**UNIVERSIDADE FEDERAL DE SÃO CARLOS**

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

# GERAÇÃO DE VETORES DE SENTIDO PARA O PORTUGUÊS

**JÉSSICA RODRIGUES DA SILVA**

**ORIENTADOR: PROFA. DRA. HELENA DE MEDEIROS CASELI**

São Carlos – SP

Junho/2019

**UNIVERSIDADE FEDERAL DE SÃO CARLOS**

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

# GERAÇÃO DE VETORES DE SENTIDO PARA O PORTUGUÊS

**JÉSSICA RODRIGUES DA SILVA**

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação, área de concentração: Inteligência Artificial

Orientador: Profa. Dra. Helena de Medeiros Caseli

São Carlos – SP

Junho/2019

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências Exatas e de Tecnologia
Programa de Pós-Graduação em Ciência da Computação

## Folha de Aprovação

Assinaturas dos membros da comissão examinadora que avaliou e aprovou a Defesa de Dissertação de Mestrado da candidata Jéssica Rodrigues da Silva, realizada em 03/07/2019:

Profa. Dra. Helena de Medeiros Caseli
UFSCar

Profa. Dra. Heloisa de Arruda Camargo
UFSCar

Prof. Dr. Thiago Alexandre Salgueiro Pardo
USP

A todos que fizeram parte desta jornada comigo.

# AGRADECIMENTOS

Agradeço a Prof[a] Helena Caseli pela orientação, apoio e dedicação no desenvolvimento deste projeto. Agradeço também aos amigos do LALIC pela troca de conhecimento em nossas reuniões semanais.

Agradeço ao meu esposo Ricardo pela ajuda, cuidado incondicional e dedicação a me ajudar com o que fosse necessário para concluir este projeto. Agradeço também aos meus pais e irmãos pela admiração e entusiasmo.

Agradeço aos amigos de trabalho pela troca de conhecimento diária e a parceria em tantos desafios. Agradeço também a Samsung Instituto de P&D da Amazônia pela infraestrutura oferecida e apoio financeiro ao meu mestrado.

Essa pesquisa é parte do projeto MMeaning, apoiado pela Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), processo n[o] 2016/13002-0, e foi parcialmente apoiado pela Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

*Epígrafe*

John R. Firth

You shall know a word by the company it keeps!

# RESUMO

Representações vetoriais numéricas são capazes de representar desde palavras até significados, em espaços vetoriais contínuos de baixa dimensão. Essas representações utilizam a modelagem distribucional, onde o contexto em que a palavra ocorre é levado em consideração para a geração do vetor. As representações de palavras, mais conhecidas como *word embeddings* ou *word vectors* (Word2vec, FastText, Wang2vec e Glove), muito utilizadas até então, apresentam uma importante limitação: produzem uma única representação vetorial para cada palavra, ignorando o fato de que palavras ambíguas podem assumir significados diferentes (contextos diferentes). Essa combinação de significados pode ser um problema para várias aplicações. Por exemplo, em uma tarefa de compreensão de linguagem, usando o vetor de uma palavra ambígua como "banco", todos os possíveis significados – como instituição financeira, banco de sangue ou um item de mobília – serão misturados em um único vetor numérico, causando uma interpretação semântica errada da sentença na qual ocorre. Ao longo dos últimos anos, as representações de significados (sentidos), conhecidas como *sense embeddings* ou *sense vectors* mostraram ser capazes de modelar conhecimento sintático e semântico e passaram a ser utilizadas em aplicações de PLN. Por ser capaz de transformar os vários sentidos de uma palavra ambígua em vetores numéricos, notou-se o poder desse recurso em fazer Desambiguação Lexical de Sentidos (DLS). Esta pesquisa gerou e avaliou vetores de sentido para o português (PT-BR e PT-EU), e mostrou que eles superam os vetores tradicionais em tarefas intrínsecas e extrínsecas de PLN, já que são capazes de lidar com a ambiguidade lexical. Até onde sabemos, este é o primeiro trabalho a investigar a geração e a avaliação de vetores de sentido para o português.

**Palavras-chave**: vetores de sentido, desambiguação lexical de sentidos, vetores de palavra, modelagem distribucional

# ABSTRACT

Numerical vector representations are able to represent from words to meanings, in a low-dimensional continuous space. These representations are based on distributional modeling, where the context in which the word occurs is taken into account for vector generation. The word representations, known as word embeddings or word vectors (Word2vec, FastText, Wang2vec and Glove), which have been widely used until now, have an important limitation: they produce a single vector representation for each word, ignoring the fact that ambiguous words can represent different meanings (different contexts). This mixture of meanings can be a problem for many applications. For example, in a language comprehension task, using the vector of an ambiguous word as "bank", all possible meanings –such as financial institution, blood bank, or furniture item –will be mixed into a single numerical vector, causing an erroneous semantic interpretation of the sentence in which it occurs. Over the last few years, representations of meanings, known as sense embeddings or sense vectors, have proven to be able to model syntactic and semantic knowledge and have been used in NLP applications. By being able to transform the various meanings of an ambiguous word into numerical vectors, sense vectors can be applied to Word Sense Disambiguation (WSD). Thus, this work generated and evaluated sense vectors for Portuguese (PT-BR and PT-EU), and showed that they overcome traditional vectors in intrinsic and extrinsic NLP tasks, since they are capable of dealing with lexical ambiguity. To the best of our knowledge, this is the first work to address the geneation and evaluation of sense vectors for Portuguese.

**Keywords**: sense embeddings, sense vectors, word sense disambiguation, word embeddings, word vectors, distributional modeling

# LISTA DE FIGURAS

# LISTA DE TABELAS

# SUMÁRIO

# Capítulo 1

## INTRODUÇÃO

Todas as linguagens naturais têm ambiguidades, sejam elas sintáticas, lexicais, semânticas ou de escopo. Cançado (2008) afirma que a ambiguidade é um fenômeno semântico que aparece quando uma simples palavra ou um grupo de palavras é associado a mais de um significado. O tipo de ambiguidade de maior importância para este trabalho é o lexical, que segundo Cançado (2008), ocorre quando a dupla interpretação incide somente sobre o item lexical (no caso desta pesquisa, uma palavra).

O foco da maioria dos trabalhos voltados para o tratamento automático da ambiguidade lexical está na resolução de ambiguidades e a tarefa que se preocupa em resolver esse problema é denominada Desambiguação Lexical de Sentidos (DLS) (do inglês *Word Sense Disambiguation*, WSD). A DLS é uma subárea do Processamento de Línguas Naturais (PLN) que tem como objetivo determinar qual sentido deve ser atribuído a uma palavra ambígua, de acordo com seu contexto (MANNING et al., 1999).

A DLS tem sido uma tarefa de longa data no PLN e encontra-se no cerne da compreensão da linguagem, tendo sido estudada sob vários ângulos (NAVIGLI, 2009, 2012). No entanto, o campo parece estar estagnado devido à falta de melhorias inovadoras e à dificuldade de integrar os sistemas de DLS em aplicações de PLN (LACALLE; AGIRRE, 2015). Identificar melhorias reais nos sistemas de DLS existentes tem sido um desafio e isso se deve principalmente à falta de uma estrutura unificada, que impede a comparação direta e justa entre os sistemas.

Segundo Raganato et al. (2017), embora muitos conjuntos de dados de avaliação tenham sido construídos para a tarefa, eles tendem a diferir em formato, diretrizes de construção e inventário de sentidos. Essas divergências são resolvidas individualmente usando ou construindo mapeamentos automáticos. A verificação de qualidade de tal mapeamento, no entanto, tende a ser impraticável e isso leva a erros que dão origem a inconsistências adicionais do sistema. Esse

problema também ocorre com os córpus de treinamento utilizados, onde não é possível garantir que os sistemas em comparação tenham sido treinados no mesmo córpus ou que ele tenha sido pré-processado da mesma forma.

Isso torna difícil tirar conclusões precisas sobre modelos diferentes, já que em alguns casos, melhorias ostensivas podem ter sido obtidas como consequência da natureza do córpus de treinamento, do pipeline de pré-processamento ou da versão do inventário de sentidos, ao invés do próprio modelo (RAGANATO et al., 2017).

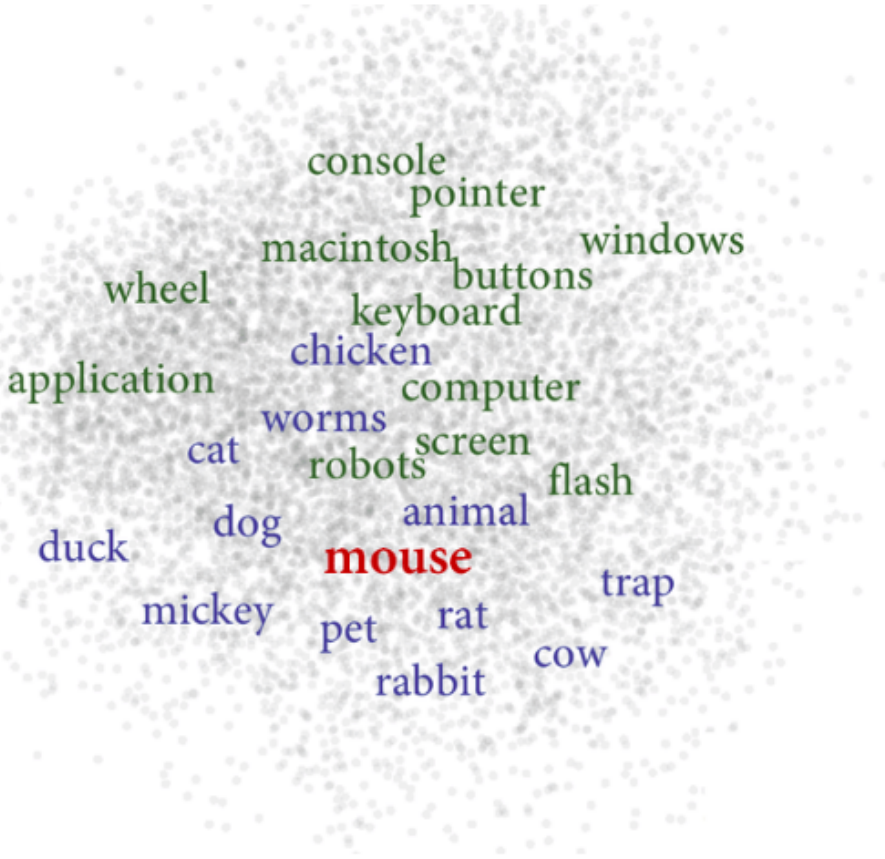
Além disso, devido a essas divergências, os sistemas atuais tendem a relatar resultados apenas em alguns conjuntos de dados, dificultando a avaliação em tarefas reais de PLN, como a própria compreensão da linguagem.

A tendência na literatura é que as soluções de DLS passem a ter como objetivo a melhoria de outras tarefas de PLN e para que isso ocorra é necessário enxergar o desafio de novos ângulos. A tarefa está intimamente relacionada com a deficiência de conflação de significado (*Meaning Conflation Deficiency*), que é a mistura de sentidos possíveis em uma única palavra (CAMACHO-COLLADOS; PILEHVAR, 2018). Por exemplo, na frase "*My mouse was broken, so I bought a new one yesterday.*" ("Meu mouse quebrou, então eu comprei um novo ontem."), o *mouse* (rato) estaria associado ao seu significado de dispositivo de computador e não ao animal. A figura 1.1 é uma ilustração dessa deficiência de conflação de significado em um espaço semântico 2D.

A conflação de significados pode ter impactos negativos adicionais na modelagem semântica e para aliviar essa deficiência, tem surgido uma nova direção de pesquisa nos últimos anos, que tenta modelar diretamente os significados individuais das palavras: as representações vetoriais de sentido ou simplesmente vetores de sentido (*sense embeddings* ou *sense vectors*).

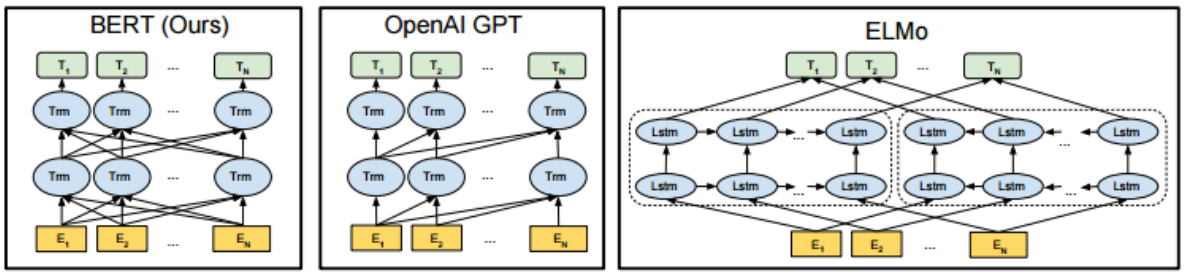
Representações vetoriais numéricas podem representar palavras ou conceitos em espaços vetoriais contínuos de baixa dimensão, reduzindo a dispersão inerente das representações tradicionais de espaços vetoriais (SALTON et al., 1975). Essas representações utilizam a modelagem distribucional, onde o contexto em que a palavra ocorre é levado em consideração para a geração do vetor (BRUNI et al., 2014). Representações de palavras ou vetores de palavra, mais conhecidos como *word embeddings* ou *word vectors* (Word2vec, FastText, Wang2vec e Glove), apresentam uma importante limitação: produzem uma única representação vetorial para cada palavra, ignorando o fato de que palavras ambíguas podem assumir significados diferentes (contextos diferentes). Essa combinação de significados pode ser um problema para várias aplicações. Para ilustrar como os vetores de sentido (gerados pelo MSSG, explicado com detalhes no cap. 4) capturam as diferenças de significado melhor do que os vetores de palavra

**Figura 1.1: Exemplo de deficiência de conflação de significado da palavra ambígua *mouse*. As palavras em azul referem-se ao sentido de animal e as em verde ao sentido de dispositivo.**




**Fonte: Retirado de (CAMACHO-COLLADOS; PILEHVAR, 2018)**

**Figura 1.2: Exemplo de deficiência de conflação de significado da palavra ambígua *mouse*. As palavras em azul referem-se ao sentido de animal e as em verde ao sentido de dispositivo.**




**Fonte: Retirado de (DEVLIN et al., 2018)**

(gerados pelo word2vec, explicado com detalhes no cap. 3), um exemplo de operação algébrica é mostrado na Figura 1.3. Este exemplo é para a palavra ambígua "banco", que em nosso domínio, tem três significados pré-dominantes: (1) banco de reservas (futebol, basquete), (2) armazenamento físico (banco de sangue, banco de dados) e (3) instituição financeira (Santander,

Pactual). Neste exemplo, mostramos os resultados de *banco* + *dados* − *dinheiro* e esperamos como resultado palavras relacionadas ao segundo significado da palavra "banco". Quando os vetores de sentido são usados (parte superior da Figura 1.3), obtemos exatamente o que esperamos. No entanto, quando os vetores de palavra são usados (parte inferior da Figura 1.3), não obtemos nenhum resultado relacionado à dados.

**Figura 1.3: Operação algébrica utilizando vetores de sentido (MSSG) e de palavra (word2vec) para a palavra ambígua "banco"**

```
sv.wv.most_similar(positive=['banco#4', 'dados#4'], negative=['dinheiro#3'])
```

```
[('odbc#2', 0.5867434144020081),
 ('mysql#2', 0.5699540972709656),
 ('postgresql#2', 0.5609346032142639),
 ('pubmed#2', 0.5564833283424377),
 ('sql#2', 0.5561952590942383),
 ('sgbd#2', 0.5543730854988098),
 ('bdp#2', 0.5477346181869507),
 ('sqlite#2', 0.5405952334403992),
 ('blockchain#2', 0.5359663367271423),
 ('sgbdr#2', 0.5341420769691467)]
```

```
wv.most_similar(positive=['banco', 'dados'], negative=['dinheiro'])
```

```
[('sisbacen', 0.49601683020591736),
 ('condepe/fidem', 0.46564042568206787),
 ('ine-instituto', 0.45371222496032715),
 ('ibge.o', 0.4466230869293213),
 ('indicadores', 0.44582003355026245),
 ('banxico', 0.44311660528182983),
 ('bancos', 0.4411255121231079),
 ('divulgados', 0.4391328692436218),
 ('agroclimático', 0.43791693449020386),
 ('couchdb', 0.4372578263282776)]
```

**Fonte: Autoria própria**

Essa simples operação algébrica mostra que faz sentido considerar e explorar os vetores de sentido como um possível recurso de linguagem capaz de resolver ambiguidades lexicais em tarefas de PLN.

## 1.1 Objetivo e Hipótese

Este projeto de mestrado teve como objetivo investigar o uso de representações vetoriais de sentido para resolver o problema da ambiguidade lexical em tarefas de PLN.

Desse modo, investigou-se a geração e aplicação de vetores de sentido de uma palavra ambígua em tarefas intrínsecas e extrínsecas de PLN, melhorando o desempenho dessas tarefas por meio da resolução das ambiguidades lexicais. A tarefa intrínseca foi a de Analogias

Sintáticas e Semânticas e as extrínsecas foram as de Similaridade Semântica e Desambiguação Lexical de Sentidos.

A hipótese investigada neste trabalho era a de que o desempenho de tarefas de PLN usando vetores de sentido seria melhor do que quando outros recursos de linguagem são utilizados (como os vetores de palavras), pois são capazes de fazer DLS. Foram investigados e implementados métodos independentes de língua, que foram avaliados para o português brasileiro (PT-BR) e europeu (PT-EU).

## 1.2 Organização do texto

Este texto está organizado da seguinte forma. No Capítulo 2, trazemos toda a fundamentação teórica da Desambiguação Lexical de Sentidos. No Capítulo 3, descrevemos alguns dos métodos mais populares de geração de vetores de palavras tradicionais, seus algoritmos e aplicações. No capítulo 4, descrevemos as abordagens existentes para a geração de vetores de sentido, algoritmos e aplicações.

No Capítulo 5, apresentamos as técnicas propostas para a geração de vetores de sentido, bem como os experimentos e as ferramentas que foram utilizadas neste projeto. Foram gerados vetores de sentido para o português, por meio de duas estratégias (agrupamento e etiquetação) baseadas em propostas da literatura. Este documento é finalizado com algumas considerações finais (Capítulo 6) sobre o trabalho e suas contribuições.

# Capítulo 2

## DESAMBIGUAÇÃO LEXICAL DE SENTIDOS

O foco da maioria dos trabalhos voltados para o tratamento automático da ambiguidade lexical está na resolução de ambiguidades e a tarefa que se preocupa em resolver esse problema é denominada Desambiguação Lexical de Sentidos (DLS) (do inglês *Word Sense Disambiguation*, WSD).

As abordagens para a DLS são frequentemente definidas de acordo com a principal fonte de conhecimento utilizada na definição do sentido correto de uma palavra ambígua. Os métodos que dependem de dicionários, tesauros e bases de conhecimento lexicais, sem usar qualquer evidência de córpus, são denominados baseados em dicionário ou **baseados em conhecimento**. Já os métodos que não fazem uso de informações externas e trabalham diretamente de córpus são denominados métodos **baseados em córpus** e realizam a DLS via aprendizado de máquina supervisionado (quando as palavras ambíguas do córpus de treinamento estão anotadas com o sentido correto), não supervisionado (quando o córpus de treinamento não possui qualquer anotação de sentido) ou semissupervisionado (quando apenas parte do córpus de treinamento possui anotação de sentido correto).

A seguir, são apresentadas as principais medidas de avaliação usadas na DLS (seção 2.1), principais abordagens baseadas em conhecimento (seção 2.2) e principais abordagens supervisionadas (seção 2.3) e não supervisionadas (seção 2.4) baseadas em córpus.

## 2.1 Avaliação

A avaliação de sistemas de DLS pode ser feita de duas formas: (1) a avaliação intrínseca, na qual os sistemas são testados considerando o objetivo específico para o qual foram desenvolvidos, neste caso, para a desambiguação lexical, independentemente da tarefa em que serão

aplicados; e (2) a avaliação extrínseca (ou validação), na qual os resultados dos sistemas são avaliados em termos da sua contribuição para o desempenho global de um sistema criado para determinada aplicação.

A avaliação intrínseca normalmente consiste em comparar a saída do sistema para determinadas entradas com os resultados esperados (corretos), obtidos por meio da atribuição manual de sentidos em um córpus de referência. Os resultados dessas comparações são reportados de acordo com diferentes medidas. Abaixo são mostradas as medidas de avaliação mais utilizadas (EDMONDS, 2002). Para entender como essas medidas são calculadas no caso investigado neste trabalho, a seguir temos algumas definições baseadas em um classificador que classifica palavras como **ambígua** ou **não ambígua**:

- Verdadeiros Positivos (VP): casos em que o sistema classificou a palavra como **ambígua** e realmente era **ambígua**.
- Falsos positivos (FP): casos em que o sistema classificou a palavra como **ambígua** e na verdade era **não ambígua**.
- Falsos Verdadeiros (FV): casos em que o sistema classificou a palavra como **não ambígua** e realmente era **não ambígua**.
- Falsos Negativos (FN): casos em que o sistema classificou a palavra como **não ambígua** e na verdade era **ambígua**.

Seguem as fórmulas:

- **Acurácia** (*accuracy*): quantidade de palavras ambíguas classificadas corretamente (VP) somado ao total de palavras não ambíguas classificadas corretamente (FV) com relação ao total de palavras do conjunto de teste. A equação 2.1 ilustra o cálculo da acurácia.

$$accuracy = \frac{VP + FV}{Total} \tag{2.1}$$

- **Precisão** (*precision*): quantidade de palavras ambíguas classificadas corretamente (VP) em relação a essa mesma quantidade somada à quantidade de palavras não ambíguas classificadas como ambíguas (FP). A equação 2.2 apresenta o cálculo da precisão.

$$precision = \frac{VP}{VP + FP} \tag{2.2}$$

- **Revocação** (*recall*): quantidade de palavras ambíguas classificadas corretamente (VP) com relação a essa mesma quantidade somada à quantidade de palavras ambíguas classificadas como não ambíguas (FN). A equação 2.3 mostra o cálculo da revocação.

$$recall = \frac{VP}{VP + FN} \tag{2.3}$$

Na avaliação extrínseca são utilizadas medidas específicas para as aplicações nas quais a DLS é empregada.

## 2.2 Métodos baseados em conhecimento

Juntamente com os métodos baseados em córpus, os métodos baseados em conhecimento englobam as principais técnicas para a desambiguação lexical de sentidos. Os métodos baseados em conhecimento para DLS geralmente são aplicáveis às palavras de qualquer texto, enquanto os baseados em córpus funcionam apenas para padrões extraídos dos córpus utilizados no treinamento.

Nos métodos baseados em conhecimento, a desambiguação é realizada através de consultas a informações previamente e explicitamente especificadas, manualmente ou a partir de recursos lexicais. Segundo Kilgarriff (1992), o processo manual de construção de regras específicas requer fontes variadas de conhecimento. Nesse processo manual, além da quantidade muito pequena de palavras mapeadas, outra crítica é a ausência de critérios claros para a seleção do subconjunto de palavras ambíguas. Os possíveis sentidos para cada palavra ambígua também são, em geral, definidos pelo pesquisador, como um recorte dos sentidos encontrados em dicionários.

A seguir, são apresentados quatro tipos principais de métodos baseados em conhecimento: (1) métodos que utilizam a sobreposição contextual em relação às definições de dicionário, (2) métodos baseados em medidas de similaridade calculadas com base em redes semânticas, (3) métodos baseados em preferências de seleção, como meio de restringir os possíveis significados das palavras em um determinado contexto e (4) métodos baseados em heurística, que dependem de propriedades da linguagem humana.

### 2.2.1 Sobreposição contextual

O algoritmo de Lesk (1986) foi um dos primeiros algoritmos desenvolvidos para a desambiguação lexical de sentidos em textos de qualquer domínio. O único recurso requerido pelo

algoritmo é um conjunto de entradas de dicionário para cada sentido possível, acompanhadas de conhecimento sobre o contexto onde a desambiguação será realizada.

A principal ideia por trás da definição original do algoritmo é desambiguar palavras encontrando a sobreposição entre suas definições de sentido. Para cada sentido $i$ da palavra $W_1$ e cada sentido $j$ da palavra $W_2$, calcula-se a maior sobreposição de palavras em comum de $W_1$ e $W_2$. Atribui-se o sentido $i$ encontrado à palavra $W_1$ e o sentido $j$ encontrado à palavra $W_2$. A Figura 2.1 ilustra os principais passos do algoritmo.

**Figura 2.1: Algoritmo de Lesk (baseado em dicionário).**

(1) for each sense $i$ of $W_1$
(2) for each sense $j$ of $W_2$
(3) compute *Overlap(i,j)*, the number of words in common between the definitions of sense $i$ and sense $j$
(4) find $i$ and $j$ for which *Overlap(i,j)* is maximized
(5) assign sense $i$ to $W_1$ and sense $j$ to $W_2$

**Fonte: Retirado de (LESK, 1986)**

Como exemplo, considere a tarefa de desambiguar as palavras *pine* e *cone*. O *Oxford Advanced Learner's Dictionary* define quatro sentidos para *pine* e três sentidos para *cone*, reproduzidos na Figura 2.2

**Figura 2.2: Exemplo de aplicação do algoritmo de Lesk (1986).**

**pine**
1* seven kinds of evergreen tree with needle-shaped leaves
2 pine
3 waste away through sorrow or illness
4 pine for something, pine to do something

**cone**
1 solid body which narrows to a point
2 something of this shape, whether solid or hollow
3* fruit of certain evergreen trees (fir, pine)

**Fonte: Retirado de (LESK, 1986)**

A primeira definição para *pine* e a terceira definição de *cone* (marcados com * na figura) têm a maior sobreposição entre todas as combinações de sentidos possíveis, com três palavras em comum: *evergreen*, *tree* e *pine* e, portanto, estes são os significados selecionados pelo algoritmo

de Lesk para o par de *pine* e *cone*.

O algoritmo de Lesk foi avaliado em uma amostra de pares de palavras ambíguas anotadas manualmente em relação ao *Oxford Advanced Learner's Dictionary*. Observou-se uma precisão de 50-70% (LESK, 1986). Há uma outra versão do algoritmo de Lesk que tenta resolver a explosão combinatória de sentidos de palavras, como uma variação simplificada que executa um processo de desambiguação separado para cada palavra ambígua no texto. Nesse algoritmo simplificado, o significado correto de cada palavra em um texto é determinado individualmente, encontrando o sentido que leva à maior sobreposição entre a definição do dicionário e o contexto atual. Em vez de tentar determinar simultaneamente os significados de todas as palavras em um determinado texto, essa abordagem processa cada palavra individualmente, independentemente do significado das outras palavras que ocorrem no mesmo contexto.

Um trabalho para a língua portuguesa que utilizou dois métodos de DLS baseados em conhecimento foi o de Nóbrega e Pardo (2012). O primeiro método consistiu na abordagem heurística de adotar o sentido mais frequente para uma palavra (explicada na seção 2.2.4.1). O segundo, foi a adaptação do algoritmo de Lesk (1986). O objetivo foi explorar métodos simples e de uso geral para desambiguar substantivos comuns em textos jornalísticos escritos em português do Brasil, usando a Wordnet de Princeton (WordNet-Pr) como repositório de sentidos e um dicionário bilíngue para fazer os mapeamentos para os *synsets* da WordNet-Pr. A avaliação ocorreu em 50 grupos de textos do córpus CSTNews (ALEIXO; PARDO, 2008), totalizando 140 textos e 4.366 palavras desambiguadas (sendo 466 palavras distintas). O método heurístico obteve precisão, cobertura e acurácia de 51%. O método Lesk (1986) adaptado foi testado com seis variações, porém, todas com desempenho inferior ao método heurístico.

Uma desvantagem do método de Lesk (1986) é o fato de ser dependente de definições de um dicionário específico, uma vez que a presença ou ausência de uma palavra na definição do dicionário pode mudar completamente os resultados. Além disso, a simples contagem das palavras pode privilegiar a escolha de sentidos com definições mais extensas. Entretanto, o método é de grande relevância para a área de DLS, pois serviu de base para vários outros trabalhos estatísticos.

### 2.2.2 Similaridade semântica

Como bem estabelecido em (HARRIS, 1954 apud ZANZOTTO et al., 2010), na chamada hipótese distribucional: palavras que compartilham contextos similares geralmente possuem significados similares. Portanto, os sentidos apropriados podem ser encontrados com base na menor distância semântica entre os contextos de ocorrência (RADA et al., 1989).

Geralmente, a similaridade semântica é calculada com base em um pequeno número de palavras encontradas na vizinhança imediata de uma palavra-alvo ou apenas nas palavras relacionadas por dependências sintáticas com a palavra-alvo. Esses métodos direcionam o contexto local de uma determinada palavra e não levam em consideração informações contextuais adicionais encontradas fora de um determinado tamanho de janela. No entanto, existem outros métodos que consideram um contexto global e tentam criar segmentos de significado olhando todo o texto, com seu escopo estendido além de uma pequena janela centrada em palavras-alvo.

Uma série de medidas de similaridade foram propostas para quantificar o grau de semelhança semântica entre duas palavras. A maioria dessas medidas depende de redes semânticas e segue a metodologia proposta por Rada et al. (1989) para computar métricas em redes semânticas.

Dada uma medida de similaridade semântica definida como:

$$score : Senses_D \times Senses_D \; [0,1] \tag{2.4}$$

onde $Senses_D$ é o conjunto completo de sentidos listados em um léxico de referência. Para desambiguar uma palavra-alvo $w_i$ em um texto $T = (w_1, \cdots, w_n)$, escolhe-se o sentido $S$ de $w_i$ que maximiza a seguinte soma:

$$S = argmax_{S \in Senses_D(w_i)} \sum_{w_j \in T : w_j <> w_i} max_{S' \in Senses_D(w_j)} score(S, S') \tag{2.5}$$

O $score(S,S')$ calcula a menor distância entre pares de sentidos de palavras (ex: o número de arestas do caminho mais curto sobre uma rede léxica). A hipótese é a de que, dado um par de palavras $w_i$ e $w_j$ que ocorram no mesmo contexto, os sentidos que minimizam a distância entre eles são os significados mais apropriados. Assim, dado um sentido $S$ da palavra-alvo $w_i$, a fórmula resume a contribuição do sentido mais apropriado de cada palavra do contexto $w_j <> w_i$. Como resultado, o sentido com a soma mais alta é escolhido.

A similaridade semântica geralmente é calculada considerando-se um contexto local. Um texto geralmente envolve mais de duas palavras ambíguas onde a distância no contexto influencia seu significado. Pesquisas nesta área consideraram o uso do contexto local como uma restrição adicional para limitar o número de palavras no conjunto de palavras ambíguas.

Um exemplo de trabalho que realizou a DLS com base em similaridade semântica em um contexto local foi (PATWARDHAN et al., 2003). Patwardhan et al. (2003) aplicaram cinco medidas de similaridade para decidir qual o sentido correto em 1.723 casos de substantivos ambíguos da amostra lexical inglesa *Senseval-2*. Eles calcularam um *score* cumulativo adici-

onando as distâncias semânticas da palavra-alvo para as palavras na sua vizinhança imediata (ou seja, uma palavra para a esquerda e uma palavra para a direita). O sentido selecionado era aquele com maior pontuação cumulativa.

As dependências semânticas e sintáticas são outra restrição possível que pode ser aplicada a palavras envolvidas em uma relação de similaridade. Stetina e Nagao (1998) desenvolveram um método que usa dependências semânticas e sintáticas entre palavras e uma medida de similaridade simples que estabelece a semelhança entre duas palavras se elas pertencerem ao mesmo *synset* da WordNet. Experimentos usando dependências semânticas e sintáticas calculadas com cerca de 100 textos do Semcor para o inglês apontaram uma precisão geral de desambiguação de 80,3%, medida em 15 arquivos de teste. No mesmo conjunto de teste, eles obtiveram 75,2% usando a heurística de sentido mais frequente (apresentada na seção 2.2.4.1).

O contexto global leva em consideração uma quantidade maior de palavras que co-ocorrem com um dado sentido de uma palavra. Não há um consenso sobre quando um contexto deixa de ser local para se tornar global. Entretanto, considera-se normalmente uma janela de várias sentenças. O tamanho dessa janela também pode variar. Yarowsky (1993, 1994) sugere uma janela de 20 a 50 palavras à esquerda e à direita da palavra-alvo. Uma desvantagem deste método é que nem sempre o contexto da palavra a ser desambiguada inclui um número suficiente de palavras. Outra desvantagem é o custo de processamento de uma grande quantidade de palavras e suas informações.

Com relação à comparação entre os dois tipos de contexto (local $\times$ global), vários autores (Yarowsky (1992), por exemplo) afirmam que o contexto global é mais indicado para a desambiguação de substantivos, pois eles requerem informações que podem estar mais distantes no texto. Já o contexto local é mais apropriado para desambiguação de verbos ou adjetivos, pois eles necessitam de informações que geralmente estão próximas, como os argumentos de um verbo ou os elementos modificados por um adjetivo. Dagan e Itai (1994) sugerem que os dois métodos são complementares e podem ser combinados para obter um conhecimento maior sobre o contexto de ocorrência da palavra ambígua.

Na literatura, não foram encontrados trabalhos de DLS para o português do Brasil que utilizem algum método baseado em similaridade semântica.

### 2.2.3 Preferências de seleção

Alguns dos primeiros algoritmos para a DLS baseiam-se em preferências de seleção como forma de restringir os significados possíveis de uma palavra em um contexto. As preferências

seletivas capturam informações sobre possíveis relações entre categorias de palavras e representam o conhecimento de senso comum sobre classes de conceitos. *comer-comida* e *beber-líquido* são exemplos de tais restrições semânticas, que podem ser usadas para descartar significados incorretos e selecionar apenas aqueles sentidos que estão em harmonia com regras de senso comum.

Embora as preferências de seleção sejam intuitivas e ocorram de forma natural, é difícil colocá-las em prática para resolver o problema de DLS. A razão principal é a relação circular entre preferências de seleção e DLS: aprender restrições semânticas requer conhecimento dos sentidos envolvidos em uma relação. A tarefa de DLS pode melhorar se grandes coleções de preferências seletivas estiverem disponíveis.

A seguir, são mostradas as estratégias mais utilizadas para tentar superar essa circularidade e aprender automaticamente as preferências de seleção com base em: (i) contagens de frequência ou (ii) relações de classe a classe adquiridas a partir de taxonomias criadas manualmente.

As contagens de frequência de relações entre palavras são medidas úteis para explicar a relação semântica entre elas. Dadas duas palavras $w_1$ e $w_2$, e a relação sintática $R$ que as conecta (por exemplo, sujeito-verbo, verbo-objeto), este método conta, em um grande córpus, o número de vezes em que as duas palavras ocorrem na relação $R$, representada aqui como $Count(w_1, w_2, R)$. Um método alternativo é usar probabilidades condicionais para estimar o ajuste semântico de uma determinada relação. Sob a mesma suposição de que as preferências de seleção são aprendidas para duas palavras $w_1$ e $w_2$ conectadas por uma relação $R$, a probabilidade condicional é determinada como na Equação 2.6, onde a palavra $w_2$ impõe as preferências de seleção em $w_1$. A restrição também pode ser expressa na outra direção, invertendo-se as funções das duas palavras.

$$P(w_1|w_2, R) = \frac{Count(w_1, w_2, R)}{Count(w_2, R)} \tag{2.6}$$

Brockmann e Lapata (2003) avaliaram cinco modelos de aquisição de preferências de seleção para verbos, seus objetos diretos, sujeitos e complementos preposicionais. Eles trabalharam com o alemão e a ideia foi mostrar que o modelo construído anteriormente para o inglês poderia ser generalizado. A avaliação dos modelos foi feita em contraposição com as avaliações humanas e a conclusão foi a de que não há um método melhor para extrair todas as preferências e que uma combinação teria um melhor desempenho.

Várias estratégias foram propostas para determinar a preferência seletiva entre duas pa-

lavras, entre uma palavra e uma classe semântica, ou entre duas classes semânticas. Uma avaliação comparativa destas estratégias em uma tarefa de DLS é relatada em (AGIRRE; MARTINEZ, 2001). Nesse trabalho, os autores observaram que o modelo classe-classe obteve uma maior cobertura na desambiguação em comparação com o modelo palavra-palavra ou palavra-classe. Em um conjunto de 8 substantivos, o *baseline* de sentido mais frequente (explicado na seção 2.2.4.1) obteve uma precisão de 69% e uma cobertura de 100%. As preferências de seleção de palavra-palavra obtiveram precisão de 95,9% e 26% de cobertura, as de palavra-classe 66,9% de precisão e 86,7% de cobertura e, finalmente, as preferências de classe-classe obtiveram uma precisão de 66,6% e uma cobertura de 97,3%.

Para a língua portuguesa, Cabezudo (2015) investigou as preferências de seleção através de buscas na web. O objetivo desse trabalho foi fazer a DLS de verbos do português brasileiro incorporando conhecimento linguístico da Verbnet.Br (repositório de verbos do PT-BR). O método seleciona pares de palavras visando desambiguar uma delas usando a outra como contexto. Constroem-se consultas formadas pelos *synsets* (presentes na WordNet-Pr) dessas palavras e utiliza-as em um motor de busca, escolhendo o *synset* com a maior quantidade de resultados retornados como o sentido correto da palavra ambígua. A avaliação teve como objetivo desambiguar os verbos do córpus CSTNews, contra *baselines* de sentido mais frequente e método cego (escolhe aleatoriamente os sentidos para cada verbo). Em todas as avaliações, o método heurístico de sentido mais frequente superou o proposto em precisão, cobertura, abrangência e acurácia. Já o método proposto superou o método cego em todas essas métricas.

### 2.2.4 Heurísticas

Uma maneira fácil e bastante precisa de prever os significados das palavras é confiar em heurísticas derivadas de propriedades linguísticas observadas em textos. Uma dessas heurísticas, muitas vezes usada como *baseline* na avaliação de sistemas de DLS, como citado anteriormente, é a heurística de sentido mais frequente. As outras duas heurísticas apresentadas nesta seção referem-se à tendência de uma palavra de exibir o mesmo significado em todas as suas ocorrências em um discurso (um sentido por discurso) ou na mesma colocação (um sentido por colocação).

#### 2.2.4.1 Sentido mais frequente

Entre todos os significados possíveis de uma palavra, um significado ocorre com mais frequência do que os outros. Os significados das palavras exibem uma distribuição Zipfiana: um sentido tem uma frequência dominante, seguido de uma diminuição significativa na frequência

para os demais (ZIPF, 1949). Logo, assumindo a disponibilidade de informação de frequência de palavras, um método de desambiguação muito simples pode ser projetado atribuindo a cada palavra o seu significado mais frequente, de acordo com essa distribuição de sentido a priori. Este método é frequentemente usado como *baseline* para DLS e, de acordo com Gale et al. (1992b), os sistemas mais razoáveis devem superar este *baseline*.

Embora conceitualmente muito simples e trivial de implementar, há uma desvantagem importante associada a este método: a heurística de sentido é aplicável somente às poucas línguas para as quais os córpus anotados com sentido estão disponíveis. Além disso, uma mudança de domínio ou gênero do córpus pode afetar significativamente as distribuições sensoriais, diminuindo consideravelmente o desempenho desta heurística.

Como já mencionado anteriormente, Nóbrega e Pardo (2012) utilizaram essa abordagem para o PT-BR, usando a WordNet-Pr como repositório de sentidos e um dicionário bilíngue para fazer os mapeamentos para os *synsets* dessa WordNet, alcançando 51% de precisão, cobertura e acurácia.

#### 2.2.4.2 Um sentido por discurso

Essa heurística foi introduzida por Gale et al. (1992a), que afirmam que uma palavra tende a preservar seu significado em todas as suas ocorrências em um determinado texto. Essa heurística possibilita a desambiguação automática de todas as instâncias de uma palavra sempre que o seu significado for identificado em pelo menos uma das ocorrências.

Gale et al. (1992a) testaram a hipótese de um sentido por discurso em 9 palavras com ambiguidade de 2 sentidos em um experimento realizado com 5 textos. Os avaliadores receberam 82 pares de sentenças e deveriam determinar se elas correspondiam ao mesmo sentido ou não. No geral, eles descobriram que com uma probabilidade de 98%, duas ocorrências de palavras no mesmo texto teriam o mesmo sentido.

Nóbrega e Pardo (2013) apresentaram o primeiro trabalho de propósito geral para desambiguação lexical do sentido para o português brasileiro, sendo o foco do trabalho a desambiguação de substantivos. Nesse trabalho, os autores adotam a heurística de um sentido por discurso no córpus CSTNews (ALEIXO; PARDO, 2008), que agrupa notícias em coleções. O repositório de sentidos utilizado foi a WordNet-Pr. Foi usado também um dicionário bilíngue para fazer os mapeamentos para os *synsets* da WordNet-Pr. Este trabalho também comparou os resultados com a aplicação da heurística de sentido mais frequente (que obteve 51% em precisão, cobertura e acurácia). Nenhum dos métodos testados superou esse *baseline*. No cenário multidocumento,

dois dos métodos conseguiram desambiguar 43,90% e 41,20% das palavras, contribuindo positivamente para a DLS.

Embora essa hipótese funcione bem para palavras ambíguas com poucos sentidos, Krovetz (1998) testou palavras com mais de dois sentidos e descobriu que essas palavras tendem a ter mais de um sentido por texto. Ele baseou sua avaliação no Semcor e no córpus DSO. Cerca de 33% das palavras nesses textos possuíam vários sentidos por discurso e, portanto, a precisão geral de desambiguação alcançada neste caso foi inferior a 70%.

#### 2.2.4.3 Um sentido por colocação

A heurística de sentido único por colocação é semelhante à de sentido único por discurso, mas tem um alcance diferente. Foi introduzida por Yarowsky (1993), que afirma que uma palavra tende a preservar seu significado quando usada na mesma colocação. Em outras palavras, as palavras próximas fornecem pistas fortes e consistentes sobre o sentido de uma palavra-alvo. Observou-se, também, que esse efeito é mais forte para as colocações adjacentes e torna-se mais fraco à medida que a distância entre as palavras aumenta.

Experimentos iniciais com essa hipótese consideraram novamente palavras ambíguas com poucos sentidos, principalmente com dois sentidos. Uma precisão global de 97% foi observada em um grande conjunto de exemplos com anotações manuais. Tal como acontece com a hipótese de um sentido por discurso, outras experiências realizadas com diferentes tipos de córpus mostraram que a força da hipótese diminui significativamente quando se usa palavras ambíguas com mais de dois sentidos possíveis.

Martinez e Agirre (2000) testaram a hipótese de sentido único por colocação em um ambiente experimental diferente, em que os córpus anotados envolviam variações de gênero e tema – *Semcor* e DSO – e os significados das palavras eram definidos em relação às entradas da WordNet. Semelhante às descobertas de Krovetz (1998) no caso da hipótese de sentido único por discurso, Martinez e Agirre (2000) descobriram que a precisão da heurística de sentido único por colocação cai significativamente para cerca de 70% quando as palavras com graus mais altos de ambiguidade são consideradas.

Por fim, outro trabalho desenvolvido com o foco em DLS para o português considerando a abordagem baseada em conhecimento foi o de Specia (2007). Nesse trabalho, a autora propôs uma abordagem de DLS para o português brasileiro voltada especificamente para a tradução automática (inglês-português), que segue uma metodologia híbrida (baseada em conhecimento e córpus) e utiliza um formalismo relacional para a representação de vários tipos de conheci-

mentos e de exemplos de desambiguação, por meio da técnica de Programação Lógica Indutiva (PLI) (MUGGLETON, 1991). O PLI combina características de aprendizado de máquina e programação lógica para fornecer mecanismos para o aprendizado supervisionado de modelos simbólicos (conjuntos de regras) a partir de exemplos de desambiguação, incluindo conhecimento relacional. Isso é possível porque a linguagem de representação utilizada possui poder de expressividade equivalente ao da lógica de primeira ordem, permitindo a representação de predicados n-ários e variáveis, possibilitando capturar relacionamentos contextuais. Para tarefas multilíngues, o PLI superou os resultados de outros algoritmos de aprendizado de máquina com as mesmas fontes de conhecimento. Para tarefas monolíngues, as abordagens de PLI obtiveram resultados comparáveis aos melhores métodos utilizados no *Senseval-3*.

## 2.3 Métodos supervisionados baseados em córpus

Nos métodos baseados em córpus, como o próprio nome revela, o conhecimento é extraído automaticamente a partir de córpus, evitando que grande quantidade de conhecimento precise ser levantada e estruturada. Essa é uma abordagem empírica que ganhou força com o avanço dos algoritmos de Aprendizado de Máquina (AM).

Nesse contexto, um córpus provê um conjunto de exemplos que explicam o resultado de uma determinada tarefa e que, quando submetidos a algoritmos de AM, permitem o desenvolvimento de modelos capazes de descrever esses exemplos e de predizer o comportamento de novos exemplos nessa tarefa. Esses algoritmos se subdividem em: supervisionados, não supervisionados e semissupervisionados.

Segundo Manning et al. (1999), a abordagem supervisionada na DLS consiste em induzir automaticamente modelos de classificação ou regras a partir de exemplos anotados. Em um córpus anotado, os exemplos possuem etiquetas de sentido, normalmente atribuídas com base no conjunto de sentidos de um dicionário ou recurso lexical. Esse córpus já é, portanto, desambiguado, e pode ser dado como entrada para algoritmos supervisionados para que generalizem esse conhecimento e consigam induzir o sentido correto de novas instâncias de ambiguidade, sem anotação prévia.

A principal vantagem da abordagem supervisionada é o fato de que os sentidos podem ser especificados previamente, provendo uma etiquetação mais adequada e refinada. A grande desvantagem é, mais uma vez, a aquisição prévia de conhecimento, nesse caso, córpus de treinamento previamente anotado com sentidos que requer um trabalho normalmente feito por humanos. Esse problema acaba por restringir a abrangência de muitos trabalhos a poucas pala-

vras, pois não há, ainda, córpus representativos com etiquetas de sentido visando a uma ampla utilização para a DLS.

A seguir são descritos os principais métodos de DLS da abordagem supervisionada baseada em córpus.

### 2.3.1 Probabilísticos

Tradicionalmente, em um problema de classificação (aprendizado supervisionado), os métodos estatísticos estimam um conjunto de parâmetros que expressam a probabilidade condicional de cada classe em um determinado contexto. Esses parâmetros podem ser combinados para atribuir o conjunto de classes que maximizam sua probabilidade em novos exemplos. O algoritmo Naive Bayes (RO; PE, 1973) é o algoritmo mais simples deste tipo, que usa a regra de Bayes e assume a independência condicional dos atributos, dado o rótulo da classe. Naive Bayes foi aplicado com sucesso em muitas investigações em DLS (GALE et al., 1992a; MILLER et al., 1993; PEDERSEN; BRUCE, 1997; ESCUDERO et al., 2000a).

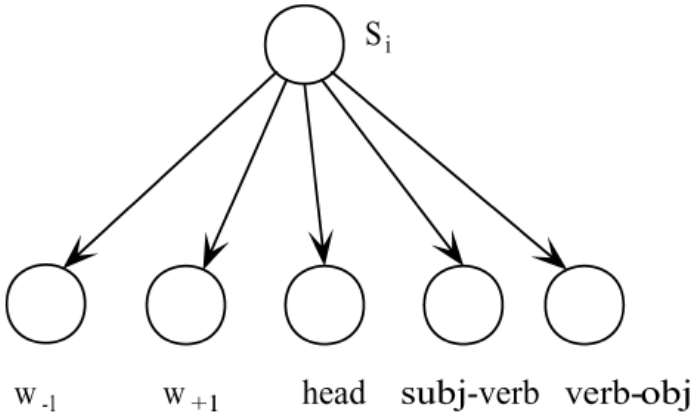
Um classificador Naive Bayes treinado para DLS baseia-se no cálculo da probabilidade condicional de cada sentido $S_i$ de uma palavra $w$ dados os atributos (*features*) $f_j$ no contexto. O sentido $S$ que maximiza a fórmula da equação 2.7 é escolhido como o sentido mais adequado no contexto.

$$\begin{aligned} S = argmax_{S_i \in Senses_D(w)} P(S_i \mid f_1, \cdots, f_m) = \\ argmax_{S_i \in Senses_D(w)} \frac{P(f_1, \cdots, f_m \mid S_i) P(S_i)}{P(f_1, \cdots, f_m)} = \\ argmax_{S_i \in Senses_D(w)} P(S_i) \prod_{j=1}^{m} P(f_j \mid S_i) \end{aligned} \tag{2.7}$$

onde $m$ é o número de atributos e a última parte da fórmula é obtida com base na suposição de que os atributos são condicionalmente independentes, dado o sentido (o denominador também é descartado, pois não influencia nos cálculos).

As probabilidades de $P(S_i)$ e $P(f_j \mid S_i)$ são estimadas, respectivamente, como as frequências de ocorrência relativas no conjunto de treinamento do sentido $S_i$ e do atributo $f_j$ na presença do sentido $S_i$. Para evitar resultados zerados, deve-se ponderar dividindo $P(S_i)$ por $N$, onde $N$ é o tamanho do conjunto de treinamento (NG, 1997; ESCUDERO et al., 2000b). Ao classificar a ocorrência do substantivo *banco* na sentença *O banco cobrou meu cheque*, a rede bayesiana é a mostrada na Figura 2.3.

**Figura 2.3: Exemplo de Rede Bayesiana.**




**Fonte: Retirado de (ENEKO; EDMONDS, 2007)**

onde os atributos, para esse exemplo específico, recebem os seguintes valores:

- $w-1 = o$
- $w+1 = cobrou$
- $head = banco$
- $subj-verb = banco$
- $verb-obj = cheque$

onde os dois últimos atributos codificam o papel gramatical do substantivo *banco* como sujeito e o do substantivo *cheque* como objeto direto na sentença. Supondo que as probabilidades dessas cinco características, dado o sentido financeiro de *banco*, são:

- $P(w-1 = o \mid banco/FINANCEIRO) = 0,66$
- $P(w+1 = cobrou \mid banco/FINANCEIRO) = 0,35$
- $P(head = banco \mid banco/FINANCEIRO) = 0,76$
- $P(subj-verb = banco \mid banco/FINANCEIRO) = 0,44$
- $P(verbo-obj = cheque \mid banco/FINANCEIRO) = 0,60$

Estima-se a probabilidade de ocorrência de $P(banco/FINANCEIRO) = 0,36$. O resultado final é:

$$score(banco/FINANCEIRO) = 0,36 * 0,66 * 0,35 * 0,76 * 0,44 * 0,6 = 0,016 \quad (2.8)$$

A probabilidade do sentido financeiro de *banco* ocorrer nesta sentença é de $0,016$. Apesar de sua simplicidade, Naive Bayes é utilizado em muitos trabalhos da literatura com uma precisão do estado da arte na DLS supervisionada (MOONEY, 1996; NG, 1997; LEACOCK et al., 1998). Mooney (1996) obtiveram uma acurácia de aproximadamente 73% com Naive Bayes, contra aproximadamente 70% de uma rede neural *Perceptron* e aproximadamente 55% de uma árvore de decisão C4.5. O experimento teve como objetivo desambiguar 6 sentidos da palavra *line* (linha), empregada em sentenças de um *dump* do *Wall Street Journal* (inglês) de 1987-89, com 25 milhões de palavras. Na literatura, não foram encontrados trabalhos de DLS para o português do Brasil baseados em métodos supervisionados probabilísticos.

### 2.3.2 Baseados em similaridade de exemplos

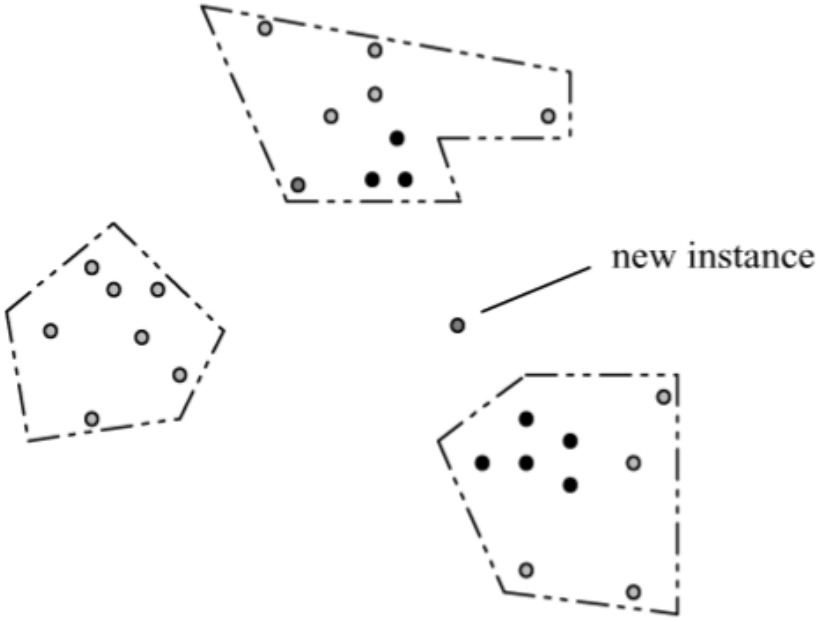
Os métodos baseados em similaridade de exemplos realizam a desambiguação aplicando métricas de similaridade, comparando novos exemplos com um conjunto de exemplos pré-cadastrados para cada sentido da palavra. Uma das formas de calcular a similaridade entre dois exemplos é através do Modelo de Espaço Vetorial, do inglês *Vector Space Model* (VSM), considerando o ângulo formado pelos vetores dos exemplos. Schutze (1992) aplicou este modelo codificando cada palavra do contexto em um vetor de palavras que representa a frequência de suas ocorrências. Desta forma, cada palavra-alvo é representada como um vetor, calculado através da soma dos vetores das palavras relacionadas ao contexto.

Miller et al. (1993) compararam VSM, redes neurais e Naive Bayes, e concluíram que os dois primeiros superam o último na DLS (para o inglês). Yarowsky et al. (2001) combinaram seis classificadores supervisionados, incluindo VSM, e obtiveram bons resultados no *dataset Senseval-2*. Para o treinamento, usaram um conjunto rico de recursos (incluindo informações sintáticas) e ponderação de tipos de características (AGIRRE et al., 2005).

Outro algoritmo representativo dos métodos baseados em similaridade de exemplos, e o mais utilizado, é o algoritmo *k-Nearest Neighbor* (kNN) (PETERSON, 2009). Neste algoritmo, a classificação de um novo exemplo é realizada com base nos $k$ exemplos mais próximos, recuperados em um conjunto pré-cadastrado e rotulado com sentidos. No caso mais simples, o passo de treinamento armazena todos os exemplos na memória e a generalização é feita na entrada

de um novo exemplo a ser classificado. Um exemplo do kNN é mostrado na Figura 2.4, onde as instâncias atribuídas ao mesmo sentido estão dentro de polígonos e os pontos pretos são os vizinhos mais próximos da nova instância. A nova instância é atribuída ao polígono inferior, pois este tem cinco pontos pretos contra três do polígono superior.

**Figura 2.4: Exemplo de algoritmo kNN.**




**Fonte: Retirado de (ENEKO; EDMONDS, 2007)**

Uma questão muito importante no emprego deste algoritmo é a definição de uma métrica de similaridade (ou distância) apropriada para a tarefa, que deve levar em consideração a importância relativa de cada atributo, sendo eficiente computacionalmente.

O esquema de combinação para decidir o sentido resultante entre os vizinhos mais próximos de $k$ também leva a vários algoritmos alternativos. Os primeiros trabalhos com kNN para DLS foram desenvolvidos por Ng e Lee (1996) e Ng (1997) no córpus DSO. Ng e Lee (1996) obtiveram uma acurácia de 68,6%, superando os 63,7% do *baseline* de sentido mais frequente. Ng (1997) identificou o valor ótimo de $k$ automaticamente para cada palavra e obteve 74,5% de acurácia, melhorando os resultados obtidos anteriormente. Na literatura, não foram encontrados trabalhos de DLS para o português do Brasil usando métodos supervisionados de similaridade de exemplos.

### 2.3.3 Baseados em regras discriminativas

Os métodos baseados em regras discriminativas aprendem condições associadas a cada sentido da palavra. Após o aprendizado baseado no córpus de treinamento, o sistema verifica regras que melhor casam com o sentido de uma palavra ambígua determinando, assim, que este é o sentido correto. Esses métodos podem ser classificados em:

- **Métodos baseados em listas de decisão**: Uma lista de decisão (RIVEST, 1987) é um conjunto de regras ordenadas que servem para atribuir o sentido apropriado a uma palavra ambígua. Trata-se de uma lista de regras do tipo *if-then-else* para a qual um conjunto de treinamento é usado para induzir um conjunto de atributos para a condição testada no *if*. Como resultado, são criadas regras do tipo (valor-atributo, sentido, *score*). A lista de decisão é, então, formada por essas regras ordenadas decrescentemente por seus *scores*. Dada uma ocorrência da palavra *w* e sua representação como um vetor de atributos, a lista de decisão é verificada e o sentido atribuído é aquele com maior *score*.

  Um exemplo simplificado de uma lista de decisão é mostrado na Tabela 2.1. A primeira regra do exemplo se aplica ao sentido de *banco* como instituição financeira (Bank/FINANCE) e espera uma *conta bancária* como contexto à esquerda (*account with bank*). A terceira regra se aplica ao *banco* no sentido de fornecimento (por exemplo, banco de sangue), e assim por diante, onde outras regras poderiam prever determinado sentido de uma palavra.

**Tabela 2.1: Exemplo de lista de decisão**

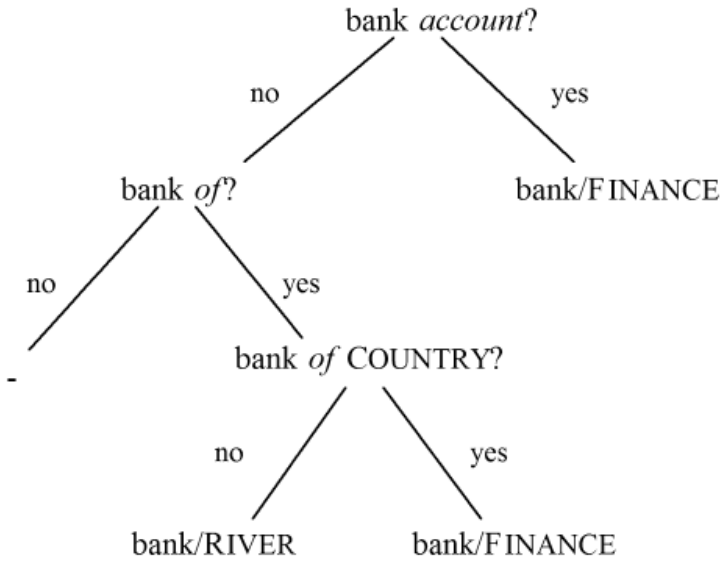
| Regra | Sentido | *Score* |
|---|---|---|
| account with bank | Bank/FINANCE | 4.83 |
| stand/V on/P...bank | Bank/FINANCE | 3.35 |
| bank of blood | Bank/SUPPLY | 2.48 |
| work/V...bank | Bank/FINANCE | 2.33 |
| the left/J bank | Bank/RIVER | 1.12 |
| of the bank | - | 0.01 |

**Fonte: Adaptado de (ENEKO; EDMONDS, 2007)**

  As listas de decisão foram a técnica mais bem sucedida nas primeiras competições de avaliação do *Senseval*, por exemplo Yarowsky (2000) obteve 73,4% de precisão na avaliação em um *dataset* com 36 palavras anotadas com sentido (do inglês). Martinez e Agirre (2000) aplicaram a técnica na tentativa de aliviar o gargalo de aquisição de conhecimento causado pela falta de córpus anotado manualmente com sentidos, porém, obtiveram apenas 70% de precisão (inglês).

- **Métodos baseados em árvores de decisão**: Uma árvore de decisão é um modelo preditivo usado para representar regras de classificação em uma estrutura de árvore que divide recursivamente o conjunto de dados de treinamento. Cada nó interno de uma árvore de decisão representa um teste em um valor de atributo, e cada ramo representa um resultado do teste. Uma predição é feita quando um nó folha é alcançado.
  Nas últimas décadas, as árvores de decisão raramente foram aplicadas à DLS. Um algoritmo popular para árvores de decisão é o C4.5 (QUINLAN, 2014), uma extensão do algoritmo ID3 (QUINLAN, 1986). Em um experimento comparativo com vários algoritmos de aprendizado de máquina para DLS, Mooney (1996) concluiu que as árvores de decisão obtidas com o algoritmo C4.5 são superadas por outras abordagens supervisionadas como métodos bayesianos e de redes neurais. Apesar de representarem o modelo preditivo de forma compacta e legível para humanos, elas apresentam diversos problemas, como a dispersão dos dados devido ao alto número de atributos e a falta de confiabilidade das previsões geradas a partir de pequenos conjuntos de treinamento. Um exemplo de uma árvore de decisão para DLS é mostrado na Figura 2.5. Por exemplo, se o substantivo "banco" deve ser classificado na sentença *Nós sentamos em um banco de areia*, a árvore é percorrida e, depois de seguir o caminho *no-yes-no*, o sentido *bank/RIVER* é escolhido.

**Figura 2.5: Exemplo de árvore de decisão.**




**Fonte: Retirado de (ENEKO; EDMONDS, 2007)**

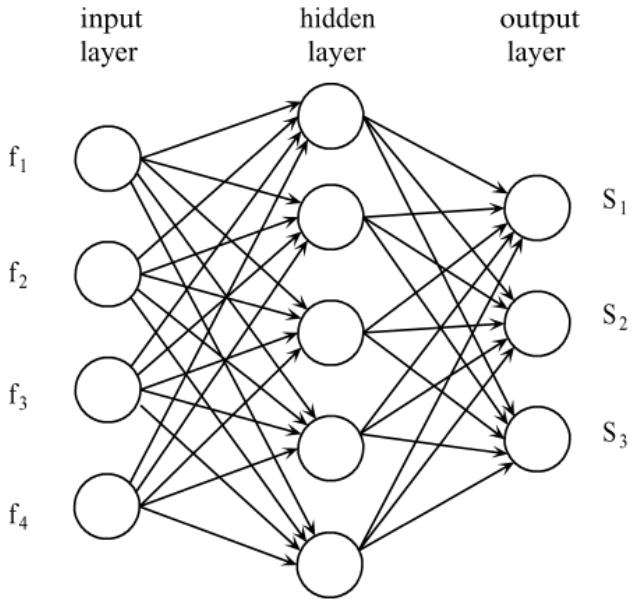
Para o português, nesse caso o europeu, Travanca (2013) propôs duas abordagens para a desambiguação lexical de sentidos de verbos. A primeira, baseada em regras, faz uso das

descrições lexicais, sintáticas e semânticas dos sentidos dos verbos presentes em um recurso lexical chamado ViPEr (MAMEDE et al., 2012). A segunda, usa aprendizado de máquina via *framework* MegaM (III, 2004), baseado nos modelos de máxima entropia (BERGER et al., 1996) com um conjunto de atributos comumente usados em DLS para determinar o sentido correto de um verbo. Para as avaliações, o *baseline* usado foi o método do sentido mais frequente, que obteve 84% de acurácia. O método proposto baseado em regras obteve 64,82% e o baseado em aprendizagem de máquina obteve 79,15%. Os dois métodos juntos obtiveram uma acurácia de 87,2%, superando o *baseline*.

### 2.3.4 Baseados em redes neurais

Uma rede neural é um grupo interconectado de neurônios artificiais que usa um modelo computacional para o processamento de dados com base em uma abordagem conexionista. Assim como nos outros métodos de AM supervisionados, pares de entrada e saída desejada são inseridos no algoritmo de aprendizagem com o objetivo de generalizar conhecimento. À medida que são fornecidos novos pares, os pesos são ajustados progressivamente de modo que a resposta desejada tenha uma ativação maior do que qualquer outra unidade de saída. A Figura 2.6 mostra uma rede neural *perceptron* multicamada (do inglês, *Multilayer Perceptron* ou MLP), alimentada com os valores de quatro atributos ($f_1, f_2, f_3, f_4$) e que exibe o *score* de três sentidos para uma palavra ($S_1, S_2, S_3$).

**Figura 2.6: Exemplo de rede neural *feedforward*.**




**Fonte: Retirado de (ENEKO; EDMONDS, 2007)**

Os pesos na rede podem ser positivos ou negativos, permitindo que evidências sejam acumuladas a favor ou contra um sentido. Cottrell (1985) utilizou redes neurais para representar palavras como nós, de modo que as palavras ativam os conceitos aos quais são semanticamente relacionadas e vice-versa. A ativação de um nó provoca a ativação de nós conectados por ligações excitatórias e a desativação de nós, conectados por ligações inibitórias (sentidos concorrentes para a mesma palavra).

Nancy e Véronis (1990) construíram uma rede neural a partir das definições do dicionário *Collins English Dictionary*. Eles conectaram cada sentido do dicionário às palavras que ocorrem no texto e obtiveram 90% de precisão contra 83% do algoritmo de Lesk (*baseline*). Towell e Voorhees (1998) descobriram que as redes neurais funcionam melhor sem o uso de camadas ocultas e utilizaram *perceptrons* para ligar atributos de entrada aos sentidos. Eles avaliaram o modelo proposto em sentenças com palavras ambíguas como *line* (linha), *serve* (servir) e *hard* (difícil). A precisão foi de 87%, 90% e 81% respectivamente.

Em vários estudos, as redes neurais mostraram-se eficazes em comparação com outros métodos supervisionados (MILLER et al., 1993; TOWELL; VOORHEES, 1998; MOONEY, 1996). Como principais desvantagens das redes neurais, há a dificuldade em interpretar os resultados, a necessidade de uma grande quantidade de dados para o treinamento e o ajuste de parâmetros.

Na literatura, não foram encontrados trabalhos de DLS para o português do Brasil usando métodos supervisionados de redes neurais.

## 2.4 Métodos não supervisionados baseados em córpus

A abordagem não supervisionada para a DLS tem a vantagem de fazer uso de córpus sem anotação prévia de sentidos, porém, demanda que esses córpus sejam representativos quanto às palavras semanticamente ambíguas. O objetivo dos algoritmos é identificar grupos de sentidos similares presentes no córpus de treinamento e transformá-los em *clusters*. Diante de uma nova instância ambígua, a maior similaridade entre a entrada e um dos *clusters* determina o sentido mais provável desta palavra, em seu contexto. O conjunto de *clusters* e, com isso, o grau de generalidade/especialização dos sentidos depende muito do algoritmo usado e dos seus parâmetros. Os grupos obtidos podem ou não corresponder a diferentes níveis de uma hierarquia lexical padrão. A desvantagem é que nem sempre os *clusters* de sentidos gerados são bem definidos, o que pode dificultar a identificação clara dos sentidos encontrados.

A seguir são apresentados os principais métodos não supervisionados para DLS.

### 2.4.1 Métodos distributivos

Os métodos distributivos assumem que o sentido de uma palavra está relacionado à distribuição de palavras em torno dela, suposição que está baseada na hipótese distribucional de (HARRIS, 1954 apud ZANZOTTO et al., 2010) e na célebre frase de Firth (1957): *Você conhecerá uma palavra através de sua companhia!*. Para ilustrar essa ideia, a seguir é mostrado um exemplo:

- *A garrafa de vinho está na mesa.*
- *Todo mundo gosta de vinho.*
- *Vinho faz você ficar bêbado.*
- *Nós fazemos vinho de uva.*

Os contextos em que a palavra *vinho* ocorre sugerem que pode ser algum tipo de bebida alcoólica feita a partir da uva. O método de distribuição chega a esse significado representando recursos do contexto de *vinho* que podem se sobrepor a características de palavras semelhantes como cerveja, licor, tequila, etc. Podemos, então, representar uma palavra $w$ como um vetor de características, onde uma característica binária $f_i$ representa cada uma das $N$ palavras no léxico $v_i$. Atribui-se 1 caso $w$ e $v_i$ ocorram em alguma janela de contexto e 0 caso contrário. O significado da palavra $w$ pode ser representado como o vetor de características:

$$w = (f_1, f_2, f_3, \cdots, f_N)$$

Se $w$ = vinho, $v_1$ = garrafa, $v_2$ = bêbado, $v_3$ = sofá e $v_4$ = mesa o vetor de características de $w$ no córpus acima seria:

$$w = (1, 1, 0, 1 \cdots)$$

Dadas duas palavras representadas por esses vetores esparsos de características, pode-se aplicar uma medida de distância e dizer que as palavras são semelhantes se seus vetores estiverem próximos no espaço vetorial. A Figura 2.7 mostra um exemplo de similaridade vetorial para as palavras "damasco" (*apricot*), "abacaxi" (*pineapple*), "digital" (*digital*) e "informações" (*information*). Com base no significado dessas quatro palavras, uma métrica de similaridade mostra que "damasco" e "abacaxi" são similares, assim como "digital" e "informação". São mostradas 8 dimensões dos vetores de co-ocorrência (*arts*, *boil*, *data*, *function*, *large*, *sugar*, *summarized* e *water*), calculados a partir de palavras que ocorrem dentro de um contexto de duas linhas no córpus *Brown*. Os vetores de "damasco" e "abacaxi" são mais similares do que

os de “damasco” e “informação”. Como os vocabulários são grandes e a maioria das palavras não ocorrem próxima às outras, os vetores são bastante esparsos.

**Figura 2.7: Vetor de co-ocorrência para quatro palavras do córpus Brown, mostradas em 8 dimensões.**

| | arts | boil | data | function | large | sugar | summarized | water |
|---|---|---|---|---|---|---|---|---|
| **apricot** | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |
| **pineapple** | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |
| **digital** | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 |
| **information** | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 |

**Fonte: Retirado de (JURAFSKY; MARTIN, 2008)**

A similaridade semântica entre duas palavras pode ser dada pelo cosseno entre seus vetores correspondentes. Em geral, o cosseno é definido como na Equação 2.9, onde $\vec{x}$ e $\vec{y}$ são os vetores sendo comparados. Esse valor mede a distância entre os diferentes contextos de ocorrência das palavras que estão sendo comparadas.

$$cos(\vec{x}|\vec{y}) = \frac{\vec{x}.\vec{y}}{|\vec{x}||\vec{y}|} \tag{2.9}$$

Schütze (1998) divide a tarefa de descoberta de sentidos a partir de contextos em dois passos. O primeiro é descobrir os diferentes significados de uma palavra dividindo os contextos em que ocorrem em *clusters*. O segundo é rotular cada *cluster* com o sentido que descreva o significado da palavra nesses contextos. Em resumo, ao invés de criar uma definição tradicional, um conjunto de contextos de palavras que estão associados a um *cluster* pode ser usado como uma aproximação do sentido. A seguir, são mostrados dois métodos de descoberta de sentido a partir do contexto: a clusterização de contextos e a clusterização de palavras.

#### 2.4.1.1 Clusterização de contextos

Na clusterização de contextos, o objetivo é agrupar os contextos nos quais uma determinada palavra-alvo ocorre, de modo que os *clusters* resultantes sejam formados por contextos nos quais a palavra-alvo ocorre com o mesmo sentido. Cada contexto no qual a palavra-alvo ocorre é um membro de um dos *clusters*. Esse método também é chamado de baseado em *token*, uma vez que cada ocorrência da palavra-alvo (ou seja, cada *token*) é preservada. As entradas para a descoberta de sentidos baseada na clusterização de contextos são contextos anotados com a palavra-alvo, como mostrado abaixo. Embora pareça semelhante à entrada de algoritmos supervisionados, vale ressaltar que nesse caso não há tags de sentido incluídas nos dados, por isso a anotação é apenas da palavra ambígua e não de qual sentido deve ser atribuído a ela.

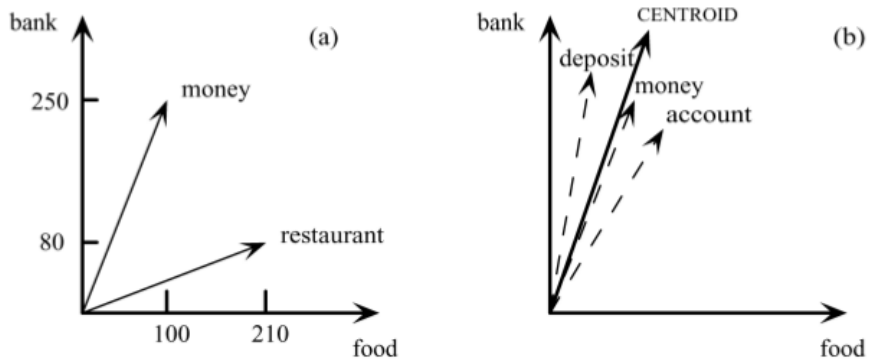
- **Cluster 1:**
  *O **banco** funciona 24 horas.*
  *O **banco** aprovou meu financiamento.*

- **Cluster 2:**
  *O **banco** está com déficit de sangue.*
  *Deve-se armazenar esses dados em um **banco**.*

A desambiguação de uma palavra pode ser feita através de uma métrica de similaridade (cosseno) que aproxima o centroide do *cluster* de seu contexto. Dessa forma, pode-se inferir que o sentido representado por esse *cluster* é o sentido correto da palavra ambígua. Na sentença *Acerte suas dívidas com o banco.*, o contexto do *token* "banco" se aproximaria mais do *cluster* 1 do que do 2, pois contém palavras que ocorrem com maior frequência no contexto de banco como instituição financeira.

Uma abordagem histórica desse tipo baseia-se na ideia de espaço de palavras (SCHUTZE, 1992), ou seja, um espaço vetorial cujas dimensões são palavras. Uma palavra $w$ em um córpus pode ser representada como um vetor cujo $j$-ésimo componente conta o número de vezes que a palavra $w_j$ co-ocorre com $w$ dentro de um contexto fixo (uma frase ou um contexto maior). A hipótese desse modelo é a de que o perfil distributivo das palavras expressa implicitamente a sua semântica. A Figura 2.8 (a) mostra dois exemplos de vetores de palavras, *restaurant* = (210, 80) e *money* = (100, 250), onde a primeira dimensão representa a contagem de co-ocorrências com *food* e a segunda conta as co-ocorrências com *bank*. A semelhança entre duas palavras $v$ e $w$ pode, então, ser medida geometricamente, por exemplo, pelo cosseno entre os vetores $v$ e $w$ correspondentes.

**Figura 2.8: Vetores de palavras em um espaço vetorial de duas dimensões.**




**Fonte: Retirado de (JURAFSKY; MARTIN, 2008)**

Um vetor é calculado para cada palavra em um córpus. Esse tipo de representação combina

os sentidos: um vetor inclui todos os sentidos da palavra que representa. Ao juntar o conjunto de vetores de cada palavra do córpus, obtém-se uma matriz de co-ocorrência. Para lidar com um grande número de dimensões, a Análise Semântica Latente (do inglês *Latent Semantic Analysis*, LSA) pode ser aplicada para reduzir a dimensionalidade do espaço multidimensional resultante através da Decomposição de Valor Singular (do inglês, *Singular Value Decomposition*, SVD) (GOLUB; LOAN, 1989). A SVD encontra os principais eixos de variância no espaço de palavras. A redução da dimensionalidade transforma os vetores de palavras no espaço de alta dimensão em um espaço de dimensão inferior, mesclando termos que possuem significados semelhantes. O objetivo é agrupar vetores de contexto, isto é, vetores que representam o contexto de ocorrências específicas de uma palavra-alvo. Um vetor de contexto é construído como o centroide (média normalizada) dos vetores das palavras que o compõem (SCHUTZE, 1992; SCHÜTZE, 1998). Um exemplo de vetor de contexto é mostrado na Figura 2.8 (b), onde a palavra *stock* co-ocorre com *deposit*, *money* e *account*. Esses vetores de contexto são vetores de segunda ordem, uma vez que eles não representam diretamente o contexto em questão. Finalmente, a discriminação de sentido pode ser realizada agrupando os vetores de contexto de uma palavra-alvo usando um algoritmo de clusterização.

Schütze (1998) propôs um algoritmo, denominado *context-group discrimination*, que agrupa as ocorrências de uma palavra ambígua em *clusters* de sentidos com base na semelhança contextual entre as ocorrências. A semelhança contextual é calculada como descrito acima, enquanto que o agrupamento é realizado com o algoritmo de *Expectation Maximization* (EM), um procedimento de estimativa de máxima verossimilhança iterativa de um modelo probabilístico (DEMPSTER et al., 1977). Uma abordagem de agrupamento diferente consiste em *agglomerative clustering* (PEDERSEN; BRUCE, 1997) na qual, inicialmente, cada instância constitui um *cluster* único. Em seguida, o *agglomerative clustering* combina o par de *clusters* mais parecido e continua com pares sucessivamente menos parecidos até atingir um limite de parada. Os autores avaliaram o método em grupos de palavras (do inglês) divididos pelas classes gramaticais (adjetivos, substantivos e verbos) em comparação com o *baseline* de sentido mais frequente, que obteve 57% de acurácia. A única classe que superou o *baseline* foi a de substantivos já que, segundo os autores, possuem uma distribuição de sentidos mais uniforme. A acurácia obtida nessa classe foi de aproximadamente 62%.

Um problema na construção de vetores de contexto é que uma grande quantidade de dados de treinamento (não anotados) são necessários para determinar uma distribuição significativa de co-ocorrências de palavras.

#### 2.4.1.2 Clusterização de palavras

Uma abordagem diferente para a indução de sentidos de palavras envolve o agrupamento de palavras que são semanticamente semelhantes e, portanto, podem transmitir um significado específico. Uma abordagem bem conhecida para o agrupamento de palavras (LIN, 1998) consiste na identificação das palavras $W = (w_1, \cdots, w_k)$ semelhantes (possivelmente sinônimos) a uma palavra alvo $w_0$. A semelhança entre $w_0$ e $w_i$ é determinada com base nas informações de suas características únicas, dadas pelas dependências sintáticas que ocorrem em um córpus (por exemplo, sujeito-verbo, verbo-objeto, adjetivo-substantivo, etc.). Quanto mais dependências as duas palavras compartilham, maior a semelhança entre elas. Lin (1998) não compara seus resultados à nenhum *baseline*.

Para discriminar entre os sentidos, um algoritmo de clusterização de palavras é aplicado. Seja $W$ a lista de palavras semelhantes ordenadas por grau de similaridade a $w_0$. Uma árvore de similaridade $T$ é criada, inicialmente, consistindo em um único nó $w_0$. Em seguida, para cada $i \in \{1, \cdots, k\}, w_i \in W$ é adicionado como filho de $w_j$ na árvore $T$ tal que $w_j$ é a palavra mais semelhante a $w_i$ entre $\{w_0, \cdots, w_{i-1}\}$. Após um passo de poda, cada sub-árvore a partir de $w_0$ é considerada como um sentido distinto de $w_0$. Abaixo tem-se um exemplo de 4 *clusters* de palavras gerados:

- **Cluster 1:** *linha, cordão, gravata, cabo*
- **Cluster 2:** *linha, telefone, ocupado, chamada*
- **Cluster 3:** *banco, financeiro, dinheiro, conta, agência*
- **Cluster 4:** *banco, areia, depósito, terreno*

Os *clusters* resultantes não incluem nenhuma informação dos contextos individuais de cada palavra. Uma palavra é considerada ambígua se ocorrer em mais de um *cluster*, como é o caso de "linha" e "banco". A desambiguação lexical de sentidos é feita através das demais palavras que estão nos *clusters*, logo, "banco" do *cluster* 3 terá o sentido de instituição financeira devido às palavras "financeiro", "dinheiro", "conta" e "agência" que ocorrem nesse *cluster*. Seguindo a mesma ideia, a ocorrência "banco" do *cluster* 4 terá o sentido de "depósito".

Na literatura, não foram encontrados trabalhos de DLS para o português do Brasil baseados em métodos não supervisionados usando métodos distributivos.

### 2.4.2 Equivalência translacional

Outra estratégia empregada para a DLS não supervisionada faz uso de textos paralelos (textos que são a tradução uns dos outros) e considera que diferentes sentidos de uma palavra são normalmente traduzidos para diferentes palavras em outra língua. Assim, a lexicalização em diferentes línguas pode ser usada para definir e estruturar distinções de sentido (RESNIK; YAROWSKY, 1997, 1999).

A estratégia consiste em desambiguar palavras rotulando-as com a tradução apropriada. Por exemplo, a palavra *sentence* do inglês pode ser traduzida para *peine* ou *phrase*, no francês, dependendo do contexto. No entanto, esse método não realiza a desambiguação completa, pois é comum que diferentes significados da mesma palavra tenham a mesma tradução (por exemplo, os sentidos de *wing* como um órgão e como parte de um edifício são ambos traduzidos para *ala*, no italiano).

Resnik e Yarowsky (1997) propuseram que fossem considerados apenas os sentidos que são lexicalizados transversalmente em um conjunto mínimo de linguagens. Por exemplo, a palavra *table* no inglês é traduzida como *table* em francês e *tavola* em italiano, tanto no sentido de mesa (móvel) como no sentido de pessoas em uma mesa. Essa ambiguidade é preservada nas três línguas e permite a identificação de um único sentido. Para implementar esta proposta, Ide (2000) sugeriu o uso de um índice de coerência para identificar a tendência de lexicalizar os sentidos de forma diferente em todos os idiomas.

Gale et al. (1992a) propuseram um método que usa córpus paralelos para a criação automática de um conjunto de dados anotado com sentidos. Dada uma palavra-alvo, cada frase no idioma de origem é anotada com a tradução da palavra no idioma de destino. Um classificador Naive Bayes é treinado com o conjunto de dados resultante e aplicado em uma tarefa de DLS. Os testes mostraram uma precisão muito alta (acima de 90%) em um pequeno conjunto de palavras.

Diab e Resnik (2002) utilizaram córpus paralelos com base no fato de que as lexicalizações da mesma palavra em duas línguas diferentes preservam algumas características semânticas do núcleo. Nesse processo, o córpus da língua-alvo também é rotulado com sentidos. Nos experimentos realizados, o francês era a língua de origem e o inglês era o idioma alvo. O córpus paralelo em inglês e o inventário de sentidos foram utilizados nos experimentos. O algoritmo é dividido em quatro etapas principais:

1. No primeiro passo, são identificadas as palavras no córpus alvo (inglês) e suas traduções correspondentes no córpus de origem (francês).

2. Na segunda etapa, *clusters* são formados a partir da similaridade das palavras do idioma alvo (inglês).

3. Na terceira etapa, dentro de cada um desses *clusters*, são selecionados os sentidos com maior similaridade semântica com todas as palavras do *cluster*.

4. Finalmente, os sentidos das palavras no idioma alvo (inglês) são projetados nas palavras correspondentes na língua de origem. Como resultado, um grande número de palavras francesas recebe sentidos do inventário de sentidos do inglês.

Esse método foi avaliado usando os dados do teste *Senseval-2* e a precisão obtida foi de 59,4% e cobertura de 54,5%. Os autores não compararam os resultados com um *baseline*.

O principal problema das abordagens multilíngues está no gargalo da aquisição de conhecimento, pois há uma falta de córpus paralelos para várias línguas.

Na literatura, não foram encontrados trabalhos de DLS para o português do Brasil baseados em métodos não supervisionados usando equivalência translacional.

## 2.5 Considerações finais

A seguir, na Tabela 2.2, são resumidas algumas características importantes dos principais trabalhos citados neste capítulo.

**Tabela 2.2: Resumo dos principais trabalhos citados neste capítulo**

| Trabalho | Metodologia | Idioma | Recurso | Avaliação |
|---|---|---|---|---|
| Lesk (1986) | Sobreposição Contextual | EN | *Oxford Advanced Learner's Dictionary* | Precisão = 50%-70% |
| Stetina e Nagao (1998) | Similaridade Semântica | EN | WordNet<br>SemCor | Precisão = 80,3% |
| Agirre e Martinez (2001) | Preferências de Seleção | EN | Brown<br>SemCor<br>WordNet | Precisão = 66,6%<br>Cobertura = 97,3% |
| Nóbrega e Pardo (2012) | Sentido mais frequente | PT-BR | CSTNews<br>WordNet Pr | Precisão = 51%<br>Cobertura = 51%<br>Acurácia = 51% |
| Krovetz (1998) | Um sentido por discurso | EN | DSO<br>SemCor | Precisão = 68% |
| Martinez e Agirre (2000) | Um sentido por colocação | EN | DSO<br>SemCor<br>WordNet-Pr | Precisão = 70% |
| Specia (2007) | Conhecimento+AM | PT-BR | Diversos | Precisão = 94,2% |
| Mooney (1996) | Naive Bayes | EN | *Wall Street Journal* | Acurácia = 73% |
| Ng (1997) | kNN | EN | DSO | Acurácia = 74,5% |
| Travanca (2013) | Conhecimento<br>AM<br>Conhecimento+AM | PT-EU | CETEMPúblico | Acurácia = 64,82%<br>Acurácia = 79,15%<br>Acurácia = 87,20% |
| Nancy e Véronis (1990) | Redes Neurais | EN | *Collins English Dictionary* | Precisão = 90% |
| Schütze (1998) | Clusterização de Contextos | EN | *Wall Street Journal* | Precisão = 62% |
| Diab e Resnik (2002) | Equivalência Translacional | FR-EN | *Senseval-2* | Precisão = 59,4%<br>Cobertura = 54,5% |

**Fonte: Autoria própria**

# Capítulo 3

## VETORES DE PALAVRAS (*Word Embeddings*)

Essa pesquisa se propôs a fazer DLS de forma indireta, onde a modelagem de sentidos de uma palavra ambígua é feita durante a geração de representações vetoriais, utilizando AM não supervisionado baseado em córpus: método distributivo (seção 2.4.1). Esses vetores gerados são posteriormente utilizados como *feature* em tarefas de PLN.

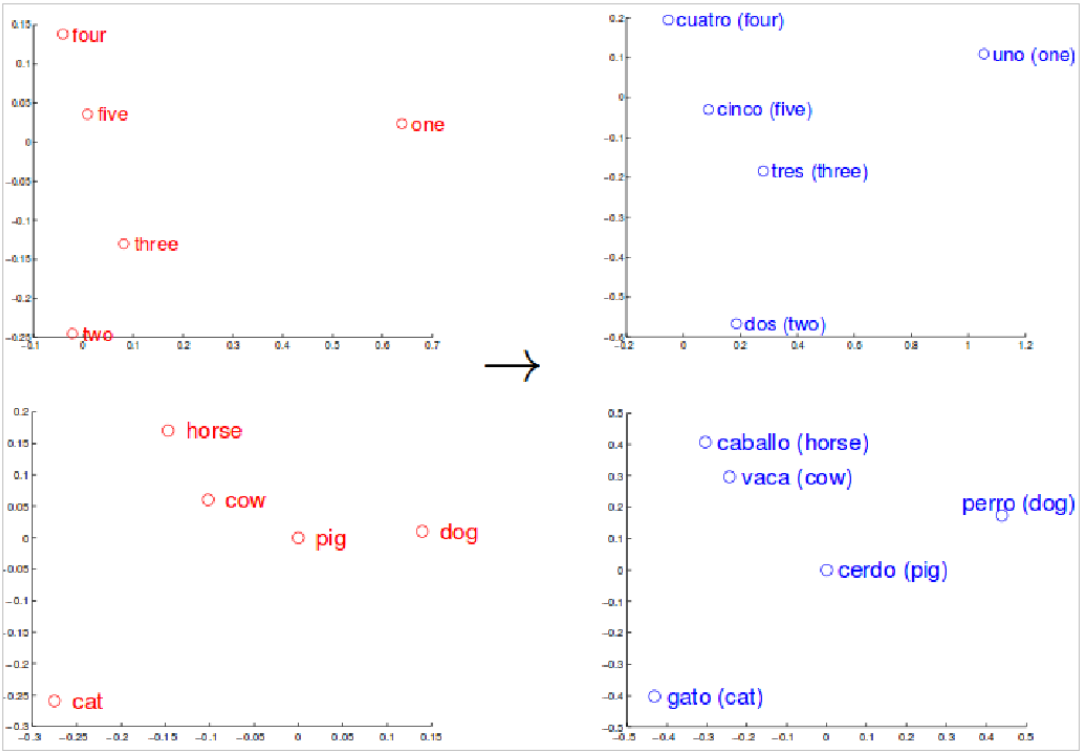
A representação semântica distribuída é baseada na hipótese distribucional que estabelece que o sentido de uma palavra é dado por seu contexto de ocorrência (BRUNI et al., 2014). Esses vetores de palavras podem ser usados como recursos em uma variedade de aplicações, tais como: classificação de documentos (SEBASTIANI, 2002), perguntas e respostas (TELLEX et al., 2003) e reconhecimento de entidade nomeada (TURIAN et al., 2010).

A representação de palavras como vetores contínuos tem uma longa história (HINTON, 1984; HINTON et al., 1985; ELMAN, 1990). Muitos tipos diferentes de modelos foram propostos para estimar representações contínuas de palavras, incluindo a Análise Semântica Latente (do inglês, *Latent Semantic Analysis* – LSA) e a Alocação Latente de Dirichlet (do inglês, *Latent Dirichlet Allocation* – LDA). Já as representações distribuídas de palavras aprendidas por redes neurais apresentam um desempenho significativamente superior ao LSA ao preservar regularidades lineares entre as palavras (MIKOLOV et al., 2013; ZHILA et al., 2013). Quanto ao LDA, sabe-se que ele é computacionalmente caro quando usado em grandes conjuntos de dados.

Uma arquitetura muito popular para estimar um modelo de linguagem neural (do inglês, *neural network language model* – NNLM) foi proposta em (BENGIO et al., 2003), onde uma rede neural *feedforward*, com uma camada linear e uma camada escondida não-linear, foi usada para aprender a representação vetorial de palavras e um modelo de linguagem estatístico. Este trabalho tem sido seguido por muitos outros.

Outra arquitetura interessante de NNLM foi apresentada por Mikolov et al. (2009), onde os vetores de palavras são primeiro aprendidos usando uma rede neural com uma única camada oculta. Os vetores de palavras são, então, usados para treinar o NNLM. Assim, os vetores de palavra são aprendidos mesmo sem construir o NNLM completo. Mikolov et al. (2013) estende diretamente essa arquitetura, focando apenas no primeiro passo onde os vetores de palavras são aprendidos usando um modelo simples. O objetivo é gerar vetores contendo números de tal forma que palavras similares de acordo com seus contextos estarão "próximas" no espaço vetorial, como ilustra a Figura 3.1. Segundo Mikolov et al. (2013), os vetores em cada língua foram projetados para 2 dimensões usando PCA e rotacionados manualmente para enfatizar a similaridade.

**Figura 3.1: Representações distribuídas para palavras em inglês (à esquerda, em vermelho) e espanhol (à direita, em azul).**




**Fonte: Retirado de (MIKOLOV et al., 2013)**

As principais ferramentas utilizadas na atualidade para modelar uma representação vetorial distribuída são: Word2Vec, GloVe, Wang2Vec e FastText. As próximas seções trazem explicações sobre essas ferramentas, com: seus algoritmos, seus processos de treinamento e algumas avaliações reportadas na literatura. Contudo, antes de apresentar as diferentes abordagens propostas para a geração de vetores de palavras, descreve-se brevemente a metodologia de avaliação empregada nessa área.

## 3.1 Avaliação

Assim como ocorre na avaliação dos métodos de DLS, os vetores de palavras também podem ser avaliados intrínseca ou extrinsecamente. Nessa avaliação, além das medidas tradicionais (precisão, acurácia, etc.) apresentadas no capítulo anterior (veja seção 2.1), outras também podem ser utilizadas. Para a avaliação intrínseca de vetores de palavras, a maioria dos autores compara a similaridade nas anotações de sentido humana (considerada correta) e automática, o que se convenciona chamar de *correlação*.

Uma das medidas estatísticas mais utilizada para calcular essa correlação é a de *Spearman*, que mede a intensidade da relação entre variáveis ordinais e usa a ordem das observações em vez do valor observado. Deste modo, este coeficiente não é sensível a assimetrias na distribuição, nem à presença de *outliers*, não exigindo, portanto, que os dados provenham de duas populações normais. O coeficiente varia entre -1 e 1. Quanto mais próximo estiver destes extremos, maior será a associação entre as variáveis, negativa ou positivamente, respectivamente. A fórmula 3.1 mostra como o cálculo do coeficiente de *Spearman* é feito (BOLBOACA; JÄNTSCHI, 2006).

$$\rho = 1 - \frac{6\sum d_i^2}{(n^3 - n)} \tag{3.1}$$

$n$: número de pares $(x_i, y_i)$

$d_i$: (posição de $x_i$ dentre os valores de $x$) - (posição de $y_i$ dentre os valores de $y$)

Outra medida utilizada na avaliação intrínseca de vetores de palavras é o coeficiente de concordância de *Kendall*, que indica o grau de concordância das avaliações feitas por diversos avaliadores nas mesmas amostras. O coeficiente de *Kendall* varia de -1 a 1. Quanto maior o coeficiente, mais forte será a associação entre as avaliações. Um coeficiente alto significa que os avaliadores estão aplicando essencialmente o mesmo padrão ao avaliarem as amostras. A fórmula 3.2 mostra como o cálculo do coeficiente de *Kendall* é feito (BOLBOACA; JÄNTSCHI, 2006).

$$\tau = \frac{C - D}{n(n-1)/2} \tag{3.2}$$

C: Quantidade de pares concordantes

D: Quantidade de pares discordantes

O denominador é o número total de combinações de pares, então, o coeficiente deve estar

no intervalo $-1 \leq \tau \leq 1$.

Já na avaliação extrínseca, os vetores de palavras são aplicados em tarefas de PLN e as medidas utilizadas na avaliação são específicas dessas aplicações.

## 3.2 Word2Vec

Partindo da premissa de que técnicas básicas como contagem de n-gramas já estão em seu limite, Mikolov et al. (2013) propõe a utilização de modelos de linguagem baseados em redes neurais para modelar representações distribuídas de palavras. O principal objetivo das técnicas propostas por Mikolov et al. (2013) é aprender vetores de palavras de alta qualidade, a partir de enormes conjuntos de dados com bilhões de palavras. De maneira surpreendente, verificou-se que a similaridade das representações de palavras vai além das simples regularidades sintáticas. Dentro de um espaço de dimensões vetoriais, usando uma simples operação algébrica nos vetores de palavras, foi mostrado por exemplo que o $vetor(rei) - vetor(homem) + vetor(mulher)$ resulta num vetor que está próximo da representação vetorial da palavra *rainha*.

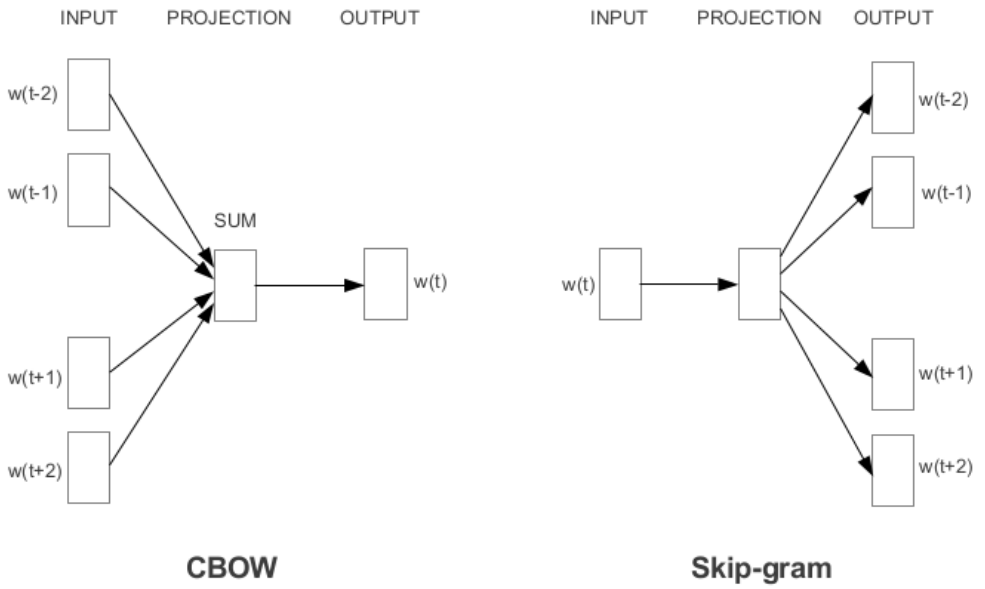
Mikolov et al. (2013) propõem duas arquiteturas de modelos para a aprendizagem de representações distribuídas de palavras que tentam minimizar a complexidade computacional: o modelo *Continuous Bag-of-Words* (CBOW) e o modelo Skip-gram.

- **CBOW** – No CBOW, a arquitetura é semelhante à do NNLM *feedforward*, onde a camada escondida não-linear é removida e a camada de projeção é compartilhada para todas as palavras (não apenas a matriz de projeção). Assim, todas as palavras são projetadas na mesma posição. Essa arquitetura é chamada de modelo de saco de palavras (*bag of words*), pois a ordem das palavras não influencia a projeção.
  O CBOW usa representação distribuída contínua do contexto. A arquitetura do modelo é mostrada na Figura 3.2, na qual pode-se observar que a matriz de pesos entre a entrada e a camada de projeção é compartilhada para todas as posições de palavras (da mesma maneira que no NNLM).

- **Skip-gram** – A arquitetura do Skip-gram é semelhante à do CBOW, mas em vez de prever a palavra atual com base no contexto, Skip-gram tenta maximizar a classificação de uma palavra com base em outra da mesma sentença. Mais precisamente, usa-se cada palavra atual como uma entrada para um classificador log-linear para prever palavras dentro de um intervalo anterior e posterior à palavra atual. O aumento do intervalo melhora a qualidade dos vetores de palavra resultantes, mas também aumenta a complexidade com-

putacional. A distância entre uma palavra do contexto e a palavra atual indica o grau de relação entre elas. Quanto mais distante, menos relacionada estará à palavra atual, podendo receber pesos menores. A arquitetura do Skip-gram também pode ser visualizada na Figura 3.2.

**Figura 3.2: A arquitetura do CBOW (à esquerda) e do Skip-gram (à direita).**




**Fonte: Retirado de (MIKOLOV et al., 2013)**

Conclui-se que quando vetores de palavras de alta dimensionalidade são treinados em uma grande quantidade de dados, os vetores resultantes podem ser usados para descobrir relações semânticas muito sutis entre palavras, como uma cidade e o país ao qual pertence. Por exemplo, *França* está para *Paris* assim como *Alemanha* está para *Berlim*. Os vetores de palavras com tais relacionamentos semânticos podem ser usados para melhorar tarefas de PLN como: Tradução Automática, Recuperação de Informações e Sistemas de Perguntas e Respostas.

Em uma avaliação desses modelos apresentada em (MIKOLOV et al., 2013), o Word2vec foi treinado em um córpus do *Google News* com 6 bilhões de *tokens*. O vocabulário foi restringido para 1 milhão de palavras mais frequentes. O modelo foi validado no conjunto de dados *Semantic-Syntatic Word Relationship* (MIKOLOV et al., 2013) que contém pares de palavras relacionadas semanticamente ou sintaticamente. A Tabela 3.1 mostra cinco pares relacionados semanticamente (nas primeiras 5 linhas) e nove pares relacionados sintaticamente (nas últimas 9 linhas).

Nessa avaliação, o modelo Skip-gram obteve melhor desempenho do que o CBOW, tanto para relações semânticas quanto sintáticas, atingindo 53,3% de acurácia total, como mostrado

**Tabela 3.1: Exemplo de pares de palavras relacionados semanticamente e sintaticamente.**

| Type of relationship | Word Pair 1 | | Word Pair 2 | |
|---|---|---|---|---|
| Common capital city | Athens | Greece | Oslo | Norway |
| All capital cities | Astana | Kazakhstan | Harare | Zimbabwe |
| Currency | Angola | kwanza | Iran | rial |
| City-in-state | Chicago | Illinois | Stockton | California |
| Man-Woman | brother | sister | grandson | granddaughter |
| Adjective to adverb | apparent | apparently | rapid | rapidly |
| Opposite | possibly | impossibly | ethical | unethical |
| Comparative | great | greater | tough | tougher |
| Superlative | easy | easiest | lucky | luckiest |
| Present Participle | think | thinking | read | reading |
| Nationality adjective | Switzerland | Swiss | Cambodia | Cambodian |
| Past tense | walking | walked | swimming | swam |
| Plural nouns | mouse | mice | dollar | dollars |
| Plural verbs | work | works | speak | speaks |

**Fonte: Retirado de (MIKOLOV et al., 2013)**

na Tabela 3.2. A validação foi feita analisando relações semânticas e sintáticas presentes no conjunto de dados *Semantic-Syntatic Word Relationship*.

Para o português, Hartmann et al. (2017) geraram vetores de palavras usando quatro ferramentas: Word2vec, Wang2vec, FastText e Glove. Os vetores gerados foram, então, avaliados intrinsecamente em tarefas de analogia sintática e semântica e extrinsecamente nas tarefas de *PoS tagging* e similaridade semântica sentencial.

O córpus para treinamento foi composto a partir de diversas fontes do português do Brasil (PT-BR) e português europeu (PT-EU), como: o LX-Corpus (RODRIGUES et al., 2016), a Wikipédia (*dump* de 20/10/2016), notícias do GoogleNews, IMDB-PT, G1, dentre outros. Ao todo, obteve-se mais de 1 bilhão de *tokens*.

Para a avaliação intrínseca, foi utilizado um *dataset* de analogias sintática e semânticas de Rodrigues et al. (2016) onde o melhor desempenho foi obtido pelo GloVe, tanto no PT-BR como no PT-EU, como mostrado na Tabela 3.3. Os piores resultados foram obtidos pelo Word2vec, tanto CBOW como *Skip-gram*.

A avaliação extrínseca foi feita nas tarefas de etiquetação morfossintática e de similaridade semântica. Na tarefa de etiquetação morfossintática o objetivo é predizer, através de uma rede neural *perceptron* multicamada, etiquetas morfossintáticas em novas sentenças. A rede recebe

**Tabela 3.2: Comparação do desempenho dos modelos propostos por Mikolov et al. (2013) com métodos *baseline*.**

| Model | Vector Dimensionality | Training words | Accuracy [%] | | |
|---|---|---|---|---|---|
| | | | Semantic | Syntactic | Total |
| Collobert-Weston NNLM | 50 | 660M | 9.3 | 12.3 | 11.0 |
| Turian NNLM | 50 | 37M | 1.4 | 2.6 | 2.1 |
| Turian NNLM | 200 | 37M | 1.4 | 2.2 | 1.8 |
| Mnih NNLM | 50 | 37M | 1.8 | 9.1 | 5.8 |
| Mnih NNLM | 100 | 37M | 3.3 | 13.2 | 8.8 |
| Mikolov RNNLM | 80 | 320M | 4.9 | 18.4 | 12.7 |
| Mikolov RNNLM | 640 | 320M | 8.6 | 36.5 | 24.6 |
| Huang NNLM | 50 | 990M | 13.3 | 11.6 | 12.3 |
| Our NNLM | 20 | 6B | 12.9 | 26.4 | 20.3 |
| Our NNLM | 50 | 6B | 27.9 | 55.8 | 43.2 |
| Our NNLM | 100 | 6B | 34.2 | **64.5** | 50.8 |
| CBOW | 300 | 783M | 15.5 | 53.1 | 36.1 |
| Skip-gram | 300 | 783M | **50.0** | 55.9 | **53.3** |

**Fonte: Retirado de (MIKOLOV et al., 2013)**

representações de palavras em forma de vetores numéricos (vetores de palavras) e aprende a predizer um novo exemplo a partir do treinamento em um córpus já rotulado com etiquetas morfossintáticas. Nessa tarefa, Wang2vec obteve a melhor acurácia com o algoritmo Skip-gram com 1.000 dimensões. O resultado pode ser explicado com o fato de que Wang2vec leva em consideração a ordem das palavras, capturando mais informação sintática.

Na tarefa de similaridade semântica o objetivo é predizer, através de uma regressão linear, o *score* de similaridade entre duas sentenças. O modelo recebe as sentenças e converte as palavras para seus respectivos vetores de palavras. O treinamento é feito em um *dataset* já anotado com *score* de similaridade e a predição ocorre no *dataset* de testes, sem anotação de *score*. Nessa tarefa, os melhores resultados para o PT-EU foram alcançados pelo Word2Vec (CBOW) com 1.000 dimensões. O melhor resultado para o PT-BR foi obtido pelo Skip-Gram do Wang2Vec usando 1.000 dimensões (veja Tabela 3.4).

## 3.3 GloVe

Pennington et al. (2014) propuseram o GloVe, um modelo de regressão log-bilinear global que combina as vantagens de duas principais abordagens de modelos da literatura: métodos de

**Tabela 3.3: Avaliação intrínseca dos vetores de palavras gerados em Hartmann et al. (2017) para o português.**

| Embedding Models | | Size | PT-BR | | | PT-EU | | |
|---|---|---|---|---|---|---|---|---|
| | | | Syntactic | Semantic | All | Syntactic | Semantic | All |
| FastText | CBOW | 50 | 35.2 | 4.2 | 19.6 | 35.2 | 4.6 | 19.8 |
| | | 100 | 45.0 | 6.1 | 25.5 | 45.1 | 6.4 | 25.7 |
| | | 300 | 52.0 | 8.4 | 30.1 | 52.0 | 9.1 | 30.5 |
| | | 600 | 52.6 | 5.9 | 29.2 | 52.4 | 6.5 | 29.4 |
| | | 1,000 | 50.6 | 4.8 | 27.7 | 50.4 | 5.4 | 27.9 |
| | Skip-Gram | 50 | 36.8 | 18.4 | 27.6 | 36.5 | 17.1 | 26.8 |
| | | 100 | 50.8 | 30.0 | 40.4 | 50.7 | 28.9 | 39.8 |
| | | 300 | **58.7** | 32.2 | 45.4 | **58.5** | 31.1 | 44.8 |
| | | 600 | 55.1 | 24.3 | 39.6 | 55.0 | 23.9 | 39.4 |
| | | 1,000 | 45.1 | 14.6 | 29.8 | 45.2 | 13.8 | 29.4 |
| GloVe | | 50 | 28.7 | 13.7 | 27.4 | 28.5 | 12.8 | 27.7 |
| | | 100 | 39.7 | 28.7 | 34.2 | 39.9 | 26.6 | 33.2 |
| | | 300 | 45.8 | 45.8 | **46.7** | 45.9 | 42.3 | **46.2** |
| | | 600 | 42.3 | **48.5** | 45.4 | 42.3 | **43.8** | 43.1 |
| | | 1,000 | 39.4 | 45.9 | 42.7 | 39.8 | 42.5 | 41.1 |
| Wang2Vec | CBOW | 50 | 28.4 | 9.2 | 18.8 | 28.4 | 8.9 | 18.6 |
| | | 100 | 40.9 | 26.2 | 33.5 | 40.8 | 24.4 | 32.6 |
| | | 300 | 49.9 | 40.3 | 45.1 | 50.0 | 36.9 | 43.5 |
| | Skip-Gram | 50 | 30.6 | 12.2 | 21.3 | 30.6 | 11.5 | 21.0 |
| | | 100 | 43.9 | 22.2 | 33.0 | 44.0 | 21.2 | 32.6 |
| | | 300 | 53.3 | 33.9 | 42.8 | 53.4 | 32.3 | 43.6 |
| | | 600 | 52.9 | 35.0 | 43.9 | 53.0 | 33.2 | 43.1 |
| | | 1,000 | 47.3 | 33.2 | 40.2 | 47.6 | 30.9 | 39.2 |
| Word2Vec | CBOW | 50 | 9.8 | 2.2 | 6.0 | 9.7 | 1.9 | 5.8 |
| | | 100 | 16.2 | 3.6 | 9.9 | 16.0 | 3.5 | 9.7 |
| | | 300 | 24.7 | 4.6 | 23.9 | 24.5 | 4.5 | 23.6 |
| | | 600 | 25.8 | 5.2 | 23.1 | 25.4 | 5.1 | 22.9 |
| | | 1,000 | 26.2 | 4.9 | 22.9 | 26.2 | 4.5 | 22.7 |
| | Skip-Gram | 50 | 17.0 | 5.4 | 11.2 | 16.9 | 4.8 | 10.8 |
| | | 100 | 25.2 | 8.0 | 16.6 | 24.8 | 7.4 | 16.1 |
| | | 300 | 33.0 | 15.6 | 29.2 | 32.2 | 14.1 | 29.8 |
| | | 600 | 35.6 | 20.0 | 33.4 | 35.3 | 17.6 | 33.5 |
| | | 1,000 | 34.1 | 21.3 | 32.6 | 33.6 | 18.1 | 31.9 |

**Fonte: Retirado de (HARTMANN et al., 2017)**

fatoração de matriz global, como o LSA (DEERWESTER et al., 1990), e métodos de janela de contexto local, como o modelo Skip-gram de Mikolov et al. (2013). O modelo aproveita eficientemente a informação estatística utilizando apenas os elementos diferentes de zero de uma matriz de co-ocorrência de palavras, em vez de usar toda a matriz esparsa ou janelas de contexto individuais de um grande córpus.

As estatísticas de ocorrências de palavras em um córpus são a principal fonte de informação disponível para os métodos não supervisionados aprenderem representações de palavras. Essas estatísticas são basicamente calculadas considerando-se uma matriz de contagem de co-ocorrência (ou simplesmente matriz de co-ocorrência). Seja $X$ a matriz de co-ocorrência de palavra-palavra com cada entrada $X_{ij}$ representando o número de vezes que a palavra $j$ ocorre no contexto da palavra $i$ e $X_i = \sum_k X_{ik}$ o número de vezes que qualquer palavra aparece no contexto da palavra $i$. Tem-se que $P_{ij} = P(j|i) = \frac{X_{ij}}{X_i}$ é a probabilidade da palavra $j$ aparecer no contexto da palavra $i$.

**Tabela 3.4: Avaliação extrínseca dos vetores gerados em Hartmann et al. (2017) na tarefa de similaridade semântica.**

| Embedding Models | | Size | PT-BR | | PT-EU | |
|---|---|---|---|---|---|---|
| | | | $\rho$ | MSE | $\rho$ | MSE |
| FastText | CBOW | 50 | 0.36 | 0.66 | 0.34 | 1.05 |
| | | 100 | 0.37 | 0.66 | 0.36 | 1.04 |
| | | 300 | 0.38 | 0.65 | 0.37 | 1.03 |
| | | 600 | 0.33 | 0.68 | 0.38 | 1.02 |
| | | 1,000 | 0.39 | 0.64 | 0.41 | 0.99 |
| | Skip-Gram | 50 | 0.45 | 0.61 | 0.43 | 0.98 |
| | | 100 | 0.49 | 0.58 | 0.47 | 0.94 |
| | | 300 | 0.55 | 0.53 | 0.40 | 1.02 |
| | | 600 | 0.40 | 0.64 | 0.40 | 1.01 |
| | | 1,000 | 0.52 | 0.56 | 0.54 | 0.86 |
| Wang2Vec | CBOW | 50 | 0.53 | 0.55 | 0.51 | 0.89 |
| | | 100 | 0.56 | 0.52 | 0.54 | 0.85 |
| | | 300 | 0.53 | 0.55 | 0.51 | 0.89 |
| | Skip-Gram | 50 | 0.51 | 0.56 | 0.47 | 0.92 |
| | | 100 | 0.54 | 0.54 | 0.50 | 0.89 |
| | | 300 | 0.58 | 0.50 | 0.53 | 0.85 |
| | | 600 | 0.59 | **0.49** | 0.54 | **0.83** |
| | | 1,000 | **0.60** | **0.49** | 0.54 | 0.85 |
| GloVe | | 50 | 0.42 | 0.62 | 0.38 | 1.01 |
| | | 100 | 0.45 | 0.60 | 0.42 | 0.98 |
| | | 300 | 0.49 | 0.58 | 0.45 | 0.95 |
| | | 600 | 0.50 | 0.57 | 0.45 | 0.94 |
| | | 1,000 | 0.51 | 0.56 | 0.46 | 0.94 |
| Word2Vec | CBOW | 50 | 0.47 | 0.59 | 0.46 | 0.95 |
| | | 100 | 0.50 | 0.57 | 0.49 | 0.91 |
| | | 300 | 0.55 | 0.53 | 0.54 | 0.87 |
| | | 600 | 0.57 | 0.51 | **0.55** | 0.86 |
| | | 1,000 | 0.58 | 0.50 | **0.55** | 0.86 |
| | Skip-Gram | 50 | 0.46 | 0.60 | 0.43 | 0.97 |
| | | 100 | 0.48 | 0.58 | 0.45 | 0.95 |
| | | 300 | 0.52 | 0.56 | 0.48 | 0.93 |
| | | 600 | 0.53 | 0.54 | 0.50 | 0.92 |
| | | 1,000 | 0.54 | 0.54 | 0.50 | 0.91 |

**Fonte: Retirado de (HARTMANN et al., 2017)**

Para ilustrar esse método, considere um exemplo simples que mostra como certos aspectos do significado podem ser extraídos diretamente das probabilidades de co-ocorrência: considere duas palavras $i$ e $j$ que exibem um aspecto particular de interesse, sendo $i = ice$ (gelo) e $j = steam$ (vapor). A relação dessas palavras pode ser examinada estudando a proporção de suas probabilidades de co-ocorrência com várias palavras $k$ como ilustrado na Tabela 3.5.

**Tabela 3.5: Probabilidades de co-ocorrência para palavras-alvo *ice* e *steam* com palavras de contexto.**

| Probability and Ratio | $k = solid$ | $k = gas$ | $k = water$ | $k = fashion$ |
|---|---|---|---|---|
| $P(k\|ice)$ | $1.9 \times 10^{-4}$ | $6.6 \times 10^{-5}$ | $3.0 \times 10^{-3}$ | $1.7 \times 10^{-5}$ |
| $P(k\|steam)$ | $2.2 \times 10^{-5}$ | $7.8 \times 10^{-4}$ | $2.2 \times 10^{-3}$ | $1.8 \times 10^{-5}$ |
| $P(k\|ice)/P(k\|steam)$ | 8.9 | $8.5 \times 10^{-2}$ | 1.36 | 0.96 |

**Fonte: Retirado de (PENNINGTON et al., 2014)**

Para as palavras $k$ relacionadas a *ice*, mas não a *steam*, como k = *solid*, espera-se um valor grande para $P_{ik}/P_{jk}$. De modo análogo, para as palavras $k$ relacionadas a *steam*, mas não a *ice*, como k = *gas*, o valor deve ser pequeno. Para palavras $k$ como *water* ou *fashion*, que estão relacionadas tanto a *ice* quanto a *steam*, ou a nenhuma delas, a proporção deve ser próxima de um. Em comparação com as probabilidades, a relação é capaz de distinguir palavras relevantes (*solid* e *gas*) de palavras irrelevantes (*water* e *fashion*) e também é capaz de discriminar entre as duas palavras relevantes.

O argumento acima sugere que o ponto de partida apropriado para a aprendizagem de vetores de palavras deve ser com os índices de probabilidades de co-ocorrência em vez das próprias probabilidades. Observando que a relação $Pik/P_{jk}$ depende de três palavras $i$, $j$ e $k$, o modelo mais geral assume a forma da equação 3.3:

$$F(w_i, w_j, w_k) = \frac{P_{ik}}{P_{jk}} \tag{3.3}$$

Em uma avaliação apresentada em (PENNINGTON et al., 2014), o GloVe foi treinado em cinco córpus de tamanhos variados: Wikipedia 2010 com 1 bilhão de *tokens*; Wikipedia 2014 com 1,6 bilhões de *tokens*; Gigaword 5 com 4,3 bilhões de *tokens*; a combinação Gigaword 5 + Wikipedia 2014, com 6 bilhões de *tokens*; e em 42 bilhões de *tokens* de dados da Web. Após o treinamento, o GloVe foi validado em uma tarefa de analogia de palavras e obteve uma acurácia de 75% (em negrito), superando os demais métodos usados na comparação (veja Tabela 3.6[1]).

**Tabela 3.6: Resultados de acurácia na tarefa de analogia de palavras**

| Model | Dim. | Size | Sem. | Syn. | Tot. |
|---|---|---|---|---|---|
| ivLBL | 100 | 1.5B | 55.9 | 50.1 | 53.2 |
| HPCA | 100 | 1.6B | 4.2 | 16.4 | 10.8 |
| GloVe | 100 | 1.6B | 67.5 | 54.3 | 60.3 |
| SG | 300 | 1B | 61 | 61 | 61 |
| CBOW | 300 | 1.6B | 16.1 | 52.6 | 36.1 |
| vLBL | 300 | 1.5B | 54.2 | 64.8 | 60.0 |
| ivLBL | 300 | 1.5B | 65.2 | 63.0 | 64.0 |
| GloVe | 300 | 1.6B | 80.8 | 61.5 | 70.3 |
| SVD | 300 | 6B | 6.3 | 8.1 | 7.3 |
| SVD-S | 300 | 6B | 36.7 | 46.6 | 42.1 |
| SVD-L | 300 | 6B | 56.6 | 63.0 | 60.1 |
| CBOW† | 300 | 6B | 63.6 | 67.4 | 65.7 |
| SG† | 300 | 6B | 73.0 | 66.0 | 69.1 |
| GloVe | 300 | 6B | 77.4 | 67.0 | 71.7 |
| CBOW | 1000 | 6B | 57.3 | 68.9 | 63.7 |
| SG | 1000 | 6B | 66.1 | 65.1 | 65.6 |
| SVD-L | 300 | 42B | 38.4 | 58.2 | 49.2 |
| GloVe | 300 | 42B | **81.9** | **69.3** | **75.0** |

**Fonte: Retirado de (PENNINGTON et al., 2014)**

Para o português, no trabalho de Hartmann et al. (2017), os autores concluíram que o GloVe

[1]Os valores sublinhados são os melhores desempenhos de modelos de tamanho similar. Os valores em negrito são os melhores desempenhos. Os vetores HPCA estão disponíveis publicamente, os resultados de vLBL são de (MNIH; KAVUKCUOGLU, 2013), os resultados de Skip-Gram (SG) e CBOW são de (MIKOLOV et al., 2013, 2013), SG† e CBOW† foram treinados usando a ferramenta word2vec.

obteve o melhor desempenho em tarefas de analogias sintáticas e semânticas (Tabela 3.3) e os piores resultados, junto com FastText, nas tarefas de etiquetação morfossintática e similaridade semântica.

## 3.4 Wang2Vec

Segundo Ling et al. (2015), uma das ferramentas mais utilizadas para a geração de vetores de palavras é o Word2Vec (MIKOLOV et al., 2013). Embora tenham sido propostas abordagens mais sofisticadas (DHILLON et al., 2011; HUANG et al., 2012; FARUQUI; DYER, 2014; GOLDBERG; LEVY, 2014; YANG; EISENSTEIN, 2015), o Word2Vec continua a ser uma escolha popular devido a sua eficiência e simplicidade.

No entanto, como o Skip-gram e o CBOW implementados no Word2Vec não consideram a ordem das palavras, os vetores criados não obtêm um bom desempenho em tarefas que envolvem sintaxe, como etiquetação morfossintática ou análise de dependência. Em um modelo onde a ordem das palavras é descartada, as relações sintáticas entre as palavras não podem ser capturadas adequadamente. Por exemplo, enquanto a maioria das palavras ocorre com o artigo *o*, apenas os substantivos tendem a ocorrer exatamente após o artigo (por exemplo, *o gato*). Isto é embasado por evidências empíricas que sugerem que não considerar a ordem conduz a representações sintáticas precárias (ANDREAS; KLEIN, 2014; BANSAL et al., 2014).

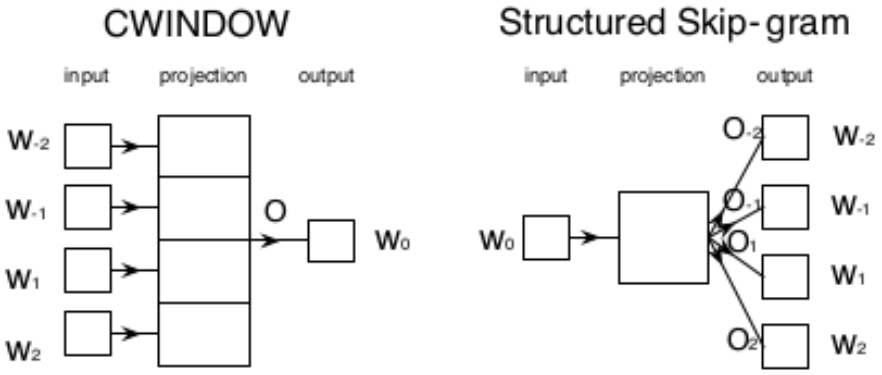
Com o objetivo de possibilitar que o Word2Vec tenha um bom desempenho em tarefas baseadas em sintaxe, Ling et al. (2015) propuseram duas modificações simples no Word2Vec, uma para o modelo Skip-gram e outra para o modelo CBOW. O objetivo dos autores é melhorar o vetor final gerado, mantendo a simplicidade e a eficiência dos modelos originais. Para tanto, Ling et al. (2015) propuseram o modelo Skip-gram estruturado e o CWINDOW (*Continuous Window Model*).

O Skip-gram de Mikolov et al. (2013) usa uma única matriz de saída para prever todas as palavras de contexto dado o vetor da palavra alvo. O Skip-gram estruturado é uma alteração do original que faz com que ele seja sensível ao posicionamento das palavras. Ele define um conjunto de matrizes de saída onde cada uma tem como objetivo prever a saída de uma posição específica de uma palavra de contexto para a palavra alvo.

O CBOW, também de Mikolov et al. (2013), define uma janela de palavras onde a predição da matriz de saída é alimentada com a soma dos vetores das palavras de contexto. No CWINDOW, são definidas diferentes matrizes de saída que recebem vetores de palavras de contexto concatenados na ordem em que as palavras ocorrem. Este modelo CWINDOW se baseia na

proposta de janela de Collobert et al. (2011). Ambos os modelos são mostrados na Figura 3.3.

**Figura 3.3: Modelos CWINDOW e Skip-gram estruturado propostos por Ling et al. (2015).**




**Fonte: Retirado de (LING et al., 2015)**

Em uma avaliação apresentada em (LING et al., 2015), o Wang2vec foi treinado em três córpus distintos: um *dump* da Wikipédia em inglês contendo 1.897 milhões de palavras (WIKI(S)), um do Twitter contendo 56 milhões de *tweets* em inglês com 847 milhões de palavras (TWITTER) e outro da Wikipédia com 16 milhões de palavras, fornecido no pacote Word2Vec (WIKI(L)).

Uma das tarefas usadas para validação do Wang2vec foi a etiquetação morfossintática feita no conjunto de dados *Penn Treebank* (PTB) e em um conjunto de dados de 500 *tweets*, em inglês, rotulados. Como mostrado na Tabela 3.7, o Skip-gram estruturado obteve 97,05% de acurácia no conjunto de dados PTB e 89,79% no conjunto de *tweets*, superando o CBOW e o Skip-gram originais.[2]

**Tabela 3.7: Avaliação do modelo proposto por Ling et al. (2015) na tarefa de etiquetação morfossintática.**

| | **PTB** | | **Twitter** | |
|---|---|---|---|---|
| | Dev | Test | Dev | Test |
| CBOW | 95.89 | 96.13 | 87.85 | 87.54 |
| Skip-gram | 96.62 | 96.68 | 88.84 | 88.73 |
| CWindow | **96.99** | 97.01 | **89.72** | 89.63 |
| Structured Skip-gram | 96.62 | **97.05** | 89.69 | **89.79** |
| SENNA | 96.54 | 96.58 | 84.96 | 84.85 |

**Fonte: Retirado de (LING et al., 2015)**

[2]CBOW e *Skip-gram* são os originais de Mikolov et al. (2013). CWINDOW e *Structured Skip-gram* são as propostas de Ling et al. (2015). SENNA é o modelo de Collobert et al. (2011).

Para o português, segundo Hartmann et al. (2017), o Wang2vec obteve a melhor acurácia com o algoritmo Skip-gram com 1.000 dimensões na avaliação extrínseca na tarefa de etiquetação morfossintática. O resultado pode ser explicado com o fato de que Wang2vec leva em consideração a ordem das palavras, capturando mais informação sintática. Por exemplo, Wang2vec tende a agrupar palavras relacionadas sintaticamente, como *putting, turning, sticking, pulling e picking agrupadas a partir da entrada breaking.*

## 3.5 FastText

Para propor o FastText, Bojanowski et al. (2016) se baseou no fato de que representações contínuas de palavras, treinadas em um grande córpus não rotulado, são úteis para várias tarefas de PLN. Entretanto, muitos modelos treinados para gerar tais representações ignoram a morfologia das palavras, gerando um vetor totalmente distinto para cada palavra, mesmo quando elas possuem as mesmas características morfológicas (ex: "quebrar" e "quebrado"). Isso é uma limitação principalmente para línguas com um extenso vocabulário e muitas palavras raras, como é o caso do português, espanhol, francês, turco, finlandês, etc.

A partir dessa premissa, o FastText foi proposto com base no Skip-gram de Mikolov et al. (2013). Por usar uma representação vetorial distinta para cada palavra, o Skip-gram ignora a estrutura interna das palavras. Para reter essa informação, o FastText representa cada palavra *w* como uma sacola de n-gramas de caracteres. São adicionados símbolos de limite $<$ e $>$ no início e no final das palavras, permitindo distinguir prefixos e sufixos de outras sequências de caracteres. A própria palavra *w* também é incluída no conjunto de seus n-gramas, a fim de aprender uma representação para cada palavra (além dos n-gramas de caracteres). Por exemplo, para a palavra *where* e n = 3, a representação em trigramas fica da seguinte forma:

$< wh, whe, her, ere, re >$

Veja que a sequência $< her >$, correspondente à palavra *her*, é diferente do trigrama *her* presente na palavra *where*. Na prática, extrai-se todos os n-gramas para *n* maior ou igual à 3 e menor ou igual à 6.

A representação vetorial é associada a cada n-grama de caracteres e as palavras são representadas pelas somas dessas representações. A formação de palavras costuma seguir regras, o que torna possível gerar representações vetoriais de palavras para línguas ricas morfologicamente usando informação a nível de caractere.

Este método simples permite o compartilhamento das representações vetoriais das palavras,

permitindo aprender representações confiáveis para palavras raras.

Em uma avaliação apresentada em (BOJANOWSKI et al., 2016), o FastText foi treinado em *dumps* da Wikipédia em nove idiomas: árabe, tcheco, alemão, inglês, espanhol, francês, italiano, romeno e russo. O algoritmo foi avaliado em uma tarefa de analogia de palavras, onde a palavra A está para a palavra B assim como a palavra C está para D, onde D deve ser predita pelo modelo. Foram utilizados conjuntos de dados de analogia de palavras para o inglês, tcheco, alemão e italiano. Os resultados aparecem na Tabela 3.8[3].

**Tabela 3.8: Acurácia do FastText e outros sistemas na tarefa de analogia de palavras (sintaxe e semântica) para tcheco (CS), alemão (DE), inglês (EN) e italiano (IT).**

| | | sg | cbow | sisg |
|---|---|---|---|---|
| CS | Semantic | 25.7 | 27.6 | 27.5 |
| | Syntactic | 52.8 | 55.0 | 77.8 |
| DE | Semantic | 66.5 | 66.8 | 62.3 |
| | Syntactic | 44.5 | 45.0 | 56.4 |
| EN | Semantic | 78.5 | 78.2 | 77.8 |
| | Syntactic | 70.1 | 69.9 | 74.9 |
| IT | Semantic | 52.3 | 54.7 | 52.3 |
| | Syntactic | 51.5 | 51.8 | 62.7 |

**Fonte: Retirado de (BOJANOWSKI et al., 2016)**

Observa-se que a informação morfológica levou aos melhores resultados para as tarefas sintáticas, onde FastText obtém 77,8% de acurácia para o tcheco, contra 52,8% do Skip-gram original e 55% do CBOW original, por exemplo. O desempenho superior continua nos demais idiomas avaliados. Em contrapartida, o FastText não obteve um bom desempenho nas analogias semânticas, tendo acurácias inferiores aos dos outros sistemas em comparação.

Para o português, de acordo com Hartmann et al. (2017), o FastText não teve um bom desempenho nas tarefas de analogias sintáticas e semânticas, como detalhado na Tabela 3.3. Na etiquetação morfossintática, o FastText também obteve o pior desempenho, fato inesperado já que o modelo foi proposto para modelar bem informações morfossintáticas, característica da tarefa. Na tarefa de similaridade semântica também notou-se um desempenho abaixo dos demais.

[3]Nesta tabela, sg indica o Skip-gram de Mikolov et al. (2013); cbow, o CBOW de Mikolov et al. (2013) e sisg, o Skip-gram do FastText.

## 3.6 Considerações finais

A seguir, na Tabela 3.9, apresenta-se um resumo das principais características dos trabalhos relacionados descritos neste capítulo.

**Tabela 3.9: Resumo das principais características dos trabalhos de geração de vetores de palavras de sentido único apresentados neste capítulo.**

| **Trabalho** | **Ferramenta** | **Idioma** | **Recurso** | **Avaliação** |
|---|---|---|---|---|
| Mikolov et al. (2013) | Word2Vec | EN | *Google News*<br>*Semantic-Syntatic*<br>*Word Relationship* | Acurácia = 53,3% |
| Pennington et al. (2014) | GloVe | EN | Wikipédia<br>*Gigaword*<br>Dados da Web | Acurácia = 75% |
| Ling et al. (2015) | Wang2Vec | EN | Wikipédia<br>Twitter<br>*Penn Treebank* | Acurácia = 97% |
| Bojanowski et al. (2016) | FastText | EN | Wikipédia<br>*Analogy Task* | Acurácia = 74,9% |
| Hartmann et al. (2017) | Word2Vec | PT-BR | LX-Corpus<br>Wikipédia<br>Analogias sintática<br>e semânticas<br>Dentre outros | *Spearman* = 32,6% |
| Hartmann et al. (2017) | GloVe | PT-BR | idem | *Spearman* = 42,7% |
| Hartmann et al. (2017) | Wang2Vec | PT-BR | idem | *Spearman* = 40,2% |
| Hartmann et al. (2017) | FastText | PT-BR | idem | *Spearman* = 29,8% |

**Fonte: Autoria própria**

# Capítulo 4

## VETORES DE SENTIDO (*Sense Embeddings*)

Apesar de muito úteis em diversas aplicações, os vetores de palavras citados no capítulo anterior têm limitações. Por serem gerados sem diferenciar contextos, geram uma única representação vetorial para cada palavra, não levando em consideração que palavras ambíguas podem assumir significados que as colocariam em espaços vetoriais distintos. Conclui-se que há perda de informação ao representar uma ambiguidade lexical em um vetor único, já que este trará apenas o sentido mais comumente utilizado para a palavra (ou aquele que ocorre no córpus a partir do qual os vetores de palavras foram gerados).

Pesquisas iniciais apontaram que os vetores de palavras poderiam modelar aspectos do sentido da palavra (SCHÜTZE, 1998; KINTSCH, 2001) e pesquisas recentes propuseram uma série de modelos que representam cada sentido da palavra separadamente, cada um associado a um vetor específico (REISINGER; MOONEY, 2010; NEELAKANTAN et al., 2015; HUANG et al., 2012; PINA; JOHANSSON, 2014; WU; GILES, 2015; LIU et al., 2015a). Modelos vetoriais de sentidos oferecem grande potencial para melhorar muitas tarefas de compreensão da linguagem. Nessa linha, neste trabalho investigou-se diferentes métodos para a geração desses vetores com informação de sentido da palavra, conhecidos como vetores de sentido (do inglês, *sense embeddings* ou *sense vectors*).

Para ilustrar esse conceito, considere como exemplo a palavra "letra" ocorrendo nas seguintes sentenças:

- *A* ***letra***$^{sentido1}$ *da canção é excêntrica.*
- *A música tem uma ótima* ***letra***$^{sentido1}$.
- *Sua* ***letra***$^{sentido2}$ *é muito bonita.*
- *A minha* ***letra***$^{sentido2}$ *cursiva é uma obra de arte.*

Nesse exemplo, o sentido1 especifica letra de música (palavras ditas em uma música) enquanto o sentido2 refere-se à caligrafia (características pessoais de como escrever). Assim, os métodos que geram vetores de palavra modelam todos os sentidos (presentes no córpus) de uma palavra em um único vetor, fazendo com que haja a perda dessa informação e o sentido mais frequente se sobressaia. No exemplo acima, o sentido1 seria misturado ao sentido2 e ambos poderiam ser perdidos no vetor final da palavra "letra". Os vetores de sentido, por sua vez, seriam capazes de gerar vetores diferentes: um para o sentido1 e outro para o sentido2.

Vale ressaltar que mesmo palavras ambíguas podem ter apenas um vetor de sentido aprendido para elas se as ocorrências da palavra no córpus usado para aprendizado dos vetores não forem representativas de todos os sentidos possíveis. Assim, a limitação na geração de vetores de sentido não está relacionada necessariamente à estratégia de geração, mas também está vinculada ao córpus usado no aprendizado. Contudo, enquanto os métodos de geração de vetores de sentido têm a possibilidade de modelar vários sentidos de uma palavra, os métodos tradicionais (apresentados no capítulo 3) não têm essa capacidade considerando que não capturam evidências de sentido, como a ocorrência da mesma palavra em contexto distintos.

De acordo com Camacho-Collados e Pilehvar (2018), os métodos que geram vetores de sentido se enquadram em duas categorias: baseados em conhecimento e não supervisionados.

Os métodos baseados em conhecimento (IACOBACCI et al., 2015; ROTHE; SCHÜTZE, 2015; CHEN et al., 2014) definem os sentidos possíveis com base em um repositório de sentidos existente, como a WordNet, Wikipédia, BabelNet ou Freebase. Uma das vantagens desses métodos é o fato de que representações de sentido aprendidas em um córpus não etiquetado estarão ligadas a um repositório de sentidos criado por especialistas, possibilitando que os sentidos descobertos sejam condizentes com os sentidos conhecidos. As desvantagens são a indisponibilidade e a baixa cobertura, uma vez que esse repositório de sentidos pode não estar disponível ou não existir para um dado idioma; e sentidos que não estão no repositório não serão modelados na geração dos vetores de sentido.

Já os métodos não supervisionados podem ser divididos em técnicas que exploram córpus monolíngue (4.1) e as que exploram córpus multilíngue ou etiquetado (4.2).

A seguir são descritos os principais métodos de geração de vetores de sentido que fazem a indução de sentido de modo não supervisionado, visto que esta é a abordagem escolhida para ser investigada neste trabalho. Escolhemos esta abordagem por não depender de um repositório de sentidos, recurso limitado e as vezes inexistente para alguns idiomas.

## 4.1 Representações que exploram córpus monolíngue

Ainda segundo Camacho-Collados e Pilehvar (2018), a abordagem que explora somente córpus monolíngue pode ser dividida em três categorias principais: modelos baseados em clusterização (*clustering-based or two-stage*), modelos baseados em treinamento unificado (*joint training*) e modelos contextualizados (*contextualized embeddings*).

### 4.1.1 Modelos baseados em clusterização de contextos (dois estágios)

Schütze (1998) foi um dos primeiros trabalhos a identificar a deficiência dos vetores de palavras e a propor a indução de significados através do agrupamento de contextos nos quais uma palavra ambígua ocorre. Muitos outros trabalhos seguiram essa ideia. Algoritmos de clusterização são utilizados para agrupar vetores de contexto, onde cada *cluster* representará um sentido encontrado no córpus de treinamento. A vantagem desta técnica é que é totalmente sem supervisão, não sendo necessário qualquer repositório de sentidos ou base de conhecimento. A desvantagem é o fato das representações não estarem ligadas a sentidos conhecidos, tornando sua interpretação mais complexa e abstrata. A seguir são descritos os principais trabalhos que geram vetores de sentido através da clusterização de contextos em dois estágios.

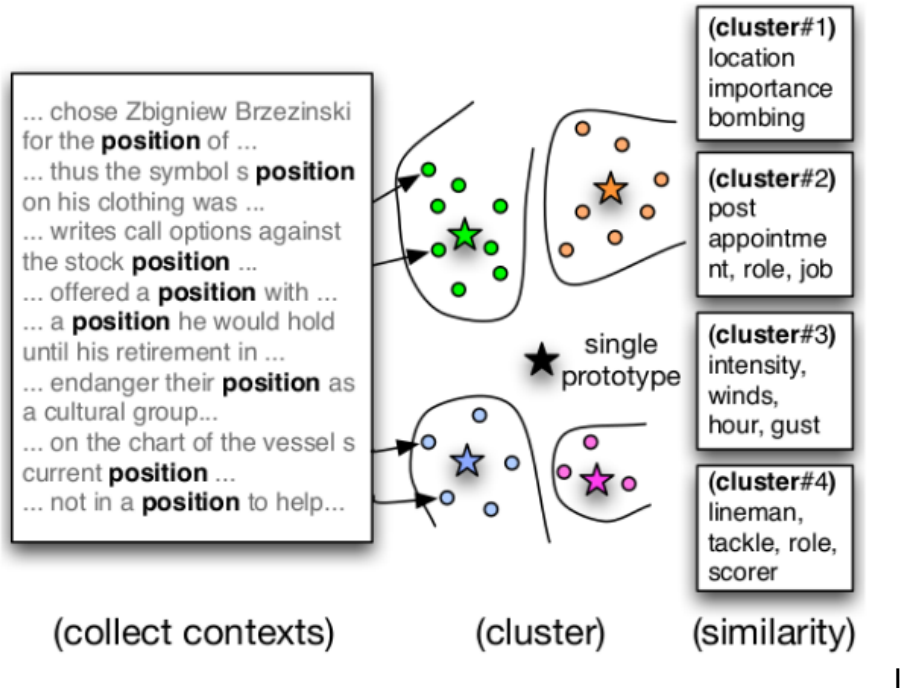
#### 4.1.1.1 Reisinger e Mooney (2010)

Um dos primeiros trabalhos utilizando redes neurais foi o de Reisinger e Mooney (2010), no qual a identificação dos sentidos das palavras foi realizada como uma tarefa de pré-processamento com a posterior geração de vetores de sentido (dois estágios). A Figura 4.1 ilustra o processo implementado em (REISINGER; MOONEY, 2010).

Primeiro, os contextos de cada palavra são definidos considerando-se uma janela de 10 palavras anteriores e 10 posteriores à palavra alvo. Em seguida, cada contexto possível de uma palavra é representado pela média ponderada dos vetores das palavras que o compõem. Esses vetores de contexto (os círculos na Figura 4.1) são, então, agrupados e cada centroide (as estrelas na Figura 4.1) é selecionado para representar o sentido do *cluster*. Por exemplo, na Figura 4.1, as palavras *location*, *importance* e *bombing* são palavras de contexto que ocorrem no centroide do *cluster* 1.

Considerando-se o exemplo da palavra ambígua “letra” apresentado anteriormente, em (REISINGER; MOONEY, 2010) dois *clusters* seriam gerados (um para o sentido1 e outro para o sentido2) como a média ponderada dos vetores de palavas presentes em cada contexto.

**Figura 4.1: Representação do processo de clusterização para modelagem de sentidos.**




**Fonte: Retirado de (REISINGER; MOONEY, 2010)**

Formados os *clusters*, cada palavra do córpus é etiquetada com o *cluster* de sentido mais próximo de seu contexto, como mostrado acima. Como "letra" é ambígua e ocorre em contextos distintos, cada ocorrência receberá um *cluster* diferente, possibilitando a geração de um vetor por sentido. Após esse pré-processamento, uma rede neural é treinada a partir do córpus etiquetado, gerando os vetores de sentido.

Para treinar os vetores de sentido, Reisinger e Mooney (2010) usaram dois córpus: (1) um *dump* da Wikipédia, em inglês, contendo 2.8M artigos com um total de 2.05B de palavras e (2) a terceira edição, em inglês, do córpus *Gigaword*, com 6.6M de artigos e 3.9B palavras. A Wikipédia é mais representativa quanto a variedade de sentidos, enquanto a *Gigaword* contém apenas textos de notícias e tende a empregar menos sentidos de palavras ambíguas.

Os autores apresentaram comparações de similaridade semântica através de avaliações feitas por humanos, tanto para palavras isoladas quanto para palavras em contexto sentencial. O conjunto de dados utilizado foi o *WordSim-353* (FINKELSTEIN et al., 2001), que contém 353 pares de palavras cuja a similaridade foi avaliada por humanos, em uma escala de 1 a 10 (por exemplo, nessa escala "copo" e "bebida" receberam uma nota de similaridade igual a 7,25). Os pares incluem palavras ambíguas e não ambíguas. Essas pontuações são dadas sem qualquer informação contextual.

A similaridade entre dois vetores de sentido pode ser calculada através de medidas de similaridade semântica, como AvgSim e MaxSim, como ilustrado a seguir em 4.1 e 4.2, respectivamente, para duas palavras, $w$ e $w'$.

$$AvgSim(w,w') =^{def} \frac{1}{K^2} \sum_{j=1}^{K} \sum_{k=1}^{K} d(\pi_k(w), \pi_j(w')) \tag{4.1}$$

$$MaxSim(w,w') =^{def} max_{1<=j<=K,1<=k<=K} d(\pi_k(w), \pi_j(w')) \tag{4.2}$$

Onde d $(\cdot,\cdot)$ é uma medida padrão de similaridade distributiva (como o cosseno). AvgSim calcula a similaridade média de todos os vetores de sentido de uma palavra, ignorando informações de contexto. MaxSim calcula a similaridade de cada par de vetor de sentido com o contexto atual.

Os autores obtiveram uma correlação de *Spearman* (veja seção 3.1) de aproximadamente 62,5% no *WordSim-353*, para os córpus da Wikipédia e *Gigaword*. Reisinger e Mooney (2010) não validaram a técnica extrinsecamente (em tarefas de PLN), ponto que contou como uma desvantagem para o método proposto.

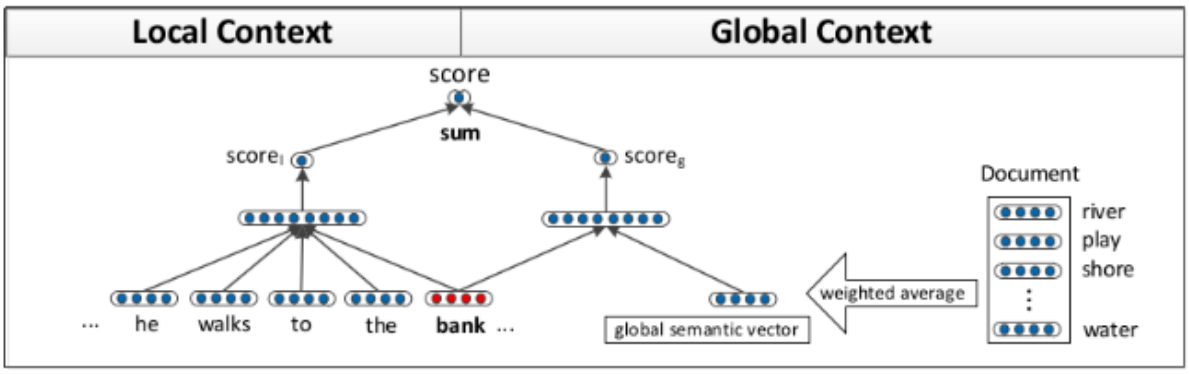
#### 4.1.1.2 Huang et al. (2012)

Huang et al. (2012) estendem a abordagem de Reisinger e Mooney (2010) incorporando um contexto global na geração dos vetores de sentido. Segundo eles, agregar informação de um contexto maior melhora a qualidade das representações vetoriais de palavras ambíguas, que possuem mais de um contexto local possível. Para fornecer a representação vetorial do contexto global, o modelo proposto utiliza todas as palavras do documento em que a palavra alvo ocorre, incorporando essa representação ao contexto local. Cada representação de contexto, local e global, é gerada por uma rede neural com camadas ocultas.

A representação do contexto local é modelada a partir de uma janela de 5 palavras anteriores e 5 posteriores à palavra alvo, respeitando a ordem das palavras. Ao final, obtém-se um peso para cada contexto, sendo que o local captura mais informações sintáticas e o global mais informações semânticas. Esses pesos são somados e atualizados a cada iteração da rede via *backpropagation*. A Figura 4.2 ilustra esse processo.

O modelo proposto aprende vetores de palavras ao mesmo tempo que aprende a discriminar a próxima palavra dada a sequência de palavras ao redor da palavra-alvo (contexto local) e o documento (contexto global) em que a sequência ocorre.

**Figura 4.2: Arquitetura das redes neurais que geram contextos local e global para a palavra "*bank*".**




**Fonte: Retirado de (HUANG et al., 2012)**

Para aprender múltiplos vetores, os autores reuniram janelas de contexto de tamanho fixo (5 palavras antes e 5 depois) de todas as ocorrências de uma palavra no córpus. O contexto de cada palavra é representado pela média ponderada dos vetores das palavras que nele ocorrem (como em Reisinger e Mooney (2010)), onde usa-se uma ponderação *tf-idf* para identificar relevância entre as palavras. Em seguida, para agrupar esses vetores de contexto, usa-se um *k-means* esférico, que mostrou-se capaz de modelar bem as relações semânticas (DHILLON; MODHA, 2001). Finalmente, como em Reisinger e Mooney (2010), cada palavra do córpus é associada a um *cluster* e são usadas para gerar vetores de sentido.

Os autores treinaram o modelo em um *dump* de abril de 2010 da Wikipédia em inglês, por ser representativa quanto ao uso das palavras ambíguas e os artigos serem organizados em tópicos. Foram 2 milhões de artigos e 990 milhões de *tokens*.

Os autores avaliaram o modelo no *WordSim-353* considerando a correlação de *Spearman* (veja seção 3.1). Os resultados foram comparados com *baselines* e são mostrados na Tabela 4.1[1].

As pontuações do *WordSim-353* são dadas sem qualquer informação contextual, fato que motivou os autores a criarem um novo *dataset*, com 2.003 pares de palavras em contextos sentenciais. Os autores usaram a Wikipédia para capturar pares de palavras com diferentes sentidos. O *dataset* inclui verbos, adjetivos e alguns substantivos. Na Tabela 4.2 está a correlação

[1]Os *baselines* foram: C&W* são os vetores já treinados de Collobert e Weston (2008); C&W são os modelos de Collobert e Weston (2008) treinados por Huang et al. (2012); *Our Model** é o modelo de Huang et al. (2012) sem *stopwords*, enquanto *Our Model-g* usa apenas o contexto global; *Pruned tf-idf* é Reisinger e Mooney (2010); ESA é (Gabrilovich e Markovitch, 2007); HLBL é o modelo hierárquico *log-bilinear* de (Mnih and Hinton, 2008). O córpus RCV1 contém um ano do portal de notícias *Reuters English newswire*.

**Tabela 4.1: Correlação de *Spearman* no WordSim-353 para o modelo proposto (Our Model) e *baselines*.**

| Model | Corpus | $\rho \times 100$ |
|---|---|---|
| Our Model-g | Wiki. | 22.8 |
| C&W | RCV1 | 29.5 |
| HLBL | RCV1 | 33.2 |
| C&W* | Wiki. | 49.8 |
| C&W | Wiki. | 55.3 |
| Our Model | Wiki. | 64.2 |
| Our Model* | Wiki. | 71.3 |
| Pruned *tf-idf* | Wiki. | 73.4 |
| ESA | Wiki. | 75 |
| Tiered Pruned *tf-idf* | Wiki. | 76.9 |

**Fonte: Retirado de (HUANG et al., 2012)**

de *Spearman* usando esse novo *dataset*. [2]

**Tabela 4.2: Correlação de *Spearman* no novo *dataset*.**

| Model | $\rho \times 100$ |
|---|---|
| C&W-S | 57.0 |
| Our Model-S | 58.6 |
| Our Model-M AvgSim | 62.8 |
| Our Model-M AvgSimC | **65.7** |
| *tf-idf*-S | 26.3 |
| Pruned *tf-idf*-S | 62.5 |
| Pruned *tf-idf*-M AvgSim | 60.4 |
| Pruned *tf-idf*-M AvgSimC | 60.5 |

**Fonte: Retirado de (HUANG et al., 2012)**

Um dos modelos de Huang et al. (2012) que usa vetores de sentido (em negrito) supera o modelo de vetor de palavra (Model-S) e também os modelos de palavra e sentido dos *baselines* Collobert e Weston (2008) e Reisinger e Mooney (2010).

[2]Os modelos terminados em -S representam vetores de palavra enquanto os terminados em -M são vetores de sentido. *Our Model* é Huang et al. (2012). C&W é o de Collobert e Weston (2008). *tf-idf* e Pruned são de Reisinger e Mooney (2010). AvgSim calcula a similaridade com cada vetor contribuindo igualmente, enquanto o AvgSimC leva em consideração a probabilidade da palavra pertencer ao *cluster* desse vetor.

### 4.1.2 Modelos baseados em treinamento unificado

A abordagem baseada em clusterização de contexto sofre por ocorrer em dois estágios, e esses estágios não tirarem vantagens um do outro. Outra limitação é precisar de grande capacidade computacional. Surgiu então a abordagem unificada, onde muitos pesquisadores propuseram extensões do modelo Skip-gram (MIKOLOV et al., 2013), realizando todo o treinamento em um único estágio. Essa abordagem é eficiente computacionalmente. A seguir são descritos os principais trabalhos que geram vetores de sentido através do treinamento unificado.

#### 4.1.2.1 Neelakantan et al. (2015)

Baseando-se em Huang et al. (2012), Neelakantan et al. (2015) propuseram a geração de vetores de sentido sem a etapa de pré-processamento das abordagens anteriores (dois estágios), seguindo uma abordagem diferente que é uma adaptação do Skip-Gram de Mikolov et al. (2013). Nessa abordagem, a identificação dos sentidos ocorre em conjunto com o treinamento para a geração dos vetores, tornando o processo eficiente e escalável (treinamento em conjunto).

Os autores propuseram dois métodos, o MSSG (*Multiple-sense Skip-gram*) e NP-MSSG (*Non-parametric Multiple-sense Skip-gram*), onde o primeiro implementa uma quantidade fixa de sentidos possíveis para cada palavra e o segundo faz essa descoberta em tempo de execução.

O sentido é predito como nas abordagens anteriores, onde a representação vetorial do contexto é dada pela média ponderada dos vetores que representam as palavras que o compõem. Esses vetores de contexto são agrupados e associados às palavras do córpus por aproximação a seu contexto. Após prever o sentido, realiza-se a atualização de gradiente sobre o centroide do *cluster* e o treinamento continua.

O modelo Skip-gram original (MIKOLOV et al., 2013) prevê palavras de contexto a partir de uma palavra-alvo. Nesse modelo, $v(w) \in R^d$ é a representação vetorial da palavra $w \in W$, onde $W$ é o vocabulário e $d$ é a dimensionalidade do vetor.

Dado um par de palavras $(w_t, c)$, a probabilidade da palavra $c$ ser observada no contexto da palavra $w_t$ é dada pela fórmula 4.3.

$$P(D = 1|v(w_t), v(c)) = \frac{1}{1 + e^{-v(w_t)^T v(c)}} \tag{4.3}$$

A probabilidade da palavra $c$ não ser observada no contexto de $w_t$ é dada pela fórmula 4.4.

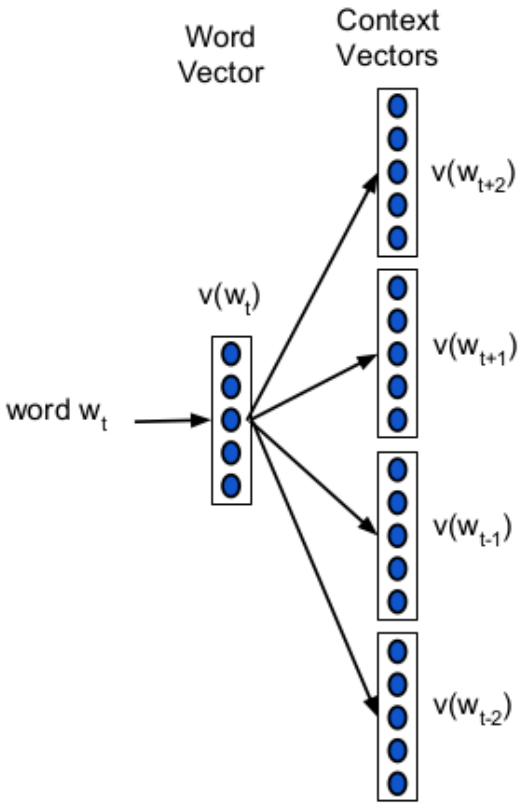
$$P(D = 0|v(w_t), v(c)) = 1 - P(D = 1|v(w_t), v(c)) \tag{4.4}$$

Dado um conjunto de treinamento com a sequência de palavras $w_1, w_2, \cdots, w_T$, vetores de palavras são aprendidos maximizando a função objetivo apresentada na equação 4.5.

$$J(\theta) = \sum_{(w_t,c_t)\in D_+} \sum_{c\in c_t} logP(D=1|v(w_t),v(c)) + \sum_{(w_t,c'_t)\in D_-} \sum_{c'\in c'_t} logP(D=0|v(w_t),v(c')) \tag{4.5}$$

onde $w_t$ é a $t$-ésima palavra no conjunto de treinamento, $c_t$ é o conjunto observado de palavras do contexto da palavra $w_t$ e $c'_t$ é o conjunto de palavras de contexto ruidosas escolhidas aleatoriamente para a palavra $w_t$. $D_+$ consiste do conjunto de todos os pares de palavras de contexto observadas $(w_t, c_t)(t = 1,2,\cdots,T)$. $D_-$ consiste de pares $(w_t, c'_t)(t = 1,2,\cdots,T)$ onde $c'_t$ é o conjunto de palavras de contexto ruidosas mostradas aleatoriamente para a palavra $w_t$. Para cada treinamento da palavra $w_t$, o conjunto de palavras de contexto $c_t = W_t - R_t, \cdots, W_{t-1}$, $W_{t+1}, \cdots, W_t + R_t$ inclui palavras $R_t$ à esquerda e à direita da palavra-alvo, como mostrado na Figura 4.3. A arquitetura tem janela $R_t = 2$ e contexto $c_t$ da palavra $w_t$ consistindo de $w_{t-1}, w_{t-2}, w_{t+1}, w_{t+2}$.

**Figura 4.3: Arquitetura do modelo Skip-gram estendido de Neelakantan et al. (2015).**

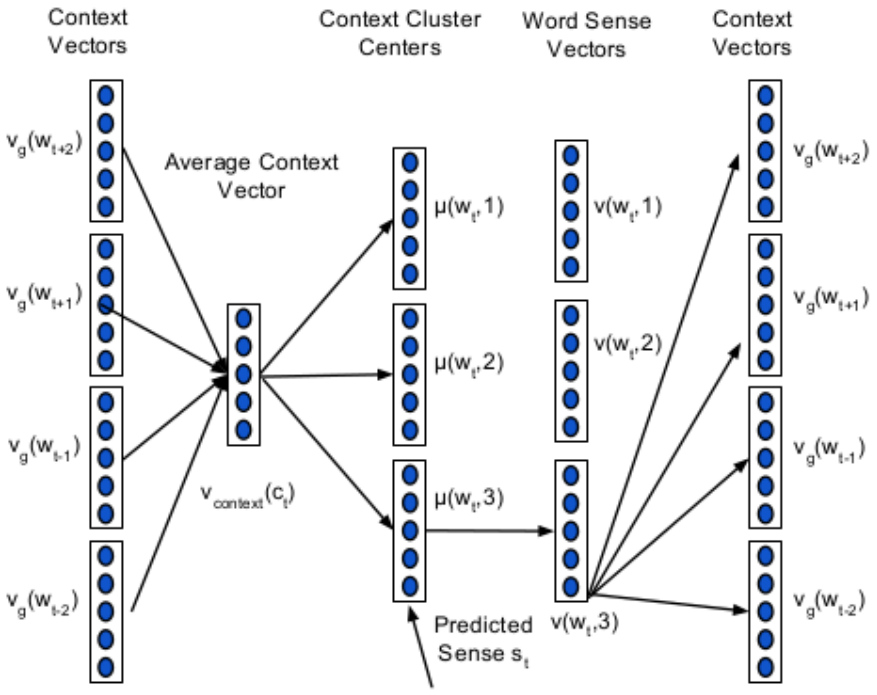



**Fonte: Retirado de (NEELAKANTAN et al., 2015)**

As extensões do modelo Skip-gram, MSSG e NP-MSSG, aprendem múltiplos vetores por palavra e foram baseadas em trabalhos como (HUANG et al., 2012) e (REISINGER; MOONEY, 2010). No modelo MSSG, cada palavra $w \in W$ é associada a um vetor global $v_g(w)$ e cada sentido da palavra tem um vetor de sentido $v_s(w,k)(k = 1,2,\cdots,K)$ e um *cluster* de contexto com centroide $u(w,k)(k = 1,2,\cdots,K)$. O vetor de sentido $K$ e os vetores globais são de dimensão $d$ e $K$ é um hiperparâmetro.

Considerando a palavra $w_t$ e $c_t = W_t - R_t,\cdots,W_{t-1},W_{t+1},\cdots,W_t + R_t$ como o conjunto de palavras de contexto observadas, a representação vetorial do contexto é definida como a média da representação vetorial global das palavras no contexto. Seja $v_{context}(c_t) = \frac{1}{2*R_t}\sum_{c \in c_t} v_g(c)$ a representação vetorial do contexto $c_t$. Usa-se os vetores globais das palavras de contexto em vez de seus vetores de sentido para evitar a complexidade computacional associada à predição dos sentidos das palavras do contexto. É possível, então, predizer o sentido da palavra $w_t$, $s_t$, quando observada com o contexto $c_t$. A Figura 4.4 mostra esse processo. A arquitetura tem janela $R_t = 2$, $K = 3$ e o contexto $c_t$ da palavra $w_t$ consiste de $w_{t-1}, w_{t-2}, w_{t+1}, w_{t+2}$.

**Figura 4.4: Arquitetura do modelo MSSG de Neelakantan et al. (2015).**




**Fonte: Retirado de (NEELAKANTAN et al., 2015)**

O algoritmo usado na construção dos *clusters* é semelhante ao *k-means*. O centroide do *cluster* é a média das representações vetoriais de todos os contextos que pertencem a esse *cluster*. Como medida de similaridade, usa-se a similaridade de cosseno.

No MSSG, a probabilidade de que a palavra $c$ seja observada no contexto da palavra $w_t$, dado o sentido e a probabilidade de que não seja observada, tem a adição de $s_t$ (sentido de $w_t$) nas fórmulas do *Skip-gram* original (fórmulas 4.3 e 4.4). A função objetivo (4.5) também passa a considerar $(w_t, s_t)$ ao invés de apenas $(w_t)$. $D_+$ e $D_-$ são construídos da mesma maneira que no modelo de *Skip-gram* original. Depois de prever o sentido da palavra $w_t$, atualiza-se o vetor de sentido gerado para a palavra $w_t(v_s(w_t, s_t))$, o vetor global das palavras do contexto e o vetor global das palavras de contexto ruidosas escolhidas aleatoriamente. O centroide do *cluster* de contexto $s_t$ para a palavra $w_t(u(w_t, s_t))$ é atualizado quando o contexto $c_t$ é adicionado ao *cluster* $s_t$.

O modelo NP-MSSG tem a vantagem de aprender um número variável de sentidos por palavra. A proposta desse método de clusterização não-paramétrica online está baseada em (MEYERSON, 2001), onde cria-se um novo *cluster* (sentido) para uma palavra através da probabilidade proporcional à distância de seu contexto ao *cluster* mais próximo (sentido). O número de sentidos para uma palavra é desconhecido e é aprendido durante o treinamento. Inicialmente, as palavras não têm vetores de sentido e *clusters* de contexto. Cria-se o primeiro vetor de sentido e *cluster* de contexto para cada palavra em sua primeira ocorrência nos dados de treinamento. Um novo *cluster* de contexto e um vetor de sentido são criados online durante o treinamento quando a distância do vetor do contexto de uma palavra observada e o centroide de cada *cluster* existente é menor que $\lambda$, que é um hiperparâmetro do modelo.

Neelakantan et al. (2015) usaram o mesmo córpus que Huang et al. (2012) para treinamento dos vetores específicos de sentido: um *dump* de abril de 2010 da Wikipédia em inglês com 2 milhões de artigos e 990 milhões de *tokens*.

Os autores avaliaram a proposta nos córpus em inglês *WordSim-353* e *Stanford's Contextual Word Similarities* (SCWS). O SCWS é a adaptação do *WordSim-353* desenvolvida por Huang et al. (2012). Foram utilizadas algumas medidas de similaridade, como: (i) a avgSim, que calcula a similaridade média de todos os vetores de uma palavra, ignorando informações de contexto; (ii) a avgSimC, que calcula a similaridade de cada par de vetor de sentido com o contexto atual; (iii) a globalSim, que usa o vetor global de contexto de cada palavra com o contexto atual e (iv) a localSim, que seleciona um único sentido para cada palavra com o contexto atual.

Na Tabela 4.3, são apresentados os valores para a correlação de *Spearman* (veja seção 3.1) entre os julgamentos de similaridade de cada modelo e o julgamento humano, presente no conjunto de dados SCWS.[3] Nota-se que os modelos propostos (MSSG e NP-MSSG) superam os

[3]Os primeiros três modelos, Collobert e Weston (2008) e Mikolov et al. (2013), usam a abordagem de vetor de palavra, portanto, avgSim, avgSimC e localSim não são calculadas. *Pruned tf-idf* é o modelo de vetor de sentido de Reisinger e Mooney (2010) e *Huang et al-50d* é o modelo de vetor de sentido de Huang et al. (2012). Os

*baselines*.

**Tabela 4.3: Correlação de *Spearman* no córpus SCWS.**

| Model | globalSim | avgSim | avgSimC | localSim |
|---|---|---|---|---|
| TF-IDF | 26.3 | - | - | - |
| Collobort & Weston-50d | 57.0 | - | - | - |
| Skip-gram-50d | 63.4 | - | - | - |
| Skip-gram-300d | 65.2 | - | - | - |
| Pruned TF-IDF | 62.5 | 60.4 | 60.5 | - |
| Huang et al-50d | 58.6 | 62.8 | 65.7 | 26.1 |
| MSSG-50d | 62.1 | 64.2 | 66.9 | 49.17 |
| MSSG-300d | 65.3 | 67.2 | **69.3** | 57.26 |
| NP-MSSG-50d | 62.3 | 64.0 | 66.1 | 50.27 |
| NP-MSSG-300d | **65.5** | **67.3** | 69.1 | **59.80** |

**Fonte: Retirado de (NEELAKANTAN et al., 2015)**

O trabalho de Neelakantan et al. (2015) foi escolhido como base de uma das abordagens deste trabalho, a que chamados de **agrupamento**.

#### 4.1.2.2 Li e Jurafsky (2015)

Li e Jurafsky (2015) alegam que vetores de sentido levam a modelos mais poderosos de representações no espaço vetorial, porém, não se sabe se eles levam realmente a melhorias nas tarefas de compreensão da linguagem.

Os autores baseiam-se em Huang et al. (2012) e Neelakantan et al. (2015) e propõem um algoritmo no qual uma palavra só é associada a um novo vetor de sentido quando uma evidência no contexto (por exemplo, palavras vizinhas, estatísticas de co-ocorrência) sugere que ela é suficientemente diferente dos sentidos existentes até então.

Essa linha de pensamento aponta naturalmente para o uso de Processos de Restaurantes Chineses (*Chinese Restaurant Processes* (CRP) (GRIFFITHS et al., 2004; TEH et al., 2005), que foram aplicados no campo relacionado à indução de sentido de palavras. Na analogia do CRP, a palavra atual pode estar em uma das mesas existentes (pertencendo a um dos sentidos existentes) ou escolher uma nova mesa (um novo sentido). A decisão é tomada medindo a relação semântica (com base em informações de contexto local e informações de documentos globais) e o número de clientes (palavras) já sentados nessa mesa (a popularidade dos sentidos das palavras).

modelos MSSG e NP-MSSG são os de vetor de sentido de Neelakantan et al. (2015) e superam os demais.

O cálculo do sentido ótimo global (ideal) exige procurar pelo espaço de todos os sentidos para todas as palavras, o que pode ser caro. Por isso, os autores escolheram duas abordagens heurísticas simples:

- ***Greedy Search***: atribui o rótulo de sentido ótimo local para cada *token* e o representa com o vetor de palavra associado a esse sentido.
- ***Expectation***: calcula a probabilidade de cada sentido possível para a palavra atual e representa a palavra com o vetor de expectativa:

$$e_w = \sum_{z \in Z_w} p(w|z, context) * e_w^z \tag{4.6}$$

Os autores avaliaram os modelos propostos no *dataset* SCWS de Huang et al. (2012), onde as avaliações humanas foram feitas em pares de palavras em contexto sentencial. Os modelos foram treinados na Wikipédia com 1,1 bilhão de *tokens* e em uma combinação da Wikipédia, *Gigaword* e *Common Crawl Dataset*, composto por 120 bilhões de *tokens*.

O resultado pode ser visto na Tabela 4.4, onde os modelos propostos foram comparados ao Skip-gram de Mikolov et al. (2013), superando esse *baseline* em todas as configurações testadas. SG+Greedy é o modelo proposto *Greedy Search* e SG+Expect é o *Expectation*.

**Tabela 4.4: Correlação de *Spearman* entre a avaliação de similaridade de cada modelo e as avaliações humanas do SCWS.**

| Model | Dataset | SCWS Correlation |
|---|---|---|
| SkipGram | 1.1B (wiki) | 64.6 |
| SG+Greedy | 1.1B (wiki) | 66.4 |
| SG+Expect | 1.1B (wiki) | 67.0 |
| SkipGram | 120B | 66.4 |
| SG+Greedy | 120B | 69.1 |
| SG+Expect | 120B | 69.7 |

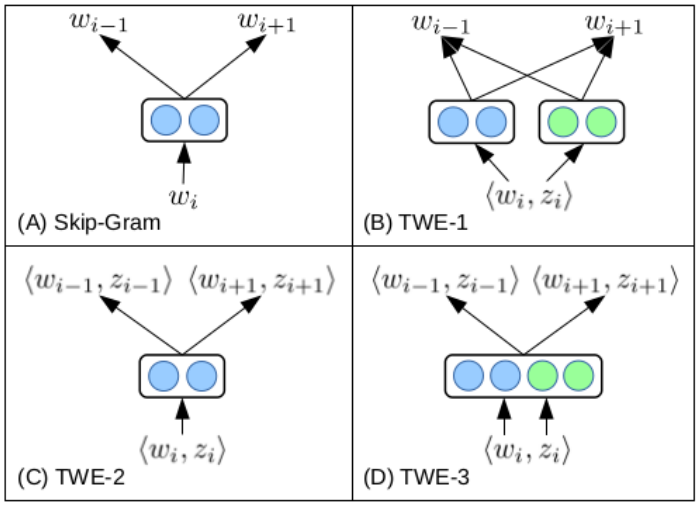
**Fonte: Retirado de (LI; JURAFSKY, 2015)**

#### 4.1.2.3 Liu et al. (2015b)

Liu et al. (2015b) propõem a geração de vetores de sentido utilizando tópicos extraídos do córpus como informação adicional. A ideia é gerar vetores com base nas duas informações: a palavra e seu tópico. Os autores deram a esta estratégia o nome de *Topical Word Embeddings* (TWE). Por exemplo, a palavra "*apple*" pode referenciar a fruta sob o tópico *food* ou a empresa de TI, que aparece sob o tópico *information technology* (IT).

Os tópicos são extraídos por meio da aplicação do algoritmo LDA (BLEI et al., 2003), onde dada uma sequência de palavras $D = w_1, \cdots, w_M$ depois do processamento do LDA, cada *token* $w_i$ é discriminado em um tópico $z_i$, formando um par palavra/tópico $(w_i, z_i)$, que pode ser usado para gerar TWE. Os autores apresentam três modelos de TWE, mostrados na Figura 4.5, onde o tamanho da janela é 1 e $w_{i-1}$ e $w_{i+1}$ são palavras contextuais de $w_i$.

**Figura 4.5: Comparação entre os modelos Skip-Gram e TWE. Os círculos azuis indicam vetores de palavras e os círculos verdes indicam vetores de tópicos.**




**Fonte: Retirado de (LIU et al., 2015b)**

- TWE-1: Esse modelo considera cada tópico como uma pseudo palavra e gera vetores de tópicos e vetores de palavras separadamente. O treinamento de $(w_i, z_i)$ é, então, realizado de acordo com os vetores de $w_i$ e $z_i$.
- TWE-2: Esse modelo considera cada par de palavra e tópico $w_i, z_i$ como uma pseudo palavra e gera o TWE.
- TWE-3: Esse modelo gera vetores de palavras e tópicos separadamente. Em seguida, ele constrói o vetor de cada par de palavra e tópico concatenando seus respectivos vetores e gera o TWE.

Como mostrado na Tabela 4.5, os autores avaliaram os modelos com *baselines* que incluem modelos de vetores de palavra (C&W, TFIDF, *Pruned* TFIDF, LDA-S, LDA-C e Skip-Gram) e de vetores de sentido (*Pruned* TFIDF-M, o modelo de Huang e o modelo de Tian).[4]

[4]C&W é Collobert e Weston (2008), TFIDF (incluindo *Pruned* TFIDF e *Pruned* TFIDF-M) são Reisinger e Mooney (2010), o modelo de Huang é Huang et al. (2012), o modelo de Tian é Tian et al. (2014), LDA é Blei et al. (2003) e Skip-Gram é Mikolov et al. (2013)

**Tabela 4.5: Correlação de Spearman no *dataset* SCWS.**

| Model | $\rho \times 100$ | |
|---|---|---|
| C&W | 57.0 | |
| TFIDF | 26.3 | |
| Pruned TFIDF | 62.5 | |
| LDA-S | 56.9 | |
| LDA-C | 50.4 | |
| Skip-Gram | 65.7 | |
| | AvgSimC | MaxSimC |
| Pruned TFIDF-M | 60.5 | 60.4 |
| Tian | 65.4 | 63.6 |
| Huang | 65.3 | 58.6 |
| TWE-1 | **68.1** | **67.3** |
| TWE-2 | 67.9 | 63.6 |
| TWE-3 | 67.1 | 65.5 |

**Fonte: Retirado de (LIU et al., 2015b)**

Todos os modelos TWE superam os *baselines* de vetores de sentido e de palavra. Já o modelo TWE-1 (em negrito) é o que alcança melhor desempenho.

### 4.1.3 Modelos de representações contextualizadas

Dado que os modelos não supervisionados são frequentemente gerados através de clusterização de contextos, os sentidos gerados não são claros e seu mapeamento para conceitos bem definidos não é direto. Recentemente, uma linha emergente de pesquisa concentrou-se na integração direta de representações vetoriais em tarefas de PLN. As representações contextualizadas são sensíveis ao contexto, ou seja, sua representação muda dinamicamente dependendo do contexto em que elas aparecem.

O *tagger* de sequência de Li e McCallum (2005) é um dos trabalhos pioneiros que empregam representações contextualizadas. O modelo infere variáveis sensíveis ao contexto para cada palavra e integra-as a um *tagger* de sequência de CRF (*Conditional Random Field*). Context2vec (MELAMUD et al., 2016) é uma das primeiras e mais proeminentes propostas no novo ramo das representações contextualizadas. O modelo representa o contexto de uma palavra extraindo o vetor de saída de um perceptron multicamadas construído sobre um modelo de linguagem LSTM bidirecional. Context2vec constitui a base para muitos dos trabalhos subsequentes.

O modelo Context Vectors (CoVe) de McCann et al. (2017) calcula representações contextualizadas usando uma rede LSTM bidirecional de duas camadas, em um modelo de tradução automática sequência-a-sequência (*seq2seq*) com atenção. A técnica ELMo (*Embeddings from*

*Language Model*) (PETERS et al., 2018) usa um modelo de linguagem bidirecional LSTM de múltiplas camadas em textos monolíngues e alguns pesos são compartilhados entre as duas direções do modelo.

Seguindo a ideia de ELMo, OpenAI GPT (*Generative Pre-training Transformer*) (RADFORD et al., 2018), expande o modelo de linguagem não supervisionado para uma escala muito maior, treinando em uma gigantesca coleção de texto puro[5]. Difere do ELMo na arquitetura e no uso das representações contextualizadas em tarefas de PLN.

O BERT (*Bidirectional Encoder Representations from Transformers*) (DEVLIN et al., 2018) é descendente direto do GPT: um modelo de linguagem grande treinado em texto puro e ajustado em tarefas específicas de PLN, sem arquiteturas de rede personalizadas. Em comparação com o GPT, a maior diferença e melhoria do BERT é tornar o treinamento bidirecional.

Uma limitação dessa abordagem de representação contextualizada é o alto custo computacional devido ao tamanho/complexidade dos modelos e do córpus.

## 4.2 Representações que exploram córpus multilíngue ou etiquetado

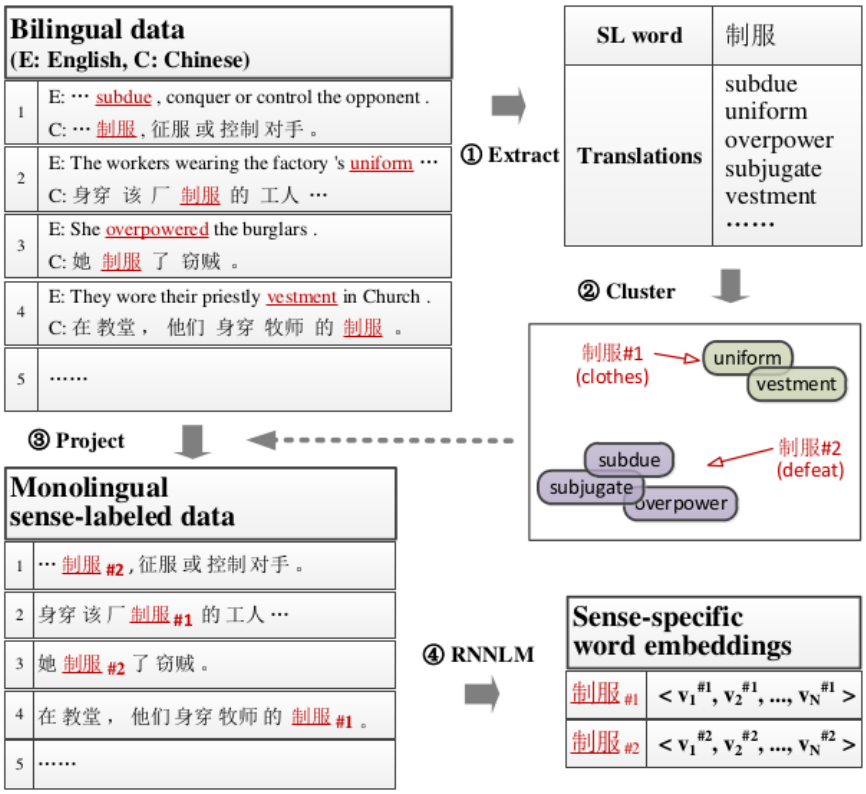
A abordagem de equivalência translacional (córpus multilíngue) baseia-se no fato de que diferentes sentidos de uma palavra são normalmente traduzidos para diferentes palavras em outra língua. Assim, a lexicalização em diferentes línguas pode ser usada para definir e estruturar sentidos. A abordagem que usa texto etiquetado ao invés de texto puro, faz a identificação de sentidos a partir do conhecimento trazido por essas etiquetas (ex: etiquetas morfossintáticas, de sentimentos ou de entidade nomeada). A seguir são descritos os principais trabalhos que geram vetores de sentido através de equivalência translacional ou córpus etiquetado.

### 4.2.1 Guo et al. (2014)

Guo et al. (2014) propuseram a geração de vetores de sentido a partir de dados paralelos bilíngues. Os autores utilizaram a tarefa de indução de sentido da palavra, do inglês *Word Sense Induction* (WSI), para descobrir o sentido da palavra antes da geração de vetores. Inspirados por Gale et al. (1992b) e Chan e Ng (2005), que usaram dados bilíngues para gerar automaticamente exemplos de treinamento de DLS, Guo et al. (2014) apresentam uma abordagem bilíngue para WSI não supervisionado, como mostrado na Figura 4.6.

[5]Texto sem etiquetas

**Figura 4.6: Ilustração do método proposto. SL é o idioma de origem.**




**Fonte: Retirado de (GUO et al., 2014)**

Primeiro, são identificadas as traduções de cada palavra do idioma fonte no idioma alvo. Uma vez que podem haver várias traduções para uma única palavra do idioma fonte, essas traduções são agrupadas por sentido, exibindo diferentes sentidos em diferentes *clusters*, como melhor detalhado a seguir. O próximo passo é etiquetar as palavras do idioma fonte com cada *cluster*, induzindo o sentido correto nas palavras ambíguas e gerando o córpus de treinamento para os vetores de sentido. A proposta de Guo et al. (2014) utilizou córpus paralelo bilíngue nos idiomas chinês (fonte) e inglês (alvo).

Os autores apresentam uma maneira de extrair traduções para as palavras do idioma fonte explorando a probabilidade de tradução produzida por modelos de alinhamento de palavras (BROWN et al., 1993; OCH; NEY, 2003; LIANG et al., 2006). Mais formalmente, a sentença chinesa é representada como $c = (c_1, \cdots, c_I)$ e a sentença em inglês como $e = (e_1, \cdots, e_J)$. Os modelos de alinhamento podem ser considerados como:

$$p(c|e) = \sum_a p(a, c|e) \tag{4.7}$$

$$p(a, c|e) = \pi_{j=1}^{J} Pd(a_j|a_{j-}, j) Pt(c_j|e_{a_j}) \tag{4.8}$$

onde $a$ é o alinhamento que especifica a posição de uma palavra do inglês alinhada a cada palavra do chinês, $Pd(a_j|a_{j-}, j)$ é a probabilidade de distorção e $Pt(c_j|e_{a_j})$ é a probabilidade de tradução.

Para cada palavra do idioma fonte, as traduções são agrupadas por sentidos. No momento da clusterização, cada tradução é representada como um vetor de palavra, com o objetivo de medir a similaridade entre eles. A representação vetorial de palavra é escolhida por dois motivos: (1) codifica bem a semântica lexical e pode ser usada diretamente para medir a similaridade das palavras e (2) leva a agrupamentos extremamente eficientes. Para a clusterização, foi utilizado o algoritmo *Affinity Propagation* (AP) (FREY; DUECK, 2007). O AP tem como principal vantagem o fato de que o número de *clusters* resultantes é dinâmico e depende principalmente da distribuição dos dados. Em comparação com outras possíveis abordagens de clusterização, como o *Hierarchical Agglomerative Clustering* (KARTSAKLIS et al., 2013), o AP determina o número de *clusters* resultantes automaticamente, sem usar nenhum critério de partição.

Os *clusters* produzidos são, então, projetados de volta ao idioma fonte, identificando os sentidos das palavras. Para cada ocorrência $w^o$ da palavra $w$ no idioma fonte, primeiro seleciona-se a palavra alinhada como sua tradução. Identifica-se, então, o sentido de $w^o$ com base na similaridade da tradução com cada exemplar dos *clusters*, selecionando aquele com a máxima similaridade. Quando $w^o$ está alinhado com NULL (ou seja, não há palavra correspondente no lado alvo), atribui-se o sentido mais frequente de $w$ presente no conjunto de dados bilíngue. Depois de projetar os sentidos das palavras no idioma fonte, obtém-se um córpus anotado com sentidos, que é usado para treinar os vetores de sentido com RNNLM (*Recurrent Neural Network Language Model*). O processo de treinamento é exatamente o mesmo que os de vetores de palavras, exceto pelo fato de que as palavras foram previamente rotuladas com etiquetas de sentido.

A proposta foi avaliada em tarefas intrínsecas e tarefas de PLN. Para as intrínsecas, os autores construíram manualmente uma versão do WordSim-353 para o chinês, contendo 401 pares de palavras ambíguas, cuja a similaridade também foi avaliada por humanos. Para avaliar a capacidade de identificar similaridade entre palavras, a proposta usou as medidas MaxSim e AvgSim, em conjunto com a similaridade de cosseno. Os números da Tabela 4.6 mostram a correlação de *Spearman* e *Kendall* de três propostas, incluindo a dos autores (Ours), com o conjunto de dados de pares ambíguos criado manualmente. O modelo proposto pelos autores (em negrito) supera o *baseline* de vetor de palavra (SingleEmb) e o de vetor de sentido de Huang et al. (2012) (Multi-prototype).

Os autores também avaliaram o modelo proposto na tarefa de Reconhecimento de Entidades Nomeadas (NER) para o chinês. O experimento foi realizado no conjunto de dados Diário

**Tabela 4.6: Correlação de *Spearman* ($\rho$ x 100) e *Kendall* ($\tau$ x 100) de três propostas, incluindo a dos autores.**

| System | MaxSim | | AvgSim | |
|---|---|---|---|---|
| | $\rho \times 100$ | $\tau \times 100$ | $\rho \times 100$ | $\tau \times 100$ |
| **Ours** | **55.4** | **40.9** | 49.3 | 35.2 |
| SingleEmb | 42.8 | 30.6 | 42.8 | 30.6 |
| Multi-prototype | 40.7 | 29.1 | 38.3 | 27.4 |

**Fonte: Retirado de (GUO et al., 2014)**

do Povo (edições de janeiro e junho de 1998) (*People's Daily*). Foram selecionados três tipos mais comuns de entidades: pessoa, localização e organização. A rede foi treinada nos dados de janeiro. Os dados de junho foram divididos em conjunto de desenvolvimento e teste. A Tabela 4.7[6] mostra o desempenho do modelo no conjunto de teste, utilizando métricas de Precisão (P), Recall (R) e F1-score (F). Um dos modelos dos autores que usa vetores de sentido (em negrito) supera os demais modelos, de sentido (SenseEmb (greedy)) e de palavra (SingleEmb e *Baseline*).

**Tabela 4.7: Performance de propostas aplicadas a tarefa de NER para o chinês.**

| System | P | R | F |
|---|---|---|---|
| Baseline | 93.27 | 81.46 | 86.97 |
| +SingleEmb | 93.55 | 82.32 | 87.58 |
| +SenseEmb (greedy) | 93.38 | 83.56 | 88.20 |
| +SenseEmb (beam search) | **93.59** | **84.05** | **88.56** |

**Fonte: Retirado de (GUO et al., 2014)**

Limitações dessa proposta são a necessidade de córpus paralelos bilíngues representativos quanto aos sentidos das palavras ambíguas e o custo computacional.

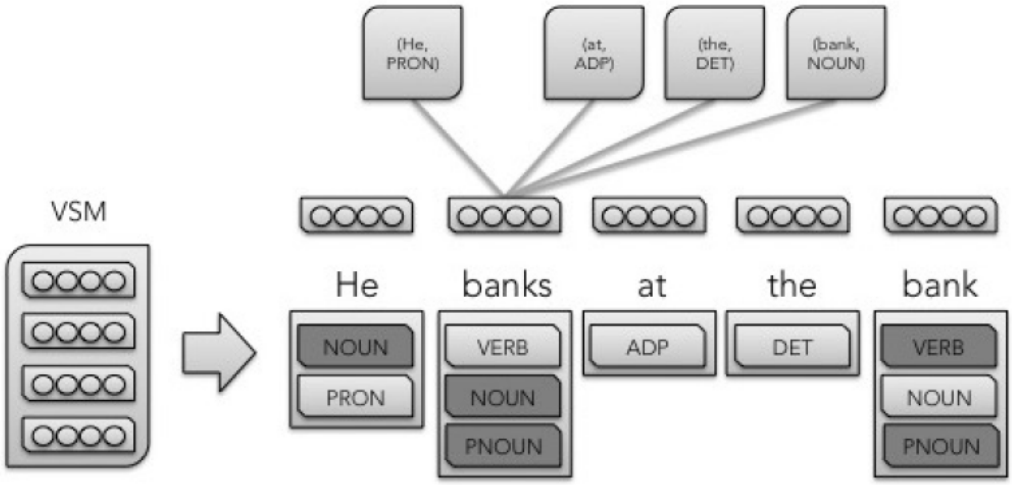
### 4.2.2 Trask et al. (2015)

Trask et al. (2015) propõem a geração de vetores de sentido a partir de um córpus anotado com etiquetas morfossintáticas (*part-of-speech* ou PoS), possibilitando identificar palavras ambíguas a partir da quantidade de etiquetas gramaticais que recebem (por exemplo, "livro" substantivo e "livro" verbo). A estrutura do sense2vec é mostrada na Figura 4.7.

Os autores sugeriram que a estratégia de anotar o córpus com etiquetas de *part-of-speech* é vantajosa por descobrir, de forma pouco custosa, palavras empregadas em mais de um contexto,

[6] O *baseline* é Che et al. (2013). As demais propostas são adaptadas do modelo proposto em (GUO et al., 2014).

**Figura 4.7: Estrutura do Sense2vec de Trask et al. (2015).**




**Fonte: Retirado de (TRASK et al., 2015)**

já que essas palavras terão uma etiqueta gramatical diferente em cada contexto. Essa estratégia possibilita criar uma representação de sentido para cada uso. O passo final concerne em treinar um modelo word2vec (CBOW ou Skip-Gram) (MIKOLOV et al., 2013) com o córpus etiquetado, de forma que, ao invés de prever uma palavra dadas as palavras vizinhas, prediz um sentido dados os sentidos circundantes.

A Tabela 4.8 mostra a similaridade de cosseno da palavra "*bank*" (banco) desambiguada pelo método proposto. O córpus utilizado é do inglês e sem anotação prévia, utilizado na tarefa *Google Word Analogy Task* e disponibilizado pelo Google.

**Tabela 4.8: Similaridade de cosseno para a palavra "*bank*", desambiguada por etiquetas de *part-of-speech*.**

| bank | NOUN | 1.0 | bank | PROPN | 1.0 | bank | VERB | 1.0 |
|---|---|---|---|---|---|---|---|---|
| banks | NOUN | .786 | bank | NOUN | .570 | gamble | VERB | .533 |
| banking | NOUN | .629 | hsbc | PROPN | .536 | earn | VERB | .485 |
| lender | NOUN | .619 | citibank | PROPN | .523 | invest | VERB | .470 |
| bank | PROPN | .570 | wachovia | PROPN | .503 | reinvest | VERB | .466 |
| ubs | PROPN | .535 | grindlays | PROPN | .492 | donate | VERB | .466 |

**Fonte: Retirado de (TRASK et al., 2015)**

Para a análise de sentimento, o sense2vec foi treinado com um córpus anotado com etiquetas de PoS e adjetivos com etiquetas de sentimento. A palavra "*bad*" foi desambiguada entre sentimento positivo e negativo onde de um lado apareceram palavras como "*terrible*", "*horrible*" e "*awful*", indicando o sentimento negativo de "*bad*". Já do lado positivo apareceram

"*good*", "*wrong*" e "*funny*", indicando um sentido mais sarcástico de "*bad*".

Na tarefa de Reconhecimento de Entidade Nomeada (do inglês, *Named Entity Recognition* ou NER), o sense2vec foi treinado com um córpus anotado com etiquetas de PoS e NER. A palavra "*Washington*" foi desambiguada entre as categorias de entidade nomeada PERSON_NAME (nome de pessoa) e GPE (geolocalização). Na categoria PERSON_NAME apareceram palavras como "*George_Washington*", "*Henry_Knox*" e "*Philip_Schuyler*" e na categoria GPE apareceram "*Washington_DC*", "*Seattle*" e "*Maryland*". A palavra "*Hillary_Clinton*" também foi desambiguada entre as categorias TITLE (título) e ORG_NAME (organização). Para TITLE apareceram palavras como "*Secretary_of_State*", "*Senator*" e "*Chief*" e para ORG_NAME apareceram "*Senate*", "*White_House*" e "*Congress*".

A limitação dessa proposta está em não ser capaz de lidar com ambiguidades onde os sentidos de uma mesma palavra pertencem à mesma classe gramatical, como em "banco" instituição financeira e "banco" de uma praça, que são ambos substantivos.

O trabalho de Trask et al. (2015) foi escolhido como base de uma das abordagens deste trabalho, a que chamados de **etiquetação**.

## 4.3 Considerações finais

A seguir, na Tabela 4.9, apresenta-se um resumo das principais características dos trabalhos relacionados descritos neste capítulo. Vale ressaltar, também, para o português do Brasil, até onde se tem conhecimento, não existem propostas de geração de vetores de sentido.

**Tabela 4.9: Resumo das principais características dos trabalhos de geração de vetores de sentido apresentados neste capítulo**

| **Trabalho** | **Metodologia** | **Idioma** | **Recurso** | **Avaliação** |
|---|---|---|---|---|
| Reisinger e Mooney (2010) | Vetores de sentido | EN | Wikipédia<br>*Gigaword*<br>*WordSim-353* | *Spearman* = 62,5% |
| Huang et al. (2012) | Vetores de sentido | EN | Wikipédia<br>*WordSim-353*<br>SCWS | *Spearman* = 65,7% |
| Neelakantan et al. (2015) | Vetores de sentido | EN | Wikipédia<br>*WordSim-353*<br>SCWS | *Spearman* = 67,3% |
| Li e Jurafsky (2015) | Vetores de sentido | EN | Wikipédia<br>Wikipédia+*Gigaword*+<br>*Common Crawl Dataset* | *Spearman* = 69,7% |
| Liu et al. (2015b) | Vetores de sentido | EN | Wikipédia<br>SCWS | *Spearman* = 68,1% |
| Guo et al. (2014) | Vetores de sentido | CH-EN | *WordSim-353* Chinês<br>*People's Daily* | *Spearman* = 55,4%<br>*Kendall* = 40,9% |
| Trask et al. (2015) | Vetores de sentido | EN | *Google Word*<br>*Analogy Task* | - |

**Fonte: Autoria própria**

# Capítulo 5

## EXPERIMENTOS

Com base nos trabalhos relacionados citados anteriormente, duas estratégias não supervisionadas baseadas em córpus foram utilizadas para encontrar os diferentes sentidos de cada palavra ambígua e gerar seus respectivos vetores. As duas estratégias são:

1. **Agrupamento** – Agrupamento de sentidos a partir de um córpus não anotado, tendo como base o trabalho de Neelakantan et al. (2015). O objetivo é gerar vetores de sentido a partir de um córpus sem anotação prévia (texto puro) e representativo quanto a ambiguidades lexicais. Assim como em Neelakantan et al. (2015), gerou-se representações vetoriais para cada sentido aplicando o algoritmo *Multi-Sense Skip-Gram* (MSSG)[1];

2. **Etiquetação** – Identificação de sentidos a partir de um córpus anotado com etiquetas de PoS com base na proposta de Trask et al. (2015). O objetivo é gerar vetores de sentido a partir de um córpus anotado com etiquetas morfossintáticas usando um PoS *tagger*[2] , possibilitando identificar palavras ambíguas a partir da quantidade de etiquetas gramaticais que recebem (por exemplo, "livro" substantivo e "livro" verbo). A partir do córpus anotado, o algoritmo sense2vec[3] (TRASK et al., 2015) foi aplicado para a geração dos vetores de sentido.

A seguir, são apresentados o córpus utilizado para gerar os vetores de sentido (5.1), os parâmetros de rede utilizados para o treinamento (5.2) e os experimentos realizados para avaliar as duas abordagens: MSSG e Sense2Vec (5.3).

[1]Disponível em ⟨https://bitbucket.org/jeevan_shankar/multi-sense-skipgram⟩. Acesso em: 07 jan. 2018.
[2]nlpnet. Disponível em: ⟨nilc.icmc.usp.br/nlpnet/⟩. Acesso em: 06 jan. 2018.
[3]Disponível em: ⟨https://github.com/explosion/sense2vec⟩. Acesso em: 08 jan. 2018.

## 5.1 Córpus de Treinamento

O córpus utilizado para o treinamento dos vetores de sentido foi o mesmo de Hartmann et al. (2017) que é composto por textos do português brasileiro (PT-BR) e português europeu (PT-EU). Nesse trabalho, um grande córpus foi coletado para obter variedade de gênero e representatividade da língua portuguesa. A Tabela 5.1 resume as informações sobre esse córpus: nome, quantidade de *tokens* e *types* e uma breve descrição do gênero.

**Tabela 5.1: Estatísticas do córpus de treinamento**

| Córpus | *Tokens* | *Types* | Gênero |
|---|---|---|---|
| LX-Corpus (RODRIGUES et al., 2016) | 714.286.638 | 2,605,393 | Misto |
| Wikipedia | 219.293.003 | 1.758.191 | Enciclopédico |
| GoogleNews | 160.396.456 | 664.320 | Informativo |
| SubIMDB-PT | 129.975.149 | 500.302 | Falas |
| G1 | 105.341.070 | 392.635 | Informativo |
| PLN-Br (BRUCKSCHEN et al., 2008) | 31.196.395 | 259.762 | Informativo |
| Alfabetização de domínio público | 23.750.521 | 381.697 | Prosa |
| Lacio-web (ALUÍSIO et al., 2003) | 8.962.718 | 196.077 | Misto |
| Portuguese e-books | 1.299,008 | 66.706 | Prosa |
| Mundo Estranho | 1.047.108 | 55.000 | Informativo |
| CHC | 941.032 | 36.522 | Informativo |
| FAPESP | 499.008 | 31.746 | Ciência |
| Textbooks | 96.209 | 11.597 | Didático |
| Folhinha | 73.575 | 9.207 | Informativo |
| NILC subcorpus | 32.868 | 4.064 | Informativo |
| Para Seu Filho Ler | 21.224 | 3.942 | Informativo |
| SARESP | 13.308 | 3.293 | Didático |
| **Total** | **1.395.926.282** | **3.827.725** | |

**Fonte: Autoria própria**

Similar ao que foi feito em Hartmann et al. (2017) para gerar vetores de palavras, o córpus foi pré-processado para reduzir o tamanho do vocabulário, sob a premissa de que a redução do vocabulário fornece mais elementos representativos. Assim, as palavras com menos de cinco ocorrências foram substituídas por um símbolo especial DESCONHECIDO; os numerais foram normalizados para zeros; as URLs foram mapeadas para o *token* URL e os emails para o *token* EMAIL. O texto foi tokenizado com base em espaços em branco e sinais de pontuação, com atenção especial à hifenização.

Para o modelo sense2vec, o córpus foi anotado com etiquetas de PoS usando a ferramenta nlpnet (FONSECA; ROSA, 2013), que é considerada o estado da arte em anotação morfossintática para o PT-BR. Um exemplo de uma sentença com e sem etiqueta é mostrado na tabela 5.2. Nesse exemplo, a palavra ambígua "marca" ocorre em sentenças com dois significados distintos: marca registrada de um fabricante (sentido1) e o infinitivo do verbo marcar (sentido2).

**Tabela 5.2: Exemplo de sentença com e sem anotação da palavra ambígua *marca***

| | |
|---|---|
| marca$_{sense1}$ | também virou modelo de uma **marca** famosa de roupas .<br>também\|PDEN virou\|V modelo\|N de\|PREP uma\|ART **marca**\|**N** famosa\|ADJ de\|PREP roupas\|N .\|PU |
| marca$_{sense2}$ | o relógio **marca** 00h , e o filme já vai começar .<br>o\|ART relógio\|N **marca**\|**V** 00\|NUM h\|N ,\|PU e\|KC o\|ART filme\|N já\|ADV vai\|V começar\|V .\|PU |

**Fonte: Autoria própria**

Ambas as abordagens para gerar vetores de sentido foram treinadas com este córpus. A única diferença é que a entrada para o MSSG é a sentença sem qualquer anotação de PoS, enquanto a entrada para o sense2vec é a sentença anotada.

## 5.2 Parametrização

Para todos os treinamentos, incluindo os *baselines*, os vetores foram gerados com 300 dimensões, usando o modelo Skip-Gram, com janela de contexto de cinco palavras à esquerda e cinco palavras à direita da palavra alvo. A taxa de aprendizado foi definida para 0,025 e a frequência mínima para cada palavra foi definida para 10. O MSSG foi escolhido por gerar uma quantidade fixa de sentidos por palavra (o número máximo foi definido como 3) e tornar justa a comparação com o sense2vec.

## 5.3 Avaliação

Os vetores de sentido gerados foram avaliados de forma intrínseca e extrínseca. Para a avaliação intrínseca, foi utilizado o *dataset* de analogias sintáticas e semânticas de Rodrigues et al. (2016). Para a avaliação extrínseca, os vetores de sentido foram utilizados em tarefas de Similaridade Semântica e Desambiguação Lexical de Sentidos (DLS).

### 5.3.1 Avaliação Intrínseca

Baseado em Hartmann et al. (2017), este experimento é uma tarefa de analogias sintáticas e semânticas onde o uso de vetores de sentido é avaliado. Vetores de palavra foram escolhidos como *baselines*.

**Dataset.** O *dataset* das Analogias Sintáticas e Semânticas de Rodrigues et al. (2016) tem analogias no português brasileiro (PT-BR) e europeu (PT-EU). Nas analogias sintáticas, temos as seguintes categorias: *adjective-to-adverb, opposite, comparative, superlative, present-participle, nationality-adjective, past-tense, plural, and plural-verbs*. Nas analogias semânticas, temos as seguintes categorias: *capital-common-countries, capital-world, currency, city-in-state and family*. Em cada categoria, temos exemplos de analogias com quatro palavras:

1. ***adjective-to-adverb:***
    - fantástico fantasticamente aparente aparentemente **(syntactic)**
2. ***capital-common-countries:***
    - Berlim Alemanha Lisboa Portugal **(semantic)**

**Algoritmo.** O algoritmo recebe as três primeiras palavras da analogia e deve prever a quarta. Considerando o exemplo anterior, o algoritmo receberia Berlim (a), Alemanha (b) e Lisboa (c) e deveria prever Portugal (d). Internamente, a seguinte operação algébrica é executada entre vetores:

$$v(b) + v(c) - v(a) = v(d) \tag{5.1}$$

**Medidas de avaliação.** A medida usada neste caso é a acurácia, que calcula a porcentagem de palavras corretamente rotuladas em relação à quantidade total de palavras no *dataset*.

**Discussão dos resultados.** A Tabela 5.3 mostra os resultados gerais da acurácia obtida nas analogias. A Tabela 5.4 mostra os resultados em cada categoria das analogias sintáticas. A Tabela 5.5 mostra os resultados em cada categoria das analogias semânticas. O Word2vec, o GloVe e o FastText foram adotados como *baselines*, uma vez que tiveram bom desempenho nos experimentos de Hartmann et al. (2017). Note que os vetores gerados pelo nosso modelo sense2vec superam os *baselines* nos níveis sintático e semântico.

Nas analogias sintáticas, os vetores de sentido gerados pelo sense2vec superam os vetores de palavra gerados pelo word2vec em *opposite, nationality-adjective, past-tense, plural and plural-verbs*, como mostrado na tabela 5.4. Um exemplo da categoria *adjective-to-adverb* é mostrado na tabela 5.6. Podemos explicar este tipo de sucesso através de uma operação algébrica de vetores. Ao calcular *v(aparentemente)* + *v(completo)* - *v(aparente)* o vetor resultante do word2vec é *v(incompleto)* quando deveria ser *v(completamente)*. A opção correta aparece como o segundo vizinho mais próximo no word2vec.

**Tabela 5.3: Valores de acurácia obtidos na avaliação intrínseca em analogias sintáticas e semânticas**

| ***Embedding*** | **PT-BR** | | | **PT-EU** | | |
|---|---|---|---|---|---|---|
| | **Sintático** | **Semântico** | **Todos** | **Sintático** | **Semântico** | **Todos** |
| Word2Vec (word) | 49,4 | 42,5 | 45,9 | 49,5 | 38,9 | 44,3 |
| GloVe (word) | 34,7 | 36,7 | 35,7 | 34,9 | 34,0 | 34,4 |
| FastText (word) | 39,9 | 8,0 | 24,0 | 39,9 | 7,6 | 23,9 |
| MSSG (sense) | 23,0 | 6,6 | 14,9 | 23,0 | 6,3 | 14,7 |
| Sense2Vec (sense) | **52,4** | **42,6** | **47,6** | **52,6** | **39,5** | **46,2** |

**Fonte: Autoria própria**

**Tabela 5.4: Valores de acurácia obtidos em cada categoria nas analogias sintáticas**

| ***Categoria*** | **PT-BR** | | | | | **PT-EU** | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **Word2Vec** | **GloVe** | **FastText** | **MSSG** | **Sense2Vec** | **Word2Vec** | **GloVe** | **FastText** | **MSSG** | **Sense2Vec** |
| *adjective-to-adverb* | 14,4 | 5,1 | **23,4** | 5,9 | 11,0 | 13,1 | 5,1 | **24,6** | 6,2 | 11,0 |
| *opposite* | 22,0 | 17,2 | 17,2 | 4,5 | **23,1** | 22,0 | 17,2 | 17,2 | 4,5 | **23,1** |
| *comparative* | 66,7 | 60,0 | 56,7 | **73,3** | 60,0 | 66,7 | 60,0 | 56,7 | **73,3** | 60,0 |
| *superlative* | 17,7 | 1,7 | **34,8** | 1,3 | 14,1 | 17,7 | 1,7 | **34,8** | 1,3 | 14,1 |
| *present-participle* | **82,8** | 59,8 | 81,9 | 67,1 | 80,6 | **83,1** | 58,5 | 82,4 | 66,8 | 80,9 |
| *nationality-adjective* | 69,2 | 49,0 | 10,6 | 2,6 | **73,7** | 69,3 | 49,3 | 10,7 | 2,6 | **73,2** |
| *past-tense* | 53,3 | 41,2 | 58,4 | 38,7 | **62,9** | 53,3 | 41,2 | 58,4 | 38,7 | **62,9** |
| *plural* | 44,0 | 33,3 | 38,9 | 15,0 | **49,4** | 44,0 | 33,4 | 36,3 | 14,5 | **49,1** |
| *plural-verbs* | 47,8 | 30,5 | 50,2 | 35,5 | **51,6** | 47,3 | 30,4 | 50,7 | 34,9 | **52,3** |
| *total* | 49,4 | 34,7 | 39,9 | 14,9 | **52,4** | 49,5 | 34,9 | 39,9 | 23,0 | **52,6** |

**Fonte: Autoria própria**

**Tabela 5.5: Valores de acurácia obtidos em cada categoria nas analogias semânticas**

| ***Categoria*** | **PT-BR** | | | | | **PT-EU** | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **Word2Vec** | **GloVe** | **FastText** | **MSSG** | **Sense2Vec** | **Word2Vec** | **GloVe** | **FastText** | **MSSG** | **Sense2Vec** |
| *capital-common-countries* | **75,5** | 70,9 | 9,5 | 6,9 | 74,9 | **78,4** | 68,8 | 11,3 | 8,2 | 75,8 |
| *capital-world* | 53,9 | 36,8 | 9,1 | 6,6 | **54,0** | 50,0 | 34,7 | 8,4 | 6,2 | **51,0** |
| *currency* | 6,0 | 1,6 | 0,3 | 1,0 | **6,6** | 5,7 | 1,4 | 0,4 | 1,0 | **6,9** |
| *city-in-state* | 22,7 | **35,6** | 4,5 | 5,1 | 24,4 | 17,7 | **31,2** | 3,6 | 3,9 | 19,0 |
| *family* | **64,7** | 56,3 | 27,3 | 23,8 | 60,5 | **62,8** | 56,3 | 29,2 | 24,7 | 61,4 |
| *total* | 42,5 | 36,7 | 8,0 | 6,6 | **42,6** | 38,9 | 34,0 | 7,6 | 6,3 | **39,5** |

**Fonte: Autoria própria**

Ainda analisando a categoria sintática *adjective-to-adverb*, que também contém analogias como: *v(calmamente) + v(alegre) - v(calmo) = v(alegremente)* e *v(incerto) + v(conveniente) - v(certo) = v(inconveniente)*, constatamos que o FastText tem a melhor acurácia (23,4% no PT-BR e 24,6% no PT-EU) devido ao seu treinamento ter como principal característica considerar a morfologia das palavras, gerando vetores com pesos compartilhados entre palavras

**Tabela 5.6: Exemplos de analogias sintáticas preditas pelo word2vec e sense2vec**

| | |
|---|---|
| word2vec | aparente aparentemente completo : completamente **(esperado)**<br>aparente aparentemente completo : incompleto **(predito)** |
| sense2vec | aparente\|ADJ aparentemente\|ADV completo\|ADJ : completamente\|ADV **(esperado)**<br>aparente\|ADJ aparentemente\|ADV completo\|ADJ : completamente\|ADV **(predito)** |

**Fonte: Autoria própria**

de mesma morfologia (ex: "quebrar" e "quebrado"). O mesmo efeito ocorre com a categoria sintática *superlative* que contém analogias como: *v(brilhantíssimo) + v(fácil) - v(brilhante) = v(facílimo)*.

Outra análise interessante é com relação a categoria sintática *past-tense* onde o sense2vec tem a melhor acurácia (62,9% no PT-BR e PT-EU) e o FastText fica em segundo lugar (58,4% no PT-BR e PT-EU). Essa categoria contém analogias como: *v(dançou) + v(descrevendo) - v(dançando) = v(descreveu)* e *v(foi) + v(dizendo) - v(indo) = v(disse)*. Descobrimos que o FastText acerta mais casos onde há o compartilhamento de morfologia (primeiro exemplo) e erra mais casos onde não há (segundo exemplo), que é onde há uma certa adição de semântica na analogia sintática (uso de verbos irregulares). Já o sense2vec tem um desempenho equilibrado em ambos os casos, o que quer dizer que ele considera a morfologia e também entende a semântica. O mesmo efeito ocorre com a categoria sintática *plural-verbs*, onde também há o uso de verbos irregulares (*v(vão) + v(ver) - v(ir) = v(vêem)*).

Nas analogias semânticas, os vetores de sentido gerados pelo sense2vec superam os vetores de palavra gerados pelo word2vec em *capital-world, currency e city-in-state*. Exemplos de *city-in-state* são mostrados na tabela 5.7. Neste caso, as etiquetas de PoS são sempre a mesma para todas as palavras: N (*noun* - substantitvo). Isso indica que o sucesso do sense2vec está relacionado à qualidade dos vetores de sentido como um todo. Como todas as palavras são etiquetadas, esse recurso acaba melhorando a inferência de todos os espaços vetoriais durante o treinamento.

Ainda analisando a categoria *city-in-state*, observamos que o GloVe tem a melhor acurácia (35,6% no PT-BR e 31,2% no PT-EU). Essa categoria contém analogias como: *v(Colorado) + v(Orlando) - v(Denver) = v(Flórida)*. Acreditamos que o bom desempenho ocorre devido a característica principal do GloVe que é utilizar informações de contexto global, ou seja, olhar para a matriz de co-ocorrência das palavras no córpus de treinamento e acrescentar essa informação ao vetor de palavra.

Concluímos, então, que as etiquetas de PoS do sense2vec funcionam como um recurso extra

**Tabela 5.7: Exemplos de analogias semânticas preditas pelo word2vec e sense2vec**

| | |
|---|---|
| word2vec | arlington texas akron : kansas **(predito)** ohio **(esperado)** |
| sense2vec | arlington\|N texas\|N akron\|N : ohio\|N **(predito)(esperado)** |
| word2vec | bakersfield califórnia madison : pensilvânia **(predito)** wisconsin **(esperado)** |
| sense2vec | bakersfield\|N califórnia\|N madison\|N : wisconsin\|N **(predito)(esperado)** |
| word2vec | worcester massachusetts miami : seattle **(predito)** flórida **(esperado)** |
| sense2vec | worcester\|N massachusetts\|N miami\|N : flórida\|N **(predito)(esperado)** |

**Fonte: Autoria própria**

no treinamento dos vetores de sentido, gerando vetores numéricos mais precisos que permitem a obtenção dos melhores resultados. Também verificamos que o sense2vec lida bem com analogias de cunho morfológico e de cunho semântico, absorvendo um pouco das características principais do FastText e GloVe.

O modelo MSSG teve um desempenho abaixo do esperado nesta tarefa, provavelmente por não possuir espaços vetoriais tão bem definidos quanto o sense2vec (acreditamos que esteja relacionado ao algoritmo que gera os vetores).

### 5.3.2 Avaliação Extrínseca

*Embeddings* são recursos que podem ser integrados em tarefas de PLN. Quanto melhor a qualidade dos vetores, melhor é o desempenho dessas tarefas. Logo, os vetores de sentido gerados foram avaliados em tarefas de similaridade semântica e DLS.

#### 5.3.2.1 Similaridade Semântica

Baseado em Hartmann et al. (2017), este experimento é uma tarefa de similaridade semântica entre sentenças onde o uso de vetores de sentido como *feature* é avaliado. Vetores de palavra foram escolhidos como *baselines*.

**Dataset.** O ASSIN (Avaliação de Similaridade Semântica e Inferência Textual) foi um *workshop* realizado no PROPOR-2016 que englobou duas tarefas de avaliação: (i) *semantic similarity* e ii) *entailment*. Escolhemos a primeira para avaliar os vetores de sentido gerados em uma tarefa extrínseca e de similaridade semântica. A tarefa requer que participantes atribuam um *score* de similaridade de 1 a 5 a pares de sentenças. O *workshop* disponibilizou *dataset* de treinamento e testes para o português brasileiro (PT-BR) e europeu (PT-EU).

**Algoritmo.** O objetivo desta tarefa é predizer, através de uma regressão linear, o *score* de similaridade entre duas sentenças. O modelo é treinado no *dataset* de treinamento, que contém pares de sentenças com o *score* de ouro (*gold score*). A predição ocorre no *dataset* de testes, que contém pares de sentenças sem o *gold score*. Como temos este mesmo *dataset* de testes com o *gold score*, é possível calcular a Correlação de Pearson ($\rho$) e o *Mean Squared Error* (MSE) entre eles. Os resultados mostram o quanto a predição automática se aproxima da predição humana.

**Medidas de avaliação.** O coeficiente de correlação de Pearson mede a relação linear entre dois *datasets*: um anotado pelos participantes e outro produzido pelo sistema. Como outros coeficientes de correlação, ele varia entre -1 e +1, com 0 significando nenhuma correlação. Correlações de -1 ou +1 significam uma relação linear exata. O *Mean Squared Error* (MSE) calcula a média dos quadrados dos erros, ou seja, a diferença entre o valor estimado e o esperado.

**Discussão dos resultados.** A Tabela 5.8 mostra o desempenho de nossos vetores de sentido para os *datasets* de testes do PT-BR e PT-EU, através da Correlação de Pearson ($\rho$) e do MSE. O resultado obtido pelo sense2vec supera os *baselines* no PT-BR, obtendo maior coeficiente de correlação e menor erro (resultados em negrito). Isso se deve à qualidade superior dos vetores gerados pelo sense2vec, que levam em conta as etiquetas de PoS das palavras. Como o córpus foi anotado com PoS usando a ferramenta nlnet (que foi treinada para o PT-BR), o mesmo sucesso não foi alcançado para o PT-EU.

**Tabela 5.8: Valores para o coeficiente de Pearson ($\rho$) e MSE obtidos na avaliação extrínseca na tarefa de similaridade semântica. A setas indicam se menor (↓) ou se maior (↑) é melhor.**

| ***Embedding*** | **Tipo** | **Dim.** | **PT-BR** | | **PT-EU** | |
|---|---|---|---|---|---|---|
| | | | $\rho$ (↑) | **MSE** (↓) | $\rho$ (↑) | **MSE** (↓) |
| Word2Vec | word | 300 | 0,55 | 0,53 | 0,55 | 0,84 |
| GloVe | word | 300 | 0,46 | 0,60 | 0,47 | 0,93 |
| FastText | word | 300 | 0,53 | 0,55 | 0,51 | 0,89 |
| MSSG | sense | 300 | 0,39 | 0,65 | 0,35 | 1,04 |
| Sense2Vec | sense | 300 | **0,57** | **0,51** | 0,53 | 0,87 |

**Fonte: Autoria própria**

Na tabela 5.9 temos dois pares de sentenças (PT-BR) com o *gold score*, a estimativa feita quando vetores de palavra são usados como *feature* (word2vec) e a estimativa quando vetores de sentido são usados (sense2vec). Em ambos os exemplos, a estimativa mais próxima do *gold*

*score* é aquela usando vetores de sentido e, no segundo par, temos a palavra ambígua "banco" ocorrendo no sentido do banco de reservas de futebol.

**Tabela 5.9: Exemplos de pares de sentenças com *scores* de similaridade**

| | |
|---|---|
| gold = **3.25**<br>word2vec = 3.1344962<br>sense2vec = **3.2150722** | -No Brasil, 809 instituições participarão da Primavera dos Museus neste ano.<br>-Começa nesta segunda-feira, em todo o Brasil, a Primavera dos Museus. |
| gold = **2.0**<br>word2vec = 2.8379555<br>sense2vec = **2.0541897** | -No primeiro tempo, os donos da casa abafaram o Santos.<br>-Ele foi expulso do **banco** ainda no fim do primeiro tempo. |

**Fonte: Autoria própria**

Novamente, o modelo MSSG teve um desempenho abaixo do esperado nesta tarefa, provavelmente por não possuir espaços vetoriais tão bem definidos quanto o sense2vec (acreditamos que esteja relacionado ao algoritmo que gera os vetores).

#### 5.3.2.2 Desambiguação Lexical de Sentidos (DLS)

Baseado em Pelevina et al. (2016) e Nóbrega (2013), este experimento é uma tarefa de *lexical sample* (desambiguação de uma amostra de palavra) onde a precisão de um método de desambiguação usando vetores de sentido (MSSG) é avaliada. O método de sentido mais frequente (*Most Frequent Sense*) (MFS) foi escolhido como *baseline*.

**Dataset.** O *dataset* CSTNews (CARDOSO et al., 2011) contém 50 coleções de documentos jornalísticos (PT-BR) com 72.148 palavras dispostas em 140 documentos agrupados de acordo com o tópico abordado. De acordo com Nóbrega (2013), o *dataset* CSTNews contém 466 substantivos anotados com significados, a partir dos synsets da wordnet. Desse total, 361 têm dois ou mais significados e, destes, 196 têm possibilidades de desambiguação acima da média encontrada no córpus, que é aproximadamente 6 sentidos diferentes. Assim, 77% das palavras são ambíguas (com mais de dois sentidos) e aproximadamente 42% são altamente ambíguas, com resultados acima da média.

**Algoritmo.** Para desambiguar uma palavra em um contexto, seguimos o algoritmo proposto por Pelevina et al. (2016), baseado na similaridade entre sentido e contexto. Dada uma palavra $w$ e suas palavras de contexto $C = c_1, ..., c_k$ primeiro mapeamos $w$ para um conjunto de vetores de sentido: $S = s_1, ..., s_n$. Em vez de mapear as palavras de contexto $C$ para vetores de sentido, mapeamos para seus respectivos vetores de palavra (word2vec), evitando um problema de

dimensionalidade. Para melhorar o desempenho da DLS, também aplicamos filtragem de contexto. Normalmente, apenas algumas palavras do contexto são relevantes para desambiguação, como "cadeiras" e "cozinha" são para "mesa" em "Eles compraram mesa e cadeiras para a cozinha". Para cada palavra $c_j$ no contexto $C = (c_1, ..., c_k)$ calcula-se um *score* que quantifica quão bem ele discrimina os sentidos:

$$max_i f(s_i, c_j) - min_i f(s_i, c_j) \tag{5.2}$$

onde $s_i$ itera os sentidos da palavra ambígua e $f$ é nossa estratégia de desambiguação: $sim(s_i, c_j)$. As $p$ palavras mais discriminativas do contexto são usadas para desambiguação.

**Medidas de avaliação.** Como em Pelevina et al. (2016), para calcular o desempenho do método de desambiguação, foi necessário realizar um mapeamento entre o inventário de sentidos do CSTNews e os vetores de sentido gerados pelo método MSSG. Por causa desse mapeamento, tornou-se possível saber quando o método classifica corretamente o significado da palavra ou não. Apenas as palavras que obtiveram um mapeamento válido foram escolhidas para a tarefa de *lexical sample*, que é quando seus sentidos no CSTNews possuem um vetor de sentido correspondente. Há casos onde o sentido presente no CSTNews não aparece como vetor de sentido e vice-versa. Como *baseline*, o método de sentido mais frequente (MFS) foi escolhido por ser amplamente utilizado na literatura.

**Discussão dos resultados.** A Tabela 5.10 mostra a precisão ponderada do método de desambiguação em um conjunto de palavras, calculada usando vetores de sentido (MSSG) e o MFS (*baseline*). Os resultados mostram que, em algumas palavras ambíguas, o método usando vetores de sentido supera o MFS (casos em negrito). A palavra "centro" assume dois significados em quatorze sentenças. Em duas frases com o sentido de instituição (centro de previsões, centro da NASA) e em doze com o sentido de área central (centro da cidade). Este é um exemplo em que a maior classe é o sentido mais frequente da palavra no córpus, uma característica que beneficia o MFS. Por este motivo, usamos a precisão ponderada (*precision weighted*), que não sofre com o desbalanceamento das classes. No final da tabela 5.10, temos uma média das precisões e descobrimos que os vetores de sentido superam o *baseline* nessa tarefa de *lexical sample*.

O sense2vec não foi utilizado nesta tarefa pois identificaria apenas ambiguidades com classes gramaticais diferentes. Neste caso, os sentidos de "centro" não seriam identificados, pois ocorrem apenas como substantivos (instituição e área central).

**Tabela 5.10: Valores de precisão ponderada obtidos na avaliação extrínseca para a tarefa *Lexical sample* para o *baseline* MFS (*Most Frequent Sense*) e nosso modelo (MSSG).**

| **Palavra ambígua** | **Frequência** | **# Sentidos** | **MFS** | **MSSG** |
|---|---|---|---|---|
| | | | **Precisão** | **Precisão** |
| obra | 13 | 2 | 71,59 | 65,81 |
| centro | 14 | 2 | 73,43 | **88,88** |
| estado | 10 | 2 | 36,00 | **80,00** |
| discurso | 9 | 2 | 30,86 | **63,88** |
| escola | 8 | 2 | 25,00 | **75,00** |
| presidente | 65 | 3 | 41,75 | 35,65 |
| pontos | 10 | 2 | 49,00 | **80,00** |
| **Média** | | | **46,80** | **69,88** |

**Fonte: Autoria própria**

# Capítulo 6

## CONSIDERAÇÕES FINAIS

Neste trabalho, exploramos diferentes estratégias para a geração de vetores de sentido com o objetivo de resolver ambiguidades lexicais em tarefas de PLN. Não só propomos a geração desses vetores para o português (BR e EU) como também avaliamos em tarefas intrínsecas e extrínsecas de PLN, concluindo que são melhores *features* do que os vetores de palavras tradicionais. Para a tarefa intrínseca de analogias sintáticas e semânticas, os vetores de sentido (sense2vec) superaram os *baselines* de vetores de palavras (word2vec, Glove, FastText). Para a tarefa de similaridade semântica de sentenças, os vetores de sentido (sense2vec) também superaram os *baselines* em PT-BR. Para a tarefa de amostra lexical de DLS, em que o objetivo foi induzir o significado correto da palavra ambígua dentro de um contexto, os vetores de sentido também obtiveram uma melhor precisão quando comparados com o *baseline* adotado: Sentido Mais Frequente (MFS).

Contribuímos com a geração de experimentos e recursos para o idioma específico do português, para o qual ainda não se tem notícias de investigações com vetores de sentido. Disponibilizamos o código fonte das avaliações e pré-processamentos junto com todos os vetores treinados no git ⟨https://github.com/LALIC-UFSCar/sense-embeddings⟩.

Visto que, a nível mundial, pesquisas sobre representações de sentido estão no início, contribuímos com um estudo relevante para esse cenário.

Como trabalho futuro, pretendemos experimentar uma combinação das duas abordagens (MSSG e sense2vec) e também explorar como as novas abordagens propostas para gerar modelos de linguagem são executadas em português. Neste caso, queremos comparar os resultados obtidos por vetores de sentido e modelos de linguagem em tarefas de PLN, levando em consideração não apenas sua precisão, mas também a facilidade de treinamento/uso e custo-benefício. Outro trabalho futuro é investigar se vetores de sentido têm a cobertura necessária

para serem usados como um recurso linguístico como a WordNet.

Um dos resultados esperados deste trabalho é fomentar a pesquisa relacionada a representações vetoriais de palavras e de sentidos no Brasil e servir como base para futuros trabalhos em inteligência artificial, PLN, modelagem distribucional e DLS.

# REFERÊNCIAS


AGIRRE, E.; LACALLE, O. L. de; MARTINEZ, D. Exploring feature spaces with svd and unlabeled data for word sense disambiguation. In: *Proceedings of the Conference on Recent Advances on Natural Language Processing (RANLP'05)*. [S.l.: s.n.], 2005.

AGIRRE, E.; MARTINEZ, D. Learning class-to-class selectional preferences. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 2001 workshop on Computational Natural Language Learning-Volume 7*. [S.l.], 2001. p. 3.

ALEIXO, P.; PARDO, T. A. S. *CSTNews: um córpus de textos jornalísticos anotados segundo a teoria discursiva multidocumento CST (cross-document structure theory*. [S.l.]: ICMC-USP, 2008.

ALUÍSIO, R. M.; PINHEIRO, G. M.; FINGER, M.; GRAÇAS, M.; NUNES, V.; TAGNIN, S. E. The lacioweb project: Overview and issues in brazilian portuguese corpora creation. In: CITESEER. *In: Proceedings of Corpus Linguistics*. [S.l.], 2003.

ANDREAS, J.; KLEIN, D. How much do word embeddings encode about syntax? In: *ACL (2)*. [S.l.: s.n.], 2014. p. 822–827.

BANSAL, M.; GIMPEL, K.; LIVESCU, K. Tailoring continuous word representations for dependency parsing. In: *ACL (2)*. [S.l.: s.n.], 2014. p. 809–815.

BENGIO, Y.; DUCHARME, R.; VINCENT, P.; JAUVIN, C. A neural probabilistic language model. *Journal of machine learning research*, v. 3, n. Feb, p. 1137–1155, 2003.

BERGER, A. L.; PIETRA, V. J. D.; PIETRA, S. A. D. A maximum entropy approach to natural language processing. *Computational linguistics*, MIT Press, v. 22, n. 1, p. 39–71, 1996.

BLEI, D. M.; NG, A. Y.; JORDAN, M. I. Latent dirichlet allocation. *Journal of machine Learning research*, v. 3, n. Jan, p. 993–1022, 2003.

BOJANOWSKI, P.; GRAVE, E.; JOULIN, A.; MIKOLOV, T. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

BOLBOACA, S.-D.; JÄNTSCHI, L. Pearson versus spearman, kendall's tau correlation analysis on structure-activity relationships of biologic active compounds. *Leonardo Journal of Sciences*, v. 5, n. 9, p. 179–200, 2006.

BROCKMANN, C.; LAPATA, M. Evaluating and combining approaches to selectional preference acquisition. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*. [S.l.], 2003. p. 27–34.

BROWN, P. F.; PIETRA, V. J. D.; PIETRA, S. A. D.; MERCER, R. L. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, MIT Press, v. 19, n. 2, p. 263–311, 1993.

BRUCKSCHEN, M.; MUNIZ, F.; SOUZA, J.; FUCHS, J.; INFANTE, K.; MUNIZ, M.; GONÇALVES, P.; VIEIRA, R.; ALUÍSIO, S. *Anotação Lingüística em XML do Corpus PLN-BR. NILC–TR–09–08*. [S.l.], 2008.

BRUNI, E.; TRAN, N.-K.; BARONI, M. Multimodal distributional semantics. *J. Artif. Intell. Res.(JAIR)*, v. 49, n. 2014, p. 1–47, 2014.

CABEZUDO, M. A. S. *Investigação de métodos de desambiguação lexical de sentidos de verbos do português do Brasil*. Tese (Doutorado) — Universidade de São Paulo, 2015.

CAMACHO-COLLADOS, J.; PILEHVAR, M. T. From word to sense embeddings: A survey on vector representations of meaning. *CoRR*, abs/1805.04032, 2018. Disponível em: ⟨http://arxiv.org/abs/1805.04032⟩.

CANÇADO, M. Manual de semântica: noções básicas e exercícios. 2008.

CARDOSO, P. C. F.; MAZIERO, E. G.; LÚCIA, M.; JORGE, R. C.; FELIPPO, A. D.; RINO, L. H. M.; GRAçAS, M. das; NUNES, V.; PARDO, T. A. S.; LUIS, R. W. Cstnews - a discourse-annotated corpus for single and multi-document summarization of news texts in brazilian portuguese. In: . [S.l.: s.n.], 2011.

CHAN, Y. S.; NG, H. T. Scaling up word sense disambiguation via parallel texts. In: *AAAI*. [S.l.: s.n.], 2005. v. 5, p. 1037–1042.

CHE, W.; WANG, M.; MANNING, C. D.; LIU, T. Named entity recognition with bilingual constraints. In: *HLT-NAACL*. [S.l.: s.n.], 2013. p. 52–62.

CHEN, X.; LIU, Z.; SUN, M. A unified model for word sense representation and disambiguation. In: *EMNLP*. [S.l.: s.n.], 2014. p. 1025–1035.

COLLOBERT, R.; WESTON, J. A unified architecture for natural language processing: Deep neural networks with multitask learning. In: ACM. *Proceedings of the 25th international conference on Machine learning*. [S.l.], 2008. p. 160–167.

COLLOBERT, R.; WESTON, J.; BOTTOU, L.; KARLEN, M.; KAVUKCUOGLU, K.; KUKSA, P. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, v. 12, n. Aug, p. 2493–2537, 2011.

COTTRELL, G. W. A connectionist approach to word sense disambiguation. 1985.

DAGAN, I.; ITAI, A. Word sense disambiguation using a second language monolingual corpus. *Computational linguistics*, MIT Press, v. 20, n. 4, p. 563–596, 1994.

DEERWESTER, S.; DUMAIS, S. T.; FURNAS, G. W.; LANDAUER, T. K.; HARSHMAN, R. Indexing by latent semantic analysis. *Journal of the American society for information science*, American Documentation Institute, v. 41, n. 6, p. 391, 1990.

DEMPSTER, A. P.; LAIRD, N. M.; RUBIN, D. B. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, JSTOR, p. 1–38, 1977.

DEVLIN, J.; CHANG, M.-W.; LEE, K.; TOUTANOVA, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

DHILLON, I. S.; MODHA, D. S. Concept decompositions for large sparse text data using clustering. *Machine learning*, Springer, v. 42, n. 1, p. 143–175, 2001.

DHILLON, P.; FOSTER, D. P.; UNGAR, L. H. Multi-view learning of word embeddings via cca. In: *Advances in Neural Information Processing Systems*. [S.l.: s.n.], 2011. p. 199–207.

DIAB, M.; RESNIK, P. An unsupervised method for word sense tagging using parallel corpora. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. [S.l.], 2002. p. 255–262.

EDMONDS, P. Senseval: The evaluation of word sense disambiguation systems. *ELRA newsletter*, v. 7, n. 3, p. 5–14, 2002.

ELMAN, J. L. Finding structure in time. *Cognitive science*, Wiley Online Library, v. 14, n. 2, p. 179–211, 1990.

ENEKO, A.; EDMONDS, P. Word sense disambiguation: algorithms and applications. *Springer Publishing Company, Incorporated, ISBN*, v. 1402068700, p. 9781402068706, 2007.

ESCUDERO, G.; MÀRQUEZ, L.; RIGAU, G. Naive bayes and exemplar-based approaches to word sense disambiguation revisited. In: IOS PRESS. *Proceedings of the 14th european conference on artificial intelligence*. [S.l.], 2000. p. 421–425.

ESCUDERO, G.; MÀRQUEZ, L.; RIGAU, G.; SALGADO, J. G. On the portability and tuning of supervised word sense disambiguation systems. Citeseer, 2000.

FARUQUI, M.; DYER, C. Improving vector space word representations using multilingual correlation. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. [S.l.], 2014.

FINKELSTEIN, L.; GABRILOVICH, E.; MATIAS, Y.; RIVLIN, E.; SOLAN, Z.; WOLFMAN, G.; RUPPIN, E. Placing search in context: The concept revisited. In: ACM. *Proceedings of the 10th international conference on World Wide Web*. [S.l.], 2001. p. 406–414.

FIRTH, J. R. A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis*, Basil Blackwell, 1957.

FONSECA, E.; ROSA, J. A two-step convolutional neural network approach for semantic role labeling. In: . [S.l.: s.n.], 2013. p. 1–7. ISBN 978-1-4673-6128-6.

FREY, B. J.; DUECK, D. Clustering by passing messages between data points. *science*, American Association for the Advancement of Science, v. 315, n. 5814, p. 972–976, 2007.

GALE, W. A.; CHURCH, K. W.; YAROWSKY, D. A method for disambiguating word senses in a large corpus. *Computers and the Humanities*, Springer, v. 26, n. 5, p. 415–439, 1992.

GALE, W. A.; CHURCH, K. W.; YAROWSKY, D. Using bilingual materials to develop word sense disambiguation methods. In: *Proceedings of the International Conference on Theoretical and Methodological Issues in Machine Translation*. [S.l.: s.n.], 1992. p. 101–112.

GOLDBERG, Y.; LEVY, O. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*, 2014.

GOLUB, G. H.; LOAN, C. F. V. Matrix computations. johns hopkins series in the mathematical sciences. *Johns Hopkins University Press, Baltimore, MD*, 1989.

GRIFFITHS, T. L.; JORDAN, M. I.; TENENBAUM, J. B.; BLEI, D. M. Hierarchical topic models and the nested chinese restaurant process. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2004. p. 17–24.

GUO, J.; CHE, W.; WANG, H.; LIU, T. Learning sense-specific word embeddings by exploiting bilingual resources. In: *COLING*. [S.l.: s.n.], 2014. p. 497–507.

HARRIS, Z. Distributional structure. *Word*, v. 23, n. 10, p. 146–162, 1954.

HARTMANN, N.; FONSECA, E.; SHULBY, C.; TREVISO, M.; RODRIGUES, J.; ALUISIO, S. Portuguese word embeddings: Evaluating on word analogies and natural language tasks. *arXiv preprint arXiv:1708.06025*, 2017.

HINTON, G.; RUMELHART, D.; WILLIAMS, R. Learning internal representations by back-propagating errors. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, v. 1, 1985.

HINTON, G. E. Distributed representations. 1984.

HUANG, E. H.; SOCHER, R.; MANNING, C. D.; NG, A. Y. Improving word representations via global context and multiple word prototypes. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*. [S.l.], 2012. p. 873–882.

IACOBACCI, I.; PILEHVAR, M. T.; NAVIGLI, R. Sensembed: Learning sense embeddings for word and relational similarity. In: *ACL (1)*. [S.l.: s.n.], 2015. p. 95–105.

IDE, N. Cross-lingual sense determination: Can it work? *Computers and the Humanities*, Springer, v. 34, n. 1, p. 223–234, 2000.

III, H. D. Notes on cg and lm-bfgs optimization of logistic regression. *Paper available at http://pub. hal3. name# daume04cg-bfgs, implementation available at http://hal3. name/megam*, v. 198, p. 282, 2004.

JURAFSKY, D.; MARTIN, J. H. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. [S.l.]: MIT Press, 2008.

KARTSAKLIS, D.; SADRZADEH, M.; PULMAN, S. Separating disambiguation from composition in distributional semantics. In: SOFIA ,BULGARIA. *Proceedings of 17th Conference on Natural Language Learning (CoNLL)*. [S.l.], 2013.

KILGARRIFF, A. Polysemy. Citeseer, 1992.

KINTSCH, W. Predication. *Cognitive science*, Elsevier, v. 25, n. 2, p. 173–202, 2001.

KROVETZ, R. More than one sense per discourse. *NEC Princeton NJ Labs., Research Memorandum*, 1998.

LACALLE, O. Lopez de; AGIRRE, E. A methodology for word sense disambiguation at 90% based on large-scale CrowdSourcing. In: *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*. Denver, Colorado: Association for Computational Linguistics, 2015. p. 61–70. Disponível em: ⟨https://www.aclweb.org/anthology/S15-1007⟩.

LEACOCK, C.; MILLER, G. A.; CHODOROW, M. Using corpus statistics and wordnet relations for sense identification. *Computational Linguistics*, MIT Press, v. 24, n. 1, p. 147–165, 1998.

LESK, M. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In: ACM. *Proceedings of the 5th annual international conference on Systems documentation*. [S.l.], 1986. p. 24–26.

LI, J.; JURAFSKY, D. Do multi-sense embeddings improve natural language understanding? *arXiv preprint arXiv:1506.01070*, 2015.

LI, W.; MCCALLUM, A. Semi-supervised sequence modeling with syntactic topic models. In: *AAAI*. [S.l.: s.n.], 2005.

LIANG, P.; TASKAR, B.; KLEIN, D. Alignment by agreement. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*. [S.l.], 2006. p. 104–111.

LIN, D. Automatic retrieval and clustering of similar words. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 17th international conference on Computational linguistics-Volume 2*. [S.l.], 1998. p. 768–774.

LING, W.; DYER, C.; BLACK, A. W.; TRANCOSO, I. Two/too simple adaptations of word2vec for syntax problems. In: *HLT-NAACL*. [S.l.: s.n.], 2015. p. 1299–1304.

LIU, W.; MEI, T.; ZHANG, Y.; CHE, C.; LUO, J. Multi-task deep visual-semantic embedding for video thumbnail selection. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2015. p. 3707–3715.

LIU, Y.; LIU, Z.; CHUA, T.-S.; SUN, M. Topical word embeddings. In: *AAAI*. [S.l.: s.n.], 2015. p. 2418–2424.

MAMEDE, N.; BAPTISTA, J.; DINIZ, C.; CABARRÃO, V. String: An hybrid statistical and rule-based natural language processing chain for portuguese. Citeseer, 2012.

MANNING, C. D.; SCHÜTZE, H. et al. *Foundations of statistical natural language processing*. [S.l.]: MIT Press, 1999. v. 999.

MARTINEZ, D.; AGIRRE, E. One sense per collocation and genre/topic variations. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics-Volume 13*. [S.l.], 2000. p. 207–215.

MCCANN, B.; BRADBURY, J.; XIONG, C.; SOCHER, R. Learned in translation: Contextualized word vectors. In: *Advances in Neural Information Processing Systems*. [s.n.], 2017. Disponível em: ⟨http://arxiv.org/abs/1708.00107⟩.

MELAMUD, O.; GOLDBERGER, J.; DAGAN, I. context2vec: Learning generic context embedding with bidirectional LSTM. In: *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. Berlin, Germany: Association for Computational Linguistics, 2016. p. 51–61. Disponível em: ⟨https://www.aclweb.org/anthology/K16-1006⟩.

MEYERSON, A. Online facility location. In: IEEE. *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*. [S.l.], 2001. p. 426–431.

MIKOLOV, T.; CHEN, K.; CORRADO, G.; DEAN, J. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

MIKOLOV, T.; KOPECKY, J.; BURGET, L.; GLEMBEK, O. et al. Neural network based language models for highly inflective languages. In: IEEE. *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*. [S.l.], 2009. p. 4725–4728.

MIKOLOV, T.; SUTSKEVER, I.; CHEN, K.; CORRADO, G. S.; DEAN, J. Distributed representations of words and phrases and their compositionality. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2013. p. 3111–3119.

MILLER, G. A.; LEACOCK, C.; TENGI, R.; BUNKER, R. T. A semantic concordance. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the workshop on Human Language Technology*. [S.l.], 1993. p. 303–308.

MNIH, A.; KAVUKCUOGLU, K. Learning word embeddings efficiently with noise-contrastive estimation. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2013. p. 2265–2273.

MOONEY, R. J. Comparative experiments on disambiguating word senses: An illustration of the role of bias in machine learning. *arXiv preprint cmp-lg/9612001*, 1996.

MUGGLETON, S. Inductive logic programming. *New generation computing*, Springer, v. 8, n. 4, p. 295–318, 1991.

NANCY, I.; VÉRONIS, J. Mapping dictionaries: A spreading activation approach. In: *6th Annual Conference of the Centre for the New Oxford English Dictionary*. [S.l.: s.n.], 1990. p. 52–64.

NAVIGLI, R. Word sense disambiguation: A survey. *ACM Computing Surveys (CSUR)*, ACM, v. 41, n. 2, p. 10, 2009.

NAVIGLI, R. A quick tour of word sense disambiguation, induction and related approaches. In: *Proceedings of the 38th International Conference on Current Trends in Theory and Practice of Computer Science*. Berlin, Heidelberg: Springer-Verlag, 2012. (SOFSEM'12), p. 115–129. ISBN 978-3-642-27659-0. Disponível em: ⟨http://dx.doi.org/10.1007/978-3-642-27660-6_10⟩.

NEELAKANTAN, A.; SHANKAR, J.; PASSOS, A.; MCCALLUM, A. Efficient non-parametric estimation of multiple embeddings per word in vector space. *arXiv preprint arXiv:1504.06654*, 2015.

NG, H. T. Exemplar-based word sense disambiguation: Some recent improvements. *arXiv preprint cmp-lg/9706010*, 1997.

NG, H. T.; LEE, H. B. Integrating multiple knowledge sources to disambiguate word sense: An exemplar-based approach. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 34th annual meeting on Association for Computational Linguistics*. [S.l.], 1996. p. 40–47.

NÓBREGA, F. A. A. *Desambiguação lexical de sentidos para o português por meio de uma abordagem multilíngue mono e multidocumento*. Tese (Doutorado) — Universidade de São Paulo, 2013.

NÓBREGA, F. A. A.; PARDO, T. A. S. Explorando métodos de uso geral para desambiguação lexical de sentidos para a língua portuguesa. 2012.

NÓBREGA, F. A. A.; PARDO, T. A. S. Desambiguação lexical de sentido com uso de informação multidocumento por meio de redes de co-ocorrência (word sense disambiguation with the use of multi-document information with cooccurrence nets)[in portuguese]. In: *Proceedings of the 9th Brazilian Symposium in Information and Human Language Technology*. [S.l.: s.n.], 2013.

OCH, F. J.; NEY, H. A systematic comparison of various statistical alignment models. *Computational linguistics*, MIT Press, v. 29, n. 1, p. 19–51, 2003.

PATWARDHAN, S.; BANERJEE, S.; PEDERSEN, T. Using measures of semantic relatedness for word sense disambiguation. In: SPRINGER. *CICLing*. [S.l.], 2003. v. 2588, p. 241–257.

PEDERSEN, T.; BRUCE, R. Distinguishing word senses in untagged text. *arXiv preprint cmp-lg/9706008*, 1997.

PELEVINA, M.; AREFIEV, N.; BIEMANN, C.; PANCHENKO, A. Making sense of word embeddings. In: *Proceedings of the 1st Workshop on Representation Learning for NLP*. Berlin, Germany: Association for Computational Linguistics, 2016. p. 174–183. Disponível em: ⟨https://www.aclweb.org/anthology/W16-1620⟩.

PENNINGTON, J.; SOCHER, R.; MANNING, C. Glove: Global vectors for word representation. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. [S.l.: s.n.], 2014. p. 1532–1543.

PETERS, M.; NEUMANN, M.; IYYER, M.; GARDNER, M.; CLARK, C.; LEE, K.; ZETTLEMOYER, L. Deep contextualized word representations. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, 2018. p. 2227–2237. Disponível em: ⟨https://www.aclweb.org/anthology/N18-1202⟩.

PETERSON, L. E. K-nearest neighbor. *Scholarpedia*, v. 4, n. 2, p. 1883, 2009.

PINA, L. N.; JOHANSSON, R. A simple and efficient method to generate word sense representations. *arXiv preprint arXiv:1412.6045*, 2014.

QUINLAN, J. R. Induction of decision trees. *Machine learning*, Springer, v. 1, n. 1, p. 81–106, 1986.

QUINLAN, J. R. *C4. 5: programs for machine learning*. [S.l.]: Elsevier, 2014.

RADA, R.; MILI, H.; BICKNELL, E.; BLETTNER, M. Development and application of a metric on semantic nets. *IEEE Transactions on systems, man, and cybernetics*, IEEE, v. 19, n. 1, p. 17–30, 1989.

RADFORD, A.; NARASIMHAN, K.; SALIMANS, T.; SUTSKEVER, I. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/languageunsupervised/language understanding paper. pdf*, 2018.

RAGANATO, A.; CAMACHO-COLLADOS, J.; NAVIGLI, R. Word sense disambiguation: A unified evaluation framework and empirical comparison. In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Valencia, Spain: Association for Computational Linguistics, 2017. p. 99–110. Disponível em: ⟨https://www.aclweb.org/anthology/E17-1010⟩.

REISINGER, J.; MOONEY, R. J. Multi-prototype vector-space models of word meaning. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. [S.l.], 2010. p. 109–117.

RESNIK, P.; YAROWSKY, D. A perspective on word sense disambiguation methods and their evaluation. In: *Proceedings of the ACL SIGLEX workshop on tagging text with lexical semantics: Why, what, and how*. [S.l.: s.n.], 1997. p. 79–86.

RESNIK, P.; YAROWSKY, D. Distinguishing systems and distinguishing senses: New evaluation methods for word sense disambiguation. *Natural language engineering*, Cambridge University Press, v. 5, n. 2, p. 113–133, 1999.

RIVEST, R. L. Learning decision lists. *Machine learning*, Springer, v. 2, n. 3, p. 229–246, 1987.

RO, D.; PE, H. Pattern classification and scene analysis. Wiley, 1973.

RODRIGUES, J.; BRANCO, A.; NEALE, S.; SILVA, J. Lx-dsemvectors: Distributional semantics models for portuguese. In: SPRINGER. *International Conference on Computational Processing of the Portuguese Language*. [S.l.], 2016. p. 259–270.

ROTHE, S.; SCHÜTZE, H. Autoextend: Extending word embeddings to embeddings for synsets and lexemes. *arXiv preprint arXiv:1507.01127*, 2015.

SALTON, G.; WONG, A.; YANG, C. S. A vector space model for automatic indexing. *Commun. ACM*, ACM, New York, NY, USA, v. 18, n. 11, p. 613–620, nov. 1975. ISSN 0001-0782. Disponível em: ⟨http://doi.acm.org/10.1145/361219.361220⟩.

SCHUTZE, H. Dimensions of meaning. In: IEEE. *Supercomputing'92., Proceedings*. [S.l.], 1992. p. 787–796.

SCHÜTZE, H. Automatic word sense discrimination. *Computational linguistics*, MIT Press, v. 24, n. 1, p. 97–123, 1998.

SEBASTIANI, F. Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, ACM, v. 34, n. 1, p. 1–47, 2002.

SPECIA, L. *Uma abordagem híbrida relacional para a desambiguação lexical de sentido na tradução automática*. Tese (Doutorado) — Universidade de São Paulo, 2007.

STETINA, J.; NAGAO, M. General word sense disambiguation method based on a full sentential context. *Journal of Natural Language Processing*, The Association for Natural Language Processing, v. 5, n. 2, p. 47–74, 1998.

TEH, Y. W.; JORDAN, M. I.; BEAL, M. J.; BLEI, D. M. Sharing clusters among related groups: Hierarchical dirichlet processes. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2005. p. 1385–1392.

TELLEX, S.; KATZ, B.; LIN, J.; FERNANDES, A.; MARTON, G. Quantitative evaluation of passage retrieval algorithms for question answering. In: ACM. *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*. [S.l.], 2003. p. 41–47.

TIAN, F.; DAI, H.; BIAN, J.; GAO, B.; ZHANG, R.; CHEN, E.; LIU, T.-Y. A probabilistic model for learning multi-prototype word embeddings. In: *COLING*. [S.l.: s.n.], 2014. p. 151–160.

TOWELL, G.; VOORHEES, E. M. Disambiguating highly ambiguous words. *Computational Linguistics*, MIT Press, v. 24, n. 1, p. 125–145, 1998.

TRASK, A.; MICHALAK, P.; LIU, J. sense2vec-a fast and accurate method for word sense disambiguation in neural word embeddings. *arXiv preprint arXiv:1511.06388*, 2015.

TRAVANCA, T. *Verb sense disambiguation*. Tese (Doutorado) — Master's thesis, Instituto Superior Técnico, Universidade de Lisboa, June, 2013.

TURIAN, J.; RATINOV, L.; BENGIO, Y. Word representations: a simple and general method for semi-supervised learning. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 48th annual meeting of the association for computational linguistics*. [S.l.], 2010. p. 384–394.

WU, Z.; GILES, C. L. Sense-aaware semantic analysis: A multi-prototype word representation model using wikipedia. In: *AAAI*. [S.l.: s.n.], 2015. p. 2188–2194.

YANG, Y.; EISENSTEIN, J. Unsupervised multi-domain adaptation with feature embeddings. In: *HLT-NAACL*. [S.l.: s.n.], 2015. p. 672–682.

YAROWSKY, D. Word-sense disambiguation using statistical models of roget's categories trained on large corpora. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 14th conference on Computational linguistics-Volume 2*. [S.l.], 1992. p. 454–460.

YAROWSKY, D. One sense per collocation. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the workshop on Human Language Technology*. [S.l.], 1993. p. 266–271.

YAROWSKY, D. Decision lists for lexical ambiguity resolution: Application to accent restoration in spanish and french. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*. [S.l.], 1994. p. 88–95.

YAROWSKY, D. Hierarchical decision lists for word sense disambiguation. *Computers and the Humanities*, Springer, v. 34, n. 1, p. 179–186, 2000.

YAROWSKY, D.; CUCERZAN, S.; FLORIAN, R.; SCHAFER, C.; WICENTOWSKI, R. The johns hopkins senseval2 system descriptions. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *The Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems*. [S.l.], 2001. p. 163–166.

ZANZOTTO, F. M.; KORKONTZELOS, I.; FALLUCCHI, F.; MANANDHAR, S. Estimating linear models for computational distributional semantics. In: *Proceedings of COLING 2010*. [S.l.: s.n.], 2010. p. 1263–1271.

ZHILA, A.; YIH, W.-t.; MEEK, C.; ZWEIG, G.; MIKOLOV, T. Combining heterogeneous models for measuring relational similarity. In: *HLT-NAACL*. [S.l.: s.n.], 2013. p. 1000–1009.

ZIPF, G. K. *Human Behaviour and the Principle of Least Effort*. [S.l.]: Addison-Wesley, 1949.