0
apreço	100,00	0	4	0	0	medos	100,00	0	1	0	0
bestas	100,00	0	1	0	0	meta	100,00	189	1	0	0
bola	100,00	501	0	0	0	metas	100,00	113	0	0	0
bolas	83,33	15	0	0	3	modelo	99,57	0	234	1	0
cerca	98,30	3	343	0	6	namoro	100,00	0	18	0	0
choro	100,00	1	10	0	0	olho	98,91	1	90	1	0
colher	100,00	3	14	0	0	pega	89,47	31	3	0	4
colheres	100,00	1	0	0	0	pego	100,00	3	20	0	0
começo	99,62	6	257	1	0	pelas	100,00	0	52	0	0
concerto	100,00	0	6	0	0	peso	99,54	0	216	1	0
conforto	100,00	0	26	0	0	piloto	100,00	0	119	0	0
conserto	100,00	0	16	0	0	posto	98,39	0	183	3	0
consolo	100,00	0	7	0	0	reforço	100,00	0	53	0	0
contorno	100,00	0	10	0	0	relevo	100,00	0	3	0	0
controle	100,00	0	142	0	0	rola	60,00	3	0	0	2
cor	100,00	0	70	0	0	rolha	100,00	0	1	0	0
coro	100,00	0	22	0	0	rolo	100,00	0	3	0	0
corte	87,31	36	81	11	6	rota	100,00	74	0	0	0
desemprego	100,00	0	140	0	0	rotas	100,00	37	0	0	0
desespero	100,00	0	28	0	0	seca	98,11	0	52	0	1
desgosto	100,00	0	2	0	0	secas	100,00	0	17	0	0
desses	100,00	0	266	0	0	seco	100,00	0	70	0	0
deste	99,58	0	707	3	0	sede	89,71	304	1	0	35
destes	100,00	0	69	0	0	selo	100,00	0	21	0	0
emprego	100,00	0	217	0	0	sobre	100,00	0	302	0	0
endosso	100,00	0	3	0	0	sopro	100,00	0	1	0	0
enredo	100,00	0	6	0	0	suborno	100,00	0	29	0	0
enterro	100,00	0	26	0	0	sufoco	100,00	0	6	0	0
erro	100,00	0	145	0	0	termos	100,00	0	203	0	0
esforço	99,37	0	158	0	1	toco	100,00	4	2	0	0
esmero	100,00	0	1	0	0	topo	100,00	0	48	0	0
este	100,00	0	329	0	0	torno	100,00	0	165	0	0
fora	99,53	208	2	0	1	torre	96,55	0	28	1	0
força	99,03	0	102	1	0	torres	100,00	0	9	0	0
forma	100,00	25	0	0	0	travessa	75,00	3	0	0	1
gelo	100,00	0	47	0	0	troco	100,00	0	5	0	0
gosto	89,77	42	37	0	9	zelo	100,00	0	6	0	0
governo	100,00	0	995	0	0						

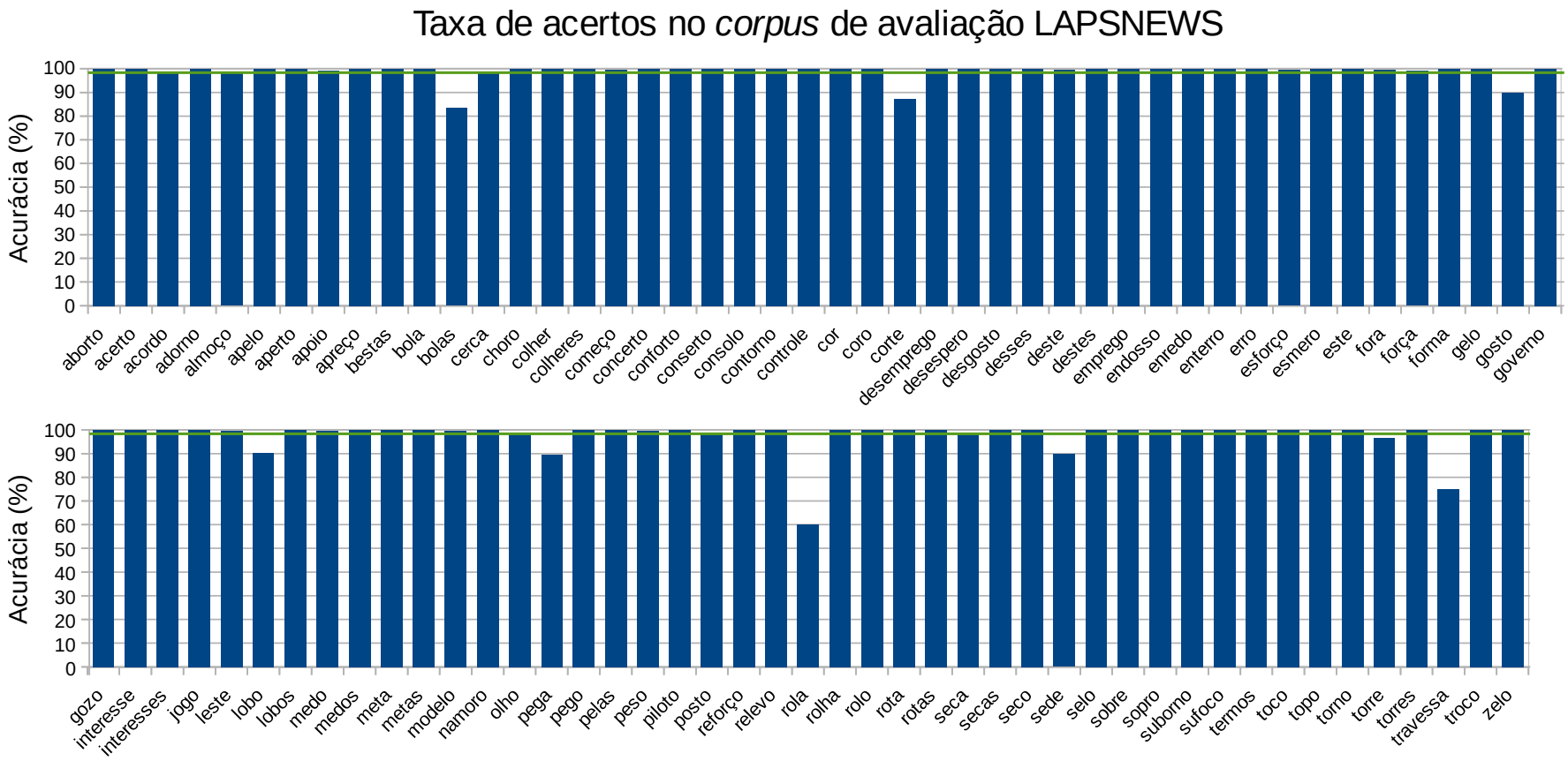

Figura 13 – A avaliação do desambiguador de HH no *corpus* LAPSNEWS teve um acerto médio de 98,32%.

As palavras “gosto”, “travessa” e “rola” são desambiguados pelos algoritmos de indução de regras JRip, Ridor e o classificador RandomForest, respectivamente. No intuito de inspecionar com mais detalhes, as regras induzidas para “gosto” e “travessa” serão discutidas a seguir.

A Tabela 8 é um mapa dos símbolos usados nas regras induzidas listadas no Apêndice B, e a correspondente notação utilizada na definição dos vetores de atributos.

Tabela 8 – Descrição dos símbolos usados nas listagens de regras induzidas. As posições são relativas ao HH considerado.

Símbolo	Descrição	No vetor de atributos
POS_left_n	enésima etiqueta gramatical à esquerda	$p_{-3},\ p_{-2},\ p_{-1}$
POS_right_n	enésima etiqueta gramatical à direita	$p_{+1},\ p_{+2},\ p_{+3}$
Word_left_n	enésima palavra à esquerda	$w_{-3},\ w_{-2},\ w_{-1}$
Word_right_n	enésima palavra à direita	$p_{+1},\ p_{+2},\ p_{+3}$
Prefix_left_n	prefixo da enésima palavra à esquerda	$pre_{-3}, pre_{-2}, pre_{-1}$
Prefix_right_n	prefixo da enésima palavra à direita	$pre_{+1}, pre_{+2}, pre_{+3}$
Sufix_left_n	sufixo da enésima palavra à esquerda	$suf_{-3}, suf_{-2}, suf_{-1}$
Sufix_right_n	sufixo da enésima palavra à direita	$suf_{+1}, suf_{+2}, suf_{+3}$
pronuncia	pronúncia prevista	0=fechada, 1=aberta
=IF= e =if=	indicador de início da frase	
=FF= e =ff=	indicador de fim da frase	

Na Tabela 9 é exibida a sequência de regras induzidas pelo algoritmo JRip a partir dos exemplares presentes no *corpus* de treino, utilizada para determinar se o HH “gosto” em uma frase possui pronúncia aberta ou fechada. Para observar o funcionamento, analisaremos algumas frases classificadas incorretamente.

Tabela 9 – Regras induzidas pelo algoritmo JRip para desambiguação do HH “gosto”.

```
1 (Word_left_3 = =if=) and (Word_left_1 = =if=) => pronuncia=1 (40.0/0.0)
2 (Word_left_1 = não) => pronuncia=1 (58.0/0.0)
3 (POS_left_1 = PROPESS) => pronuncia=1 (36.0/1.0)
4 (Word_left_3 = =if=) and (POS_right_2 = N) => pronuncia=1 (13.0/4.0)
5 (Word_right_1 = muito) => pronuncia=1 (12.0/1.0)
6 (Word_left_1 = que) and (POS_left_2 = N) => pronuncia=1 (5.0/0.0)
7 (Word_right_1 = de) and (Word_right_2 = ti) => pronuncia=1 (4.0/0.0)
8  => pronuncia=0 (623.0/24.0)
```

Por exemplo, a representação da instância de entrada (ou vetor de atributo) do HH “gosto” na frase: “Mas para me organizar melhor, **gosto** de ter os jogadores o mais rápido possível” para o algoritmo JRip é:

$$\underset{w_{-3}}{\text{me}},\ \underset{w_{-2}}{\text{dorme}},\ \underset{w_{-1}}{\text{melhor}},\ \underset{w_{+1}}{\text{de}},\ \underset{w_{+2}}{\text{ter}},\ \underset{w_{+3}}{\text{os}},\ \underset{p_{-3}}{\text{PROPESS}},\ \underset{p_{-2}}{\text{V}},\ \underset{p_{-1}}{\text{ADJ}},\ \underset{p_{+1}}{\text{PREP}},\ \underset{p_{+2}}{\text{V}},\ \underset{p_{+3}}{\text{ART}},\ \underset{pre_{-3}}{\text{?}},$$

$$\underset{pre_{-2}}{\text{?}},\ \underset{pre_{-1}}{\text{?}},\ \underset{pre_{+1}}{\text{?}},\ \underset{pre_{+2}}{\text{?}},\ \underset{pre_{+3}}{\text{o-}},\ \underset{suf_{-3}}{\text{?}},\ \underset{suf_{-2}}{\text{?}},\ \underset{suf_{-1}}{\text{-or}},\ \underset{suf_{+1}}{\text{?}},\ \underset{suf_{+2}}{\text{?}},\ \underset{suf_{+3}}{\text{-s}}$$

Note que no vetor de atributos, o verbo "organizar" foi automaticamente substituído por "dorme". Isso porque na base de treino correspondente ao HH "gosto" não existe a palavra "organizar" na posição relativa w_{-2}. As regras induzidas na Tabela 9 não capturam esta instância e a mesma recai na regra padrão (linha 8), indicando incorretamente a pronúncia fechada (0). Intuitivamente, sabe-se que a pronúncia da palavra "gosto" na frase em questão deveria ser aberta, mas o pronome "eu", que o antecede, está oculto, logo, não é percebido pelo desambiguador.

Já a frase "Hoje estou jogando bem e vou atuar no piso rápido, que é o que eu mais **gosto** disse Thiago Alves" teve sua instância representada por:

$$\underset{w_{-3}}{\text{que}},\ \underset{w_{-2}}{\text{eu}}\ ,\ \underset{w_{-1}}{\text{mais}},\ \underset{w_{+1}}{?}\ ,\ \underset{w_{+2}}{\text{david}},\ \underset{w_{+3}}{?}\ ,\ \underset{p_{-3}}{\text{PRO-KS}},\ \underset{p_{-2}}{\text{PROPESS}},\ \underset{p_{-1}}{\text{PROADJ}},\ \underset{p_{+1}}{\text{V}}\ ,\ \underset{p_{+2}}{\text{NPROP}},$$

$$\underset{p_{+3}}{\text{NPROP}},\ \underset{pre_{-3}}{?}\ ,\ \underset{pre_{-2}}{\text{e-}}\ ,\ \underset{pre_{-1}}{?}\ ,\ \underset{pre_{+1}}{\text{dis-}}\ ,\ \underset{pre_{+2}}{?}\ ,\ \underset{pre_{+3}}{\text{a-}}\ ,\ \underset{suf_{-3}}{?}\ ,\ \underset{suf_{-2}}{\text{-u}}\ ,$$

$$\underset{suf_{-1}}{\text{-is}}\ ,\ \underset{suf_{+1}}{?}\ ,\ \underset{suf_{+2}}{?}\ ,\ \underset{suf_{+3}}{\text{-es}}$$

Nesse segundo exemplo, o pronome "eu" é seguido da palavra "mais", ainda assim, nenhuma das regras casou com a instância acima, recaindo novamente na resposta incorreta de pronúncia fechada. Na posição relativa w_{+2} houve uma substituição do pronome pessoal "Thiago" por "david", possibilitado devido que o etiquetador morfossintático conseguiu anotar corretamente esta informação. No geral, casos semelhantes com pronome oculto ou explicito mas não imediatamente antecedente ao HH são mais frequentes e confusos para o desambiguador.

As regras listadas na Tabela 10 foram induzidas pelo algoritmo Ridor para desambiguar o HH "travessa" a partir das amostras no *corpus* de treino.

Tabela 10 – Regras induzidas pelo algoritmo Ridor para desambiguação do HH "travessa".

```
1 pronuncia = 0  (128.0/81.0)
2 Except (POS_right_2 = N) => pronuncia = 1  (21.0/1.0) [24.0/2.0]
3 Except (POS_left_1 = PREP+ART) => pronuncia = 1  (6.0/0.0) [9.0/1.0]
4 Except (POS_right_1 = N) and (Word_right_1 = j) => pronuncia = 1  (5.0/0.0)
      [4.0/0.0]
```

Tomando a frase "A **travessa** se encontra fechada desde as cinco horas" como entrada, temos a instância a seguir:

$$\underset{w_{-3}}{\text{=if=}},\ \underset{w_{-2}}{\text{=if=}},\ \underset{w_{-1}}{\text{a}}\ ,\ \underset{w_{+1}}{\text{se}}\ ,\ \underset{w_{+2}}{\text{faceira}},\ \underset{w_{+3}}{?}\ ,\ \underset{p_{-3}}{\text{=IF=}},\ \underset{p_{-2}}{\text{=IF=}},\ \underset{p_{-1}}{\text{ART}},\ \underset{p_{+1}}{\text{PROPESS}},\ \underset{p_{+2}}{\text{V}}\ ,\ \underset{p_{+3}}{\text{PCP}},\ \underset{pre_{-3}}{?}\ ,$$

$$\underset{pre_{-2}}{?}\ ,\ \underset{pre_{-1}}{?}\ ,\ \underset{pre_{+1}}{?}\ ,\ \underset{pre_{+2}}{?}\ ,\ \underset{pre_{+3}}{?}\ ,\ \underset{suf_{-3}}{?}\ ,\ \underset{suf_{-2}}{?}\ ,\ \underset{suf_{-1}}{?}\ ,\ \underset{suf_{+1}}{?}\ ,\ \underset{suf_{+2}}{\text{-eira}}\ ,\ \underset{suf_{+3}}{\text{-ada}}$$

Para a frase em questão, não houve regra que cobrisse o vetor correspondente, recaindo na regra padrão (linha 1) de pronúncia fechada, que para esse exemplo é equivocada. Já a comparação da palavra à direita do HH com a letra "j" (linha 4), é porque muitos nomes de ruas possuem abreviações, o que sugere outro melhoramento, que seria consolidar um marcador uniforme para estes casos.

A frase "A confusão **rola** solta nos filmes da seção midnight madness do festival de Toronto, que começou literalmente à meia noite da sexta feira...", teve a pronúncia emitida incorretamente pelo algoritmo atribuído RandomForest. Sua representação de entrada ao desambiguador é:

$$\frac{\text{N}}{p_{-1}} \, , \, \frac{\text{questões}}{w_{-1}}$$

A instância representada é compacta, consistindo em apenas dois atributos selecionados. A palavra à esquerda ao HH, "confusão", foi substituída por "questões", que consta na base de treino. A distinção comum é entre a forma do verbo "rolar" e o substantivo "rola" (espécie de ave). Este caso é um candidato em potencial para revisão das frases com este HH no *corpus* de treino, analisando as anotações à procura de exemplos incorretos ou confusos para o algoritmo.

Apesar da dificuldade com alguns HH conforme os exemplos apresentados, a técnica de aprendizado supervisionado aplicada tem a possibilidade de ter um bom desempenho em aplicações direcionadas como em TTS. As regras induzidas ajudam a desvendar algumas características da base de treino atual e fornece um indicador para tentar melhorá-lo. No caso dos classificadores atribuídos, fornecem uma possibilidade de maior generalização que as regras, mas para isso, ainda cabe melhorias na base de treino e na definição do vetor de atributos.

5 Conclusão

O uso de técnicas de aprendizado de máquina para tratar o problema de ambiguidade entre palavras não é novidade em várias línguas, porém, no que tange ao PB, as referências são escassas. A maioria das pesquisas usam regras linguísticas formuladas com base em análise contextual, contudo, não descrevem claramente a metodologia usada para compor as regras, além de ser uma estratégia de difícil implementação do ponto de visto semântico. Outros estudos baseiam-se de forma limitada em etiquetas gramaticais, onde as colocações destas nas vizinhanças dos HH determinam a pronúncia adequada.

Assim sendo, o método de aprendizagem supervisionada aqui desenvolvido para tratar do problema de pronúncias de pares de HH em PB é viável até certo limite. A adoção do etiquetador morfossintático traz diversas vantagens mas seu desempenho também tem influência sobre o desambiguador.

Apesar da elevada acurácia em texto típico, o desambiguador de HH desenvolvido sofre de pelo menos duas limitações:

(i) Em geral, existe escassez do emprego de um dos pares de pronúncias do HH no *corpus* de treino, o que em princípio acarreta dificuldade na coleta de exemplos variados e possíveis do HH em PB;

(ii) Para HH de mesma classe gramatical, a abordagem de contexto local tem queda no desempenho, para que sejam corretamente desambiguado com precisão, estes parecem requerer em princípio uma interpretação semântica, um problema WSD clássico.

O estado atual do desambiguador poderá ser melhorado para alguns HH em que necessitam de mais exemplos. Um dos motivos é a ocorrência desbalanceada da pronúncia de muitos HH. Somando a isto, existe também a necessidade de escolher o *corpus* de treino, visto que o desempenho em um domínio isolado não cobre facilmente outros, por isso, a decisão dependerá do contexto da aplicação final.

5.1 Recursos gerados

Durante o desenvolvimento deste trabalho para desambiguação automática de HH, gerou-se os seguintes recursos principais:

(i) Uma base textual com os HH anotados manualmente com possibilidade de futuras ampliações;

(ii) Modelos de classificadores e regras induzidas a partir da base de treino pronto para desambiguar HH;

(iii) Códigos fontes empregados durante o desenvolvimento nas linguagens Python, Lua e Java que podem ser adaptados para novos projetos.

5.2 Objetivos atingidos

Os seguintes objetivos foram atingidos neste trabalho:

(i) Aplicação de diferentes algoritmos de aprendizado de máquina na desambiguação de cada HH;

(ii) Análise das características da base de conhecimento (tamanho, domínio e etc.) no desempenho do sistema;

(iii) Disponibilização de um desambiguador de HH funcional e de código-livre para a comunidade (parcialmente atingido).

5.3 Trabalhos futuros

Nos trabalhos futuros, planeja-se as seguintes atividades:

(i) Avaliar o desambiguador em um TTS, para verificar se a voz sintética sofre algum tipo de alteração na qualidade de acordo com a resposta dos voluntários submetido ao experimento de escuta;

(ii) Aplicar em um ambiente embarcado, para isto, é preferível a uniformização dos programas desenvolvidos para uma linguagem de programação, por exemplo o Java, e solucionar a portabilidade das ferramentas como o etiquetador morfossintático e reduzir do número de dependências externas incorporando e/ou reimplementado os mesmos;

(iii) Disponibilização completa os códigos fontes, para facilitar a aquisição das ferramentas desenvolvidas neste trabalho por outros pesquisadores da área e interessados, incluindo a base de treino usada para produzir os modelos pré-treinados;

(iv) Avaliar do grau de diferença estatística, para observar as diferenças entre os classificadores usando testes estatísticos.

Referências

ABL. *Espaço Machado de Assis na Web da Academia Brasileira de Letras*. [S.l.]: Centro Cultural da ABL, 2011. Disponível em: <http://www.machadodeassis.org.br>. Acesso em 6 de agosto de 2015. Citado na página 29.

ABNEY, S. Understanding the Yarowsky algorithm. *Comput. Linguist.*, MIT Press, Cambridge, MA, USA, v. 30, n. 3, p. 365–395, set. 2004. ISSN 0891-2017. Disponível em: <http://dx.doi.org/10.1162/0891201041850876>. Citado na página 23.

AIRES, R. et al. Combining multiple classifiers to improve part of speech tagging: A case study for brazilian portuguese. In: *The Proceeding of the Brazilian AI Symposium*. [S.l.: s.n.], 2000. (SBIA 2000). Citado na página 31.

ALUÍSIO, S. et al. An account of the challenge of tagging a reference corpus for brazilian portuguese. In: *6th Workshop on Computational Processing of the Portuguese Language*. Berlin, Heidelberg: Springer-Verlag, 2003. (PROPOR'2003), p. 110–117. Citado na página 28.

ARLOT, S.; CELISSE, A. A survey of cross-validation procedures for model selection. *Statist. Surv.*, The American Statistical Association, the Bernoulli Society, the Institute of Mathematical Statistics, and the Statistical Society of Canada, v. 4, p. 40–79, 2010. Disponível em: <http://dx.doi.org/10.1214/09-SS054>. Citado na página 39.

BARBOSA, F.; FERRARI, L.; RESENDE JR, F. G. V. A distinção entre homógrafos heterófonos em sistemas de conversão texto-fala. In: *Processamento da Linguagem, Cultura e Cognição: Estudos de Linguística Cognitiva*. Braga, Portugal: [s.n.], 2003. Citado na página 24.

BARBOSA, F.; FERRARI, L.; RESENDE JR, F. G. V. A methodology to analyze homographs for a brazilian portuguese TTS system. In: *6th Workshop on Computational Processing of the Portuguese Language*. Berlin, Heidelberg: Springer-Verlag, 2003. (PROPOR'2003), p. 57–61. Citado na página 24.

BASILE, P. et al. Combining knowledge-based methods and supervised learning for effective italian word sense disambiguation. In: *Proceedings of the 2008 Conference on Semantics in Text Processing*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2008. (STEP '08), p. 5–16. Disponível em: <http://dl.acm.org/citation.cfm?id=1626481.1626483>. Citado na página 22.

BERNERS-LEE, T.; HENDLER, J.; LASSILA, O. *The Semantic Web*: A new form of web content that is meaningful to computers will unleash a revolution of new possibilities. 2001. Disponível em: <http://www-sop.inria.fr/acacia/cours/essi2006/Scientific%20American_%20Feature%20Article_%20The%20Semantic%20Web_%20May%202001.pdf>. Acesso em 24 de junho de 2016. Citado na página 20.

BHARGAVA, N. et al. Decision tree analysis on j48 algorithm for data mining. *International Journal of Advanced Research in Computer Science and Software Engineering*, v. 3, p. 1114–1119, June 2013. ISSN 2277 128X. Disponível em:

<http://www.ijarcsse.com/docs/papers/Volume_3/6_June2013/V3I6-0408.pdf>. Citado na página 34.

BIELZA, C.; LARRAÑAGA, P. Discrete Bayesian network classifiers: A survey. *ACM Computing Surveys*, v. 47, n. 1, p. 43, April 2014. DOI: <http://dx.doi.org/10.1145/2576868>. Citado 2 vezes nas páginas 33 e 34.

BRAGA, D.; MARQUES, M. A. In: _____. *Diacrítica*. Braga: Centro de Estudos Humanísticos/Universidade do Minho, 2007. (Ciências da Linguagem, 21), cap. Desambiguador de Homógrafos Heterófonos para Sistemas de Conversão Texto-Fala em Português, p. 25–49. Citado 3 vezes nas páginas 15, 16 e 24.

BREIMAN, L. Random Forests. *Machine Learning*, v. 45, n. 1, p. 5–32, 2001. ISSN 1573-0565. Disponível em: <http://dx.doi.org/10.1023/A:1010933404324>. Citado 2 vezes nas páginas 33 e 34.

CAMBRIA, E.; WHITE, B. Jumping nlp curves: A review of natural language processing research [review article]. *IEEE Computational Intelligence Magazine*, v. 9, n. 2, p. 48–57, May 2014. ISSN 1556-603X. Citado na página 15.

CARDIE, C. Embedded machine learning system for natural language processing: A general framework. In: . Berlin, Heidelberg: Springer, 1996, (Lecture Notes in Computer Science, v. 1040). cap. Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing, p. 315–328. Citado na página 15.

CLEARY, J. G.; TRIGG, L. E. K*: An instance-based learner using an entropic distance measure. In: *12th International Conference on Machine Learning*. [S.l.]: Morgan Kaufmann, 1995. p. 108–114. Citado na página 33.

COHEN, W. W. Fast effective rule induction. In: *In Proceedings of the Twelfth International Conference on Machine Learning*. [S.l.]: Morgan Kaufmann, 1995. p. 115–123. Citado na página 36.

COLLOBERT, R. et al. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, JMLR.org, v. 12, p. 2493–2537, nov. 2011. ISSN 1532-4435. Disponível em: <http://dl.acm.org/citation.cfm?id=1953048.2078186>. Citado na página 31.

COSTA, E. S. et al. Um framework para desenvolvimento de sistemas TTS personalizados no Português do Brasil. In: *XXX Simpósio Brasileiro de Telecomunicações*. Brasília, DF: [s.n.], 2012. (SBrT'12). Citado 2 vezes nas páginas 9 e 16.

CUNHA, C.; CINTRA, L. *Nova gramática do português contemporâneo*. 6. ed. Rio de Janeiro: Lexikon Editorial, 2013. Citado na página 37.

DOMINGOS, P. *Machine Learning*: Week three: Learning sets of rules and logic programs. 2016. <https://class.coursera.org/machlearning-001/lecture>. Acesso em 15 de Abril de 2016. Citado na página 36.

FERRARI, L.; BARBOSA, F.; RESENDE JR, F. G. V. Construções gramaticais e sistemas de conversão texto-fala: O caso dos homógrafos. In: *Proceedings of the International Conference on Cognitive Linguistics*. [S.l.: s.n.], 2003. Citado na página 24.

FONSECA, E. R.; ROSA, J. L. G. Mac-Morpho revisited: Towards robust part-of-speech tagging. In: *Proceedings of the 9th Brazilian Symposium in Information and Human Language Technology*. Fortaleza, Ceará: Sociedade Brasileira de Computação, 2013. p. 98–107. Citado 3 vezes nas páginas 28, 31 e 32.

FREUND, Y.; SCHAPIRE, R. E. Experiments with a new boosting algorithm. In: SAITTA, L. (Ed.). *Proceedings of the Thirteenth International Conference on Machine Learning (ICML 1996)*. Morgan Kaufmann, 1996. p. 148–156. ISBN 1-55860-419-7. Disponível em: <http://www.biostat.wisc.edu/~kbroman/teaching/statgen/2004/refs/freund.pdf>. Citado na página 33.

GALE, W. A.; CHURCH, K. W.; YAROWSKY, D. A method for disambiguating word senses in a large corpus. *Computers and the Humanities*, v. 26, n. 5, p. 415–439. ISSN 1572-8412. Disponível em: <http://dx.doi.org/10.1007/BF00136984>. Citado na página 39.

GALVES, C.; FARIA, P. *Tycho Brahe Parsed Corpus of Historical Portuguese*. 2010. Disponível em: <http://www.tycho.iel.unicamp.br/~tycho/corpus/en/index.html>. Acesso em 6 de agosto de 2015. Citado na página 29.

HALE, S. A. Global connectivity and multilinguals in the Twitter network. In: *Proceedings of the 2014 ACM Annual Conference on Human Factors in Computing Systems*. Montreal, Canada: ACM, 2014. Citado na página 30.

HALL, M. et al. The WEKA data mining software: An update. *SIGKDD Explorations*, v. 11, n. 1, p. 10–18, 2009. Citado 3 vezes nas páginas 33, 36 e 51.

HAMADA, L.; NETO, N. Desambiguação de homógrafos-heterófonos por aprendizado de máquina em português brasileiro. In: *X Symposium in Information and Human Language Technology*. Natal, Rio Grande do Norte: [s.n.], 2015. (STIL 2015), p. 181–190. Citado 3 vezes nas páginas 17, 43 e 44.

HAN, J.; KAMBER, M.; PEI, J. *Data Mining: Concepts and Techniques*. 3. ed. [S.l.]: Morgan Kaufmann, 2011. Citado na página 32.

HÜHN, J.; HÜLLERMEIER, E. Furia: an algorithm for unordered fuzzy rule induction. *Data Mining and Knowledge Discovery*, v. 19, n. 3, p. 293–319, 2009. ISSN 1573-756X. Disponível em: <http://dx.doi.org/10.1007/s10618-009-0131-8>. Citado na página 36.

HUSAIN, M. S.; BEG, M. R. Word sense ambiguity: A survey. *International Journal of Computer and Information Technology*, v. 2, p. 1161–1168, November 2013. ISSN 2279-0764. Disponível em: <http://www.ijcit.com>. Citado 4 vezes nas páginas 15, 21, 22 e 34.

IDE, N.; VÉRONIS, J. Introduction to the special issue on word sense disambiguation: The state of the art. *Comput. Linguist.*, MIT Press, Cambridge, MA, USA, v. 24, n. 1, p. 2–40, mar. 1998. ISSN 0891-2017. Disponível em: <http://dl.acm.org/citation.cfm?id=972719.972721>. Citado na página 17.

JURMAN, G.; RICCADONNA, S.; FURLANELLO, C. A comparison of MCC and CEN error measures in multi-class prediction. *PLoS ONE*, Public Library of Science, v. 7, n. 8, p. 1–8, August 2012. Disponível em: <http://dx.doi.org/10.1371%2Fjournal.pone.0041882>. Citado 2 vezes nas páginas 38 e 39.

KOHAVI, R. A study of cross-validation and bootstrap for accuracy estimation and model selection. In: *INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE*. [S.l.: s.n.], 1995. p. 1137–1143. Citado na página 39.

MADARIAGA, R. S. de; CASTILLO, J. R. F. del. The bootstrapping of the Yarowsky algorithm in real corpora. *Information Processing & Management*, v. 45, n. 1, p. 55–69, 2009. ISSN 0306-4573. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0306457308000794>. Citado na página 23.

MÀRQUEZ, L. *Machine Learning and Natural Language Processing*. Barcelona, Spain, 2000. 53 p. Technical report: Centre de recerca TALP, Departament de Llenguatges i Sistemes Informàtics, LSI, Universitat Politècnica de Catalunya, UPC. Citado na página 34.

MÀRQUEZ, L. et al. Supervised corpus-based methods for WSD. In: _____. *Word sense disambiguation*: Algorithms and applications. New York, NY: Springer Netherlands, 2006. (Text, Speech and Language Technology, v. 33), p. 167–216. ISBN 978-1-4020-4809-8. Citado 3 vezes nas páginas 21, 23 e 33.

MCCARTHY, D. Word Sense Disambiguation: An overview. *Language and Linguistics Compass*, Blackwell Publishing Ltd, v. 3, n. 2, p. 537–558, 2009. ISSN 1749-818X. Disponível em: <http://dx.doi.org/10.1111/j.1749-818X.2009.00131.x>. Citado 4 vezes nas páginas 15, 17, 20 e 23.

MONTOYO, A.; SUÁREZ, A.; PALOMAR, M. Combining supervised-unsupervised methods for word sense disambiguation. In: *Proceedings of the Third International Conference on Computational Linguistics and Intelligent Text Processing*. London, UK, UK: Springer-Verlag, 2002. (CICLing '02), p. 156–164. ISBN 3-540-43219-1. Disponível em: <http://dl.acm.org/citation.cfm?id=647344.724010>. Citado na página 22.

MONTOYO, A. et al. Combining knowledge- and corpus-based word-sense-disambiguation methods. *J. Artif. Int. Res.*, AI Access Foundation, USA, v. 23, n. 1, p. 299–330, mar. 2005. ISSN 1076-9757. Disponível em: <http://dl.acm.org/citation.cfm?id=1622503.1622510>. Citado na página 22.

MORAIS, E.; VIOLARO, F. Data-driven text-to-speech synthesis. In: *XXII Simpósio Brasileiro de Telecomunicações*. Campinas, SP: [s.n.], 2005. (SBrT'05). Citado na página 16.

NAVIGLI, R. Word sense disambiguation: A survey. *ACM Comput. Surv.*, ACM, New York, NY, USA, v. 41, n. 2, p. 10:1–10:69, February 2009. ISSN 0360-0300. Disponível em: <http://doi.acm.org/10.1145/1459352.1459355>. Citado 5 vezes nas páginas 20, 22, 28, 34 e 39.

NAVIGLI, R. SOFSEM 2012: Theory and practice of computer science: 38th conference on current trends in theory and practice of computer science, špindlerův mlýn, Czech Republic, january 21-27, 2012. proceedings. In: _____. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. cap. A Quick Tour of Word Sense Disambiguation, Induction and Related Approaches, p. 115–129. ISBN 978-3-642-27660-6. Disponível em: <http://dx.doi.org/10.1007/978-3-642-27660-6_10>. Citado 4 vezes nas páginas 15, 16, 20 e 22.

NETO, N. et al. Free tools and resources for brazilian portuguese speech recognition. *Journal of the Brazilian Computer Society*, v. 17, p. 53–68, 2011. Citado na página 29.

NNAMOKO, N. A. et al. Evaluation of filter and wrapper methods for feature selection in supervised machine learning. In: MERABTI, M.; ABUELMA'ATTI, O. (Ed.). *The 15th Annual Postgraduate Symposium on the convergence of Telecommunication, Networking and Broadcasting*. Liverpool, United Kingdom: Liverpool John Moores University, School of Computing & Mathematical Sciences, 2014. (PGNet 2014), p. 296. ISBN 9781902560281. Citado na página 51.

NORDMAN, A. *Data Mining*: Classification: Decision trees. 2016. <http://staffwww.itn.liu.se/~aidvi/courses/06/dm/lectures/lec3.pdf>. Acesso em 29 de junho de 2016. Citado na página 34.

PAIVA, V. de; RADEMAKER, A.; MELO, G. de. Openwordnet-pt: An open Brazilian Wordnet for reasoning. In: *Proceedings of COLING 2012: Demonstration Papers*. Mumbai, India: The COLING 2012 Organizing Committee, 2012. p. 353–360. Published also as Techreport http://hdl.handle.net/10438/10274. Disponível em: <http://www.aclweb.org/anthology/C12-3044>. Citado na página 22.

Portal S. F. *Portal São Francisco*. 1998. Disponível em: <http://www.portalsaofrancisco.com.br/>. Acesso em 6 de agosto de 2015. Citado na página 29.

POWERS, D. M. W. Evaluation: From precision, recall and f-measure to ROC, informedness, markedness & correlation. *Journal of Machine Learning Technologies*, v. 2, n. 1, p. 37–63, 2011. Citado 2 vezes nas páginas 38 e 39.

RESNIK, P. Word sense disambiguation: Algorithms and applications. In: _____. Dordrecht: Springer Netherlands, 2006. cap. WSD in NLP Applications, p. 299–337. ISBN 978-1-4020-4809-8. Disponível em: <http://dx.doi.org/10.1007/978-1-4020-4809-8_11>. Citado 2 vezes nas páginas 17 e 20.

REZENDE, S. O. (Ed.). *Sistemas inteligentes: fundamentos e aplicações*. 1. ed. Barueri, SP: Manole Ltda., 2005. Citado na página 36.

RIBEIRO, R.; OLIVEIRA, L.; TRANCOSO, I. Using morphosyntactic information in TTS systems: Comparing strategies for european portuguese. In: *6th Workshop on Computational Processing of the Portuguese Language*. Berlin, Heidelberg: Springer-Verlag, 2003. (PROPOR'2003), p. 143–150. Citado 2 vezes nas páginas 24 e 31.

SEARA, I. et al. Considerações sobre os problemas de alternância vocálica das formas verbais do português falado no Brasil para aplicação em um sistema de conversão texto-fala. In: *XIX Simpósio Brasileiro de Telecomunicações*. Fortaleza, CE: [s.n.], 2001. (SBrT 2001). Citado 2 vezes nas páginas 14 e 24.

SEARA, I. et al. Alternância vocálica das formas verbais e nominais do português brasileiro para aplicação em conversão texto-fala. *Revista da Sociedade Brasileira de Telecomunicações*, v. 17, n. 1, p. 79–85, June 2002. Citado na página 24.

SHULBY, C.; MENDONÇA, G.; MARQUIAFÁVEL, V. Automatic disambiguation of homographic heterophone pairs containing open and closed mid vowels. In: *Proceedings of the 9th Brazilian Symposium in Information and Human Language Technology*. Fortaleza, Ceará: [s.n.], 2013. p. 126–137. Citado na página 24.

SIDOROV, G. et al. Syntactic n-grams as machine learning features for natural language processing. *Expert Systems with Applications*, v. 41, n. 3, p. 853 – 860, 2014. ISSN 0957-4174. Methods and Applications of Artificial and Computational Intelligence. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0957417413006271>. Citado na página 35.

SILVA, D. C.; BRAGA, D.; RESENDE JR, F. G. V. Conjunto de regras para desambiguação de homógrafos heterófonos no português brasileiro. In: *XXVII Simpósio Brasileiro de Telecomunicações*. Blumenau, SC: [s.n.], 2009. (SBrT 2009). Citado 2 vezes nas páginas 17 e 27.

SILVA, D. C.; BRAGA, D.; RESENDE JR, F. G. V. A rule-based method for homograph disambiguation in brazilian portuguese text-to-speech systems. *Journal of Communications and Information Systems*, v. 1, n. 1, April 2012. Citado 4 vezes nas páginas 14, 23, 24 e 40.

TEODORESCU, L. R. et al. Part of speech tagging for romanian text-to-speech system. In: *13th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*. [s.n.], 2011. (SYNASC 2011), p. 153–159. ISBN 978-0-7695-4630-8. Disponível em: <http://doi.ieeecomputersociety.org/10.1109/SYNASC.2011.55>. Citado na página 23.

WEBB, I. G.; BOUGHTON, R. J.; WANG, Z. Not so naive bayes: Aggregating one-dependence estimators. *Machine Learning*, v. 58, n. 1, p. 5–24, january 2005. ISSN 1573-0565. Disponível em: <http://dx.doi.org/10.1007/s10994-005-4258-6>. Citado na página 33.

WITTEN, H.; FRANK, E. *Data Mining: Practical Machine Learning Tools and Techniques*. 2. ed. [S.l.]: Morgan Kaufmann, 2005. Citado na página 33.

YAROWSKY, D. Progress in speech synthesis. In: _____. New York, NY: Springer New York, 1997. cap. Homograph Disambiguation in Text-to-Speech Synthesis, p. 157–172. ISBN 978-1-4612-1894-4. Disponível em: <http://dx.doi.org/10.1007/978-1-4612-1894-4_12>. Citado na página 23.

ZAMPIERI, M. Evaluating knowledge-poor and knowledge-rich features in automatic classification: A case study in WSD. In: *Computational Intelligence and Informatics (CINTI), 2012 IEEE 13th International Symposium on*. [S.l.: s.n.], 2012. p. 359–363. Citado 2 vezes nas páginas 21 e 22.

Apêndices

APÊNDICE A – Dados finais para geração dos modelos WEKA

Tabela 11 – Atributos selecionados para desambiguação por regras induzidas para cada HH; ao todo são 73 HH agrupados conforme a tabela. A definição do vetor é mostrado primeiro, seguido de todos os HH que utilizam esta definição. O algoritmo para desambiguação correspondente ao HH e seus parâmetros estão definidos na Tabela 13.

w_{-3}, w_{-2}, w_{-1}, w_{+1}, w_{+2}, w_{+3}, p_{-3}, p_{-2}, p_{-1}, p_{+1}, p_{+2}, p_{+3}, $pre_{-3}, pre_{-2}, pre_{-1}, pre_{+1}, pre_{+2}, pre_{+3}$, $suf_{-3}, suf_{-2}, suf_{-1}, suf_{+1}, suf_{+2}, suf_{+3}$: acerto, adorno, apoio, apreço, besta, bestas, bola, cerro, colher, colheres, controle, cor, coro, corte, desemprego, desespero, desgosto, endosso, enredo, erro, esforço, esmero, gelo, gosto, governo, interesses, namoro, olho, pela, pelas, pelo, piloto, reforço, rego, rogo, rolha, seca, secas, soco, sopro, suborno, sufoco, termos, tola, topo, torres, travessa, vede. (Total=48)
w_{-2}, w_{-1}, p_{-3}, p_{-1}, p_{+2}, p_{+3}, $pre_{-3}, pre_{-1}, suf_{-3}, suf_{+1}$: este
suf_{-1}: formas
w_{-1}, p_{-3}, p_{+2}, suf_{-3}, suf_{+3}: forma
w_{-1}, p_{-1}, p_{+1}: rota
w_{-1}, p_{-1}: rotas, toco, torno. (Total=3)
w_{+1}, p_{+1}, suf_{+1}: lobos
w_{+1}, p_{-1}, p_{+1}, suf_{+1}: lobo
w_{+1}, p_{-1}: aperto
w_{+2}: molho
p_{-2}, p_{-1}, pre_{-2}: consolo
p_{-1}, p_{+1}, pre_{-1}, suf_{-2}, suf_{-1}, suf_{+1}: sobre
p_{-1}, p_{+1}, pre_{-1}, suf_{-1}: aborto
p_{-1}, p_{+1}: concerto, contorno, peso. (Total=3)
p_{-1}: almoço, conforto, enterro, flagelo, jogo, metas, pego, rolo.(Total=8)

Tabela 12 – Atributos definitivos selecionados para desambiguação por classificadores para cada HH; ao todo são 33 HH agrupados conforme a tabela. A definição do vetor é mostrado primeiro, seguido de todos os HH que utilizam esta definição. O algoritmo para desambiguação correspondente ao HH e seus parâmetros estão definidos na Tabela 13.

$p_{-3},\ p_{-2},\ p_{-1},\ p_{+1},\ p_{+2},\ p_{+3},\ w_{-1},\ w_{+1},\ (w_{-2},\ w_{-1}),\ (w_{-1},\ w_{+1}),\ (w_{+1},\ w_{+2}),$ $(w_{-3},\ w_{-2},\ w_{-1}),\ (w_{-2},\ w_{-1},\ w_{+1}),\ (w_{-1},\ w_{+1},\ w_{+2}),\ (w_{+1},\ w_{+2},\ w_{+3})$: acordo, apego, apelo, bolas, cerca, cortes, deste, destes, força, interesse, leste, medo, meta, modelo, pega, seco, torre, zelo. (Total=18)
$p_{-3},\ p_{-1},\ p_{+1}$: emprego
$p_{-3},\ p_{-1},\ p_{+2},\ w_{-1}$: selo
$p_{-2},\ p_{-1},\ (w_{-1},\ w_{+1})$: medos
$p_{-2},\ p_{-1},\ p_{+1}$: desses, posto. (Total=2)
p_{-1}: gozo
$p_{-1},\ p_{+1},\ w_{-1}$: conserto, troco. (Total=2)
$p_{-1},\ p_{+1},\ w_{-1},\ w_{+1}$: fora
$p_{-1},\ p_{+2},\ w_{-1},\ w_{+1}$: começo
$p_{-1},\ w_{-1}$: boto, choro, rola. (Total=3)
$p_{+1},\ w_{-1},\ w_{+1},\ (w_{-1},\ w_{+1}),\ (w_{+1},\ w_{+2})$: sede
w_{-1}: relevo

Tabela 13 – Lista de configurações paramétricas dos algoritmos selecionados para desambiguar HH. Os algoritmos de indução são: FURIA, Ridor e JRip. Os classificadores são: AdaBoostM1, J48, NaiveBayes, KStar e RandomForest. Os atributos selecionados para cada HH constam na Tabela 11 e Tabela 12. A Acurácia e MCC correspondem a valores gerados na avaliação 10–CV.

HH	Algoritmo	Parâmetros	Conjunto para treino	Acurácia	MCC
aborto	FURIA	-F 1 -p 0 -N 0.0 -S 1 -O 1 -s 0	aborto, almoço	1,000	1,000
acerto	Ridor	-F 3 -N 2.0 -S 2 -A	acerto	0,971	0,911

Continua na página seguinte

Tabela 13 – *Continuação da página anterior*

HH	Algoritmo	Parâmetros	Conjunto para treino	Acurácia	MCC
acordo	AdaBoostM1	-Q -P 50 -S 1 -I 30 -W weka.classifiers.trees.J48 – -B -C 0.1 -M 0	acordo, boto, pega, reforço	0,977	0,921
adorno	Ridor	-F 4 -N 4.0 -S 5 -A	adorno	1,000	0,000
almoço	Ridor	-F 2 -N 0.0 -S 2	almoço, enterro, esforço, interesses	0,973	0,829
apego	J48	-B -C 0.1 -M 0	almoço, apego, concerto	0,995	0,984
apelo	J48	-B -C 0.3 -M 3	apelo, aperto, cerro, erro	0,992	0,841
aperto	FURIA	-F 3 -p 0 -N 0.0 -S 1 -O 1 -s 0	aborto, aperto, emprego, rego	0,982	0,905
apoio	FURIA	-F 2 -p 0 -N 0.0 -S 1 -O 1 -s 0	apoio, começo	0,993	0,854
apreço	Ridor	-F 4 -N 4.0 -S 5 -A	apreço	1,000	0,000
besta	FURIA	-F 2 -p 1 -E -N 1.0 -S 1 -O 2 -s 0	besta	0,990	0,704
bestas	Ridor	-F 4 -N 4.0 -S 5 -A	bestas	1,000	0,000
bola	Ridor	-F 4 -N 4.0 -S 5 -A	bola	1,000	0,000
bolas	J48	-B -C 0.1 -M 0	bola, bolas, cortes, soco	0,965	0,776
boto	NaiveBayes		boto, olho, peso	0,929	0,855
cerca	AdaBoostM1	-P 50 -S 1 -I 10 -W weka.classifiers.trees.J48 – -B -C 0.2 -M 2	cerca	0,999	0,975
cerro	Ridor	-F 4 -N 4.0 -S 5 -A	cerro	1,000	0,000
choro	NaiveBayes		choro, jogo, secas, soco	0,941	0,807
colher	FURIA	-F 2 -p 1 -E -N 2.0 -S 1 -O 2 -s 0	colher	0,932	0,825
colheres	Ridor	-F 4 -N 4.0 -S 5 -A	colheres	1,000	0,000

Continua na página seguinte

Tabela 13 – *Continuação da página anterior*

HH	Algoritmo	Parâmetros	Conjunto para treino	Acurácia	MCC
começo	KStar	-E -B 5 -M a	adorno, começo	0,975	0,933
concerto	FURIA	-F 1 -p 0 -N 0.0 -S 1 -O 1 -s 0	aborto, almoço, concerto	1,000	1,000
conforto	FURIA	-F 1 -p 0 -N 0.0 -S 1 -O 1 -s 0	aborto, conforto	1,000	1,000
conserto	NaiveBayes		besta, boto, concerto, conserto	0,986	0,915
consolo	FURIA	-F 1 -p 0 -N 2.0 -S 1 -O 1 -s 0	almoço, consolo, força	1,000	1,000
contorno	JRip	-F 5 -E -N 1.0 -S 1 -O 2 -P	contorno	1,000	1,000
controle	Ridor	-F 4 -N 4.0 -S 5 -A	controle	1,000	0,000
cor	Ridor	-F 3 -N 2.0 -S 4	cor	0,976	0,552
coro	Ridor	-F 4 -N 4.0 -S 5 -A	coro	1,000	0,000
corte	Ridor	-F 4 -N 2.0 -S 1	corte	0,943	0,857
cortes	RandomForest	-K 6 -I 1 -S 1	cortes	0,960	0,479
desemprego	Ridor	-F 4 -N 4.0 -S 5 -A	desemprego	1,000	0,000
desespero	JRip	-F 5 -E -N 4.0 -S 1 -O 2 -P	desespero	1,000	1,000
desgosto	Ridor	-F 2 -N 0.0 -S 1	boto, desgosto, força	1,000	1,000
desses	KStar	-B 5 -M a	aborto, apego, aperto, desses	0,998	0,912
deste	AdaBoostM1	-P 50 -S 1 -I 30 -W weka.classifiers.trees.J48 – -B -C 0.25 -M 2	apego, apreço, cerro, deste	0,986	0,876
destes	J48	-B -C 0.1 -M 0	aborto, destes	0,990	0,710
emprego	KStar	-B 5 -M a	conserto, emprego	1,000	1,000
endosso	FURIA	-F 1 -p 0 -N 0.0 -S 1 -O 1 -s 0	acerto, endosso	1,000	1,000
enredo	Ridor	-F 4 -N 4.0 -S 5 -A	enredo	1,000	0,000

Continua na página seguinte

Tabela 13 – *Continuação da página anterior*

HH	Algoritmo	Parâmetros	Conjunto para treino	Acurácia	MCC
enterro	Ridor	-F 2 -N 1.0 -S 5 -A	enterro	1,000	1,000
erro	JRip	-F 2 -N 2.0 -S 1 -O 1	almoço, erro	0,998	0,969
esforço	Ridor	-F 4 -N 4.0 -S 5 -A	esforço	1,000	0,000
esmero	Ridor	-F 4 -N 4.0 -S 5 -A	esmero	1,000	0,000
este	Ridor	-F 4 -N 4.0 -S 5 -A	este	0,999	0,000
flagelo	FURIA	-F 1 -p 0 -N 0.0 -S 1 -O 1 -s 0	aborto, flagelo	1,000	1,000
fora	NaiveBayes		bestas, bola, corte, fora	0,943	0,790
forma	Ridor	-F 4 -N 4.0 -S 5 -A	forma	0,998	0,000
formas	Ridor	-F 4 -N 4.0 -S 5 -A	formas	0,997	0,000
força	NaiveBayes		adorno, apreço, controle, força	0,990	0,284
gelo	Ridor	-F 4 -N 4.0 -S 5 -A	gelo	1,000	0,000
gosto	JRip	-F 5 -N 2.0 -S 1 -O 2	gosto	0,960	0,886
governo	Ridor	-F 4 -N 4.0 -S 5 -A	governo	1,000	0,000
gozo	KStar	-B 5 -M a	acerto, almoço, gozo	0,964	0,615
interesse	J48	-B -C 0.1 -M 0	bestas, cerro, conforto, interesse	0,998	0,967
interesses	Ridor	-F 4 -N 4.0 -S 5 -A	interesses	1,000	0,000
jogo	Ridor	-F 4 -N 0.0 -S 3	acerto, almoço, jogo	0,984	0,838
leste	AdaBoostM1	-Q -P 100 -S 1 -I 30 -W weka.classifiers.trees.J48 – -B -C 0.1 -M 0	leste	0,978	0,921
lobo	JRip	-F 1 -N 0.0 -S 1 -O 1 -P	aborto, apelo, concerto, lobo	0,994	0,927
lobos	JRip	-F 2 -N 2.0 -S 1 -O 2	apreço, lobos	0,955	0,819

Continua na página seguinte

Tabela 13 – *Continuação da página anterior*

HH	Algoritmo	Parâmetros	Conjunto para treino	Acurácia	MCC
medo	RandomForest	-K 6 -I 1 -S 1	almoço, medo	0,995	0,497
medos	J48	-B -C 0.1 -M 0	adorno, esmero, medos	0,947	0,889
meta	J48	-C 0.5 -M 0	colheres, formas, leste, meta	0,992	0,921
metas	Ridor	-F 2 -N 0.0 -S 1	fora, metas	1,000	1,000
modelo	J48	-B -C 0.1 -M 0	cerca, modelo	1,000	1,000
molho	JRip	-F 2 -N 0.0 -S 1 -O 1	apelo, molho	0,991	0,630
namoro	FURIA	-F 3 -p 0 -N 0.0 -S 1 -O 2 -s 0	consolo, desemprego, namoro	0,949	0,835
olho	FURIA	-F 2 -p 0 -E -N 2.0 -S 1 -O 2 -s 0	boto, cerro, flagelo, olho	0,984	0,910
pega	J48	-B -C 0.1 -M 0	colheres, pega, toco	0,995	0,955
pego	JRip	-F 2 -N 0.0 -S 1 -O 1	endosso, pego	0,944	0,882
pela	Ridor	-F 4 -N 4.0 -S 5 -A	pela	1,000	0,000
pelas	Ridor	-F 4 -N 4.0 -S 5 -A	pelas	1,000	0,000
pelo	Ridor	-F 4 -N 4.0 -S 5 -A	pelo	1,000	0,000
peso	JRip	-F 3 -N 0.0 -S 1 -O 1	aborto, desgosto, peso	0,996	0,893
piloto	Ridor	-F 4 -N 4.0 -S 5 -A	piloto	1,000	0,000
posto	NaiveBayes		aborto, adorno, posto	0,990	0,964
reforço	Ridor	-F 4 -N 4.0 -S 5 -A	reforço	1,000	0,000
rego	Ridor	-F 4 -N 4.0 -S 5 -A	rego	1,000	0,000
relevo	AdaBoostM1	-P 100 -S 1 -I 20 -W weka.classifiers.trees.J48 – -B -C 0.2 -M 2	aborto, acerto, apreço, relevo	0,961	0,871
rogo	Ridor	-F 4 -N 4.0 -S 5 -A	rogo	1,000	1,000

Continua na página seguinte

Tabela 13 – *Continuação da página anterior*

HH	Algoritmo	Parâmetros	Conjunto para treino	Acurácia	MCC
rola	RandomForest	-K 6 -I 10 -S 1	bestas, rola, sufoco	0,959	0,882
rolha	Ridor	-F 4 -N 4.0 -S 5 -A	rolha	1,000	0,000
rolo	FURIA	-F 1 -p 0 -N 0.0 -S 1 -O 1 -s 0	aborto, rolo	1,000	1,000
rota	JRip	-F 1 -N 2.0 -S 1 -O 1 -P	rota, tola	0,969	0,903
rotas	JRip	-F 2 -E -N 0.0 -S 1 -O 1	cortes, rotas	0,943	0,800
seca	FURIA	-F 2 -p 0 -N 0.0 -S 1 -O 2 -s 0	concerto, seca, seco	0,970	0,672
secas	Ridor	-F 4 -N 4.0 -S 5 -A	secas	1,000	0,000
seco	AdaBoostM1	-P 100 -S 1 -I 5 -W weka.classifiers.trees.J48 – -B -C 0.2 -M 2	concerto, contorno, seco	0,996	0,864
sede	J48	-B -C 0.5 -M 7	sede	0,839	0,659
selo	NaiveBayes		desespero, selo	1,000	1,000
sobre	Ridor	-F 4 -N 4.0 -S 5 -A	sobre	1,000	0,000
soco	JRip	-F 5 -E -N 4.0 -S 1 -O 2 -P	soco	1,000	1,000
sopro	Ridor	-F 4 -N 4.0 -S 5 -A	sopro	1,000	0,000
suborno	Ridor	-F 4 -N 4.0 -S 5 -A	suborno	1,000	0,000
sufoco	Ridor	-F 4 -N 4.0 -S 5 -A	sufoco	1,000	0,000
termos	Ridor	-F 4 -N 4.0 -S 5 -A	termos	1,000	0,000
toco	JRip	-F 2 -N 0.0 -S 1 -O 1	acerto, acordo, adorno, toco	0,932	0,820
tola	Ridor	-F 4 -N 4.0 -S 5 -A	tola	1,000	0,000
topo	Ridor	-F 2 -N 0.0 -S 2	acerto, apoio, conforto, topo	0,983	0,857
torno	JRip	-F 2 -N 0.0 -S 1 -O 1	bola, boto, lobos, torno	1,000	1,000

Continua na página seguinte

Tabela 13 – *Continuação da página anterior*

HH	Algoritmo	Parâmetros	Conjunto para treino	Acurácia	MCC
torre	J48	-C 0.4 -M 0	torre	0,983	0,886
torres	Ridor	-F 4 -N 4.0 -S 5 -A	torres	1,000	0,000
travessa	Ridor	-F 2 -N 2.0 -S 1	travessa	0,859	0,714
troco	RandomForest	-K 6 -I 20 -S 1	troco	0,951	0,902
vede	Ridor	-F 4 -N 4.0 -S 5 -A	vede	1,000	0,000
zelo	J48	-C 0.4 -M 0	adorno, conserto, contorno, zelo	0,985	0,887

APÊNDICE B – Regras finais induzidas

Listagem B.1 – Listagem das regras induzidas produzidas por diferentes algoritmos de indução para desambiguação de HH. A nomenclatura para os símbolos estão definidos na Tabela 8 – página 57.

```
Regra induzida pelo algoritmo FURIA para desambiguação do HH "aborto":

(POS_left_1 = ART) => pronuncia=0 (CF = 1.0)
(POS_left_1 = PREP+ART) => pronuncia=0 (CF = 1.0)
(POS_left_1 = PREP) => pronuncia=0 (CF = 1.0)
(POS_left_1 = V) => pronuncia=0 (CF = 1.0)
(POS_left_1 = ADJ) => pronuncia=0 (CF = 0.95)
(POS_left_1 = PROADJ) => pronuncia=0 (CF = 0.99)
(POS_left_1 = KC) => pronuncia=0 (CF = 0.99)
(POS_left_1 = =IF=) => pronuncia=0 (CF = 0.9)
(POS_left_1 = PCP) => pronuncia=0 (CF = 0.98)
(POS_left_1 = KS) => pronuncia=0 (CF = 0.97)
(POS_left_1 = PROPESS) => pronuncia=1 (CF = 0.86)
(POS_left_1 = ADV) and (POS_right_1 = ADV) => pronuncia=1 (CF = 0.62)
(POS_left_1 = ADV) and (Prefix_left_1 = a-) => pronuncia=1 (CF = 0.53)
(POS_left_1 = ADV) and (Sufix_left_1 = -ão) => pronuncia=1 (CF = 0.62)

Number of Rules : 14
------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "acerto":

pronuncia = 1  (865.0/688.0)
           Except (POS_left_1 = ART) => pronuncia = 0  (210.0/0.0) [94.0/1.0]
           Except (POS_left_1 = PREP) and (POS_left_2 = N) => pronuncia = 0
               (65.0/0.0) [48.0/0.0]
           Except (POS_left_1 = PREP+ART) => pronuncia = 0  (45.0/0.0)
               [21.0/0.0]
           Except (POS_left_1 = PROADJ) and (Sufix_left_1 = -eu) => pronuncia =
               0  (14.0/0.0) [3.0/0.0]
           Except (POS_right_1 = PREP) and (Word_right_1 = de) => pronuncia = 0
                (27.0/0.0) [6.0/1.0]
           Except (POS_left_1 = V) => pronuncia = 0  (26.0/0.0) [15.0/2.0]
           Except (POS_left_1 = ADJ) and (Sufix_left_2 = -eu) => pronuncia = 0
               (7.0/0.0) [2.0/0.0]
           Except (POS_left_1 = PREP) and (Sufix_left_1 = -m) => pronuncia = 0
               (16.0/0.0) [5.0/0.0]
           Except (POS_left_1 = ADJ) and (POS_left_2 = ART) => pronuncia = 0
               (10.0/0.0) [8.0/1.0]
           Except (POS_left_1 = PROADJ) => pronuncia = 0  (15.0/0.0) [9.0/1.0]
           Except (POS_left_2 = PREP) => pronuncia = 0  (12.0/1.0) [1.0/0.0]
           Except (POS_left_1 = PREP) => pronuncia = 0  (8.0/0.0) [3.0/1.0]
           Except (POS_left_2 = N) and (POS_left_1 = KC) and (POS_left_3 = PREP)
                => pronuncia = 0  (3.0/0.0) [1.0/0.0]
           Except (POS_left_1 = ADJ) => pronuncia = 0  (6.0/2.0) [1.0/0.0]
           Except (POS_left_1 = N) => pronuncia = 0  (6.0/3.0) [1.0/0.0]

Total number of rules (incl. the default rule): 16
```

```
40 |------------------------------------------------------------------------------
41 |Regra induzida pelo algoritmo Ridor para desambiguação do HH "adorno":
42 |
43 |pronuncia = 0  (26.0/0.0)
44 |
45 |Total number of rules (incl. the default rule): 1
46 |------------------------------------------------------------------------------
47 |Regra induzida pelo algoritmo Ridor para desambiguação do HH "almoço":
48 |
49 |pronuncia = 1  (669.0/638.0)
50 |           Except (POS_left_1 = ART) => pronuncia = 0  (88.0/0.0) [88.0/0.0]
51 |           Except (POS_left_1 = PREP+ART) => pronuncia = 0  (78.0/0.0)
   |               [85.0/0.0]
52 |           Except (POS_left_1 = PREP) => pronuncia = 0  (54.0/0.0) [64.0/0.0]
53 |           Except (POS_left_1 = PROADJ) => pronuncia = 0  (31.0/0.0) [40.0/0.0]
54 |           Except (POS_left_1 = ADJ) => pronuncia = 0  (28.0/1.0) [20.0/0.0]
55 |           Except (POS_left_1 = V) => pronuncia = 0  (16.0/0.0) [13.0/0.0]
56 |           Except (POS_left_1 = KC) => pronuncia = 0  (6.0/0.0) [5.0/0.0]
57 |           Except (POS_left_1 = =IF=) => pronuncia = 0  (7.0/0.0) [4.0/1.0]
58 |
59 |Total number of rules (incl. the default rule): 9
60 |------------------------------------------------------------------------------
61 |Regra induzida pelo algoritmo FURIA para desambiguação do HH "aperto":
62 |
63 |(POS_left_1 = ART) => pronuncia=0 (CF = 1.0)
64 |(POS_left_1 = PREP+ART) => pronuncia=0 (CF = 1.0)
65 |(POS_left_1 = PREP) => pronuncia=0 (CF = 1.0)
66 |(POS_left_1 = ADJ) => pronuncia=0 (CF = 0.98)
67 |(POS_left_1 = NPROP) => pronuncia=0 (CF = 1.0)
68 |(POS_left_1 = V) => pronuncia=0 (CF = 1.0)
69 |(POS_left_1 = PROADJ) => pronuncia=0 (CF = 1.0)
70 |(POS_left_1 = KC) => pronuncia=0 (CF = 0.93)
71 |(POS_left_1 = N) => pronuncia=0 (CF = 0.94)
72 |(Word_right_1 = no) => pronuncia=0 (CF = 1.0)
73 |(POS_left_1 = PCP) => pronuncia=0 (CF = 0.99)
74 |(POS_left_1 = PREP+PROADJ) => pronuncia=0 (CF = 0.99)
75 |(Word_right_1 = é) => pronuncia=0 (CF = 0.99)
76 |(Word_right_1 = monetário) => pronuncia=0 (CF = 0.99)
77 |(Word_right_1 = de) => pronuncia=0 (CF = 1.0)
78 |(Word_right_1 = =ff=) => pronuncia=0 (CF = 1.0)
79 |(Word_right_1 = horrível) => pronuncia=0 (CF = 0.97)
80 |(Word_right_1 = gil) => pronuncia=0 (CF = 0.99)
81 |(POS_left_1 = PROPESS) => pronuncia=1 (CF = 0.86)
82 |(POS_left_1 = KS) and (Word_right_1 = a) => pronuncia=1 (CF = 0.52)
83 |(POS_left_1 = ADV) => pronuncia=1 (CF = 0.58)
84 |(POS_left_1 = KS) and (Word_right_1 = flash) => pronuncia=1 (CF = 0.52)
85 |(POS_left_1 = =IF=) and (Word_right_1 = o) => pronuncia=1 (CF = 0.36)
86 |(POS_left_1 = ADV-KS) => pronuncia=1 (CF = 0.52)
87 |(POS_left_1 = KS) and (Word_right_1 = parece) => pronuncia=1 (CF = 0.36)
88 |(POS_left_1 = =IF=) and (Word_right_1 = a) => pronuncia=1 (CF = 0.36)
89 |(Word_right_1 = esta) => pronuncia=1 (CF = 0.36)
90 |(Word_right_1 = meu) => pronuncia=1 (CF = 0.36)
91 |(POS_left_1 = N) and (Word_right_1 = o) => pronuncia=1 (CF = 0.36)
92 |(Word_right_1 = então) => pronuncia=1 (CF = 0.36)
93 |(Word_right_1 = ele) and (POS_left_1 = ADJ) => pronuncia=1 (CF = 0.36)
94 |(Word_right_1 = 1) and (POS_left_1 = =IF=) => pronuncia=1 (CF = 0.36)
95 |(Word_right_1 = para) and (POS_left_1 = N) => pronuncia=1 (CF = 0.36)
```

```
96
97  Number of Rules : 33
98  ------------------------------------------------------------------------------
99  Regra induzida pelo algoritmo FURIA para desambiguação do HH "apoio":
100
101 (POS_right_1 = PREP+ART) => pronuncia=0 (CF = 1.0)
102 (POS_left_1 = ART) => pronuncia=0 (CF = 1.0)
103 (POS_right_2 = N) and (POS_right_1 = PREP) and (POS_left_2 = N) => pronuncia=0 (
        CF = 0.99)
104 (POS_left_1 = PREP) => pronuncia=0 (CF = 1.0)
105 (POS_left_1 = PREP+ART) => pronuncia=0 (CF = 1.0)
106 (POS_left_1 = V) => pronuncia=0 (CF = 0.94)
107 (POS_right_2 = N) and (POS_right_1 = PREP) => pronuncia=0 (CF = 0.98)
108 (POS_left_1 = PROADJ) => pronuncia=0 (CF = 0.93)
109 (POS_left_1 = ADJ) => pronuncia=0 (CF = 0.95)
110 (POS_right_1 = ADJ) => pronuncia=0 (CF = 0.99)
111 (Word_right_1 = a) and (POS_right_2 = V) => pronuncia=1 (CF = 0.96)
112 (Word_left_2 = =if=) and (Word_left_1 = =if=) => pronuncia=1 (CF = 0.79)
113 (Word_left_1 = eu) => pronuncia=1 (CF = 0.92)
114 (POS_left_1 = ADV) and (POS_right_3 = N) => pronuncia=1 (CF = 0.72)
115 (POS_right_1 = ADV) and (POS_right_2 = PREP) => pronuncia=1 (CF = 0.72)
116 (Word_left_1 = que) => pronuncia=1 (CF = 0.85)
117 (Word_left_3 = entenda) => pronuncia=1 (CF = 0.43)
118 (Word_left_1 = só) => pronuncia=1 (CF = 0.43)
119 (POS_left_1 = N) and (Word_right_1 = a) => pronuncia=1 (CF = 0.83)
120 (Word_left_3 = acha) => pronuncia=1 (CF = 0.43)
121 (Word_right_1 = desde) => pronuncia=1 (CF = 0.43)
122
123 Number of Rules : 21
124 ------------------------------------------------------------------------------
125 Regra induzida pelo algoritmo Ridor para desambiguação do HH "apreço":
126
127 pronuncia = 0  (138.0/0.0)
128
129 Total number of rules (incl. the default rule): 1
130 ------------------------------------------------------------------------------
131 Regra induzida pelo algoritmo FURIA para desambiguação do HH "besta":
132
133 (Word_right_3 = =ff=) => pronuncia=0 (CF = 1.0)
134 (POS_left_1 = ART) => pronuncia=0 (CF = 0.98)
135 (POS_left_1 = N) => pronuncia=0 (CF = 1.0)
136 (POS_left_1 = PREP+ART) => pronuncia=0 (CF = 1.0)
137 (POS_left_1 = V) => pronuncia=0 (CF = 1.0)
138 (POS_right_1 = V) => pronuncia=0 (CF = 1.0)
139 (POS_left_2 = V) => pronuncia=0 (CF = 0.98)
140 (POS_left_1 = ADV) => pronuncia=0 (CF = 1.0)
141 (Word_left_2 = tiro) => pronuncia=1 (CF = 0.51)
142 (Word_left_3 = bora) => pronuncia=1 (CF = 0.35)
143 (Word_left_2 = arcos) => pronuncia=1 (CF = 0.35)
144
145 Number of Rules : 11
146 ------------------------------------------------------------------------------
147 Regra induzida pelo algoritmo Ridor para desambiguação do HH "bestas":
148
149 pronuncia = 0  (33.0/0.0)
150
151 Total number of rules (incl. the default rule): 1
```

```
152 ------------------------------------------------------------------------------
153 Regra induzida pelo algoritmo Ridor para desambiguação do HH "bola":
154
155 pronuncia = 1  (442.0/0.0)
156
157 Total number of rules (incl. the default rule): 1
158 ------------------------------------------------------------------------------
159 Regra induzida pelo algoritmo Ridor para desambiguação do HH "cerro":
160
161 pronuncia = 0  (67.0/0.0)
162
163 Total number of rules (incl. the default rule): 1
164 ------------------------------------------------------------------------------
165 Regra induzida pelo algoritmo FURIA para desambiguação do HH "colher":
166
167 (POS_left_1 = V) => pronuncia=0 (CF = 0.95)
168 (Sufix_right_1 = -s) => pronuncia=0 (CF = 0.98)
169 (POS_left_1 = PREP) and (Sufix_left_1 = -a) => pronuncia=0 (CF = 0.99)
170 (POS_right_1 = ART) => pronuncia=0 (CF = 0.97)
171 (POS_left_1 = PREP) and (POS_right_2 = PREP) => pronuncia=0 (CF = 0.89)
172 (POS_left_1 = KC) => pronuncia=0 (CF = 0.91)
173 (Word_left_1 = de) and (POS_left_3 = PREP+ART) => pronuncia=0 (CF = 0.95)
174 (POS_right_1 = ADV) => pronuncia=0 (CF = 0.95)
175 (Word_left_2 = para) => pronuncia=0 (CF = 0.96)
176 (POS_right_1 = PROADJ) => pronuncia=0 (CF = 0.95)
177 (Prefix_left_2 = de-) => pronuncia=0 (CF = 0.88)
178 (Sufix_right_1 = -es) => pronuncia=0 (CF = 0.95)
179 (Word_right_2 = que) => pronuncia=0 (CF = 0.96)
180 (Word_left_1 = de) and (Word_right_1 = 1) => pronuncia=0 (CF = 0.91)
181 (POS_left_1 = ADJ) and (POS_left_2 = V) => pronuncia=0 (CF = 0.93)
182 (Word_left_1 = uma) => pronuncia=1 (CF = 0.98)
183 (Word_right_1 = de) => pronuncia=1 (CF = 0.97)
184 (Word_left_1 = a) and (POS_left_1 = PROPESS) => pronuncia=1 (CF = 0.79)
185 (Word_right_1 = =ff=) and (POS_left_3 = PREP) => pronuncia=1 (CF = 0.69)
186 (Word_left_1 = 1) => pronuncia=1 (CF = 0.89)
187 (Prefix_left_2 = met-) => pronuncia=1 (CF = 0.85)
188 (Word_left_1 = de) and (Prefix_left_3 = e-) => pronuncia=1 (CF = 0.69)
189 (POS_right_1 = V) and (POS_left_1 = PREP) => pronuncia=1 (CF = 0.73)
190 (Word_left_1 = com) => pronuncia=1 (CF = 0.84)
191 (POS_left_1 = PROADJ) => pronuncia=1 (CF = 0.82)
192 (POS_left_1 = PREP+ART) => pronuncia=1 (CF = 0.62)
193 (Word_left_2 = garfo) => pronuncia=1 (CF = 0.63)
194 (Sufix_right_2 = -eira) => pronuncia=1 (CF = 0.71)
195 (Word_right_1 = chá) => pronuncia=1 (CF = 0.79)
196
197 Number of Rules : 29
198 ------------------------------------------------------------------------------
199 Regra induzida pelo algoritmo Ridor para desambiguação do HH "colheres":
200
201 pronuncia = 1  (47.0/0.0)
202
203 Total number of rules (incl. the default rule): 1
204 ------------------------------------------------------------------------------
205 Regra induzida pelo algoritmo FURIA para desambiguação do HH "concerto":
206
207  => pronuncia=0 (CF = 0.0)
208 (POS_left_1 = PROPESS) => pronuncia=1 (CF = 0.87)
```

```
(POS_left_1 = ADV) => pronuncia=1 (CF = 0.65)

Number of Rules : 3
------------------------------------------------------------------------------
Regra induzida pelo algoritmo FURIA para desambiguação do HH "conforto":

 => pronuncia=0 (CF = 0.0)
(POS_left_1 = PROPESS) => pronuncia=1 (CF = 0.5)

Number of Rules : 2
------------------------------------------------------------------------------
Regra induzida pelo algoritmo FURIA para desambiguação do HH "consolo":

 => pronuncia=0 (CF = 0.0)
(POS_left_1 = PROPESS) => pronuncia=1 (CF = 0.71)
(POS_left_1 = ADV) => pronuncia=1 (CF = 0.62)

Number of Rules : 3
------------------------------------------------------------------------------
Regra induzida pelo algoritmo JRip para desambiguação do HH "contorno":

(POS_right_1 = ART) => pronuncia=1 (3.0/0.0)
(POS_left_1 = PROPESS) => pronuncia=1 (2.0/0.0)
 => pronuncia=0 (79.0/0.0)

Number of Rules : 3
------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "controle":
RIpple DOwn Rule Learner(Ridor) rules
--------------------------------------

pronuncia = 0  (215.0/0.0)

Total number of rules (incl. the default rule): 1
------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "cor":

pronuncia = 0  (450.0/0.0)

Total number of rules (incl. the default rule): 1
------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "coro":

pronuncia = 0  (38.0/0.0)

Total number of rules (incl. the default rule): 1
------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "corte":

pronuncia = 1  (277.0/203.0)
           Except (POS_left_1 = PREP+ART) and (Sufix_left_1 = -a) and (
               Word_left_1 = da) => pronuncia = 0  (32.0/0.0) [13.0/0.0]
           Except (Word_left_1 = a) => pronuncia = 0  (50.0/0.0) [12.0/0.0]
           Except (POS_left_1 = PREP+ART) and (POS_left_2 = V) => pronuncia = 0
                (15.0/0.0) [7.0/0.0]
           Except (Sufix_left_1 = -a) => pronuncia = 0  (38.0/2.0) [12.0/2.0]
```

```
263             Except (POS_left_1 = PREP+ART) and (Word_left_1 = à) => pronuncia = 0
                     (5.0/0.0) [4.0/0.0]
264             Except (Sufix_left_1 = -ela) => pronuncia = 0  (5.0/0.0) [2.0/0.0]
265
266 Total number of rules (incl. the default rule): 7
267 ------------------------------------------------------------------------------
268 Regra induzida pelo algoritmo Ridor para desambiguação do HH "desemprego":
269
270 pronuncia = 0  (175.0/0.0)
271
272 Total number of rules (incl. the default rule): 1
273 ------------------------------------------------------------------------------
274 Regra induzida pelo algoritmo JRip para desambiguação do HH "desespero":
275
276 (Word_left_1 = me) => pronuncia=1 (7.0/0.0)
277  => pronuncia=0 (560.0/0.0)
278
279 Number of Rules : 2
280 ------------------------------------------------------------------------------
281 Regra induzida pelo algoritmo Ridor para desambiguação do HH "desgosto":
282
283 pronuncia = 1  (638.0/594.0)
284             Except (POS_left_1 = PREP) => pronuncia = 0  (70.0/0.0) [50.0/0.0]
285             Except (POS_left_1 = ART) => pronuncia = 0  (79.0/0.0) [64.0/0.0]
286             Except (Sufix_left_1 = -a) => pronuncia = 0  (71.0/0.0) [57.0/0.0]
287             Except (POS_left_1 = PROADJ) => pronuncia = 0  (27.0/0.0) [22.0/0.0]
288             Except (POS_left_1 = PREP+ART) => pronuncia = 0  (25.0/0.0)
                    [18.0/0.0]
289             Except (POS_left_1 = V) and (Sufix_left_2 = -ão) => pronuncia = 0
                    (8.0/0.0) [4.0/0.0]
290             Except (POS_right_1 = KC) => pronuncia = 0  (11.0/0.0) [10.0/0.0]
291             Except (Word_right_1 = =ff=) and (POS_left_1 = ADJ) => pronuncia = 0
                     (7.0/0.0) [1.0/0.0]
292             Except (POS_left_1 = V) and (POS_right_1 = PREP) => pronuncia = 0
                    (5.0/0.0) [2.0/0.0]
293             Except (POS_left_2 = N) and (POS_left_1 = KC) => pronuncia = 0
                    (8.0/0.0) [6.0/1.0]
294             Except (POS_left_1 = V) => pronuncia = 0  (8.0/0.0) [10.0/3.0]
295             Except (POS_left_1 = ADJ) and (POS_left_3 = V) => pronuncia = 0
                    (4.0/0.0) [2.0/0.0]
296             Except (POS_left_2 = NPROP) => pronuncia = 0  (4.0/0.0) [1.0/0.0]
297             Except (POS_left_1 = N) => pronuncia = 0  (5.0/0.0) [3.0/1.0]
298             Except (POS_left_1 = ADJ) => pronuncia = 0  (3.0/0.0) [3.0/1.0]
299
300 Total number of rules (incl. the default rule): 16
301 ------------------------------------------------------------------------------
302 Regra induzida pelo algoritmo FURIA para desambiguação do HH "endosso":
303
304 (POS_left_1 = ART) and (Word_left_1 = o) => pronuncia=0 (CF = 1.0)
305 (POS_left_1 = PREP) and (POS_left_2 = N) => pronuncia=0 (CF = 1.0)
306 (Sufix_left_1 = -m) and (Word_left_1 = um) => pronuncia=0 (CF = 1.0)
307 (Sufix_left_1 = -o) and (POS_left_1 = PREP+ART) => pronuncia=0 (CF = 0.99)
308 (POS_left_1 = PROADJ) and (Sufix_left_1 = -eu) => pronuncia=0 (CF = 0.98)
309 (POS_left_1 = V) and (POS_left_2 = N) => pronuncia=0 (CF = 0.98)
310 (POS_left_1 = ADJ) and (Sufix_left_1 = -or) => pronuncia=0 (CF = 0.97)
311 (POS_left_1 = PREP) and (Sufix_left_1 = -m) => pronuncia=0 (CF = 0.99)
```

```
(POS_right_1 = PREP) and (Word_right_1 = de) and (POS_right_2 = N) => pronuncia
    =0 (CF = 1.0)
(POS_left_1 = ADJ) and (POS_right_1 = PREP) => pronuncia=0 (CF = 0.97)
(POS_left_1 = PROADJ) and (Word_left_2 = com) => pronuncia=0 (CF = 0.96)
(POS_left_1 = ADJ) and (Sufix_left_1 = -o) => pronuncia=0 (CF = 0.96)
(POS_left_1 = V) and (POS_left_2 = ADV) => pronuncia=0 (CF = 0.97)
(POS_left_1 = PROADJ) and (Word_right_3 = =ff=) => pronuncia=0 (CF = 0.97)
(POS_left_1 = ADJ) and (POS_left_2 = ART) => pronuncia=0 (CF = 0.94)
(POS_left_1 = PREP) and (POS_left_3 = N) => pronuncia=0 (CF = 0.98)
(POS_left_1 = V) and (POS_right_3 = V) => pronuncia=0 (CF = 0.96)
(POS_left_1 = PROADJ) and (Word_left_3 = com) => pronuncia=0 (CF = 0.92)
(POS_left_1 = PROADJ) and (POS_right_1 = V) => pronuncia=0 (CF = 0.96)
(POS_left_1 = V) and (POS_left_2 = NPROP) => pronuncia=0 (CF = 0.92)
(POS_left_2 = N) and (POS_left_1 = KC) and (POS_left_3 = PREP) => pronuncia=0 (
    CF = 0.95)
(POS_left_2 = PREP) and (POS_left_3 = V) => pronuncia=0 (CF = 0.98)
(POS_left_1 = N) and (POS_left_2 = N) => pronuncia=0 (CF = 0.93)
(Word_left_1 = 1) => pronuncia=0 (CF = 0.93)
(Word_right_1 = entre) => pronuncia=0 (CF = 0.98)
(POS_left_1 = KC) and (Word_left_2 = erro) => pronuncia=0 (CF = 0.9)
(Word_left_2 = o) => pronuncia=0 (CF = 0.98)
(Prefix_right_2 = in-) => pronuncia=0 (CF = 0.94)
(Sufix_left_1 = -em) => pronuncia=0 (CF = 0.95)
(Sufix_right_2 = -ão) and (POS_right_1 = KC) => pronuncia=0 (CF = 0.93)
(POS_left_1 = PROPESS) => pronuncia=1 (CF = 0.98)
(POS_left_1 = ADV) => pronuncia=1 (CF = 0.95)
(POS_right_1 = ART) and (POS_left_1 = KS) => pronuncia=1 (CF = 0.8)
(POS_left_1 = KC) and (POS_left_3 = V) => pronuncia=1 (CF = 0.82)
(POS_left_1 = KS) and (Sufix_right_2 = -a) => pronuncia=1 (CF = 0.82)
(Word_left_1 = =if=) and (POS_right_3 = N) => pronuncia=1 (CF = 0.8)
(POS_left_1 = KS) and (Word_right_2 = =ff=) => pronuncia=1 (CF = 0.68)
(POS_left_1 = PDEN) => pronuncia=1 (CF = 0.68)
(Sufix_right_1 = -s) and (Word_left_3 = os) => pronuncia=1 (CF = 0.6)
(POS_left_1 = KC) and (Sufix_right_2 = -u) => pronuncia=1 (CF = 0.6)
(POS_left_1 = KS) and (POS_right_3 = N) => pronuncia=1 (CF = 0.73)
(Word_left_3 = maior) => pronuncia=1 (CF = 0.47)
(Word_left_3 = ver) => pronuncia=1 (CF = 0.6)
(Word_left_2 = nem) => pronuncia=1 (CF = 0.47)

Number of Rules : 44
------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "enredo":

pronuncia = 0  (40.0/0.0)

Total number of rules (incl. the default rule): 1
------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "enterro":

pronuncia = 1  (49.0/45.0)
           Except (POS_left_1 = PREP+ART) => pronuncia = 0  (10.0/0.0) [4.0/0.0]
           Except (POS_left_1 = ART) => pronuncia = 0  (9.0/0.0) [5.0/0.0]
           Except (POS_left_1 = PROADJ) => pronuncia = 0  (5.0/0.0) [2.0/0.0]
           Except (POS_left_1 = PREP) => pronuncia = 0  (3.0/0.0) [5.0/0.0]

Total number of rules (incl. the default rule): 5
------------------------------------------------------------------------------
```

```
Regra induzida pelo algoritmo JRip para desambiguação do HH "erro":

(POS_left_1 = ADV) => pronuncia=1 (21.0/3.0)
(POS_left_1 = PROPESS) => pronuncia=1 (17.0/0.0)
(POS_left_1 = N) and (Sufix_right_2 = -o) => pronuncia=1 (3.0/0.0)
 => pronuncia=0 (909.0/6.0)

Number of Rules : 4
------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "esforço":

pronuncia = 0  (137.0/0.0)

Total number of rules (incl. the default rule): 1
------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "esmero":

pronuncia = 0  (24.0/0.0)

Total number of rules (incl. the default rule): 1
------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "este":

pronuncia = 0  (1769.0/0.0)

Total number of rules (incl. the default rule): 1
------------------------------------------------------------------------------
Regra induzida pelo algoritmo FURIA para desambiguação do HH "flagelo":

 => pronuncia=0 (CF = 0.0)
(POS_left_1 = PROPESS) => pronuncia=1 (CF = 0.5)

Number of Rules : 2
------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "forma":

pronuncia = 1  (582.0/0.0)

Total number of rules (incl. the default rule): 1
------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "formas":

pronuncia = 1  (336.0/0.0)

Total number of rules (incl. the default rule): 1
------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "gelo":

pronuncia = 0  (342.0/0.0)

Total number of rules (incl. the default rule): 1
------------------------------------------------------------------------------
Regra induzida pelo algoritmo JRip para desambiguação do HH "gosto":

(Word_left_3 = =if=) and (Word_left_1 = =if=) => pronuncia=1 (40.0/0.0)
(Word_left_1 = não) => pronuncia=1 (58.0/0.0)
(POS_left_1 = PROPESS) => pronuncia=1 (36.0/1.0)
```

```
(Word_left_3 = =if=) and (POS_right_2 = N) => pronuncia=1 (13.0/4.0)
(Word_right_1 = muito) => pronuncia=1 (12.0/1.0)
(Word_left_1 = que) and (POS_left_2 = N) => pronuncia=1 (5.0/0.0)
(Word_right_1 = de) and (Word_right_2 = ti) => pronuncia=1 (4.0/0.0)
 => pronuncia=0 (623.0/24.0)

Number of Rules : 8
-------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "governo":

pronuncia = 0  (1171.0/0.0)

Total number of rules (incl. the default rule): 1
-------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "interesses":

pronuncia = 0  (154.0/0.0)

Total number of rules (incl. the default rule): 1
-------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "jogo":

pronuncia = 1  (1693.0/1463.0)
           Except (POS_left_1 = ART) => pronuncia = 0  (439.0/0.0) [139.0/1.0]
           Except (POS_left_1 = PREP+ART) => pronuncia = 0  (200.0/0.0)
               [72.0/0.0]
           Except (POS_left_1 = PREP) => pronuncia = 0  (197.0/0.0) [63.0/1.0]
           Except (POS_left_1 = ADJ) => pronuncia = 0  (77.0/3.0) [28.0/1.0]
           Except (POS_left_1 = PROADJ) => pronuncia = 0  (69.0/1.0) [27.0/0.0]
           Except (POS_left_1 = V) => pronuncia = 0  (58.0/2.0) [22.0/1.0]
           Except (POS_left_1 = PREP+PROADJ) => pronuncia = 0  (14.0/0.0)
               [4.0/0.0]
           Except (POS_left_1 = NUM) => pronuncia = 0  (8.0/0.0) [2.0/0.0]
           Except (POS_left_1 = =IF=) => pronuncia = 0  (23.0/10.0) [7.0/3.0]
           Except (POS_left_1 = N) => pronuncia = 0  (14.0/5.0) [4.0/2.0]
           Except (POS_left_1 = KC) => pronuncia = 0  (18.0/8.0) [7.0/5.0]

Total number of rules (incl. the default rule): 12
-------------------------------------------------------------------------------
Regra induzida pelo algoritmo JRip para desambiguação do HH "lobo":

(POS_left_1 = PROPESS) and (Sufix_right_1 = -a) => pronuncia=1 (6.0/0.0)
(Sufix_right_1 = -al) and (Word_right_1 = frontal) and (POS_left_1 = PREP+ART)
    => pronuncia=1 (6.0/0.0)
(POS_left_1 = PROPESS) and (Word_right_1 = =ff=) => pronuncia=1 (3.0/0.0)
(Sufix_right_1 = -or) and (POS_right_1 = ADJ) and (Word_right_1 = posterior) =>
    pronuncia=1 (5.0/0.0)
(Sufix_right_1 = -al) and (Word_right_1 = temporal) => pronuncia=1 (5.0/0.0)
(POS_left_1 = PROPESS) and (POS_right_1 = ADV) => pronuncia=1 (2.0/0.0)
(POS_left_1 = PROPESS) and (POS_right_1 = ART) => pronuncia=1 (2.0/0.0)
(Sufix_right_1 = -al) and (Word_right_1 = frontal) and (POS_left_1 = ART) =>
    pronuncia=1 (4.0/0.0)
(Word_right_1 = anterior) => pronuncia=1 (5.0/0.0)
(POS_left_1 = PROPESS) and (POS_right_1 = PDEN) => pronuncia=1 (2.0/0.0)
(POS_left_1 = ADV) and (POS_right_1 = PREP) and (Word_right_1 = em) => pronuncia
    =1 (1.0/0.0)
(Sufix_right_1 = -al) and (Word_right_1 = parietal) => pronuncia=1 (3.0/0.0)
```

```
475 (POS_left_1 = =IF=) and (Sufix_right_1 = -u) and (Word_right_1 = eu) =>
        pronuncia=1 (1.0/0.0)
476 (POS_left_1 = ADV) and (Word_right_1 = do) => pronuncia=1 (1.0/0.0)
477 (Word_right_1 = occipital) => pronuncia=1 (3.0/0.0)
478 (POS_left_1 = =IF=) and (Word_right_1 = para) => pronuncia=1 (1.0/0.0)
479 (POS_left_1 = PROPESS) and (Word_right_1 = eu) => pronuncia=1 (1.0/0.0)
480 (POS_left_1 = ADV) and (Word_right_1 = pra) => pronuncia=1 (1.0/0.0)
481 (POS_left_1 = =IF=) and (Word_right_1 = aos) => pronuncia=1 (1.0/0.0)
482 (Word_right_1 = superior) => pronuncia=1 (2.0/0.0)
483 (POS_left_1 = PROPESS) and (Word_right_1 = pro) => pronuncia=1 (1.0/0.0)
484 (POS_left_1 = =IF=) and (Word_right_1 = suplicou) => pronuncia=1 (1.0/0.0)
485 (POS_left_1 = ADV) and (Word_right_1 = só) => pronuncia=1 (1.0/0.0)
486  => pronuncia=0 (2525.0/13.0)
487
488 Number of Rules : 24
489 ------------------------------------------------------------------------------
490 Regra induzida pelo algoritmo JRip para desambiguação do HH "lobos":
491
492 (POS_right_1 = ADJ) and (Sufix_right_1 = -is) => pronuncia=1 (17.0/1.0)
493 (Sufix_right_1 = -al) => pronuncia=1 (5.0/1.0)
494  => pronuncia=0 (274.0/2.0)
495
496 Number of Rules : 3
497 ------------------------------------------------------------------------------
498 Regra induzida pelo algoritmo Ridor para desambiguação do HH "metas":
499
500 pronuncia = 1  (1149.0/183.0)
501            Except (POS_left_1 = KS) => pronuncia = 0  (22.0/3.0) [10.0/1.0]
502            Except (POS_left_1 = PRO-KS) => pronuncia = 0  (15.0/1.0) [13.0/2.0]
503            Except (POS_left_1 = PROPESS) => pronuncia = 0  (11.0/4.0) [9.0/1.0]
504
505 Total number of rules (incl. the default rule): 4
506 ------------------------------------------------------------------------------
507 Regra induzida pelo algoritmo JRip para desambiguação do HH "molho":
508
509 (Word_right_2 = chaves) => pronuncia=1 (2.0/0.0)
510 (Word_right_2 = barbas) => pronuncia=1 (1.0/0.0)
511 (Word_right_2 = cabos) => pronuncia=1 (1.0/0.0)
512 (Word_right_2 = isso) => pronuncia=1 (2.0/0.0)
513 (Word_right_2 = ossos) => pronuncia=1 (1.0/0.0)
514 (Word_right_2 = ele) => pronuncia=1 (1.0/0.0)
515 (Word_right_2 = todo) => pronuncia=1 (1.0/0.0)
516 (Word_right_2 = césar) => pronuncia=1 (1.0/0.0)
517  => pronuncia=0 (1396.0/26.0)
518
519 Number of Rules : 9
520 ------------------------------------------------------------------------------
521 Regra induzida pelo algoritmo FURIA para desambiguação do HH "namoro":
522
523 (POS_left_1 = ART) => pronuncia=0 (CF = 1.0)
524 (POS_left_1 = PREP) => pronuncia=0 (CF = 1.0)
525 (Sufix_left_1 = -o) => pronuncia=0 (CF = 0.98)
526 (POS_left_1 = PROADJ) => pronuncia=0 (CF = 0.97)
527 (POS_left_1 = V) => pronuncia=0 (CF = 0.96)
528 (POS_left_1 = N) => pronuncia=0 (CF = 0.99)
529 (Word_left_1 = =if=) => pronuncia=0 (CF = 0.91)
530 (POS_left_1 = KC) => pronuncia=0 (CF = 0.86)
```

```
(POS_left_1 = ADJ) => pronuncia=0 (CF = 0.94)
(POS_left_1 = PCP) => pronuncia=0 (CF = 0.97)
(Sufix_left_2 = -a) => pronuncia=0 (CF = 0.97)
(POS_left_1 = PROPESS) => pronuncia=1 (CF = 0.85)
(Word_left_1 = não) => pronuncia=1 (CF = 0.79)
(Word_left_2 = eu) => pronuncia=1 (CF = 0.79)
(POS_left_1 = KS) => pronuncia=1 (CF = 0.52)
(Word_left_1 = e) and (Word_right_1 = um) => pronuncia=1 (CF = 0.54)
(Word_left_3 = sou) => pronuncia=1 (CF = 0.38)
(Word_left_3 = cara) => pronuncia=1 (CF = 0.38)
(Word_left_2 = gata) => pronuncia=1 (CF = 0.38)

Number of Rules : 19
------------------------------------------------------------------------------
Regra induzida pelo algoritmo FURIA para desambiguação do HH "olho":

(POS_left_1 = ART) => pronuncia=0 (CF = 1.0)
(POS_left_1 = PREP) => pronuncia=0 (CF = 1.0)
(POS_left_1 = PREP+ART) => pronuncia=0 (CF = 1.0)
(POS_left_1 = PROADJ) => pronuncia=0 (CF = 0.99)
(POS_right_1 = ADJ) => pronuncia=0 (CF = 0.99)
(POS_left_1 = ADJ) => pronuncia=0 (CF = 1.0)
(Sufix_right_1 = -u) => pronuncia=0 (CF = 0.99)
(Word_right_1 = d) => pronuncia=0 (CF = 1.0)
(POS_left_1 = V) => pronuncia=0 (CF = 0.84)
(POS_left_2 = ART) => pronuncia=0 (CF = 1.0)
(Sufix_left_1 = -m) => pronuncia=0 (CF = 1.0)
(POS_left_1 = N) and (POS_left_3 = N) => pronuncia=0 (CF = 0.97)
(POS_left_2 = NPROP) => pronuncia=0 (CF = 0.99)
(POS_right_1 = PCP) => pronuncia=0 (CF = 1.0)
(Word_right_1 = de) => pronuncia=0 (CF = 1.0)
(POS_left_2 = ADJ) => pronuncia=0 (CF = 0.99)
(Sufix_right_1 = -or) => pronuncia=0 (CF = 0.99)
(POS_right_1 = PREP+PROPESS) => pronuncia=0 (CF = 0.98)
(Sufix_left_2 = -is) => pronuncia=0 (CF = 0.97)
(POS_left_1 = PROPESS) => pronuncia=1 (CF = 0.97)
(Word_right_1 = para) and (Word_left_2 = =if=) => pronuncia=1 (CF = 0.94)
(POS_left_1 = KS) => pronuncia=1 (CF = 0.9)
(Word_left_1 = =if=) and (POS_right_1 = ART) => pronuncia=1 (CF = 0.85)
(POS_left_1 = ADV) and (Word_left_1 = não) => pronuncia=1 (CF = 0.92)
(Word_left_1 = =if=) and (POS_right_1 = ADV) => pronuncia=1 (CF = 0.78)
(Word_left_3 = =if=) and (Word_right_1 = pro) => pronuncia=1 (CF = 0.85)
(Word_left_2 = eu) => pronuncia=1 (CF = 0.93)
(POS_right_1 = ART) and (Word_left_1 = e) => pronuncia=1 (CF = 0.7)
(POS_left_1 = ADV) and (Sufix_right_1 = -a) => pronuncia=1 (CF = 0.92)
(Word_left_1 = =if=) and (Sufix_right_1 = -a) => pronuncia=1 (CF = 0.84)
(POS_left_1 = N) and (Sufix_right_1 = -a) and (POS_right_1 = PREP) => pronuncia
    =1 (CF = 0.75)
(Word_left_3 = =if=) and (Word_right_1 = mais) => pronuncia=1 (CF = 0.75)
(POS_left_1 = N) and (POS_left_2 = PROADJ) => pronuncia=1 (CF = 0.54)
(Word_right_1 = p) => pronuncia=1 (CF = 0.78)
(POS_right_1 = PROSUB) => pronuncia=1 (CF = 0.46)
(Word_right_1 = fé) => pronuncia=1 (CF = 0.78)
(Word_left_1 = mas) => pronuncia=1 (CF = 0.55)
(Word_right_1 = pro) => pronuncia=1 (CF = 0.91)
(Word_right_2 = para) and (POS_right_1 = ADV) => pronuncia=1 (CF = 0.8)
(Word_left_1 = =if=) and (Sufix_right_1 = -s) => pronuncia=1 (CF = 0.78)
```

```
587
588 Number of Rules : 40
589 ------------------------------------------------------------------------------
590 Regra induzida pelo algoritmo JRip para desambiguação do HH "pego":
591
592 (POS_left_1 = PROPESS) => pronuncia=1 (42.0/0.0)
593 (POS_left_1 = ADV) => pronuncia=1 (19.0/3.0)
594 (POS_left_1 = =IF=) => pronuncia=1 (10.0/0.0)
595 (POS_left_1 = KC) => pronuncia=1 (6.0/1.0)
596 (POS_left_1 = PDEN) => pronuncia=1 (4.0/0.0)
597 (POS_left_1 = KS) => pronuncia=1 (7.0/2.0)
598 (POS_left_1 = N) => pronuncia=1 (9.0/3.0)
599 (POS_left_1 = PRO-KS) => pronuncia=1 (4.0/0.0)
600 (POS_left_1 = PREP+ART) => pronuncia=1 (3.0/0.0)
601 (POS_left_1 = NPROP) => pronuncia=1 (1.0/0.0)
602  => pronuncia=0 (161.0/5.0)
603
604 Number of Rules : 11
605 ------------------------------------------------------------------------------
606 Regra induzida pelo algoritmo Ridor para desambiguação do HH "pela":
607
608 pronuncia = 0  (3404.0/0.0)
609
610 Total number of rules (incl. the default rule): 1
611 ------------------------------------------------------------------------------
612 Regra induzida pelo algoritmo Ridor para desambiguação do HH "pelas":
613
614 pronuncia = 0  (509.0/0.0)
615
616 Total number of rules (incl. the default rule): 1
617 ------------------------------------------------------------------------------
618 Regra induzida pelo algoritmo Ridor para desambiguação do HH "pelo":
619
620 pronuncia = 0  (2090.0/0.0)
621
622 Total number of rules (incl. the default rule): 1
623 ------------------------------------------------------------------------------
624 Regra induzida pelo algoritmo JRip para desambiguação do HH "peso":
625
626 (POS_left_1 = PROPESS) => pronuncia=1 (4.0/0.0)
627 (POS_left_1 = KS) and (POS_right_1 = ADV) => pronuncia=1 (1.0/0.0)
628 (POS_left_1 = KC) and (POS_right_1 = PREP) => pronuncia=1 (2.0/0.0)
629  => pronuncia=0 (475.0/0.0)
630
631 Number of Rules : 4
632 ------------------------------------------------------------------------------
633 Regra induzida pelo algoritmo Ridor para desambiguação do HH "piloto":
634
635 pronuncia = 0  (88.0/0.0)
636
637 Total number of rules (incl. the default rule): 1
638 ------------------------------------------------------------------------------
639 Regra induzida pelo algoritmo Ridor para desambiguação do HH "reforço":
640
641 pronuncia = 0  (46.0/0.0)
642
643 Total number of rules (incl. the default rule): 1
```

```
644 ------------------------------------------------------------------------------
645 Regra induzida pelo algoritmo Ridor para desambiguação do HH "rego":
646
647 pronuncia = 0  (101.0/0.0)
648
649 Total number of rules (incl. the default rule): 1
650 ------------------------------------------------------------------------------
651 Regra induzida pelo algoritmo Ridor para desambiguação do HH "rogo":
652
653 pronuncia = 1  (26.0/7.0)
654            Except (Word_left_2 = o) => pronuncia = 0  (5.0/0.0) [2.0/0.0]
655
656 Total number of rules (incl. the default rule): 2
657 ------------------------------------------------------------------------------
658 Regra induzida pelo algoritmo Ridor para desambiguação do HH "rolha":
659
660 pronuncia = 0  (42.0/0.0)
661
662 Total number of rules (incl. the default rule): 1
663 ------------------------------------------------------------------------------
664 Regra induzida pelo algoritmo FURIA para desambiguação do HH "rolo":
665
666  => pronuncia=0 (CF = 0.0)
667 (POS_left_1 = PROPESS) => pronuncia=1 (CF = 0.5)
668
669 Number of Rules : 2
670 ------------------------------------------------------------------------------
671 Regra induzida pelo algoritmo JRip para desambiguação do HH "rota":
672
673 (POS_right_1 = =FF=) and (POS_left_1 = N) => pronuncia=0 (6.0/0.0)
674 (POS_right_1 = =FF=) and (Word_left_1 = toda) => pronuncia=0 (5.0/0.0)
675 (POS_left_1 = N) => pronuncia=0 (11.0/1.0)
676 (POS_right_1 = =FF=) and (POS_left_1 = V) => pronuncia=0 (5.0/0.0)
677 (POS_left_1 = ADV) => pronuncia=0 (5.0/0.0)
678  => pronuncia=1 (82.0/9.0)
679
680 Number of Rules : 6
681 ------------------------------------------------------------------------------
682 Regra induzida pelo algoritmo JRip para desambiguação do HH "rotas":
683
684 (POS_left_1 = N) => pronuncia=0 (13.0/2.0)
685 (Word_left_1 = =if=) => pronuncia=0 (2.0/0.0)
686 (Word_left_1 = tooooodas) => pronuncia=0 (1.0/0.0)
687 (Word_left_1 = estão) => pronuncia=0 (2.0/0.0)
688 (Word_left_1 = promesas) => pronuncia=0 (1.0/0.0)
689  => pronuncia=1 (133.0/4.0)
690
691 Number of Rules : 6
692 ------------------------------------------------------------------------------
693 Regra induzida pelo algoritmo FURIA para desambiguação do HH "seca":
694
695 (POS_left_1 = ART) => pronuncia=0 (CF = 1.0)
696 (POS_right_3 = N) => pronuncia=0 (CF = 1.0)
697 (POS_left_1 = PREP+ART) => pronuncia=0 (CF = 1.0)
698 (POS_left_1 = N) => pronuncia=0 (CF = 0.99)
699 (POS_left_1 = PREP) => pronuncia=0 (CF = 1.0)
700 (POS_left_1 = ADJ) => pronuncia=0 (CF = 1.0)
```

```
(POS_left_1 = V) => pronuncia=0 (CF = 1.0)
(POS_left_3 = N) => pronuncia=0 (CF = 0.99)
(Word_left_1 = =if=) => pronuncia=0 (CF = 0.9)
(POS_left_1 = ADV) => pronuncia=0 (CF = 0.93)
(POS_left_1 = PROADJ) => pronuncia=0 (CF = 1.0)
(POS_left_2 = ADJ) => pronuncia=0 (CF = 1.0)
(POS_right_1 = PROADJ) and (Word_left_3 = =if=) => pronuncia=1 (CF = 0.76)
(POS_left_1 = PROPESS) => pronuncia=1 (CF = 0.67)
(Word_left_1 = e) and (POS_left_2 = V) => pronuncia=1 (CF = 0.51)
(Word_right_2 = lágrima) => pronuncia=1 (CF = 0.51)
(Sufix_left_3 = -o) and (Word_left_2 = o) and (POS_left_1 = N) => pronuncia=1 (
    CF = 0.51)
(Word_left_3 = isto) => pronuncia=1 (CF = 0.35)
(Word_left_3 = ligado) => pronuncia=1 (CF = 0.35)
(Word_left_2 = telhados) => pronuncia=1 (CF = 0.35)
(Word_left_2 = espírito) => pronuncia=1 (CF = 0.35)
(Word_left_3 = corta) => pronuncia=1 (CF = 0.35)
(Word_left_2 = msm) => pronuncia=1 (CF = 0.35)

Number of Rules : 23
------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "secas":

pronuncia = 0  (213.0/0.0)

Total number of rules (incl. the default rule): 1
------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "sobre":

pronuncia = 0  (2714.0/0.0)

Total number of rules (incl. the default rule): 1
------------------------------------------------------------------------------
Regra induzida pelo algoritmo JRip para desambiguação do HH "soco":

(POS_left_1 = PROPESS) => pronuncia=1 (8.0/0.0)
 => pronuncia=0 (184.0/0.0)

Number of Rules : 2
------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "sopro":

pronuncia = 0  (78.0/0.0)

Total number of rules (incl. the default rule): 1
------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "suborno":

pronuncia = 0  (101.0/0.0)

Total number of rules (incl. the default rule): 1
------------------------------------------------------------------------------
Regra induzida pelo algoritmo Ridor para desambiguação do HH "sufoco":

pronuncia = 0  (55.0/0.0)

Total number of rules (incl. the default rule): 1
```

```
757 ------------------------------------------------------------------------------
758 Regra induzida pelo algoritmo Ridor para desambiguação do HH "termos":
759
760 pronuncia = 0  (1349.0/0.0)
761
762 Total number of rules (incl. the default rule): 1
763 ------------------------------------------------------------------------------
764 Regra induzida pelo algoritmo JRip para desambiguação do HH "toco":
765
766 (POS_left_1 = PROPESS) => pronuncia=1 (115.0/0.0)
767 (POS_left_1 = ADV) => pronuncia=1 (99.0/4.0)
768 (POS_left_1 = KS) => pronuncia=1 (54.0/3.0)
769 (Word_left_1 = =if=) => pronuncia=1 (40.0/8.0)
770 (POS_left_1 = KC) and (Word_left_1 = e) => pronuncia=1 (34.0/9.0)
771 (Word_left_1 = que) => pronuncia=1 (6.0/0.0)
772 (POS_left_1 = PDEN) => pronuncia=1 (7.0/0.0)
773 (POS_left_1 = KC) and (Word_left_1 = mas) => pronuncia=1 (2.0/0.0)
774 (Word_left_1 = alguém) => pronuncia=1 (1.0/0.0)
775 (Word_left_1 = dobrados) => pronuncia=1 (1.0/0.0)
776 (Word_left_1 = uma) => pronuncia=1 (1.0/0.0)
777 (Word_left_1 = isso) => pronuncia=1 (1.0/0.0)
778 (Word_left_1 = amanha) => pronuncia=1 (1.0/0.0)
779 (Word_left_1 = calcinha) => pronuncia=1 (1.0/0.0)
780 (Word_left_1 = doente) => pronuncia=1 (1.0/0.0)
781 (Word_left_1 = dificil) => pronuncia=1 (1.0/0.0)
782 (Word_left_1 = daí) => pronuncia=1 (1.0/0.0)
783 (Word_left_1 = dormir) => pronuncia=1 (1.0/0.0)
784 (Word_left_1 = qnd) => pronuncia=1 (1.0/0.0)
785 (Word_left_1 = dias) => pronuncia=1 (1.0/0.0)
786  => pronuncia=0 (1356.0/20.0)
787
788 Number of Rules : 21
789 ------------------------------------------------------------------------------
790 Regra induzida pelo algoritmo Ridor para desambiguação do HH "tola":
791
792 pronuncia = 0  (20.0/0.0)
793
794 Total number of rules (incl. the default rule): 1
795 ------------------------------------------------------------------------------
796 Regra induzida pelo algoritmo Ridor para desambiguação do HH "topo":
797
798 pronuncia = 1  (1382.0/1194.0)
799             Except (POS_left_1 = ART) => pronuncia = 0  (215.0/0.0) [221.0/1.0]
800             Except (POS_left_1 = PREP) => pronuncia = 0  (130.0/0.0) [146.0/1.0]
801             Except (POS_left_1 = PREP+ART) => pronuncia = 0  (81.0/0.0)
                    [68.0/0.0]
802             Except (POS_left_1 = V) and (POS_right_1 = PREP) => pronuncia = 0
                    (22.0/0.0) [14.0/0.0]
803             Except (POS_left_1 = PROADJ) => pronuncia = 0  (49.0/0.0) [42.0/1.0]
804             Except (POS_left_1 = V) and (POS_left_2 = N) => pronuncia = 0
                    (12.0/0.0) [16.0/0.0]
805             Except (POS_left_1 = ADJ) => pronuncia = 0  (32.0/1.0) [39.0/2.0]
806             Except (POS_left_1 = V) and (POS_left_3 = N) => pronuncia = 0
                    (10.0/0.0) [3.0/0.0]
807             Except (POS_left_1 = KC) and (POS_left_2 = N) => pronuncia = 0
                    (13.0/0.0) [17.0/3.0]
```

```
808             Except (POS_left_1 = V) and (POS_right_1 = ADJ) => pronuncia = 0
                    (5.0/0.0) [1.0/0.0]
809             Except (POS_left_1 = V) and (POS_left_2 = ADV) => pronuncia = 0
                    (4.0/0.0) [5.0/0.0]
810             Except (POS_left_1 = N) and (POS_left_2 = N) => pronuncia = 0
                    (5.0/0.0) [3.0/0.0]
811             Except (POS_left_1 = V) and (POS_left_2 = PREP) => pronuncia = 0
                    (3.0/0.0) [1.0/0.0]
812             Except (POS_left_1 = V) => pronuncia = 0  (8.0/0.0) [6.0/3.0]
813             Except (POS_left_1 = NUM) => pronuncia = 0  (5.0/0.0) [1.0/0.0]
814             Except (POS_left_1 = N) => pronuncia = 0  (7.0/2.0) [4.0/1.0]
815             Except (POS_left_1 = KC) and (POS_left_3 = N) and (POS_right_2 = N)
                    => pronuncia = 0  (4.0/1.0) [1.0/0.0]
816             Except (Word_left_1 = =if=) and (POS_right_1 = PREP) => pronuncia = 0
                      (3.0/0.0) [2.0/1.0]
817
818 Total number of rules (incl. the default rule): 19
819 ------------------------------------------------------------------------------
820 Regra induzida pelo algoritmo JRip para desambiguação do HH "torno":
821
822 (Word_left_1 = em) => pronuncia=0 (193.0/0.0)
823 (POS_left_1 = N) => pronuncia=0 (45.0/6.0)
824 (Word_left_1 = o) => pronuncia=0 (18.0/0.0)
825 (Word_left_1 = os) => pronuncia=0 (27.0/7.0)
826 (POS_left_1 = PREP) and (Word_left_1 = com) => pronuncia=0 (23.0/9.0)
827 (Word_left_1 = do) => pronuncia=0 (8.0/1.0)
828 (POS_left_1 = NPROP) => pronuncia=0 (6.0/1.0)
829 (POS_left_1 = PREP) and (Word_left_1 = sobre) => pronuncia=0 (3.0/0.0)
830 (Word_left_1 = dos) => pronuncia=0 (10.0/2.0)
831 (POS_left_1 = NUM) and (Word_left_1 = dois) => pronuncia=0 (5.0/0.0)
832 (POS_left_1 = V) and (Word_left_1 = são) => pronuncia=0 (4.0/0.0)
833  => pronuncia=1 (558.0/61.0)
834
835 Number of Rules : 12
836 ------------------------------------------------------------------------------
837 Regra induzida pelo algoritmo Ridor para desambiguação do HH "torres":
838
839 pronuncia = 0  (89.0/0.0)
840
841 Total number of rules (incl. the default rule): 1
842 ------------------------------------------------------------------------------
843 Regra induzida pelo algoritmo Ridor para desambiguação do HH "travessa":
844
845 pronuncia = 0  (128.0/81.0)
846             Except (POS_right_2 = N) => pronuncia = 1  (21.0/1.0) [24.0/2.0]
847             Except (POS_left_1 = PREP+ART) => pronuncia = 1  (6.0/0.0) [9.0/1.0]
848             Except (POS_right_1 = N) and (Word_right_1 = j) => pronuncia = 1
                    (5.0/0.0) [4.0/0.0]
849
850 Total number of rules (incl. the default rule): 4
851 ------------------------------------------------------------------------------
852 Regra induzida pelo algoritmo Ridor para desambiguação do HH "vede":
853
854 pronuncia = 0  (52.0/0.0)
855
856 Total number of rules (incl. the default rule): 1
```