UNIVERSIDADE DE SÃO PAULO

Instituto de Ciências Matemáticas e de Computação

SlimRank: um modelo de seleção de respostas para perguntas de consumidores

Marcelo Criscuolo

Tese de Doutorado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Marcelo Criscuolo

SlimRank: um modelo de seleção de respostas para perguntas de consumidores

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientadora: Profa. Dra. Sandra Maria Aluísio

USP – São Carlos
Novembro de 2017

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados fornecidos pelo(a) autor(a)

C932s Criscuolo, Marcelo
SlimRank: um modelo de seleção de respostas para
perguntas de consumidores / Marcelo Criscuolo;
orientadora Sandra Maria Aluísio. -- São Carlos,
2017.
143 p.

Tese (Doutorado - Programa de Pós-Graduação em
Ciências de Computação e Matemática Computacional) --
Instituto de Ciências Matemáticas e de Computação,
Universidade de São Paulo, 2017.

1. seleção de respostas. 2. redes neurais
convolutivas. 3. vetores distribucionais. 4. grafos
semânticos. I. Aluísio, Sandra Maria, orient. II.
Título.

Marcelo Criscuolo

**SlimRank: an answer selection model for consumer
questions**

Doctoral dissertation submitted to the Institute of
Mathematics and Computer Sciences – ICMC-USP, in
partial fulfillment of the requirements for the degree of
the Doctorate Program in Computer Science and
Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and
Computational Mathematics

Advisor: Profa. Dra. Sandra Maria Aluísio

**USP – São Carlos
November 2017**

Dedicado a Pedro e Luísa.

AGRADECIMENTOS

A Deus, por Sua paciência e grande misericórdia.

À minha esposa Ana, por tantos motivos que não os posso contar.

Aos meus pais, pelo constante apoio e forte incentivo.

À Profa. Sandra, minha professora no primeiro semestre de graduação e agora orientadora no doutorado. É admirável sua disposição em abraçar o novo. Meus agradecimentos pelo incentivo e pelos conselhos em momentos cruciais desta trajetória.

À Embrapa Gado de Leite (CNPGL), pelos dados que foram essenciais para a realização deste trabalho.

Aos amigos Alexander Antônio Benedicto (HP), Edilson José Davoglio Candido (IFSP), Ednilson Geraldo Rossi (IFSP) e Mauro de Lucca (IFSP), pela compreensão e pelo apoio que manifestaram em importantes momentos desse período de doutoramento.

Aos professores e aos funcionários do ICMC. Principalmente, ao pessoal do SVPG, pela prontidão com que sempre nos atenderam.

Aos autores deste *template* LaTeX, para teses e dissertações do ICMC.

Ao Instituto Federal de São Paulo (IFSP), pela concessão do benefício de afastamento para realização de pós-graduação *stricto sensu*, que possibilitou dedicação integral a este projeto nos últimos dois anos.

- *Espresso? Mas eu pedi um Cappuccino!*
- *Não se preocupe. A distância cosseno entre eles é tão pequena que os dois são quase a mesma coisa.*

(Adaptado de *Data Monsters, Twitter, 2017*
<bit.ly/monsters-w2v-espresso>)

RESUMO

CRISCUOLO, M. **SlimRank: um modelo de seleção de respostas para perguntas de consumidores**. 2017. 143 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2017.

A disponibilidade de conteúdo gerado por usuários em *sites* colaborativos de perguntas e respostas tem impulsionado o avanço de modelos de *Question Answering* (QA) baseados em reuso. Essa abordagem pode ser implementada por meio da tarefa de seleção de respostas (*Answer Selection*, AS), que consiste em encontrar a melhor resposta para uma dada pergunta em um conjunto pré-selecionado de respostas candidatas. Nos últimos anos, abordagens baseadas em vetores distribucionais e em redes neurais profundas, em particular em redes neurais convolutivas (CNNs), têm apresentado bons resultados na tarefa de AS. Contudo, a maioria dos modelos é avaliada sobre *corpus* de perguntas objetivas e bem formadas, contendo poucas palavras. Raramente estruturas textuais complexas são consideradas. Perguntas de consumidores, comuns em *sites* colaborativos, podem ser bastante complexas. Em geral, são representadas por múltiplas frases inter-relacionadas, que apresentam pouca objetividade, vocabulário leigo e, frequentemente, contêm informações em excesso. Essas características aumentam a dificuldade da tarefa de AS. Neste trabalho, propomos um modelo de seleção de respostas para perguntas de consumidores. São contribuições deste trabalho: (i) uma definição para o objeto de pesquisa “perguntas de consumidores”; (ii) um novo *dataset* desse tipo de pergunta, chamado MilkQA; e (iii) um modelo de seleção de respostas, chamado SlimRank. O MilkQA foi criado a partir de um arquivo de perguntas e respostas coletadas pelo serviço de atendimento de uma renomada instituição pública de pesquisa agropecuária (Embrapa). Anotadores guiados pela definição de perguntas de consumidores proposta neste trabalho selecionaram 2,6 mil pares de perguntas e respostas contidas nesse arquivo. A análise dessas perguntas levou ao desenvolvimento do modelo SlimRank, que combina representação de textos na forma de grafos semânticos com arquiteturas de CNNs. O SlimRank foi avaliado no *dataset* MilkQA e comparado com *baselines* e dois modelos do estado da arte. Os resultados alcançados pelo SlimRank foram bastante superiores aos resultados dos *baselines*, e compatíveis com resultados de modelos do estado da arte; porém, com uma significativa redução do tempo computacional. Acreditamos que a representação de textos na forma de grafos semânticos combinada com CNNs seja uma abordagem promissora para o tratamento dos desafios impostos pelas características singulares das perguntas de consumidores.

Palavras-chave: seleção de respostas, redes neurais convolutivas, vetores distribucionais, grafos semânticos.

ABSTRACT

CRISCUOLO, M. **SlimRank: an answer selection model for consumer questions.** 2017. 143 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2017.

The increasing availability of user-generated content in community Q&A sites has led to the advancement of Question Answering (QA) models that relies on reuse. Such approach can be implemented by the task of Answer Selection (AS), which consists in finding the best answer for a given question in a pre-selected pool candidate answers. Recently, good results have been achieved by AS models based on distributed word vectors and deep neural networks that are used to rank answers for a given question. Convolutinal Neural Networks (CNNs) are particularly succesful in this task. Most of the AS models are built over datasets that contains only short and objective questions expressed as interrogative sentences containing few words. Complex text structures are rarely considered. However, consumer questions may be really complex. This kind of question is the main form of seeking information in community Q&A sites, forums and customer services. Consumer questions have characteristics that increase the difficulty of the answer selection task. In general, they are composed of multiple interrelated sentences that are usually subjective, and contains layman’s terms and excess of details that may be not particulary relevant. In this work, we propose an answer selection model for consumer questions. Specifically the contributions of this work are: (i) a definition for the “consumer questions” research object; (ii) a new dataset of this kind of question, which we call MilkQA; and (iii) an answer selection model, named SlimRank. MilkQA was created from an archive of questions and answers collected by the customer service of a well-known public agricultural research institution (Embrapa). It contains 2.6 thousand question-answer pairs selected and anonymized by human annotators guided by the definition proposed in this work. The analysis of questions in MilkQA led to the development of SlimRank, which combines semantic textual graphs with CNN architectures. SlimRank was evaluated on MilkQA and compared to baselines and two state-of-the-art answer selection models. The results achieved by our model were much higher than the baselines and comparable to the state of the art, but with significant reduction of computational time. Our results suggest that combining semantic text graphs with convolutional neural networks are a promising approach for dealing with the challenges imposed by consumer questions unique characteristics.

Keywords: answer selection, convolutional neural networks, distributed word vectors, semantic graphs.

LISTA DE ILUSTRAÇÕES

Figura 1 – Etapas essenciais de um modelo de QA	37
Figura 2 – QA baseada em recuperação de informação (factoides)	42
Figura 3 – Protótipo de ILN para agenda de defesas do ICMC-USP	45
Figura 4 – Arquitetura de modelo neural de língua	65
Figura 5 – Textos representados como médias de vetores (centroides)	72
Figura 6 – Camadas de entrada: redes neurais tradicionais e redes convolutivas . .	73
Figura 7 – Extração de <i>features</i> em uma CNN	74
Figura 8 – Visão modular de uma CNN	75
Figura 9 – Exemplo de CNN para classificação de textos	76
Figura 10 – Modelo CNN-std	79
Figura 11 – Modelo CNN-LDC	80
Figura 12 – Exemplo de grafo	82
Figura 13 – Grafo representando uma frase	83
Figura 14 – Imagem representada como grafo	84
Figura 15 – <i>Framework</i> Patchy-SAN	85
Figura 16 – Ferramenta de limpeza e seleção de mensagens	93
Figura 17 – Frequência dos rótulos no MilkQA	94
Figura 18 – Agrupamento de respostas similares	99
Figura 19 – Modelo SlimRank	102
Figura 20 – Exemplo de MST	105
Figura 21 – Convolução no modelo SlimRank	109
Figura 22 – Desempenho do SlimRank no <i>dataset</i> MilkQA	120

LISTA DE QUADROS

Quadro 1 – Perguntas do tipo factóide	38
Quadro 2 – Pergunta do tipo não-factóide	39
Quadro 3 – Pergunta não-factóide do tipo DEFINIÇÃO	39
Quadro 4 – Perguntas similares, extraídas do <i>site</i> AskUbuntu	48
Quadro 5 – Perguntas formuladas para o <i>dataset</i> SelQA	52
Quadro 6 – Perguntas contidas no <i>dataset</i> WikiQA	52
Quadro 7 – Exemplo de pergunta encontrada no <i>dataset</i> InsuranceQA	54
Quadro 8 – Exemplo de pergunta de consumidor	57
Quadro 9 – Interpretação estruturada para pergunta de consumidor	58
Quadro 10 – Relações identificadas por meio de vetores de palavras	68
Quadro 11 – Perguntas objetivas (não-consumidor)	95
Quadro 12 – Perguntas de consumidor extraídas do MilkQA	96
Quadro 13 – Perguntas de consumidor sem contexto explícito	96
Quadro 14 – Pergunta extraída do <i>dataset</i> MilkQA	106
Quadro 15 – Parâmetros do modelo SlimRank	109
Quadro 16 – Analogias resolvidas por vetores treinados no domínio	112

LISTA DE ALGORITMOS

Algoritmo 1 – Algoritmo de Kruskal	105
Algoritmo 2 – Identificação de vértice representativo em uma MST	106
Algoritmo 3 – Seleção de campos receptivos	107
Algoritmo 4 – Normalização de campos receptivos	108

LISTA DE TABELAS

Tabela 1 – Estado da arte para seleção de respostas	56
Tabela 2 – <i>Datasets</i> de seleção de respostas	90
Tabela 3 – Tamanhos de Perguntas: comparação de estatísticas	96
Tabela 4 – Tamanhos de Respostas: comparação de estatísticas	97
Tabela 5 – Partições do <i>dataset</i> MilkQA	114
Tabela 6 – Desempenho da seleção de respostas nos <i>datasets</i> MilkQA e WikiQA .	116
Tabela 7 – Seleção de respostas no <i>dataset</i> MilkQA usando diferentes modelos e vetores	117
Tabela 8 – Quantidades-limite de campos receptivos	118
Tabela 9 – Desempenho do SlimRank no <i>dataset</i> MilkQA	119

LISTA DE ABREVIATURAS E SIGLAS

ACL	<i>Association for Computational Linguistics</i>
AS	<i>Answer Selection</i>
BOW	<i>Bag of Words</i>
CNN	<i>Convolutional Neural Network</i>
DNN	<i>Deep Neural Network</i>
DSM	<i>Distributional Semantic Models</i>
GLN	Geração de Língua Natural
GPU	<i>Graphics Processing Unit</i>
ICMC	Instituto de Ciências Matemáticas e de Computação
ILN	Interface em Língua Natural
KB	<i>Knowledge Base</i>
MAP	<i>Mean Average Precision</i>
MRR	<i>Mean Reciprocal Rank</i>
MST	<i>Minimum Spanning Tree</i>
NER	<i>Named-Entity Recognition</i>
NILC	Núcleo Interinstitucional de Linguística Computacional
PLN	Processamento de Línguas Naturais
QA	<i>Question Answering</i>
QALD	<i>Question Answering over Linked Data</i>
RAP	Resposta Automática a Perguntas
RI	Recuperação de Informação
SAC	Serviço de Atendimento ao Cidadão
SGBD	Sistema Gerenciador de Bancos de Dados
SQL	<i>Structured Query Language</i>
TREC	<i>Text REtrieval Conference</i>
VQA	<i>Visual Question Answering</i>
VSM	<i>Vector Space Model</i>

SUMÁRIO

1	INTRODUÇÃO	29
1.1	Hipóteses	32
1.2	Objetivos	32
1.3	Métodos e resultados	33
1.4	Organização do documento	34
2	SISTEMAS DE PERGUNTAS E RESPOSTAS	35
2.1	A área de perguntas e respostas	35
2.1.1	<i>O processo de responder perguntas automaticamente</i>	36
2.2	Classificação de modelos	38
2.2.1	<i>Tipos de resposta</i>	38
2.2.2	<i>Interação com usuário</i>	40
2.2.3	<i>Domínio</i>	40
2.3	Paradigmas de implementação	41
2.3.1	<i>Modelos baseados em recuperação de informação</i>	41
2.3.2	<i>Modelos baseados em bases de conhecimento</i>	44
2.3.3	<i>Modelos baseados em reúso</i>	47
2.3.4	<i>Outros paradigmas</i>	49
2.4	Tarefa de seleção de respostas	50
2.4.1	<i>Datasets</i>	51
2.4.2	<i>Método de avaliação</i>	54
2.4.3	<i>Estado da arte</i>	56
2.5	Perguntas de consumidores	56
2.6	Considerações finais	59
3	MODELOS NEURAIS E REPRESENTAÇÕES TEXTUAIS	61
3.1	Considerações iniciais	61
3.2	Modelos neurais de língua	64
3.3	Vetores de palavras	66
3.3.1	<i>Propriedades</i>	67
3.3.2	<i>Modelos de geração de vetores</i>	68
3.3.3	<i>Vetores pré-treinados</i>	70
3.3.4	<i>Representação de segmentos de texto</i>	71

3.4	Redes neurais convolutivas	73
3.4.1	<i>Redes convolutivas para processamento de textos</i>	76
3.4.1.1	<i>Treinamento do modelo</i>	78
3.4.2	<i>Modelos de seleção de respostas</i>	78
3.5	Grafos textuais	82
3.5.1	<i>Redes convolutivas sobre grafos</i>	84
3.6	Considerações finais	87
4	O DATASET MILKQA	89
4.1	Considerações iniciais	89
4.1.1	<i>Perguntas de consumidores: uma definição</i>	91
4.2	Origem e preparação do <i>dataset</i>	91
4.3	Perguntas de consumidores no MilkQA	95
4.4	Conjuntos de respostas candidatas	98
4.5	Considerações finais	100
5	MODELO SLIMRANK	101
5.1	Considerações iniciais	101
5.2	Modelo SlimRank	102
5.2.1	<i>Construção do grafo semântico</i>	103
5.2.2	<i>Seleção e normalização de campos receptivos</i>	104
5.2.3	<i>Convolução com campos receptivos normalizados</i>	108
5.3	Considerações finais	110
6	EXPERIMENTOS E DISCUSSÃO	111
6.1	Preparação dos experimentos	111
6.1.1	<i>Pré-processamento de textos</i>	111
6.1.2	<i>Treinamento de vetores de palavras</i>	112
6.1.3	<i>Modelos e baselines de seleção de respostas</i>	113
6.2	Avaliação do <i>dataset</i> MilkQA	115
6.3	Avaliação do modelo SlimRank	118
6.4	Discussão	121
6.5	Considerações finais	123
7	CONCLUSÕES	125
7.1	Principais contribuições	125
7.1.1	<i>Definição do objeto de pesquisa</i>	126
7.1.2	<i>Dataset MilkQA</i>	126
7.1.3	<i>Modelo SlimRank</i>	127
7.2	Publicações	127
7.3	Trabalhos futuros	128

REFERÊNCIAS 131

INTRODUÇÃO

Sistemas de Perguntas e Respostas (QA – do inglês, *Question Answering*) apresentam respostas objetivas para perguntas feitas em língua natural. Podem ser identificadas três etapas essenciais no processo de QA: (i) processamento de uma pergunta de entrada; (ii) busca por informações; e (iii) preparação de uma resposta. As informações podem ser buscadas em bases de dados estruturados, como ontologias ou bancos de dados relacionais, ou em textos não-estruturados, como documentos da *web*.

Nos casos em que são usados dados estruturados, a busca por informações é normalmente delegada a algum componente especializado, como um sistema gerenciador de bancos de dados, por exemplo, enquanto, na preparação de respostas, é comum o uso de modelos de texto (*templates*) contendo lacunas que são preenchidas com os dados encontrados a partir da pergunta. O emprego de *templates* visa evitar a complexidade da tarefa de Geração de Língua Natural.

Nos sistemas de QA que têm documentos não-estruturados como fontes de dados, é bastante comum a abordagem extrativa, em que a preparação da resposta consiste em identificar um segmento de texto contendo a informação solicitada. Pode-se, por exemplo, segmentar um documento relacionado ao tópico da pergunta em frases, e então empregar um método de seleção para identificar uma dessas frases que contenha a resposta desejada. Outra abordagem, impulsionada nos últimos anos pelo aumento dos serviços colaborativos de perguntas e respostas – como o StackOverflow¹, por exemplo – é o desenvolvimento de modelos de QA baseados em reuso. Nessa abordagem, a etapa de busca por informações não envolve a segmentação de documentos em frases, mas a busca por respostas que possam estar relacionadas à pergunta, gerando assim um conjunto de respostas candidatas que, na etapa seguinte, passam por um filtro destinado a identificar a melhor resposta para uma nova pergunta, recebida como entrada no início desse processo.

¹ StackOverflow: <<https://stackoverflow.com/>>

Observa-se assim, tanto na abordagem extrativa quanto na abordagem baseada em reuso, o papel fundamental da tarefa de seleção de respostas para os sistemas de QA. Este trabalho está concentrado especificamente na tarefa de *seleção de respostas* (AS –, do inglês, *Answer Selection*), que consiste em encontrar uma resposta para uma dada pergunta em um conjunto de respostas candidatas pré-selecionado.

Há, na literatura, diversos trabalhos dedicados à tarefa de seleção de respostas (SEVERYN; MOSCHITTI, 2015; SANTOS *et al.*, 2015; WANG; MI; ITTYCHERIAH, 2016; RAO; HE; LIN, 2016; WANG; HAMZA; FLORIAN, 2017, e outros). A maior parte deles é dedicada ao estudo de perguntas objetivas, como, por exemplo²:

(1.1) Onde são encontrados os ursos panda gigantes?

(1.2) Como funciona um motor a vapor?

Entretanto, nem todas as perguntas formuladas em língua natural são diretas e bem formadas. São comuns em fóruns e em Serviços de Atendimento ao Consumidor (SACs) perguntas que apresentam estrutura bastante complexa, como, por exemplo³:

(1.3) Tenho algumas vacas leiteiras e percebi que tem umas que caíram bastante o leite. Fiz o teste c.m.t e constatei em alguns quartos de teto a presença de mastite. Nao sei qual remedio aplicar. Melhor injetavel ou via teto? Podem ajudar?

Essas perguntas, chamadas de Perguntas de Consumidores (*Consumer Questions*), são normalmente mais longas, apresentadas na forma de múltiplas frases, e contêm informações sobre o contexto em que se originaram. Também costumam conter erros ortográficos e gramaticais, além de termos leigos. As frases interrogativas costumam ser simples, como “Melhor injetável ou via teto?”, e, se analisadas isoladamente, muitas vezes não têm sentido completo, pois dependem de outras informações presentes no texto. Também são comuns perguntas apresentadas de forma indireta, como “*Eu gostaria de saber...*”, além da ocorrência de múltiplas sub-perguntas (ZHANG, 2010). As respostas para perguntas de consumidores também costumam ser mais longas e elaboradas do que as respostas dadas a perguntas objetivas.

Em trabalhos pioneiros, fenômenos linguísticos frequentes em perguntas de consumidores, como anáforas e elipses, foram abordados diretamente, com o objetivo extrair representações estruturadas de cada pergunta (KILICOGLU; FISZMAN; DEMNER-FUSHMAN, 2013). Contudo, o processamento automático de tais fenômenos está longe

² Exemplos adaptados do *dataset* WikiQA (YANG; YIH; MEEK, 2015).

³ Extraída do *dataset* MilkQA, apresentado neste trabalho. Foram mantidos os erros ortográficos e gramaticais, presentes no texto original.

de ser trivial e a abordagem direta pode ser viável apenas em circunstâncias muito específicas, como aquelas em que foi desenvolvido esse trabalho pioneiro. Além disso, o sucesso da abordagem direta também depende fortemente de ferramentas de análise sintática, que, por sua vez, dependem de uma etapa de normalização textual em que sejam corrigidos erros comuns em perguntas de consumidores (erros ortográficos, por exemplo). Essa cadeia de dependências torna ainda mais difícil o tratamento dos fenômenos linguísticos mencionados, que são abundantes em perguntas de consumidores.

Modelos de seleção de respostas baseados em Redes Neurais Profundas (DNNs – do inglês, *Deep Neural Networks*) (NIELSEN, 2015; GOODFELLOW; BENGIO; COURVILLE, 2016) são menos sensíveis a variações linguísticas, a ocorrências de fenômenos como elipses e a erros ortográficos, uma vez que, na maioria dos casos, não dependem das análises morfológica e sintática. Em vez disso, os modelos de DNNs são fortemente baseados no princípio da semântica distributiva, que propõe modelar o significado de palavras estatisticamente, a partir de regularidades observadas em determinados contextos linguísticos (LENCI, 2008 apud HARISPE *et al.*, 2015). Na prática, cada palavra é representada como um vetor em um espaço $\mathbb{R}^d$, onde relações entre vetores – medidas pela similaridade cosseno (MIKOLOV; YIH; ZWEIG, 2013), por exemplo – implicam em relações semânticas entre as palavras correspondentes. Esses vetores são gerados a partir da análise de grandes corpú de texto, normalmente por modelos neurais de língua, como o Word2vec (MIKOLOV *et al.*, 2013b).

A tarefa de seleção de respostas pode ser implementada medindo a similaridade semântica entre uma pergunta e cada uma das respostas candidatas contidas em um conjunto. A candidata para a qual for computado o maior grau de similaridade semântica é considerada a resposta correta. Arquiteturas de redes neurais profundas, em particular as redes neurais convolutivas (CNNs – do inglês, *Convolutional Neural Networks*) (LECUN *et al.*, 1990) têm sido empregadas com sucesso nessa abordagem de AS (SEVERYN; MOSCHITTI, 2015; WANG; MI; ITTYCHERIAH, 2016; RAO; HE; LIN, 2016; SANTOS *et al.*, 2016). Nas CNNs, os textos de entrada são representados por matrizes, formadas pela concatenação de vetores correspondentes às palavras do texto, na ordem em que ocorrem. Isto é, nesses modelos, os textos são modelados pela adjacência de palavras, representadas por vetores. Acreditamos que a representação baseada em adjacência seja adequada para textos curtos, como as frases que representam as perguntas objetivas. Contudo, nessa representação não se consideram explicitamente as relações semânticas intra-textuais, que são recorrentes em perguntas de consumidores.

Além disso, a representação de textos baseada na adjacência de palavras requer quantidade considerável de recursos computacionais. A representação vetorial de textos envolve um elevado número de cálculos e alto consumo de memória. Tais exigências normalmente não representam um problema quando se processa textos curtos, sendo sufici-

ente treinar os modelos em GPUs⁴. Perguntas de consumidores, por outro lado, podem facilmente conter 200 palavras, sendo que algumas chegam a ter 700 palavras ou mais. As repostas costumam ser igualmente longas, contendo centenas de palavras. Uma técnica comum em vários trabalhos (YANG; YIH; MEEK, 2015; LEI *et al.*, 2016) consiste em estabelecer um tamanho-limite para os textos, truncando o conteúdo excedente. Trata-se de uma técnica simples e eficiente, que visa evitar o desperdício de recursos computacionais causado por *outliers* em conjuntos de textos curtos. Porém, no caso de textos longos, estabelecer um ponto de corte que leve ao uso eficiente dos recursos computacionais pode resultar em descarte arbitrário de informação.

1.1 Hipóteses

Considerando a discussão anterior, a seguir, são apresentadas as hipóteses para esta tese.

Hipótese 1. Nos casos de textos compostos por múltiplas frases em que sejam abundantes as relações semânticas intra-textuais, uma representação computacional na forma de redes de relações entre palavras é mais adequada do que a tradicional representação baseada na adjacência de palavras.

Hipótese 2. Por meio da exploração das relações semânticas intra-textuais, é possível obter uma representação textual sucinta ao mesmo tempo em que se evita o descarte arbitrário de informação.

1.2 Objetivos

O principal objetivo deste trabalho é desenvolver um modelo de seleção de repostas para perguntas de consumidores, que atenda aos seguintes requisitos: (i) evitar o descarte arbitrário de informações; e (ii) modelar textos como redes de relações semânticas, de maneira que fenômenos linguísticos complexos, como elipses, possam ser interpretados indiretamente.

Uma vez que não são conhecidos *datasets* de perguntas de consumidores que estejam publicamente disponíveis, também é um objetivo deste trabalho preparar um *dataset* desse tipo, contendo perguntas em língua portuguesa.

⁴ *Graphics Processing Units* (GPUs) são circuitos eletrônicos projetados para executar cálculos e manipular memória rapidamente.

1.3 Métodos e resultados

Para a criação do *dataset* de perguntas de consumidores, chamado MilkQA, foi usado como fonte de dados um arquivo de mensagens coletadas pelo Serviço de Atendimento ao Cidadão (SAC) da Embrapa Gado de Leite⁵, contendo perguntas recebidas de milhares de consumidores entre os anos de 2003 e 2012, e suas respectivas respostas, elaboradas por um especialista. O arquivo inicial passou por duas etapas de seleção. Na primeira etapa, completamente automática, métodos estatísticos foram usados para descartar mensagens que tivessem poucas chances de aproveitamento. Foi possível reduzir, assim, o volume de textos a ser analisado na etapa seguinte, realizada manualmente. Nessa etapa, três anotadores selecionaram e anonimizaram perguntas de consumidores e suas respectivas respostas. Como resultado, foram obtidos 2.657 pares de (pergunta, resposta) do domínio de produção leiteira.

Para o desenvolvimento do modelo de seleção de respostas proposto neste trabalho, chamado SlimRank, foi elaborado um modelo de representação textual baseado em grafos, que serviu como entrada para modelos de CNNs, responsáveis por computar a similaridade semântica entre pares de (pergunta, resposta). A normalização de campos receptivos (subgrafos) em vetores numéricos que pudessem ser processados por CNNs foi realizada conforme o método apresentado no *framework* Patchy-SAN (NIEPERT; AHMED; KUTZ-KOV, 2016). Um algoritmo para seleção de campos receptivos em grafos textuais também foi desenvolvido neste trabalho, a partir da exploração de um critério de centralidade em *clusters* que é empregado na árvore métrica Slim-tree (TRAINA JR. *et al.*, 2000). Dessa maneira, foi possível atingir o objetivo de se evitar o descarte arbitrário de informações. Além de capturar corretamente diversas relações semânticas encontradas nos textos do *dataset* MilkQA, o modelo SlimRank apresentou desempenho compatível com modelos do estado da arte na tarefa de seleção de respostas, necessitando, para tanto, apenas de uma fração do tempo computacional.

De maneira pontual, as contribuições deste trabalho são:

- uma proposta de definição para o objeto de pesquisa “perguntas de consumidores”;
- um *dataset* de perguntas de consumidores, que deverá ser disponibilizado publicamente;
- um método de seleção de respostas, baseado em técnicas do estado da arte e projetado para explorar características de perguntas de consumidores.

⁵ Empresa Brasileira de Pesquisa Agropecuária (Embrapa) – <<http://www.embrapa.br/gado-de-leite/>>

1.4 Organização do documento

O restante deste documento está organizado conforme se descreve a seguir:

- No [Capítulo 2](#) é apresentada uma revisão sobre a área de QA, que, além de contextualizar o projeto de pesquisa, também tem o objetivo de prover a fundamentação teórica para o desenvolvimento do trabalho.
- O [Capítulo 3](#) é dedicado a modelos neurais de língua e de seleção de respostas e também à representação de textos, principalmente na forma de grafos. Nesse capítulo são apresentados os modelos do estado da arte da tarefa de seleção de respostas e também o *framework* Patchy-SAN, que serve de base ao modelo SlimRank.
- No [Capítulo 4](#) descreve-se o *dataset* MilkQA, criado neste projeto de pesquisa.
- No [Capítulo 5](#) é apresentado o modelo de seleção de respostas SlimRank, proposto neste trabalho. São descritos os processos de construção dos grafos semânticos, de seleção e de normalização de campos receptivos e a arquitetura baseada em redes convolutivas que compõem o modelo.
- No [Capítulo 6](#) são descritos experimentos realizados com o objetivo de avaliar a dificuldade da tarefa de seleção de respostas no *dataset* MilkQA e também o desempenho do modelo SlimRank. Nesse capítulo, descrevem-se detalhadamente as etapas de pré-processamento de textos, de treinamento de vetores de domínio e de treinamento dos modelos de seleção de respostas. Também são apresentados e discutidos os resultados obtidos.
- No [Capítulo 7](#) é apresentada a conclusão geral deste trabalho, destacando os resultados alcançados e apontando limitações e trabalhos futuros.

SISTEMAS DE PERGUNTAS E RESPOSTAS

Neste capítulo, apresenta-se uma revisão sobre a área de Sistemas de Perguntas e Respostas, com o objetivo de contextualizar o projeto de pesquisa. O capítulo se inicia com uma descrição do problema de Perguntas e Respostas e das principais técnicas encontradas na literatura. Em seguida, a discussão é direcionada para temas diretamente relacionados ao projeto, que são tratados com mais profundidade. São de particular interesse as seções sobre a tarefa de seleção de resposta e sobre perguntas de consumidores.

2.1 A área de perguntas e respostas

Na área de Perguntas e Respostas (em inglês, *Question Answering*¹ – QA), busca-se desenvolver sistemas que apresentem respostas objetivas para perguntas feitas em língua natural. Trata-se de uma disciplina de Processamento de Línguas Naturais (PLN), quase tão antiga quanto a própria computação eletrônica, com os primeiros sistemas desenvolvidos no início da década de 1960 (SIMMONS, 1965).

Em um primeiro momento, se destaca a semelhança com a área de Recuperação de Informação (RI), popularizada pelos mecanismos de busca da Internet. No entanto, a QA se distingue por dois aspectos principais: (i) receber como entrada segmentos de texto, onde são consideradas as relações entre constituintes, em vez de conjuntos de palavras-chave; e (ii) apresentar como resultado uma só resposta, em vez de coleções de documentos, como ocorre com mecanismos de RI. Contudo, é fato que alguns mecanismos de busca na *web* contemporâneos se tornaram plataformas híbridas, que vão além da recuperação de documentos, respondendo objetivamente alguns tipos de pergunta, como, por exemplo,

¹ Também se refere à área de *Question Answering* (QA), em português, como Resposta a Perguntas ou Resposta Automática a Perguntas (RAP). Esses termos são adotados principalmente no português europeu, como se observa em Santos (2012) e outros trabalhos.

“Onde fica o ICMC, da USP?”².

Várias formas de pergunta podem ser usadas na interação com sistemas de QA. Além da forma interrogativa, que ocorre naturalmente com maior frequência, também são comuns em diversos trabalhos (LOPEZ *et al.*, 2012; SIMÕES; COSTA; MOTA, 2012), a forma imperativa e mesmo expressões parciais, como se observa nos seguintes exemplos:

(2.1) Qual é o telefone da Embrapa Instrumentação Agropecuária? (interrogativa)

(2.2) Encontre o telefone da Embrapa Instrumentação Agropecuária. (imperativa)

(2.3) telefone da Embrapa Instrumentação Agropecuária (expressão parcial)

Independentemente da forma empregada, é possível verificar que esses três exemplos representam a mesma necessidade de informação³. Portanto, no contexto dos sistemas de QA, é possível generalizar a noção de *pergunta* para algo como “necessidades de informação expressas em língua natural”, de maneira que o termo não fique restrito apenas à forma interrogativa. Essa generalização do termo *pergunta* é empregada ao longo deste texto.

2.1.1 O processo de responder perguntas automaticamente

O processo de QA consiste, basicamente, em buscar dados relacionados a uma pergunta, e então usá-los para preparar uma resposta. Há, na literatura, uma grande quantidade de modelos de QA, que implementam esse processo com variados graus de sofisticação, empregando técnicas e recursos diversos. Em todos os modelos é possível identificar três etapas essenciais, ilustradas na Figura 1 e descritas a seguir.

Na etapa (1), a pergunta é processada com o objetivo de se elaborar alguma forma de consulta (*query*), que guiará a busca por informações relacionadas ao tópico da pergunta, e também com o objetivo de se preparar critérios de filtragem que, posteriormente, servirão para avaliar a adequação de possíveis respostas.

Na etapa (2), uma ou mais fontes de dados são consultadas em busca de informações relacionadas à pergunta. A escolha da fonte de dados depende do modelo implementado e do contexto em que está inserido o sistema. É comum o uso de documentos

² Essa pergunta foi respondida corretamente pelo Google, no momento em que se escreve este texto. A resposta apresentada foi: “Avenida Trabalhador Sancarlense, 400 - Centro, São Carlos - SP, 13566-590”, que é o endereço do Instituto de Ciências Matemáticas e de Computação (ICMC).

³ Considera-se *necessidade de informação* o tópico sobre o qual um usuário deseja se informar. Uma consulta (*query*) – ou pergunta, no caso de QA – é vista como uma tentativa, por parte do usuário, de comunicar ao computador uma necessidade de informação (MANNING; RAGHAVAN; SCHÜTZE, 2008, p.5).

Figura 1 – Etapas essenciais de um modelo de QA

Fonte: Elaborada pelo autor.

contendo texto em língua natural (dados não-estruturados), armazenados localmente ou buscados na *web*. Também é possível o uso de fontes de dados estruturados, como bancos de dados relacionais, por exemplo.

Por fim, na etapa (3), deve ser preparada uma resposta para a pergunta, a partir das informações obtidas na etapa anterior. Idealmente, a tarefa de elaborar respostas tange o problema de Geração de Língua Natural (GLN). Porém, métodos complexos de GLN são normalmente evitados em QA, em favor de alternativas mais simples. Três abordagens são comumente usadas na geração de respostas: (i) a extração de segmentos de texto, (ii) o uso de *templates* e (iii) a sumarização focada na pergunta.

A técnica de *extração* consiste em selecionar um segmento de texto que possa responder a pergunta, encontrado nos documentos obtidos na etapa de busca. Esse segmento de texto pode ser uma expressão curta – representando uma entidade nomeada, por exemplo –, uma frase ou mesmo um parágrafo. Os *templates*, em sua forma mais simples, são respostas previamente preparadas, contendo lacunas, que são posteriormente preenchidas pelo sistema com dados obtidos na etapa de busca. Há também formas mais elaboradas de *templates*. Em um trabalho desenvolvido para o português, [Novais, Tadeu e Paraboni \(2010\)](#), descrevem um método de realização de superfície textual linguisticamente motivado, em que respostas são geradas por meio da alteração de modelos de frases, extraídos de um corpus. O método prevê a alteração de constituintes do tipo agente, paciente e ação. Finalmente, a técnica de *sumarização* consiste em combinar informações de múltiplos documentos gerando um sumário que atenda à necessidade de informação expressa na pergunta. Essa abordagem é conhecida na literatura como sumarização focada em consultas⁴ ([DAUMÉ III; MARCU, 2006](#)).

⁴ Em inglês, *query-focused summarization*.

2.2 Classificação de modelos

Modelos de sistemas de QA podem ser classificados de várias perspectivas. Nesta seção, apresentam-se algumas classificações que ocorrem frequentemente na literatura.

2.2.1 Tipos de resposta

A classificação pelo tipo de resposta é a mais usada na caracterização de perguntas em QA. De acordo com esse critério, as perguntas são separadas em duas grandes classes: *factoides* e *não-factoides*.

Perguntas do tipo *factoide* são aquelas que podem ser respondidas por expressões textuais simples, que representem fatos, como nomes de lugares, nomes de pessoas, datas e valores numéricos. Normalmente, são perguntas que envolvem os pronomes interrogativos *quando* (*when*), *onde* (*where*), *qual* (*what*) e *quem* (*who*) – conhecidas por *wh-questions*, em inglês. No [Quadro 1](#), são apresentados alguns exemplos de factoides.

Quadro 1 – Perguntas do tipo factoide

Pergunta	Resposta	Tipo
1. Em que país afundou o navio Costa Concordia?	Itália	PAÍS
2. Qual é o telefone da Embrapa Instrumentação?	(16) 2107-2800	TELEFONE
3. Quando o Brasil foi descoberto?	22/04/1500	DATA

Fonte: Elaborada pelo autor.

Factoides são frequentemente classificados pelo tipo de resposta esperada, como se observa nos exemplos do [Quadro 1](#). Tipicamente, as respostas extraídas de documentos pertencentes a uma coleção usada como fonte de dados. A classificação por tipo de resposta é parte da estratégia usada para identificar segmentos de texto compatíveis com uma pergunta. [Li e Roth \(2002\)](#) apresentam uma taxonomia de perguntas organizada em dois níveis, adotada em diversos outros trabalhos da literatura. No primeiro nível hierárquico são previstos 6 tipos de perguntas: ABREVIACÃO, ENTIDADE, DESCRIÇÃO, HUMANO, LOCALIZAÇÃO e VALOR NUMÉRICO. Cada um desses tipos é sub-dividido em tipos de granularidade mais fina, como, por exemplo, PAÍS e CIDADE, no caso de LOCALIZAÇÃO.

Alguns factoides podem ser respondidos por métodos relativamente simples. Uma alternativa é explorar a redundância de dados existente na *web* ([BRILL; DUMAIS; BANKO, 2002](#)). Esse método se baseia na premissa de que uma informação de interesse deve ocorrer em diversos documentos da *web*. Considere-se o exemplo (3), no [Quadro 1](#). É esperado que a data desejada ocorra em muitos documentos da *web* relacionados ao tópico *descobrimento do Brasil*. Assim, após obter uma amostra desses documentos, são coletadas expressões compatíveis com a pergunta, do tipo DATA. Considera-se como a resposta correta aquela que ocorrer no maior número de documentos da amostra. Contudo, métodos

simples não são a regra para factoides. Dependendo da pergunta e das informações disponíveis, alguns casos podem requerer processos complexos que envolvam relações ontológicas e inferência (MOLDOVAN *et al.*, 2002).

São consideradas *não-factoides* perguntas que requeiram respostas mais complexas do que expressões simples, como as descritas anteriormente. Um trabalho pioneiro no tratamento desse tipo de pergunta é o de Soricut e Brill (2004). No Quadro 2 é apresentado um exemplo de pergunta do tipo não-factoides.

Quadro 2 – Pergunta do tipo não-factoides

Como um filme se qualifica para um Prêmio da Academia?

Um longa-metragem deve ser exibido em um cinema do condado de Los Angeles em 35 ou 70 mm ou em um formato digital de varredura progressiva de 24 quadros adequado para exibir em salas de cinema digital comerciais por sete dias consecutivos. (...)

Fonte: Adaptada de Soricut e Brill (2004).

Assim como ocorre com factoides, as perguntas do tipo não-factoides também são classificadas em sub-tipos. Na avaliação conjunta RespubliQA (PEÑAS *et al.*, 2009) foram propostos alguns sub-tipos para não-factoides, como, por exemplo DEFINIÇÃO e PROCEDIMENTO. Em perguntas do tipo PROCEDIMENTO, espera-se como resposta uma série de ações, que descrevam um meio oficial, ou aceito, de se fazer algo. Um exemplo desse tipo de pergunta é o apresentado no Quadro 2. Já em perguntas do tipo DEFINIÇÃO, espera-se como resposta informações que definam alguma entidade, como se observa no exemplo apresentado no Quadro 3.

Quadro 3 – Pergunta não-factoides do tipo DEFINIÇÃO

O que significa “leite integral”?

Para os propósitos deste documento, “leite integral” é o produto obtido da ordenha de uma ou mais vacas, cuja composição não tenha sido modificada desde a ordenha.

Fonte: Adaptada de Peñas *et al.* (2009).

No exemplo do Quadro 3 a resposta foi gerada por meio da seleção de um parágrafo de texto. Essa foi a estratégia adotada na avaliação conjunta RespubliQA 2009. De acordo com as regras dessa avaliação conjunta, respostas complexas, requeridas por perguntas do tipo não-factoides, deviam ser geradas por meio da seleção de um parágrafo de texto que contivesse, possivelmente entre outras informações, a resposta esperada. Em outra avaliação conjunta, a TREC-QA 2003 (VOORHEES, 2003), a estratégia adotada para

responder perguntas do tipo DEFINIÇÃO foi compilar listas de características a respeito da entidade a ser definida. Essas listas deviam ser formadas por expressões como “compositor norte-americano”, “nascido em 1900” e “filho de um imigrante judeu”, conforme se observa no trabalho de [Hildebrandt, Katz e Lin \(2004\)](#), dedicado a responder perguntas do tipo DEFINIÇÃO.

Há um tipo de pergunta não-factóide que, além de requerer respostas complexas, apresenta características bastante peculiares em sua estrutura linguística. Esse tipo de pergunta, que foi chamado neste trabalho de PERGUNTAS DE CONSUMIDORES, será abordado na [Seção 2.5](#) e depois, com mais profundidade, no [Capítulo 4](#).

2.2.2 Interação com usuário

Tradicionalmente, a interação do usuário com um sistema de QA termina quando uma resposta é retornada. Caso não tenha sua necessidade atendida, o usuário pode reformular a pergunta, fazendo uma nova tentativa, isolada da anterior. A observação de que usuários costumam apresentar várias perguntas sucessivas a sistemas de QA, motivou o surgimento da área de *QA Interativa* (*Interactive Question Answering*) ([QUARTERONI; MANANDHAR, 2009](#)).

A *QA Interativa* é originada da fusão dos sistemas de QA com sistemas de diálogo, e é definida como um processo em que o usuário participa continuamente do ciclo informativo, fazendo uma pergunta inicial e avaliando a relevância de respostas intermediárias, até atingir seu objetivo. A abordagem interativa é considerada particularmente útil nos casos em que a necessidade de informação ainda seja vaga, ou seja, nos casos em que o usuário não seja capaz de declará-la objetivamente, possivelmente por falta de familiaridade com o tópico. Nessas situações, um sistema interativo pode participar ativamente da busca por informação, solicitando ao usuário esclarecimentos que orientem a formação de um modelo interno de suas intenções ([WEBB; WEBBER, 2009](#)).

2.2.3 Domínio

Quanto ao domínio, sistemas de QA podem ser classificados como de *domínio aberto* ou de *domínio restrito*. Sistemas de *domínio aberto* são destinados a responder uma ampla variedade de perguntas, e não se limitam a determinado contexto ou área de conhecimento. Como fontes de dados, esses sistemas podem usar grandes coleções de documentos privadas ou a própria *web*. A implementação de sistemas de domínio aberto tende a ser mais desafiadora em diversos aspectos, devido à grande variedade linguística e à variedade de tópicos a que o sistema estará exposto.

Sistemas de *domínio restrito*, por outro lado, atuam em contextos limitados. Assistentes virtuais (*chatbots*), destinados a prover informações sobre instituições ([WAN-](#)

TROBA; ROMERO, 2015) e interfaces em língua natural para bases de dados estruturadas (GREEN *et al.*, 1961; WOODS; KAPLAN; NASH-WEBBER, 1972) são exemplos típicos de sistemas de domínio restrito. A principal característica da QA em domínio restrito é a integração com informações específicas do domínio (MOLLÁ; VICEDO, 2007). Ao restringir a abrangência de um sistema de QA a um contexto específico, é possível, por exemplo, definir um conjunto de padrões sintáticos suficientemente abrangente que permita o mapeamento de praticamente todas as possíveis perguntas em estruturas de dados pré-definidas ou em ontologias de domínio. A definição de ontologias de domínio, por exemplo, é uma estratégia adotada em diversos trabalhos de QA de domínio restrito (WANTROBA; ROMERO, 2015; WANG, 2011; LOPEZ *et al.*, 2007).

Se por um lado a limitação do domínio simplifica diversas tarefas, por outro, deve-se levar em consideração que o volume de dados, tipicamente menor nesses contextos, impede o uso de algumas técnicas (MOLLÁ; VICEDO, 2007), como a exploração da redundância de dados (BRILL; DUMAIS; BANKO, 2002; COSTA, 2004), por exemplo.

2.3 Paradigmas de implementação

Nesta seção, são apresentados os principais paradigmas de implementação de modelos de QA.

2.3.1 Modelos baseados em recuperação de informação

Nesse paradigma de implementação, modelos de QA são construídos em torno de um componente de Recuperação de Informação (RI), que assume papel central no processo de responder perguntas. São desenvolvidos componentes dedicados ao processamento de língua natural, de perguntas e de respostas, enquanto a busca por documentos relacionados ao tópico de interesse é integralmente delegada a um componente de RI “de prateleira”⁵.

Essa estratégia é amplamente usada na literatura, principalmente por modelos dedicados a responder factoides – apesar de não estar limitada a esse tipo de pergunta. Na verdade, modelos baseados em RI são o padrão se usam fontes de dados formadas por textos em língua natural (documentos não-estruturados). Esses textos podem ser armazenados em um repositório local ou buscados diretamente da *web*.

Não há restrições quanto ao componente de RI utilizado. Há trabalhos em que a recuperação de documentos é delegada a um Sistema Gerenciador de Bancos de Dados (SGBD) capaz de realizar a indexação de textos (SARMENTO, 2006). No entanto, observa-se que o Apache Lucene⁶ é o componente mais frequentemente adotado.

⁵ Do inglês, *off-the-shelf*. Ou seja, um componente pronto, desenvolvido por terceiros.

⁶ Apache Lucene: <<https://lucene.apache.org/>>

Figura 2 – QA baseada em recuperação de informação (factoides)

Fonte: Elaborada pelo autor.

O paradigma de QA baseado em RI é descrito a seguir, a partir da arquitetura ilustrada na Figura 2. Essa arquitetura é considerada padrão para o problema de se responder factoides, e é amplamente adotada na literatura (BRANCO *et al.*, 2008; HILDEBRANDT; KATZ; LIN, 2004; SJAFRIZAL *et al.*, 2013; BOS; GUZZETTI; CURRAN, 2007, e outros).

São ilustrados, na Figura 2, o fluxo de dados e a interação entre os componentes que formam o modelo de QA. Por se tratar de um processo centrado na RI, a primeira etapa é dedicada à elaboração de uma consulta (*query*) que possa induzir o componente de RI a retornar documentos relacionados ao tópico da pergunta. Uma maneira simples de se obter uma consulta é pela remoção de *stopwords* e sinais de pontuação. Pode-se considerar, por exemplo, a pergunta:

(2.4) Quem matou Abraham Lincoln?

Aplicando essa técnica simples, seria obtida uma consulta representada pelo conjunto de palavras-chave: {matou, abraham, lincoln}. Outra técnica, empregada no sistema AskMSR (BRILL; DUMAIS; BANKO, 2002), consiste em transformar perguntas em consultas por meio da aplicação de regras. Por meio de regras definidas manualmente, esse sistema transforma a pergunta do exemplo anterior na consulta “* matou Abraham Lincoln”, que é então passada a um mecanismo de buscas da *web*. Essa estrutura de consulta é baseada na premissa de que devem existir documentos contendo frases do tipo “<nome> matou Abraham Lincoln”, de onde seria possível extrair facilmente uma resposta.

Outra importante tarefa, realizada na primeira etapa do processo, é a classifica-

ção da pergunta pelo tipo de resposta requerido, o que é feito, na maior parte dos casos, por meio de técnicas de aprendizado de máquina. No caso da pergunta apresentada no exemplo (2.4), a resposta requerida seria do tipo PESSOA – ou seja, o nome da pessoa que matou Lincoln. Essa informação é usada como entrada na última etapa do processo (extração da resposta), onde expressões compatíveis com o tipo da pergunta serão extraídas dos documentos retornados pelo componente de RI. As expressões obtidas no processo de extração são consideradas respostas candidatas, e passam ao final por uma seleção que determinará a resposta para a pergunta. Uma vez que respostas para perguntas do tipo factóide frequentemente são entidades nomeadas, em alguns trabalhos (AMARAL *et al.*, 2005; SARMENTO, 2006) empregam-se reconhecedores de entidades (NER)⁷, com o objetivo de se identificar entidades compatíveis com o tipo de pergunta nos documentos obtidos.

A seleção de respostas, que também ocorre na etapa final, consiste em selecionar a resposta candidata que tenha recebido a maior pontuação (*score*) de compatibilidade com a pergunta. Normalmente, a pontuação de cada resposta candidata é calculada em função do número de documentos em que ela ocorre. É possível considerar alguma medida de similaridade entre o tipo de resposta requerido e o tipo verificado em cada candidata (SJA-FRIZAL *et al.*, 2013), ou simplesmente fazer uma contagem do número de documentos em que ocorra a resposta candidata (BRILL; DUMAIS; BANKO, 2002). Essa última abordagem poderia ser interpretada intuitivamente como uma medida de popularidade da resposta candidata.

Considerando que as chances de identificação de uma resposta são aumentadas com quantidades maiores de documentos, uma técnica frequentemente usada na etapa de processamento da pergunta é a *expansão de consultas*. O objetivo é ampliar a cobertura (*recall*) do componente de RI, adicionando à consulta mais palavras relacionadas à pergunta. Para implementação dessa técnica, usam-se recursos externos, como a WordNet (MILLER, 1995), que é um banco de dados léxico, em que palavras são agrupadas em conjuntos de sinônimos (*synsets*). A WordNet foi explorada por Sjafrizal *et al.* (2013) para expandir consultas geradas em seu modelo de QA, por meio da adição de palavras semanticamente relacionadas às palavras originais.

Apesar de sua grande popularidade, o paradigma de QA baseado em RI apresenta limitações. A principal crítica sofrida por esse paradigma se deve justamente à forte dependência de um componente de RI. Segundo Narayanan e Harabagiu (2004), o paradigma de QA baseado em RI é limitado pela premissa de que a resposta pode ser encontrada em textos que contenham palavras presentes na pergunta. Esses autores argumentam que respostas mais precisas podem ser encontradas se tanto a indexação de documentos quanto a geração de respostas forem realizadas por meio de estruturas semânticas, que

⁷ *Named-Entity Recognition* (NER)

permitam, inclusive, a realização de processos de inferência⁸. Eles representaram informações semânticas por meio de estruturas argumentais (*predicate-argument structures*), identificadas a partir do corpus PropBank (KINGSBURY; PALMER; MARCUS, 2002), formado por palavras anotadas com seus papéis semânticos.

Ainda, Bilotti e Nyberg (2008) estudaram a interação entre os componentes da arquitetura típica de QA. Em seus experimentos, esses autores substituíram o componente de RI de um sistema de QA por outro de alta precisão, e observaram que os melhores resultados produzidos pelo novo componente não foram aproveitados na etapa de extração de respostas. Concluíram assim que aumentar a eficiência de componentes isolados não melhora, necessariamente, o desempenho geral do sistema, recomendando que se buscassem maneiras de diminuir o isolamento entre os componentes de RI e de extração de respostas.

São abundantes, na literatura, os modelos que implementam o paradigma de QA baseado em recuperação de informação. Dois bons exemplos de implementação desse paradigma de QA para o português são os sistemas XisQuê⁹ (BRANCO *et al.*, 2008), que usa a *web* como fonte de dados, e o Priberam (AMARAL *et al.*, 2005), que usa um repositório de dados local, indexado por um componente de RI próprio.

Por fim, uma interessante revisão sobre modelos de QA baseados em recuperação de informação é apresentada em Kolomiyets e Moens (2011).

2.3.2 Modelos baseados em bases de conhecimento

Sistemas que implementam o paradigma de QA baseado em Bases de Conhecimento (KB)¹⁰ distinguem-se, principalmente, por obter informações de fontes de dados estruturados. Nesse paradigma, é comum que perguntas sejam mapeadas para uma consulta estruturada – em uma linguagem como SQL (*Structured Query Language*), por exemplo –, que será interpretada por um gerenciador de bases de dados. As respostas são normalmente apresentadas na forma de lista de valores, tal como são obtidas da fonte de dados, ou geradas a partir de *templates*. Em geral, sistemas que implementam esse paradigma funcionam como Interfaces em Língua Natural (ILN) para consulta a bases de dados estruturados.

De fato, os primeiros sistemas de QA eram ILNs. O sistema BASEBALL (GREEN *et al.*, 1961) respondia perguntas feitas em inglês sobre times, locais de jogos e pontuações de partidas de *baseball*, a partir de dados armazenados como listas estruturadas. O programa contava com um dicionário, onde palavras eram associadas a possíveis funções que pudes-

⁸ Define-se *inferência* como o processo de tornar explícita uma informação conhecida implicitamente (BLACKBURN; BOS, 2005).

⁹ Pode ser acessado em: <<http://xisque.di.fc.ul.pt/>>

¹⁰ Do inglês, *Knowledge Base* (KB).

sem assumir em uma pergunta. Dessa maneira, uma frase em inglês podia ser traduzida para uma consulta estruturada. Outro trabalho pioneiro foi o sistema LUNAR (WOODS; KAPLAN; NASH-WEBBER, 1972), que também era uma ILN, usada para consultar dados sobre análises de rochas lunares, coletadas durante as missões Apollo.

Um modelo de QA baseado em KBs é discutido em Wang (2011). Trata-se, também, de uma ILN para informações sobre serviços de uma instituição bancária. O modelo conta com uma coleção de padrões de perguntas pré-determinados. Uma ontologia de domínio é usada para identificar entidades em perguntas – **poupança** é um tipo de **conta**, por exemplo –, permitindo o preenchimento das lacunas correspondentes em uma consulta SQL, associada ao padrão de pergunta identificado. O domínio restrito facilita o desenvolvimento desse tipo de modelo, pois viabiliza a compilação de léxicos especializados e a enumeração de padrões de perguntas, que possam ser realizadas por um usuário.

No início deste projeto de pesquisa, técnicas muito semelhantes foram usadas no desenvolvimento de um protótipo de interface para a agenda de defesas do ICMC-USP¹¹. O protótipo LangUI, como foi chamado, é ilustrado na Figura 3.

Figura 3 – Protótipo de ILN para agenda de defesas do ICMC-USP

LangUI

Quando o aluno do Prof. Bergamasco vai defender?

[ajuda](#)

No dia 25/07/2014 às 11:00.

Defesa de doutorado
25/07/2014 11:00 — Sala 3002
Giuliano Angelo Zugliani
Adalberto Panobianco Bergamasco
Global solvability of systems on compact surfaces

Fonte: Elaborada pelo autor.

Assim como no modelo de Wang (2011), no protótipo LangUI, perguntas em língua natural são mapeadas para consultas SQL, por meio de um léxico de domínio, preparado

¹¹ Instituto de Ciências Matemáticas e de Computação (ICMC), da Universidade de São Paulo (USP).

manualmente. Não há o emprego de ontologias. Em vez disso, transdutores de estados finitos (JURAFSKY; MARTIN, 2009) identificam padrões sintáticos, extraindo informações que são usadas no preenchimento de lacunas da consulta. Respostas são geradas por meio de modelos (*templates*), selecionados a partir do foco da pergunta, também identificado por transdutores. Informações complementares sobre o foco da pergunta são exibidos em forma estruturada, como se observa na Figura 3.

Um protótipo de sistema de QA interativo, também ambientado no ICMC-USP, foi apresentado por Wantroba e Romero (2015). Usando uma base de dados ontológica, esse protótipo simula uma recepcionista, capaz de responder perguntas sobre professores, funcionários e eventos. Além de perguntas na forma textual, o protótipo também pode receber entradas por voz, nos idiomas inglês e português, que são transcritas e passadas a um módulo de processamento de língua natural. O sistema COMUNICA (WILKENS *et al.*, 2010), capaz de responder perguntas, em português, sobre municípios gaúchos, também recebe perguntas por reconhecimento de voz. O modelo implementado emprega técnicas semelhantes a outros descritos anteriormente, como associar padrões de perguntas a consultas em linguagem SQL, e identificar entidades por meio de ontologias.

Sistemas de QA tem sido usados como ferramenta para exploração de bases de dados ontológicos disponíveis na *web*. Grandes bases de conhecimento em forma estruturada, como a DBPedia (LEHMANN *et al.*, 2015), são interligadas em uma rede de Dados Conectados (*Linked Data*) (BIZER; HEATH; BERNERS-LEE, 2009), em que entidades contidas em diferentes bases são interligadas por meio de *links* da *web*. Tecnicamente, o termo Dados Conectados (*Linked Data*) se refere a um conjunto de padrões abertos, para estruturação e consulta de dados publicados na *web*. O padrão RDF (*Resource Description Framework*) define a forma de representação de dados, como grafos estruturados por meio de triplas do tipo <sujeito, predicado, objeto> (por exemplo, <São Carlos, altitude, 800m>), enquanto consultas a essas estruturas são realizadas por meio da linguagem SPARQL (BIZER; HEATH; BERNERS-LEE, 2009).

Sistemas de QA apresentam uma interface intuitiva para a exploração desse vasto conhecimento, enquanto escondem do usuário toda a complexidade dos modelos de dados e das linguagens de consulta. Nessa abordagem, chamada de QA sobre Dados Conectados (QALD¹²), o principal desafio é traduzir uma necessidade de informação, expressa por um usuário em língua natural, para uma consulta que possa ser processada pelos mecanismos de busca e inferência, capazes de processar esses dados estruturados (LOPEZ *et al.*, 2013).

O PowerAqua (LOPEZ *et al.*, 2012) é um sistema de QA de domínio aberto, que implementa a abordagem QALD, e é capaz de combinar dados de múltiplas fontes para responder perguntas em língua natural, como, por exemplo:

¹² Question Answering over Linked Data (QALD)

(2.5) mostre todas as cidades da Espanha

(2.6) Encontre atores ingleses que atuaram em Titanic.

O sistema apresenta respostas na forma de lista de entidades, tais como são obtidas das fontes de dados. Segundo os autores do trabalho, os maiores desafios encontrados no desenvolvimento desse tipo de sistema, além de realizar o mapeamento de perguntas em língua natural para consultas na linguagem SPARQL, são encontrar ontologias contendo informações relevantes e combinar dados complementares obtidos de múltiplas fontes.

Uma revisão bastante completa sobre ILNs para bancos de dados pode ser encontrada em (ANDROUTSOPOULOS; RITCHIE; THANISCH, 1995).

2.3.3 Modelos baseados em reúso

A disponibilidade de conteúdo gerado por usuários em *sites* colaborativos de perguntas e respostas, tais como Yahoo! Respostas e StackOverflow¹³, tem impulsionado o avanço de modelos de QA baseados em reúso. Nesse paradigma de QA, o objetivo é encontrar uma resposta, para uma dada pergunta, em uma coleção de perguntas e respostas, como as que são arquivadas diariamente em *sites* colaborativos e também em fóruns.

Para a implementação do reúso, duas estratégias de identificação de respostas são empregadas: (i) de pergunta para pergunta (*Q-to-Q*, *Question-to-Question*) e (ii) de pergunta para resposta (*Q-to-A*, *Question-to-Answer*). No caso de *Q-to-Q*, busca-se identificar uma pergunta semanticamente equivalente, assumindo-se que sua resposta também será adequada para a nova pergunta, enquanto em *Q-to-A*, a busca é feita diretamente no espaço de respostas, avaliando a adequação de cada resposta candidata à pergunta recebida.

Uma implementação da estratégia *Q-to-Q* é apresentada por Santos *et al.* (2015). Nesse trabalho, a tarefa de se recuperar perguntas semanticamente equivalentes foi implementada na forma de identificação de perguntas marcadas como duplicadas nos *sites* AskUbuntu¹⁴ e English¹⁵. Foi proposta uma arquitetura de Rede Neural Profunda (DNN)¹⁶ (GOODFELLOW; BENGIO; COURVILLE, 2016; NIELSEN, 2015), chamada BOW-CNN, que combina representações do tipo *bag of words* (BOW) com vetores de palavras criados por uma Rede Neural Convolutiva (CNN)¹⁷ (LECUN *et al.*, 1990). O

¹³ E outros da família StackExchange (<<https://stackexchange.com/sites>>).

¹⁴ AskUbuntu: <<https://askubuntu.com/>>

¹⁵ English: <<https://english.stackexchange.com/>>

¹⁶ Do inglês, *Deep Neural Network* (DNN).

¹⁷ Do inglês, *Convolutional Neural Network* (CNN).

modelo atribui pontuações (*scores*) de similaridade a pares de perguntas por meio da combinação linear:

$$s(q_1, q_2) = \beta_1 \times s_{bow}(q_1, q_2) + \beta_2 \times s_{conv}(q_1, q_2) \quad (2.7)$$

onde (q_1, q_2) é um par de perguntas, β_1 e β_2 são pesos e s_{bow} e s_{conv} são pontuações parciais, geradas a partir de uma representação *bag of words* e de uma representação gerada pela rede neural convolucional, respectivamente. De acordo com os autores, o desafio nesse tipo de cenário é a variedade linguística: usuários diferentes formulam a mesma pergunta em uma variedade de formas, usando diferentes estruturas e vocabulários.

Em outro trabalho que implementa a estratégia *Q-to-Q*, Lei *et al.* (2016) também usaram dados do *site* AskUbuntu na tarefa de identificação de perguntas semanticamente equivalentes, como as que são apresentadas no Quadro 4. Os autores apontam duas características observadas nos dados que dificultaram a tarefa: (i) escassez de anotações sobre similaridade entre perguntas e (ii) excesso de detalhes, que ofuscam informações importantes no corpo das perguntas. O segundo problema foi abordado pelo emprego de autocodificadores (*autoencoders*) (VINCENT *et al.*, 2008), treinados para gerar representações dos títulos de cada pergunta a partir de todos os pares de título e corpo. Dessa maneira, os autocodificadores funcionaram como removedores de ruído, gerando representações sucintas dos títulos a partir de informações também presentes no corpo da pergunta. A similaridade entre essas novas representações foi medida por meio da distância cosseno, e então usada para identificar perguntas semanticamente equivalentes.

Quadro 4 – Perguntas similares, extraídas do *site* AskUbuntu

Título: Como dar boot no Ubuntu pela USB?

Corpo: Eu comprei um pc Compaq com Windows 8 alguns meses atrás e agora eu quero instalar o Ubuntu mas ainda manter o Windows 8. Eu tentei o Webi, mas quando meu pc reinicia ele mostra ERROR 0x000007b. (...)

Título: Quando eu tento instalar o Ubuntu no meu laptop eu tenho que apagar todos os meus dados. A opção “Junto com o Windows” não aparece

Corpo: Eu quero instalar o Ubuntu usando um drive Usb. Ele diz que eu tenho que apagar todos os meus dados mas eu quero instalar junto com o Windows 8. A opção “Instalar com o Windows” não aparece. (...)

Fonte: Adaptada de Lei *et al.* (2016).

A estratégia *Q-to-A* tem sido foco de uma avaliação conjunta realizada nos últimos anos no *workshop* SemEval¹⁸. A tarefa, chamada *Community Question Answering*

¹⁸ O SemEval (<http://alt.qcri.org/semeval2017/>) é um *workshop* destinado à avaliação de

(CQA) (NAKOV *et al.*, 2017), é descrita como: “dada uma nova pergunta e uma grande coleção de perguntas e respostas, criada por uma comunidade de usuários, deve-se ordenar por relevância as respostas que possam ser úteis para a nova pergunta”. Em cenários como esse, a ordem de relevância normalmente é definida em termos de uma pontuação (*score*) atribuída a cada resposta candidata, ou em termos de níveis discretos de relevância, como $\langle \text{bom}, \text{útil}, \text{ruim} \rangle$.

A ordem de relevância definida por níveis discretos permite abordar o problema de reuso de respostas como um caso de classificação multi-classe. Essa abordagem foi usada por vários participantes do SemEval 2015, uma vez que a classificação nos níveis $\langle \text{bom}, \text{útil}, \text{ruim} \rangle$ era requerida pela tarefa CQA. Em dois desses trabalhos (TRAN *et al.*, 2015; BELINKOV *et al.*, 2015), diversas *features* tradicionalmente usadas em problemas de classificação de textos, como presença de determinados itens léxicos e autoria da resposta, foram combinadas com outras, extraídas de vetores de palavras¹⁹, gerados pela ferramenta *word2vec* (MIKOLOV *et al.*, 2013a), para alimentar classificadores SVM.

Apesar de permitir o emprego de classificadores, a adoção de níveis discretos de relevância não é absolutamente necessária para a implementação da estratégia *Q-to-A*, tratando-se apenas de uma opção de implementação. Há diversos trabalhos (WANG; MI; ITTYCHERIAH, 2016; SANTOS *et al.*, 2016; SEVERYN; MOSCHITTI, 2015) em que essa estratégia é implementada pela computação de uma pontuação para cada resposta.

Pode-se dizer que modelos que implementam o paradigma de reuso tem como objetivo quantificar a adequação de uma resposta candidata a uma pergunta, de maneira que tal medida possa ser usada para identificar, em uma coleção, uma resposta que possa atender às necessidades de informação de uma nova pergunta. A proposta desta pesquisa é norteadada por esse paradigma de QA.

2.3.4 Outros paradigmas

Além dos paradigmas apresentados nas subseções anteriores, há ainda outras formas de implementação, e modelos de QA.

O sistema IBM Watson (FERRUCCI *et al.*, 2010), que ganhou notoriedade pelo seu desempenho em uma competição de conhecimentos gerais (*quiz*), promovida por um programa de televisão, implementa um modelo híbrido de QA, baseado tanto em recuperação de informação quanto em bases de conhecimento estruturado.

O problema de se responder perguntas do tipo *quiz* também foi abordado em (IYYER *et al.*, 2014). Nesse trabalho, descreve-se um modelo de QA treinado para responder fac-

modelos semânticos computacionais.

¹⁹ Uma discussão detalhada sobre vetores de palavras (*word vectors*, ou ainda, *word embeddings*) será apresentada no Capítulo 3.

toides, relacionando parágrafos de texto a nomes de entidades: cenário típico nesse tipo de competição. Aproveitando-se do número limitado de respostas possíveis, os autores modelaram o problema como um caso de classificação multi-classe, em que uma rede neural recorrente, do tipo DT-RNN (SOCHER *et al.*, 2014), foi usada para identificar o nome de entidade (rótulo) com maior probabilidade de estar associado a um dado parágrafo de entrada.

Recentemente, foi proposta por Antol *et al.* (2015) a tarefa de se responder perguntas sobre imagens, chamada de VQA²⁰. Trata-se de uma tarefa bastante desafiadora, que requer a combinação de técnicas de processamento de línguas naturais com técnicas de visão computacional, para responder perguntas que também envolvem frequentemente conhecimento de senso comum.

2.4 Tarefa de seleção de respostas

A tarefa de Seleção de Respostas (AS)²¹ é executada ao final do processo de responder perguntas, e consiste em encontrar a melhor resposta para uma dada pergunta, em um conjunto pré-selecionado de respostas candidatas. Ou seja, assume-se a existência de um conjunto de respostas candidatas, contendo pelo menos uma resposta correta para uma dada pergunta, e assim, estuda-se isoladamente esse problema que é fundamental em QA.

A tarefa de seleção de respostas (AS) é descrita em diversos trabalhos (WANG; SMITH; MITAMURA, 2007; YANG; YIH; MEEK, 2015; SANTOS *et al.*, 2016; WANG; MI; ITTYCHERIAH, 2016; SEVERYN; MOSCHITTI, 2015; FENG *et al.*, 2015, e outros) e pode ser definida mais formalmente como se segue.

Definição 1 (Seleção de Respostas). Dada uma pergunta q e um conjunto de respostas candidatas $P_q = \{a_1, a_2, \dots, a_n\}$, o objetivo da tarefa de seleção de respostas é encontrar uma resposta $a_i \in G_q$, em que $G_q \subset P_q$ é o conjunto de respostas que respondem corretamente a pergunta q , com $1 \leq |G_q| < n$.

Trata-se de uma tarefa com aplicações em múltiplos modelos de QA. Em situações práticas, um cenário envolvendo uma pergunta e um conjunto de resposta candidatas pode ser encontrado, por exemplo, em modelos baseados em recuperação de informação, quando um componente de RI retorna documentos relacionados à uma pergunta, ou ainda em uma situação de reuso, quando se tem um conjunto de respostas, a partir do qual se deseja responder uma nova pergunta. Em situações de pesquisa, e de desenvolvimento de novos modelos, tal cenário é definido por *datasets* (ver Subseção 2.4.1).

²⁰ Do inglês, *Visual Question Answering* (VQA). Informações sobre a tarefa, assim como um *dataset*, estão disponíveis em: <<http://www.visualqa.org/>>

²¹ Do inglês, *Answer Selection* (AS).

A granularidade de respostas identificadas por um modelo de AS depende do tipo de pergunta, podendo variar de expressões curtas para factoides até frases ou parágrafos para perguntas mais complexas (WANG; SMITH; MITAMURA, 2007).

No Capítulo 5 deste trabalho, apresenta-se um modelo de seleção de respostas, chamado SlimRank, que foi inspirado principalmente no paradigma de reúso.

2.4.1 Datasets

Nesta seção, são apresentados alguns *datasets* de QA particularmente úteis para a tarefa de seleção de respostas.

Grande parte dos *datasets* de QA, principalmente os maiores, são formados por perguntas bastante simples, formuladas artificialmente ou derivadas de consultas submetidas a mecanismos de busca. O SelQA (JURCZYK; ZHAI; CHOI, 2016) foi introduzido como um *dataset* de *benchmark* para a tarefa de seleção de respostas, e contém aproximadamente 8 mil perguntas formuladas por trabalhadores de *crowdsourcing*, a partir de segmentos de texto extraídos da Wikipédia.

Serviços do tipo *crowdsourcing* tem sido empregados em diversos trabalhos envolvendo preparação e anotação de dados. Um desses serviços é o *Amazon Mechanical Turk*²², que distribui grandes quantidades de pequenas tarefas, chamadas *HITS*, entre diversos indivíduos remunerados por produtividade. Além dos dados, o contratante deve fornecer uma especificação, descrevendo a tarefa que deverá ser executada por cada indivíduo, como, por exemplo, dadas duas frases, classificá-las como **paráfrases**, **relacionadas** ou **não-relacionadas**.

O *dataset* SelQA foi criado a partir de segmentos de textos extraídos de 486 artigos da Wikipédia. Um a um, cada segmento foi apresentado a anotadores instruídos a formular uma pergunta que pudesse ser respondida por aquele segmento de texto. Alguns exemplos de perguntas criadas conforme esse processo são apresentados no Quadro 5.

O mesmo processo foi usado na construção do *dataset* SQuAD (RAJPURKAR *et al.*, 2016), que é bem maior, contendo 100 mil perguntas. Apesar de esse método viabilizar a criação de grandes *datasets*, observa-se que perguntas criadas a partir de textos previamente conhecidos são geralmente mais simples do que perguntas formuladas naturalmente, e que também compartilham muitas palavras com suas respectivas respostas. A quantidade significativa de palavras compartilhadas entre perguntas e respostas pode tornar a tarefa de seleção de respostas mais fácil e acabar elevando artificialmente o desempenho dos sistemas (YANG; YIH; MEEK, 2015).

Uma alternativa para evitar perguntas artificiais é coletar consultas submetidas a mecanismos de buscas. O WikiQA (YANG; YIH; MEEK, 2015) é um importante *dataset*

²² *Amazon Mechanical Turk*: <<https://www.mturk.com/>>

Quadro 5 – Perguntas formuladas para o *dataset* SelQA

Tópico: TV, **Artigo:** Série *Criminal Minds*

O episódio de estréia recebeu revisões diversas, ficando com 42 de 100 pontos no *site* de revisões Metacritic. Dorothy Rabinowitz, em sua revisão para o *Wall Street Journal*, disse que “pelas evidências dos primeiros episódios, *Criminal Minds* deve ser um sucesso, merecidamente”. Enquanto o *New York Times* foi bem menos positivo, dizendo que “o problema com *Criminal Minds* são as muitas e confusas personagens”, que podem fazer com que “a audiência – e talvez também os criadores – fique rapidamente entediada”.

Perguntas:

- (a) Como foram as revisões do episódio de estréia?
- (b) Quem achou que a audiência pode ficar entediada rapidamente?
- (c) Como foram as primeiras revisões?

Fonte: Adaptada de Jurczyk, Zhai e Choi (2016).

formado por perguntas naturais submetidas por usuários do Bing²³. Para cada pergunta há um conjunto de respostas candidatas (frases) extraídas de artigos da Wikipédia, onde as respostas corretas foram identificadas por anotadores humanos. Parte das perguntas contidas nesse *dataset* não tem resposta correta em seus respectivos conjuntos de candidatas. Esses exemplos são úteis para a tarefa de eliciação de respostas²⁴, em que o objetivo é identificar a falta de respostas corretas no conjunto de respostas candidatas de uma dada pergunta. Alguns exemplos de perguntas do *dataset* WikiQA são apresentados no Quadro 6.

Quadro 6 – Perguntas contidas no *dataset* WikiQA

- (1) Onde são encontrados os pandas gigantes?
- (2) Como funciona um motor a vapor?
- (3) Qual é a profundidade do Oceano Atlântico?

Fonte: Extraídas (e adaptadas) do *dataset* WikiQA (YANG; YIH; MEEK, 2015)

O método de coletar perguntas submetidas a mecanismos de busca também foi usado na construção do *dataset* MS MARCO (NGUYEN *et al.*, 2016), que pode ser considerado uma versão maior do WikiQA. O MS MARCO contém 100 mil perguntas do

²³ Bing: <<http://www.bing.com/>>

²⁴ Tradução livre de “*answer triggering*”.

tipo factóide, assim como perguntas sem resposta correta. Contudo, diferentemente do WikiQA, no MS MARCO não se restringiu a artigos da Wikipédia para a extração de respostas. Em vez disso, as respostas foram coletadas de milhares de documentos da *web*.

Perguntas naturais também são encontradas no *dataset* Q2AD²⁵ (AGHAEBRAHIMIAN, 2017), que é composto por 300 perguntas extraídas do *site* Quora²⁶. Nesse *dataset*, as perguntas são apresentadas como frases interrogativas diretas, contendo 10 palavras, em média. Cada pergunta é acompanhada por uma passagem de texto, contendo frases e expressões curtas que a respondem, que foram identificadas por anotadores humanos.

Outro importante *dataset* é o TREC-QA (WANG; SMITH; MITAMURA, 2007), também chamado de QASent. Esse, atualmente, é o *dataset* de referência usado em *benchmarks* da tarefa de seleção de respostas. O TREC-QA é um *dataset* pequeno, formado por 227 perguntas, selecionadas dos dados usados na trilha de QA da *Text REtrieval Conference* (TREC) (VOORHEES, 2003). Nos dados originais da TREC-QA, cada pergunta conta com uma coleção de documentos onde devem ser buscadas suas respostas. Para formar os conjuntos de respostas candidatas, os autores do *dataset* TREC-QA selecionaram frases contidas nesses documentos onde fossem encontradas uma ou mais palavras (excluídas as *stopwords*) da pergunta. Portanto, o *dataset* TREC-QA também apresenta significativa sobreposição léxica entre perguntas e respostas candidatas, que pode induzir melhores resultados em modelos baseados nessa característica.

Finalmente, o InsuranceQA (FENG *et al.*, 2015) é um *dataset* do domínio de seguros, criado a partir de dados obtidos de um *site*²⁷ da Internet, onde especialistas em seguros respondem perguntas recebidas de usuários. As respostas contidas no InsuranceQA são explicações detalhadas, mais longas do que respostas tipicamente encontradas em outros *datasets*. Porém, as perguntas são limitadas pelo *site* de origem a frases únicas, o que as faz curtas e objetivas, formadas por poucas palavras. Um exemplo de pergunta do *dataset* InsuranceQA é apresentado no Quadro 7.

Os número de *datasets* de QA disponíveis para o português é bem menor. O *dataset* Págico (SIMÕES; COSTA; MOTA, 2012) foi criado para uma avaliação conjunta sobre recuperação de informação, em língua portuguesa. Esse *dataset* contém apenas 153 perguntas do tipo factóide, elaboradas pelos organizadores da avaliação, que não são particularmente úteis para a aplicação de técnicas baseadas na quantificação da adequação entre pergunta e resposta, tipicamente usadas na tarefa de seleção de respostas. Sabe-se ainda que um conjunto de 200 perguntas, em português, foi usado na avaliação conjunta QA@CLEF (FORNER *et al.*, 2009). No entanto, essas perguntas não se encontram disponíveis publicamente.

²⁵ Quora Question Answering Dataset (Q2AD)

²⁶ Quora: <<http://www.quora.com/>>

²⁷ Insurance Library: <<http://www.insurancelibrary.com/>>

Quadro 7 – Exemplo de pergunta encontrada no *dataset* InsuranceQA

Ter um *pitbull* aumenta o preço do seguro residencial?

A maioria das companhias de seguro não cobra sobretaxas por animais. Em vez disso, elas excluem certas raças ou restringem a cobertura do seguro para outras, ou mesmo se negam fazer a apólice. Mordidas de cachorro são um grande problema para companhias de seguro, e é mais provável que elas restrinjam a cobertura ou rejeitem a proposta completamente, dependendo da raça. Seria melhor você comprar um animal seguindo as políticas de cobertura desde o início.

Fonte: Extraída (e adaptada) do *dataset* InsuranceQA (FENG *et al.*, 2015)

Nesta seção, foram apresentados *datasets* relacionados à tarefa de seleção de respostas. Dados estatísticos sobre os principais desses *datasets*, como tamanho médio de perguntas, serão apresentados no Capítulo 4, juntamente com os dados sobre o *dataset* MilkQA, criado durante este projeto de pesquisa.

2.4.2 Método de avaliação

Para a avaliação de modelos de seleção de respostas, são usados conjuntos de teste, contendo perguntas acompanhadas por seus respectivos conjuntos de respostas candidatas, os quais contém uma ou mais respostas corretas, previamente identificadas por julgadores humanos. Em termos mais formais, em um conjunto de testes $\mathcal{Q}$, cada pergunta $q \in \mathcal{Q}$ está relacionada a um conjunto de respostas candidatas $P_q = \{a_1, a_2, \dots, a_n\}$ e a um conjunto de respostas corretas $G_q \subset P_q$, com $1 \leq |G_q| < n$. Os conjuntos de teste são, usualmente, parte dos *datasets* de seleção de respostas.

O desempenho de modelos de seleção é quantificado por meio de métricas que medem sua capacidade de identificar, entre as respostas candidatas, aquelas que foram previamente marcadas como corretas, ou seja, as respostas contidas nos conjuntos G_q de cada pergunta (*ground truth sets*).

Uma métrica comum na área de Recuperação de Informação é a precisão, adaptada, a seguir, para o contexto de QA:

$$\text{Precisao} = \frac{\#(\text{respostas corretas})}{\#(\text{respostas recuperadas})} \quad (2.8)$$

A precisão representa a fração de respostas corretas contidas em um conjunto de possíveis respostas geradas por um modelo. Considere-se, por exemplo, que um modelo tenha apresentado 5 respostas a um usuário, das quais 2 estão corretas. A precisão alcançada pelo modelo, nesse caso, seria de $2/5 = 0.4$.

Como se pode verificar, a precisão é computada sobre conjuntos não ordenados. No entanto, na tarefa de seleção de respostas, deseja-se que respostas potencialmente corretas sejam apresentadas primeiro. Assim, é necessário considerar métricas que possam medir a capacidade que um modelo tem de ordenar respostas por relevância (*ranking*). Três dessas métricas são as mais usadas: *Precision@k* ($P@k$), *Mean Average Precision* (MAP) e *Mean Reciprocal Rank* (MRR).

A $P@k$ é a precisão computada apenas nos primeiros k resultados:

$$P@k = \frac{\#(\text{corretas nos primeiros } k)}{k} \quad (2.9)$$

O desempenho geral de um modelo é representado pela média aritmética da $P@k$, computada sobre todas as perguntas do conjunto de testes $\mathcal{Q}$:

$$P@k_{\text{média}} = \frac{1}{|\mathcal{Q}|} \sum_{i=1}^{|\mathcal{Q}|} P@k(q_i) \quad (2.10)$$

Contudo, a notação $P@k$ é frequentemente usada para se referir à média, ficando a desambiguação a cargo do contexto. Um caso particularmente interessante, que ocorre com bastante frequência, é a média da $P@k$ quando $k = 1$, indicada por $P@1$, que representa a taxa de respostas corretas apresentadas em primeira posição. Ou, de maneira mais simples, representa a taxa de acerto alcançada pelo modelo. Um modelo que apresente $P@1 = 0.70$ acerta 70% das perguntas.

A MAP é computada em função da *Average Precision* (AP), que é a média do valor da precisão obtido após o retorno de cada resposta correta. Seja $G_q = \{a_1, a_2, \dots, a_m\}$ o conjunto de respostas corretas da pergunta q e R_k uma lista ordenada de resultados, observada no momento em que a resposta correta $a_k \in G_q$ é retornada pelo modelo, define-se a AP como:

$$AP(q) = \frac{1}{|G_q|} \sum_{k=1}^{|G_q|} \text{Precisao}(R_k) \quad (2.11)$$

Define-se, então, a MAP, como a média aritmética da AP de todas as perguntas contidas no conjunto de testes $\mathcal{Q}$:

$$MAP(\mathcal{Q}) = \frac{1}{|\mathcal{Q}|} \sum_{i=1}^{|\mathcal{Q}|} AP(q_i) \quad (2.12)$$

A MAP é representada por valor numérico no intervalo $[0, 1]$, que indica o desempenho geral do modelo no conjunto de testes. Quanto mais próximo de 1, melhor o desempenho do sistema.

A MRR também é computada a partir de uma lista ordenada de resultados, conforme a expressão:

$$MRR = \frac{1}{|\mathcal{Q}|} \sum_{i=1}^{|\mathcal{Q}|} \frac{1}{\text{rank}_i} \quad (2.13)$$

onde rank_i representa a posição da primeira resposta correta para a pergunta q_i . A MRR representa a média do inverso da posição em que ocorre a primeira resposta correta para cada pergunta do conjunto de testes.

A ferramenta padrão para o cálculo das métricas MAP e MRR é a `trec_eval`²⁸, desenvolvida para medir o desempenho dos sistemas participantes das avaliações conjuntas promovidas na TREC. A exemplo de muitos trabalhos, a ferramenta `trec_eval` também é adotada neste trabalho para computar as métricas MAP e MRR, enquanto a métrica P@k é computada por uma ferramenta desenvolvida neste projeto.

Finalmente, pode-se encontrar uma explicação detalhada e abrangente sobre a avaliação de resultados ordenados (*ranked results*) em Manning, Raghavan e Schütze (2008, cap.8), que foi principal fonte de referência usada nesta seção.

2.4.3 Estado da arte

O histórico do estado da arte para a tarefa de seleção de respostas tem sido registrado em uma página do tipo *wiki* (ACL, 2017), mantida pela *Association for Computational Linguistics* (ACL). O estado da arte atual da tarefa, segundo essa fonte, é apresentado na Tabela 1.

Tabela 1 – Estado da arte para seleção de respostas

Modelo	Referência	MAP	MRR
LDC	Wang, Mi e Ittycheriah (2016)	0.771	0.845
Pairwise rank + MP CNN	Rao, He e Lin (2016)	0.801	0.877
BiMPM	Wang, Hamza e Florian (2017)	0.802	0.875

Fonte: ACL (2017).

O *dataset* de referência, usado para medir o desempenho dos sistemas que constam nesse histórico, é o TREC-QA (WANG; SMITH; MITAMURA, 2007). Apesar de ser considerado o *benchmark* padrão da tarefa, o TREC-QA é um *dataset* pequeno (ver Subseção 2.4.1). Acreditamos ser essa a razão de, em vários trabalhos, também serem encontradas avaliações realizadas no *dataset* WikiQA, que é maior e mais atual.

2.5 Perguntas de consumidores

Grande parte das pesquisas sobre modelos de seleção de respostas – e sobre QA de modo geral – tem sido desenvolvida sobre perguntas bem formadas e objetivas, apresentadas na forma de frases interrogativas isoladas. Pouca atenção tem sido dada a perguntas

²⁸ Disponível em: <http://trec.nist.gov/trec_eval/>

mais complexas, como as que, neste trabalho, chamamos de *Perguntas de Consumidores*. Trata-se de uma forma particular de pergunta, que normalmente ocorre em *sites* de perguntas e respostas, fóruns e Serviços de Atendimento a Consumidores (SACs).

Quadro 8 – Exemplo de pergunta de consumidor

Na semana passada eu tive uma dor no peito e meu coração deve ter batido fora do ritmo. Eu respirei fundo algumas vezes e então me senti aliviado. Nessa semana eu sinto sob minha caixa torácica esquerda como se meu estômago estivesse constantemente digerindo. *Eu devo me preocupar ou isso pode ser indigestão*. Eu tenho 29 anos e não comi nada de diferente.

Fonte: Adaptada de Zhang (2010).

Nota: Durante o processo de tradução, buscou-se reproduzir os problemas de pontuação observados no exemplo original.

Perguntas de consumidores diferem de perguntas diretas – como as do tipo factóide, por exemplo – principalmente pela sua estrutura. São perguntas compostas por múltiplas frases inter-relacionadas, principalmente em nível semântico. Uma característica marcante das perguntas de consumidores é a presença de uma descrição de algum problema (contexto), seguida por uma ou mais sub-perguntas. Essas sub-perguntas são curtas e podem parecer incompletas quando vistas isoladamente, uma vez que dependem fortemente de informações apresentadas anteriormente no contexto. Isso pode ser verificado na frase que se encontra em destaque no Quadro 8.

Alguns trabalhos pioneiros trataram especificamente de perguntas de consumidores. Em um trabalho descritivo, em que foram chamadas de perguntas sobre saúde de consumidores (*consumer health questions*), Zhang (2010) identificou características linguísticas de perguntas sobre o tópico *saúde*, submetidas ao site Yahoo! Respostas²⁹. De acordo com suas observações, essas perguntas são normalmente mais longas do que o habitual; contêm múltiplas perguntas (sub-perguntas); não são necessariamente bem formadas, ou explícitas; costumam apresentar erros ortográficos e gramaticais; e apresentam termos leigos, como “remédio para dor”, em vez de “analgésico”.

Em outro trabalho em que foram estudadas perguntas sobre saúde (KILICOGLU; FISZMAN; DEMNER-FUSHMAN, 2013), pesquisadores da Biblioteca Nacional de Medicina dos Estados Unidos (NLM)³⁰ buscaram desenvolver um método para interpretar perguntas de consumidores automaticamente. O método foi criado para ser aplicado diretamente no contexto da NLM, onde é mantido um serviço de atendimento ao público, destinado a responder perguntas sobre saúde. Esses autores propuseram uma abordagem de interpretação – idealizada a partir da análise de perguntas frequentemente recebi-

²⁹ Yahoo! Respostas: <<https://answers.yahoo.com/>>

³⁰ U.S. National Library of Medicine (NLM): <<https://www.nlm.nih.gov>>

das pela instituição – baseada na classificação de perguntas e na estruturação de dados. Primeiro, perguntas sobre doenças são classificadas em três classes: (i) tratamento, (ii) diagnóstico e (iii) prognóstico. Para cada classe, há uma estrutura de dados pré-definida (*frame*), que é preenchida por meio da aplicação de regras. Um exemplo de uma pergunta da classe *tratamento*, acompanhada da respectiva estrutura de dados, é apresentado no [Quadro 9](#). A pesquisa foi realizada sobre dados da própria NLM que, até onde se sabe, não foram disponibilizados publicamente.

Quadro 9 – Interpretação estruturada para pergunta de consumidor

Pergunta:

Eu nasci c/ atresia do esôfago e c/ dextrocardia. O coração não causa problema, mas a outra sim. Eu fico com a comida entalada o tempo todo. Minha pergunta é... *existe alguma coisa que possa consertar isso porque ultimamente eu não consigo comer nada sem ficar entalado*. Eu preciso de ajuda ou vou morrer de fome!

Frame:

Tratamento: *consertar*

Tema: Atresia do esôfago (Doença ou Síndrome)

Sinal (*cue*): *existe*

Fonte: Adaptada de [Kilicoglu, Fiszman e Demner-Fushman \(2013\)](#).

Nota: Foram reproduzidas características observadas na pergunta original, como abreviações, pontuação e informalidade.

Nesse trabalho ainda foram abordados dois fenômenos linguísticos frequentes em perguntas de consumidores: anáforas e elipses³¹. Assim como no preenchimento das estruturas de dados, no tratamento desses dois fenômenos linguísticos também foram usadas regras baseadas em informações léxicas, sintáticas e em conhecimento de domínio. Segundo uma dessas regras, por exemplo, deve-se adotar a primeira entidade nomeada que ocorrer em uma pergunta como antecedente de anáforas e elipses.

No método de interpretação de perguntas de consumidores, descrito no trabalho de [Kilicoglu, Fiszman e Demner-Fushman \(2013\)](#), problemas complexos puderam ser resolvidos de maneira simples, graças à restrição de domínio e à abordagem prática do problema, cujo objetivo principal era automatizar processos de atendimento ao público na instituição. A resolução de elipses que, em particular, está longe de ser um problema trivial na área de processamento de línguas naturais, pôde ser feita por meio de uma regra simples, derivada da observação de que, frequentemente, o nome da doença da qual trata a pergunta é a primeira entidade nomeada encontrada no texto. Apesar de ter se mostrado eficaz no contexto em que foi desenvolvido, esse método não poderia ser facilmente

³¹ Anáforas e elipses são mecanismos de coesão textual. Uma boa revisão sobre esse tema pode ser encontrada em [Halliday e Hasan \(1976\)](#).

aplicado em outras situações, que envolvessem outros domínios e objetivos. Mais ainda, o uso de ferramentas como *parsers* e reconhecedores de entidades nomeadas, das quais depende o método, exige que o texto de entrada seja normalizado, o que pode ser bastante trabalhoso no caso de perguntas de consumidores, devido às suas características.

2.6 Considerações finais

Perguntas de consumidores são abundantes em diversos contextos, e acredita-se que devam ser exploradas mais a fundo. A disponibilidade de um *dataset* é fundamental para esse propósito. Apesar de características e exemplos desse tipo de pergunta terem sido apresentadas em ambos os trabalhos pioneiros encontrados na literatura (ZHANG, 2010; KILICOGLU; FISZMAN; DEMNER-FUSHMAN, 2013), nenhum deles apresenta uma definição desse objeto de pesquisa que possa guiar a formação de um *dataset*. Neste trabalho, é proposta uma definição clara para o conceito *perguntas de consumidores*, que é apresentada no Capítulo 4, juntamente com a descrição do *dataset* MilkQA, também criado neste projeto.

A princípio, consideramos neste trabalho confrontar diretamente os aspectos linguísticos das perguntas de consumidores, a exemplo do que fizeram Kilicoglu, Fiszman e Demner-Fushman (2013). No entanto, experimentos iniciais com os dados do *dataset* MilkQA logo revelaram uma série de obstáculos envolvendo essa abordagem. Para ilustrar a situação, podemos citar a dificuldade de realizar a análise sintática desses textos, onde são abundantes os erros ortográficos e de pontuação. Em um desses experimentos, a simples falta do acento agudo na palavra *água* (substantivo) levou o *parser* Palavras (BICK, 2000) a reconhecê-la como o verbo *aguar* (“Ele *agua* as plantas.”), distorcendo completamente o sentido da frase. Procuramos ainda, nessa ocasião, normalizar o conteúdo do *dataset*, desenvolvendo um corretor ortográfico completamente automático, baseado no modelo *noisy channel* (BRILL; MOORE, 2000). Contudo, os resultados foram bastante insatisfatórios.

Diante de resultados promissores alcançados por métodos baseados na semântica distributiva (*distributional semantics*) (LENCI, 2008 apud HARISPE *et al.*, 2015) – implementados na forma de redes neurais profundas – e considerando que modelos dessa natureza tendem a ser mais tolerantes a problemas nos níveis léxico e sintático, decidimos adotar essa abordagem e assim evitar os problemas relatados, e outros ainda mais complexos, como o tratamento de anáforas e elipses, concentrando os esforços de pesquisa em um problema central: selecionar respostas para perguntas de consumidores. Assim, apresentamos no Capítulo 5 um modelo de rede neural profunda para seleção de respostas, adaptado às características de perguntas de consumidores.

A seguir, no Capítulo 3, é apresentada uma revisão sobre modelos neurais e repre-

sentação textual.

MODELOS NEURAIIS E REPRESENTAÇÕES TEXTUAIS

Neste capítulo, são apresentadas representações textuais envolvendo vetores de palavras. Também são apresentadas arquiteturas de redes neurais convolutivas para a tarefa de seleção de respostas, além de métodos de representação textual baseados em grafos. O capítulo termina com a apresentação de um *framework* para o treinamento de redes convolutivas sobre grafos.

3.1 Considerações iniciais

Em processamento de línguas naturais, as palavras são tradicionalmente tratadas como símbolos discretos¹. Essa modelagem é adotada em diversas tarefas fundamentais em processamento de línguas naturais, como estimar a probabilidade de uma palavra ocorrer em certo contexto, por exemplo. Nessa tarefa essencial para ferramentas como corretores ortográficos e reconhecedores de fala, as palavras são normalmente tratadas como símbolos discretos. Considere-se, por exemplo, a frase:

(3.1) O cachorro correu pela...

Usando um modelo estatístico de língua, é possível verificar facilmente que, nesse contexto, a palavra “casa” tem maior probabilidade de ocorrência do que “fechadura”, por exemplo. Modelos estatísticos de língua podem ser vistos como tabelas de probabilidades, em que cada palavra (ou símbolo) X é associada a uma probabilidade P de ocorrência em um dado contexto, isto é, $P(X|\text{contexto})$. No caso do exemplo anterior, a

¹ O conteúdo desta seção foi inspirado na seção de motivação de um tutorial sobre representações vetoriais de palavras, encontrado na documentação do *framework* TensorFlow ([TensorFlow](#), 2017) e também no trabalho de Mikolov, Yih e Zweig (2013).

escolha de uma das palavras teria sido guiada pela expressão $P(\text{casa}|\text{correu},\text{pela}) > P(\text{fechadura}|\text{correu},\text{pela})$, computada unicamente por meio de relações simbólicas.

A modelagem de palavras como símbolos discretos também fica evidente na tradicional representação de textos *Bag of Words* (BOW), em que apenas as ocorrências de cada palavra em um texto são consideradas, desprezando sua posição e relações de vizinhança. A representação BOW é baseada no conceito de *bag*, que é análogo ao conceito de conjunto, com a exceção de que elementos duplicados são permitidos (TURNERY; PANTEL, 2010). A seguir, são apresentados exemplos de frases com suas respectivas representações BOW.

(3.2) O cachorro correu pela casa.

{cachorro, casa, correu, o, pela}

(3.3) O cachorro fugiu para fora da casa.

{cachorro, casa, da, fora, fugiu, o, para}

(3.4) O gato andou lentamente pela sala.

{andou, gato, lentamente, o, pela, sala}

Como se pode observar, na representação BOW, as palavras de um texto são tratadas como elementos de um conjunto (ou *bag*). Na verdade, ao remover elementos duplicados de uma BOW, o que se obtém é exatamente um conjunto. Essa representação é uma escolha padrão em diversas tarefas de processamento de línguas naturais, como a classificação de textos (SEBASTIANI, 2002) e a avaliação de similaridade semântica (HARISPE *et al.*, 2015).

Na maior parte das tarefas, é mais comum que as representações BOW sejam codificadas como vetores de frequências de palavras (ou ainda, vetores tf-idf). No entanto, é possível explorar diretamente a representação BOW para computar um grau de similaridade semântica entre dois textos – tarefa que é particularmente importante para métodos de seleção de respostas. A estratégia, nesse caso, consiste em analisar a sobreposição de palavras (HARISPE *et al.*, 2015), o que pode ser feito representando cada texto como um conjunto de palavras (sem repetição), e então computando o coeficiente de Jaccard (Equação 3.5) entre os dois conjuntos.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}, \quad 0 \leq J(A,B) \leq 1 \quad (3.5)$$

Considere-se, por exemplo, a tarefa de computar o grau de similaridade semântica entre os pares de frases (3.2, 3.3) e (3.2, 3.4), apresentados anteriormente. Desconsiderando as *stopwords* (o, da, para, pela), como se faz tradicionalmente, no primeiro caso tem-se

os conjuntos $A = \{\text{cachorro, casa, correu}\}$ e $B = \{\text{cachorro, casa, fora, fugiu}\}$, e assim:

$$\text{sim}(A, B) = J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{2}{5} = 0,4 \quad (3.6)$$

E no segundo caso, com $C = \{\text{andou, gato, lentamente, sala}\}$:

$$\text{sim}(A, C) = J(A, C) = \frac{|A \cap C|}{|A \cup C|} = \frac{0}{7} = 0,0 \quad (3.7)$$

Em ambos os casos, o grau de similaridade obtido pode ser considerado pessimista. Uma situação extrema fica evidente no segundo caso (Equação 3.7), com $\text{sim}(A, C) = 0,0$. Considerando as situações descritas nas frases 3.2 (A) e 3.4 (C), seria razoável esperar que o grau de similaridade fosse maior do que zero. Em ambas as situações, descreve-se um animal doméstico, mamífero e quadrúpede, se movendo em ambientes similares. No entanto, a representação simbólica de palavras, em si, não permite identificar tais relações de similaridade em nível semântico (BENGIO *et al.*, 2003) sem que se use algum recurso externo, como uma ontologia. Usar diretamente as representações BOW na computação do grau de similaridade apenas torna essa limitação mais visível. Resultados semelhantes seriam observados se as representações BOW correspondentes a cada frase fossem codificadas na forma de vetores de frequência (ou de vetores tf-idf).

Uma alternativa às representações simbólicas é representar palavras como vetores no espaço. Nessa abordagem, conhecida como Modelo de Espaços Vetoriais (VSMs)², as palavras são representadas como pontos em um espaço contínuo (vetores no espaço). Pontos que estão próximos são semanticamente similares, enquanto pontos distantes no espaço são menos relacionados semanticamente (TURNERY; PANTEL, 2010)³. Logo, observa-se que as representações vetoriais, ao contrário das simbólicas, permitem que relações semânticas entre palavras sejam identificadas. Contudo, não se propõe aqui substituir completamente as representações simbólicas pelas vetoriais. Trata-se apenas de destacar o fato de que, dependendo da aplicação, uma pode ser mais conveniente do que a outra.

Representações de palavras como vetores em espaços contínuos são derivadas de análises de grandes corpú. Vários métodos foram desenvolvidos para a geração de vetores, todos fundamentados na *hipótese distributiva*⁴, segundo a qual “palavras que ocorrem em contextos similares tendem a ter sentidos similares” (HARRIS, 1954 apud TURNERY; PANTEL, 2010). De maneira geral, modelos baseados na hipótese distributiva são chamados de Modelos de Semântica Distributiva (DSMs)⁵, e podem ser classificados em (i) modelos baseados em contagem e (ii) modelos preditivos (BARONI; DINU; KRUSZEWSKI, 2014).

² Do inglês, *Vector Space Model* (VSM).

³ Nesse trabalho encontra-se uma ampla revisão sobre VSMs.

⁴ Em inglês, *distributional hypothesis*.

⁵ Do inglês, *Distributional Semantic Models* (DSM).

Nos modelos baseados em contagem, os vetores são derivados da frequência com que uma palavra ocorre em algum contexto, como um documento ou uma janela de palavras. Um exemplo típico de modelo baseado em contagem é o conhecido como Análise de Semântica Latente (LSA) (DEERWESTER *et al.*, 1990). Nesse modelo, a frequência de cada palavra, em cada documento de uma coleção, é registrada em uma matriz, que, após um processamento matemático, dá origem a vetores de palavras com dimensão reduzida.

Nos modelos preditivos, os vetores de palavras são aprendidos durante o processo de treinamento de redes neurais, projetadas para prever a ocorrência de uma palavra a partir de um contexto. O Word2vec (MIKOLOV *et al.*, 2013b) é um exemplo bastante popular dessa categoria de DSM. Em um estudo comparativo entre modelos baseados em contagem e modelos preditivos, foi verificado que vetores derivados de modelos preditivos proporcionaram melhores resultados em diversas tarefas, incluindo a avaliação de similaridade semântica (BARONI; DINU; KRUSZEWSKI, 2014). A seguir, na Seção 3.2, os modelos preditivos são descritos em detalhes.

3.2 Modelos neurais de língua

Representações simbólicas, tradicionalmente usadas em modelos estatísticos de língua, não permitem a identificação de relações entre palavras semelhantes, como foi discutido anteriormente. Essa característica impõe limitações a modelos estatísticos de língua, criados para estimar a probabilidade de ocorrência de sequências de palavras, modeladas como n -gramas. Devido à esparsidade de dados, frequentemente, em situações de uso, ocorrem n -gramas diferentes daqueles observados nos dados de treinamento, exigindo que várias técnicas sejam empregadas para contornar essa limitação dos modelos.

Buscando desenvolver modelos de língua com maior capacidade de generalização, isto é, que pudessem processar sequências de palavras desconhecidas tirando proveito de casos similares, Bengio *et al.* (2003) propuseram que representações de palavras fossem aprendidas por meio de modelos neurais. Assim como ocorre com modelos estatísticos, os modelos de língua neurais são ajustados para aprender uma função de probabilidade conjunta sobre sequências de n palavras, indicada como:

$$P(w_t | w_{t-n+1}^{t-1}) \quad (3.8)$$

onde w_t representa a palavra cuja probabilidade se deseja estimar e w_{t-n+1}^{t-1} representa uma sequência de $n - 1$ palavras que antecede w_t .

No modelo neural proposto por Bengio *et al.* (2003) são aprendidas simultaneamente (i) uma representação para cada palavra e (ii) a função de probabilidade, expressa em termos dessas representações. Cada palavra é representada por um vetor em $\mathbb{R}^m$ – um ponto em um espaço m -dimensional – de tamanho bem menor do que o tamanho do

vocabulário. Enquanto um vocabulário V pode ter o tamanho de $|V| = 17000$ palavras, pelo menos, alguns valores frequentes para os tamanhos dos vetores são $m = 200, 300, 600$. Uma ilustração simplificada desse modelo neural é apresentada na [Figura 4](#).

Figura 4 – Arquitetura de modelo neural de língua

Fonte: Elaborada pelo autor.

Nota: Ilustração do modelo neural descrito em [Bengio et al. \(2003\)](#).

Os vetores de palavras – também chamados de *word embeddings* – são aprendidos juntamente com os demais parâmetros do modelo, que é ajustado iterativamente, de acordo com a ocorrência de palavras observadas nos dados de treinamento. Em cada iteração, o modelo recebe como entrada uma sequência de n palavras $w_{t-n+1}, w_{t-n+2}, \dots, w_{t-1}, w_t$ e gera como saída um vetor u , do tamanho do vocabulário V , em que o elemento na posição i , u_i , representa o valor de $P(w_t | w_{t-n+1}^{t-1})$, onde i corresponde à posição da palavra w_t no vocabulário V .

Na camada de entrada, cada palavra é mapeada em um vetor embutido (*embedded*) em uma matriz W de dimensão $|V| \times m$, conforme é ilustrado na [Figura 4](#). A ordem das linhas na matriz corresponde à ordem das palavras no vocabulário, de maneira que o vetor linha $W_{i\cdot}$ corresponde à palavra V_i . Após o mapeamento, os vetores são então passados a uma camada escondida, conectada diretamente à camada de saída, onde é computada uma operação softmax, visando garantir que o vetor de saída, u , represente uma distribuição de probabilidade válida. Ou seja, que contenha apenas elementos não-negativos, cuja soma resulte em 1 (um). Considerando que as ativações computadas na camada escondida sejam representadas pelo vetor v , também de tamanho $|V|$, a operação softmax é computada como:

$$u_i = \frac{e^{v_i}}{\sum_{j=1}^{|V|} e^{v_j}} \quad (3.9)$$

onde e^x representa a função exponencial.

Considerando o tamanho dos vocabulários, geralmente da ordem de dezenas de milhares de palavras, verifica-se que o maior gargalo enfrentado por esse modelo é a com-

putação das ativações na camada de saída (softmax) (BENGIO *et al.*, 2003). No mesmo trabalho, os autores também apresentam uma proposta de implementação paralela, projetada para reduzir o tempo computacional necessário para o treinamento do modelo.

Outros modelos neurais semelhantes foram propostos (Subseção 3.3.2). Em todos os casos conhecidos, poucas são as exigências em termos de pré-processamento, que é quase sempre limitado à *tokenização* e a outras tarefas simples, executadas opcionalmente, como o tratamento de palavras com hífen e de expressões numéricas. Em geral, os *córpus* de treinamento são tratados simplesmente como sequências de palavras. A qualidade dos resultados obtidos com as representações geradas por modelos neurais depende basicamente da dimensão dos vetores e da quantidade de dados de treinamento (MIKOLOV *et al.*, 2013a).

Diversos modelos de vetores foram avaliados em dois trabalhos (MIKOLOV *et al.*, 2013a; HARTMANN *et al.*, 2017) nos quais se observam resultados semelhantes a respeito da dimensão de vetores e do tamanho dos *córpus* de treinamento. De acordo com esses trabalhos, *córpus* de treinamento maiores normalmente favorecem a geração de vetores de melhor qualidade. Quanto à dimensão dos vetores, ambos avaliaram vetores de tamanho 50, 100, 300, 600 e 1000 elementos. Pelos dados apresentados, é possível concluir que, semelhantemente, resultados melhores também tendem a ser obtidos com vetores maiores. No entanto, Mikolov *et al.* (2013a) alertam para o fato de que, depois de certo ponto, aumentar isoladamente o tamanho dos vetores ou o tamanho do *córpus* proporciona apenas melhorias marginais. Segundo esses autores, é necessário que esses valores sejam incrementados em conjunto.

Uma análise dos resultados apresentados nesses dois estudos revela que ganhos poucos significativos foram obtidos com os vetores de dimensões maiores (600 e 1000). Conclui-se, portanto, que o tamanho 300 pode representar um bom compromisso entre qualidade de resultados e consumo de recursos computacionais. Uma vez que vetores também são parâmetros a serem aprendidos pelos modelos neurais, uso de vetores de grandes dimensões implica em aumentar o número de parâmetros do modelo.

É particularmente interessante, em modelos neurais, a mínima exigência por pré-processamento dos dados de treinamento. Essa característica, além de diminuir a dependência de outras ferramentas, como *parsers*, também torna esses modelos mais robustos, mais tolerantes a ruídos presentes nos dados, como erros ortográficos e gramaticais, aos quais outras ferramentas costumam ser sensíveis.

3.3 Vetores de palavras

As representações de palavras são, muitas vezes, mais interessantes do que os próprios modelos de língua, uma vez que esses vetores apresentam propriedades que podem

ser exploradas em diversas outras tarefas não relacionadas à predição de palavras (MIKOLOV; YIH; ZWEIG, 2013).

3.3.1 Propriedades

O treinamento de modelos de vetores de palavras, geralmente, é baseado apenas na observação da co-ocorrência de palavras em janelas de tamanho n . No entanto, representações vetoriais podem capturar interessantes relações entre palavras, tanto em nível sintático quanto em nível semântico. Em um estudo conduzido por Mikolov, Yih e Zweig (2013), foi verificado que relações entre palavras podem ser identificadas por meio de operações entre os respectivos vetores. Nesse estudo, foi observada a relação `king - man + woman = queen`, que passou empregada em diversos trabalhos da área como exemplo da versatilidade dos vetores derivados de modelos neurais. Na expressão, ilustra-se o fato de que `king - man + woman` é um vetor próximo do vetor da palavra `queen`. Mais ainda, ilustra-se o fato de que é possível identificar a relação masculino/feminino (`man:woman ≡ king:queen`) diretamente pelas representações das palavras.

Especificamente, Mikolov, Yih e Zweig (2013) observaram que a diferença entre pares de vetores de palavras compartilhando a mesma relação é aproximadamente a mesma. Por exemplo, denotando o vetor da palavra w por $v(w)$, é possível observar que $v(\text{apple}) - v(\text{apples}) \approx v(\text{car}) - v(\text{cars})$, $v(\text{family}) - v(\text{families}) \approx v(\text{car}) - v(\text{cars})$ e assim por diante. A partir dessa observação, os autores propuseram um método para identificar relações entre palavras, baseado unicamente em suas representações vetoriais.

Para avaliar seu método de identificação de relações, os autores criaram um *dataset* de perguntas formuladas na forma de analogias do tipo $a:b \equiv c:d$. A tarefa de avaliação consiste em encontrar a palavra d , dadas a , b e c . Diversas relações sintáticas e semânticas foram estudadas em vetores gerados por modelos neurais semelhantes ao que foi descrito na Seção 3.2. Alguns exemplos são apresentados no Quadro 10.

O método proposto para a resolução de analogias é baseado na distância cosseno entre os vetores de palavras (MIKOLOV; YIH; ZWEIG, 2013). Para responder uma analogia do tipo $a:b \equiv c:d$, onde d é desconhecido, primeiro é computado o vetor $y = v(b) - v(a) + v(c)$, onde $v(w)$ são vetores unitários. O vetor y deve corresponder à palavra desconhecida, d . No entanto, uma vez que y representa um ponto em um espaço contínuo, é pouco provável que coincida exatamente com a posição de alguma palavra no vocabulário. Assim, busca-se a palavra w^* cujo vetor seja o mais similar ao vetor y , de acordo com a similaridade cosseno:

$$w^* = \arg \max_{w \in V'} \frac{v(w) \cdot y}{\|v(w)\| \|y\|} \quad (3.10)$$

onde $V' = V - \{a, b, c\}$.

Quadro 10 – Relações identificadas por meio de vetores de palavras

Relação	Par 1	Par 2
País/capital	France:Paris	Germany:Berlin
Capital/país	Athens:Greece	Oslo:Norway
Masculino/feminino	brother:sister	grandson:granddaughter
Nacionalidade	Switzerland:Swiss	Cambodia:Cambodian
Singular/plural	year:years	law:laws
Comparativo	good:better	rough:rougher
Passado verbal	see:saw	return:returned
Classe/instância	clothing:shirt	dish:bowl

Fonte: Mikolov, Yih e Zweig (2013) e Mikolov *et al.* (2013a).

A similaridade cosseno entre dois vetores a e b é definida como:

$$\text{sim}_{\cos}(a, b) = \frac{a \cdot b}{\|a\| \|b\|}, \quad -1 \leq \text{sim}_{\cos}(a, b) \leq 1 \quad (3.11)$$

Vetores semelhantes tendem a ter a mesma orientação no espaço. Assim, quanto menor o ângulo formado entre dois vetores, maior será a similaridade (cosseno) entre eles. Frequentemente, emprega-se essa função para medir o grau de relação entre duas palavras, a partir de seus respectivos vetores. Dadas três palavras, por exemplo, a similaridade cosseno pode ser usada para verificar quais duas estão mais fortemente relacionadas, por meio de relações como:

$$\text{sim}_{\cos}(v(\text{cachorro}), v(\text{gato})) > \text{sim}_{\cos}(v(\text{cachorro}), v(\text{abacate}))$$

É justamente nessa relação que se fundamenta o emprego da função argmax na Equação 3.10, usada no método de resolução de analogias.

Por fim, observa-se que a tarefa de resolução de analogias, pelo método descrito anteriormente, se tornou um padrão para a avaliação de vetores de palavras, adotado em diversos outros trabalhos (HARTMANN *et al.*, 2017; RODRIGUES *et al.*, 2016; MIKOLOV *et al.*, 2013a).

3.3.2 Modelos de geração de vetores

As interessantes propriedades dos vetores de palavras, que os tornam úteis em diversas aplicações de processamento de línguas naturais, motivaram o surgimento de vários modelos de treinamento baseados em grandes corpú nos últimos anos. Nesta subseção, são descritos quatro modelos que foram treinados e avaliados por Hartmann *et*

[al. \(2017\)](#). Para esse estudo, foi compilado um *cópus* de treinamento contendo textos do português brasileiro e europeu. Para avaliação intrínseca dos vetores, os autores adotaram a tarefa de resolução de analogias.

O modelo Word2vec ([MIKOLOV et al., 2013a](#); [MIKOLOV et al., 2013b](#)) foi baseado no modelo neural proposto por [Bengio et al. \(2003\)](#) (NNLM)⁶ – descrito na [Seção 3.2](#) – e é, possivelmente, o mais popular entre os modelos de computação de vetores em espaços contínuos. O Word2vec foi idealizado, principalmente, para viabilizar o treinamento eficiente de vetores em grandes *cópus* (da ordem de 1,6 bilhão de palavras), e também com o objetivo de melhorar a qualidade dos vetores. Para atingir esses objetivos, foram implementadas simplificações e técnicas alternativas com relação à arquitetura originalmente proposta para o NNLM. As mudanças afetam principalmente a camada softmax, que é sabidamente o principal gargalo da arquitetura NNLM.

Na verdade, dois modelos de arquitetura foram propostos para o Word2vec ([MIKOLOV et al., 2013a](#)): CBOW e Skip-gram. O modelo CBOW (*Continuous Bag of Words*) é treinado para prever uma palavra-alvo w_t a partir de um contexto. Nessa arquitetura, a ordem das palavras não é considerada – razão da referência à representação BOW. O contexto é definido por palavras que tipicamente ocorrem ao redor de w_t (antes e depois), não considerando sua ordem. A arquitetura Skip-gram é semelhante à CBOW, porém o modelo é treinado no sentido inverso: em vez de prever a palavra-alvo, o modelo deve prever as palavras que formam o contexto, a partir de uma palavra w_t dada como entrada.

Avaliando ambos os modelos, [Mikolov et al. \(2013a\)](#) verificaram que os vetores derivados da arquitetura CBOW superam, ou pelo menos se igualam aos vetores gerados pela arquitetura NNLM original. O modelo Skip-gram também supera o NNLM, mas apresenta resultados inferiores ao CBOW na tarefa de identificação de relações sintáticas. No entanto, o Skip-gram supera o CBOW na identificação de relações semânticas.

O modelo Wang2vec ([LING et al., 2015](#)) é uma versão do Word2vec, modificada para considerar a ordem das palavras durante o processo de treinamento. O objetivo é melhorar o desempenho dos vetores de palavras em tarefas que dependam de relações sintáticas, como etiquetagem morfossintática (*POS tagging*), mantendo, ao mesmo tempo, a eficiência do modelo original. Assim como o Word2vec, o Wang2vec também apresenta as variações CBOW e Skip-gram. Observando os resultados da avaliação conduzida por [Hartmann et al. \(2017\)](#), pode-se perceber que a versão modificada realmente proporciona resultados superiores àqueles obtidos com a versão original.

Em outro modelo inspirado no Word2vec, o FastText ([JOULIN et al., 2016](#)), vetores são treinados como n-gramas de caracteres, enquanto os vetores de palavras são computados como somas dessas representações. O objetivo dessa técnica é capturar carac-

⁶ *Neural Network Language Model (NNLM)*

terísticas morfológicas. As variações CBOW e Skip-gram também são apresentadas pelo FastText.

Finalmente, o GloVe (*Global Vectors*) (PENNINGTON; SOCHER; MANNING, 2014) é o único dos quatro modelos aqui descritos que é baseado em contagem (não é preditivo). Nesse modelo, os vetores são derivados de uma matriz de co-ocorrências usada para extrair informações estatísticas sobre o corpus. No estudo realizado por Hartmann *et al.* (2017), foi um dos que proporcionaram os melhores resultados.

As implementações dos quatro modelos apresentados nesta sub-seção foram disponibilizadas publicamente⁷ por seus respectivos autores, na forma de código-fonte. Assim, é possível experimentá-las facilmente, bastando, para isso, que se tenha um corpus de treinamento.

3.3.3 Vetores pré-treinados

Além das implementações de modelos de treinamento, também existem várias coleções de vetores disponibilizadas como recursos públicos. Vetores treinados no corpus GoogleNews (MIKOLOV *et al.*, 2013b) com o Word2vec são frequentemente usados em trabalhos dedicados ao processamento de textos em inglês. Esse modelo⁸ contém vetores de 300 dimensões treinados para um vocabulário de 3 milhões de palavras. Vetores treinados com o modelo GloVe (PENNINGTON; SOCHER; MANNING, 2014) em corpus do inglês também foram disponibilizados no *site* do projeto em diversos tamanhos⁹.

Também é possível encontrar vetores pré-treinados para outras línguas, além do inglês. O projeto Polyglot (AL-RFOU; PEROZZI; SKIENA, 2013) disponibiliza vetores para mais de 100 línguas¹⁰, treinados nas respectivas versões da Wikipédia.

Recentemente, modelos treinados especificamente para o português também foram publicados. A coleção LX-DSemVectors (RODRIGUES *et al.*, 2016) contém vetores¹¹ treinados no modelo Skip-gram, em um corpus contendo 1,7 bilhão de *tokens*. Um trabalho mais abrangente foi realizado para a criação do repositório de vetores de palavras do NILC¹² (HARTMANN *et al.*, 2017). Esse repositório abriga vetores treinados em um grande corpus contendo textos em português brasileiro e europeu, de fontes e gêneros

⁷ Word2vec: <<https://github.com/tmikolov/word2vec>>
Wang2vec: <<https://github.com/wlin12/wang2vec>>
FastText: <<https://github.com/facebookresearch/fastText>>
GloVe: <<https://nlp.stanford.edu/projects/glove/>>

⁸ Pode ser obtido em: <<https://code.google.com/archive/p/word2vec/>>

⁹ Disponíveis em: <<https://nlp.stanford.edu/projects/glove/>>

¹⁰ Disponíveis em: <<https://sites.google.com/site/rmyeid/projects/polyglot/>>

¹¹ Disponíveis em: <<https://github.com/nlx-group/lx-dsemvectors>>

¹² Repositório de *Word Embeddings* do Núcleo Interinstitucional de Linguística Computacional (NILC): <<http://nilc.icmc.usp.br/embeddings>>

variados, totalizando aproximadamente 1,4 bilhão de *tokens*. Atualmente, o repositório contém vetores de tamanhos 50, 100, 300, 600 e 1000, treinados com os modelos Word2vec, Wang2vec, FastText e GloVe.

3.3.4 Representação de segmentos de texto

Apesar de resultados surpreendentes terem sido alcançados com vetores de palavras, a representação de segmentos de texto envolvendo vetores ainda está distante de ser uma questão plenamente resolvida.

O algoritmo *Paragraph Vector* (LE; MIKOLOV, 2014) foi proposto para representar, em um único vetor, o significado de segmentos de texto de tamanho variável. O modelo de treinamento é uma variação do modelo Word2vec, com a única diferença de que um vetor adicional, representando o parágrafo de onde se originaram as palavras do contexto, também é usado como entrada. Segundo os autores, o *Paragraph Vector* foi avaliado na tarefa de classificação de textos e alcançou resultados compatíveis com o estado da arte. No entanto, em experimentos realizados neste projeto, também envolvendo classificação de textos, não foram obtidos bons resultados com esse modelo. Acreditamos que, como sugere o nome do algoritmo, a capacidade de representar o sentido de textos em um único vetor esteja limitada a textos curtos.

Em outro trabalho (KENTER; RIJKE, 2015), dedicado à avaliação de similaridade semântica entre textos curtos, são adotadas duas abordagens para a representação de segmentos de texto usando vetores de palavras (*word embeddings*, como são chamados naquele trabalho). A primeira abordagem, bastante simples, consiste em computar a média dos vetores correspondentes às palavras que compõem uma frase. De maneira mais formal, seja S o conjunto de palavras que formam uma frase, então pode-se obter um vetor c , que represente essa frase como um ponto no espaço, pela seguinte expressão:

$$c = \frac{1}{|S|} \sum_{w \in S} v(w) \quad (3.12)$$

Da perspectiva geométrica, o vetor c representa o centroide – ou baricentro (CAMARGO; BOULOS, 2005) – de um conjunto de pontos no espaço, correspondentes a cada palavra contida no conjunto S . Essa interpretação é ilustrada na Figura 5, onde estão representadas as frases 3.2 (círculos) e 3.4 (quadrados), usadas no exemplo sobre similaridade semântica, apresentado na Seção 3.1. Na figura, círculos e quadrados preenchidos representam as coordenadas de vetores de palavras, enquanto formas não preenchidas representam os respectivos centroides.

Os vetores u e v , ilustrados na figura, correspondem ao centroide de cada frase. Pode-se observar que ambos têm orientação semelhante, indicando que as duas frases são

Figura 5 – Textos representados como médias de vetores (centroides)

Fonte: Elaborada pelo autor.

próximas semanticamente. De fato, usando vetores de palavras¹³ disponíveis no repositório do NILC (HARTMANN *et al.*, 2017), verifica-se que $\text{sim}_{\cos}(u, v) = 0,69$. Em nível de complexidade, computar o grau de similaridade semântica entre duas frases pela similaridade cosseno, como foi feito neste caso, é equivalente ao uso da similaridade de Jaccard, adotada no exemplo anterior. Ambos os métodos são bastante simples. No entanto, nota-se que, graças à representação vetorial de palavras, o grau de similaridade obtido neste caso é claramente mais coerente do que o anterior.

Na segunda abordagem, proposta por Kenter e Rijke (2015), pares de frases são representados como redes semânticas, usadas para a computação de um grau de similaridade entre as duas frases. Cada palavra representa um vértice nesses grafos semânticos, e as arestas entre duas palavras w_1 e w_2 recebem pesos proporcionais à (i) similaridade cosseno entre w_1 e w_2 e à (ii) saliência de cada uma dessas palavras no corpus, medida pela IDF (*Inverse Document Frequency*) (SPARCK-JONES, 1972). Essa abordagem, empregada para dar maior importância às palavras mais significativas de cada frase, também substitui a remoção de *stopwords*, que, pelo critério de saliência, receberão pesos baixos, influenciando minimamente o grau de similaridade entre as duas frases.

O uso da IDF nesse trabalho inspirou o desenvolvimento de uma técnica empregada em nosso modelo SlimRank (Capítulo 5), e também um dos *baselines* adotados em nossos experimentos (Capítulo 6). Os autores salientam, em diversos pontos de seu trabalho, que sua proposta é destinada à representação de textos curtos. De fato, foi possível verificar, empiricamente, que centroides originados de textos maiores (maior número de palavras, e consequentemente, também de pontos no espaço) tendem a ser menos significativos.

Outra maneira de se representar segmentos de textos, usando vetores de palavras, é

¹³ Wang2vec, CBOW, 100 dimensões

empilhando os vetores de cada palavra na sequência em que ocorrem no texto, formando, assim, uma matriz. Essa organização é bastante conveniente à exploração de relações de adjacência entre palavras, e é usada principalmente nas redes neurais convolutivas, descritas a seguir, na [Seção 3.4](#).

3.4 Redes neurais convolutivas

Redes neurais convolutivas (CNNs)¹⁴ (LECUN *et al.*, 1990) foram inicialmente desenvolvidas para o reconhecimento de caracteres numéricos escritos a mão. Desde então, as CNNs vêm sendo usadas em diversas outras tarefas, não limitadas à área de visualização computacional. Particularmente, na área de QA, CNNs têm sido empregadas com sucesso em diversos trabalhos (SANTOS *et al.*, 2015; SEVERYN; MOSCHITTI, 2015; WANG; MI; ITTYCHERIAH, 2016, e outros).

Redes neurais tradicionais, do tipo *feed forward*, são organizadas em camadas, de maneira que cada neurônio de uma camada é conectado a todos os neurônios da camada seguinte. Um exemplo dessa organização em *camadas completamente conectadas* é apresentado na [Figura 6a](#), onde cada elemento do vetor de entrada x é conectado a todos os neurônios da camada escondida H .

Figura 6 – Camadas de entrada: redes neurais tradicionais e redes convolutivas

Fonte: Elaborada pelo autor.

Em CNNs, por outro lado, apenas pequenas regiões de uma matriz de entrada X , chamadas *campos receptivos locais*, são conectadas a cada neurônio. Um exemplo de campo receptivo, indicado por f , é mostrado na [Figura 6b](#). Cada neurônio, associado a um campo receptivo, se especializa em detectar uma *feature* naquela região dos exemplos de treinamento. No entanto, *features* importantes de um objeto (como a forma de um olho ou nariz, por exemplo) podem aparecer em várias posições de uma imagem (LECUN *et al.*, 1989) – ou de um texto, no caso do processamento de línguas naturais.

¹⁴ Do inglês, *Convolutional Neural Network* (CNN)

É possível detectar uma *feature* particular em qualquer posição de um exemplo de entrada (propriedade de invariância local) por meio do *compartilhamento de pesos* (*weight sharing*) (LECUN *et al.*, 1989). Por essa técnica, grupos de neurônios, que compartilham exatamente os mesmos pesos, são especializados em detectar determinada *feature*, em qualquer posição de uma matriz de entrada, e geram uma nova matriz, chamada mapa de *features* (Figura 7). Além de resultar na invariância local, o compartilhamento de pesos também torna mais rápido o treinamento do modelo, uma vez que reduz consideravelmente o número de parâmetros de treinamento, em relação a uma arquitetura completamente conectada.

Os mapas de *features* são simplificados e condensados por meio de uma operação de *pooling* (NIELSEN, 2015). Várias alternativas de implementação dessa operação são possíveis, incluindo tirar a média dos valores contidos no mapa de *features*. Porém, em trabalhos da área de processamento de línguas naturais, a alternativa mais frequentemente usada é a operação *max pooling*, que consiste em selecionar o maior valor encontrado em um mapa de *features*. Assim, cada grupo de neurônios gera um só valor para determinada *feature*.

Na prática, o compartilhamento de pesos é implementado como uma operação de convolução¹⁵. Um conjunto de pesos, compartilhado por um grupo de neurônios, é representado por pequenas matrizes, chamadas *kernels*, ou filtros de convolução (Figura 7). Informalmente, pode-se descrever a operação de convolução como um filtro f , de tamanho $F \times F$, sendo deslizado sobre uma matriz de entrada X , computando o produto escalar entre a região da matriz sobreposta pelo filtro e ele próprio. Em cada iteração, um valor escalar é obtido. Um mapa de *features* M é gerado ao se deslizar o filtro f por toda a matriz X .

Figura 7 – Extração de *features* em uma CNN

Fonte: Elaborada pelo autor.

A operação de convolução executada sobre uma matriz de entrada sem alterações – ou seja, sem preenchimento (*padding*) – resulta em um mapa de *features* de tamanho

¹⁵ Origem do nome das redes neurais convolutivas.

$H' \times W'$, menor do que o tamanho da matriz de entrada. As dimensões do mapa de *features* podem ser computadas pelas expressões:

$$H' = \frac{(H - F)}{S} + 1 \quad W' = \frac{(W - F)}{S} + 1$$

onde S representa o tamanho do passo (*stride*) de deslocamento do filtro f a cada iteração. No exemplo ilustrado na Figura 7, $S = 1$ linha (ou coluna). Portanto, considerando que a matriz X tenha tamanho $H \times W = 5 \times 4$, o mapa de *features* M terá dimensões $H' \times W' = 4 \times 3$. Considerando que o elemento $(1, 1)$, do filtro f , esteja posicionado sobre o elemento (k, l) , da matriz X , o valor de saída y do neurônio representado na figura é dado por:

$$y = \alpha \left(b + \sum_{i=1}^F \sum_{j=1}^F f_{i,j} X_{k+i-1, l+j-1} \right) \quad (3.13)$$

onde $b \in \mathbb{R}$ é um *bias* do neurônio (também compartilhado, como os demais pesos) e α representa uma função de ativação não-linear, tal como a função sigmoide ou a tangente hiperbólica. Os dois somatórios representam o produto escalar, mencionado anteriormente.

Assim como ocorre com outros pesos em redes neurais, cada filtro é iniciado com valores aleatórios, que são ajustados posteriormente, durante o treinamento do modelo. Pode-se dizer, assim, que cada filtro aprende a detectar uma *feature* automaticamente.

Modelos do tipo CNN normalmente contêm centenas de filtros. Considerando o uso da operação de *max pooling*, cada filtro resulta em um único valor, conforme foi mencionado anteriormente. Para completar o processo de extração de *features*, esses valores individuais são concatenados, formando um vetor de tamanho n , em que n corresponde ao número de filtros usados no modelo. Finalmente, o vetor de *features* resultante da concatenação é passado para uma camada completamente conectada, como a que é ilustrada na Figura 6a.

Figura 8 – Visão modular de uma CNN

Fonte: Elaborada pelo autor.

Em um modelo de CNN completo, dois componentes principais podem ser identificados: (i) um extrator de *features* – como o que é ilustrado na Figura 7 – e (ii) uma camada completamente conectada. Essa visão modular de uma CNN é representada na

Figura 8. Dependendo da aplicação para a qual foi desenvolvido o modelo, a camada completamente conectada pode implementar um classificador ou um modelo de regressão.

Na área de visão computacional, são comuns modelos de CNN formados por várias camadas de convolução, ou seja vários extratores de *features* encadeados. A cada camada, são extraídas *features* de níveis de abstração e de complexidades cada vez mais altos (LE-CUN *et al.*, 1990) (propriedade de composicionalidade), que serão, ao final, passadas à camada completamente conectada, onde poderá ser realizada a identificação de objetos, por exemplo.

Uma abordagem completa e detalhada sobre as redes neurais convolutivas pode ser encontrada em Nielsen (2015).

3.4.1 Redes convolutivas para processamento de textos

Diversos modelos de CNN destinados ao processamento de textos (WANG; MI; ITTYCHERIAH, 2016; BELINKOV; GLASS, 2016; SANTOS *et al.*, 2015; FENG *et al.*, 2015) foram influenciados pela arquitetura proposta no trabalho de Kim (2014), para a tarefa de classificação de frases. A análise dessa arquitetura, permite que se compreenda algumas características particulares de modelos CNNs destinados ao processamento de textos. Nesta subseção, descreve-se um modelo similar a uma das arquiteturas propostas por Kim (2014), destinado a identificar variantes de um idioma, tais como o português europeu e o português brasileiro.

O modelo de identificação de variantes de língua ilustrado na Figura 9 (CRISCUOLO; ALUÍSIO, 2017a), recebe uma frase como entrada e gera, como saída, uma distribuição de probabilidades sobre as possíveis classes de variantes. A maior probabilidade, indica a classe a que deve pertencer a frase de entrada. No exemplo da figura, observa-se a identificação de uma frase do português europeu (pt-PT). A seguir, descreve-se o funcionamento desse modelo.

Figura 9 – Exemplo de CNN para classificação de textos

Fonte: Adaptada de Criscuolo e Aluísio (2017a).

Seja $s = \langle w_1, w_2, w_3, \dots, w_L \rangle$ uma frase de entrada, de comprimento L . Cada palavra

w_j é mapeada em um vetor linha $v(w_j) \in \mathbb{R}^d$, embutido na matriz W , de tamanho $|V| + 1 \times d$, onde $|V|$ representa o tamanho do vocabulário formado pelas palavras das duas variantes do português. As linhas da matriz W estão na mesma ordem em que as palavras ocorrem no vocabulário V . Assim, o mapeamento de uma palavra w_j em seu respectivo vetor $v(w_j)$, é feito por uma busca pelo índice correspondente na matriz W . Palavras não encontradas no vocabulário são ignoradas.

No modelo, a frase s é representada pela matriz $S_{L \times d}$, gerada pela concatenação dos vetores de palavras $v(w_j)$. A primeira linha da matriz W representa um símbolo especial, PAD, usado para completar frases com número de palavras menor do que L . Frases comprimento maior do que L são truncadas.

Para a extração de *features*, operações de convolução são realizadas deslizando filtros f , de tamanho $h \times d$, sobre a matriz S . No caso de processamento de textos, é importante que número de colunas d em cada filtro coincida com o número de colunas na matriz S . Caso contrário, poderia haver um tratamento inconsistente dos vetores de palavras. A altura h de cada filtro, por outro lado, pode variar. Assim, é possível ter filtros abrangendo números variáveis de palavras adjacentes, de maneira análoga ao que ocorre com n-gramas.

Relações entre palavras adjacentes podem ser aprendidas pelos filtros de convolução. Na Figura 9, ilustra-se o exemplo de uma situação em que duas palavras adjacentes são processadas por um filtro (retângulo tracejado) capaz de identificar construções do tipo “ $a + <verbo\ no\ infinitivo>$ ”, típicas do português europeu.

No modelo ilustrado na figura, foram usados três filtros de convolução, que geraram três mapas de *features*. Como a largura de todos os filtros coincide com a largura da matriz S , o deslizamento ocorre apenas na direção vertical e, em vez de mapas de *features* bidimensionais (matrizes), a operação de convolução gera vetores. Formalmente, cada *feature* c_i em um mapa de *features* c é computada como:

$$c_i = \alpha(f \cdot S_{i:i+h-1} + b) \quad (3.14)$$

onde, novamente, α é uma função não-linear, tal como a tangente hiperbólica, $b \in \mathbb{R}$ é o termo de *bias* e h é a altura do filtro f . Nesse modelo, o passo de convolução (*stride*) tem tamanho 1. Ou seja, os filtros funcionam como janelas que deslizam de palavra em palavra na sequência formada pela frase s .

Na última etapa do modelo, após as operações de *max pooling* e de concatenação, é obtido um vetor de *features* final, que é processado por uma camada completamente conectada, cujas saídas passam por uma operação softmax, que resulta na distribuição de probabilidades sobre as classes de frases conhecidas (variantes do português).

3.4.1.1 Treinamento do modelo

Nos experimentos realizados por Criscuolo e Aluísio (2017a), um corpus de frases extraídas de textos jornalísticos é usado para o treinamento do modelo de identificação de variantes, descrito anteriormente. Todas as frases são *tokenizadas* e os caracteres numéricos são substituídos por zero. O tamanho máximo permitido para cada frase é $L = 80$ *tokens*.

Nesse experimentos, a matriz W é iniciada com valores aleatórios, atualizados por *backpropagation* durante o treinamento do modelo. Ou seja, nesses experimentos, os vetores são aprendidos na própria tarefa, e não são usados vetores pré-treinados.

Os demais hiper-parâmetros de treinamento são: dimensão dos vetores de palavras $d = 200$, filtros de altura 1 e 2, cada um gerando 100 mapas de *features* – ou seja, há 100 filtros de convolução de altura $h = 1$ (unigrama) e 100 filtros de altura $h = 2$ (bigrama) –, função $\alpha = \tanh$ (tangente hiperbólica) e lotes (*batches*) de treinamento de tamanho 50. Para regularização, foi aplicada a técnica *dropout* (SRIVASTAVA *et al.*, 2014) à camada completamente conectada, com probabilidade de descarte de 0,20. O modelo foi implementado no *framework* TensorFlow (ABADI *et al.*, 2016), e treinado por 3 épocas, para minimizar a entropia cruzada (*cross-entropy*) usando o otimizador Adam (KINGMA; BA, 2014).

Esse modelo alcançou precisão de 0,91 na identificação de frases do português brasileiro e de 0,93 em frases do português europeu. Para ambas as variantes, o valor da medida F1 foi de 0,92 (CRISCUOLO; ALUÍSIO, 2017a).

3.4.2 Modelos de seleção de respostas

O emprego de modelos baseados em CNNs na tarefa de seleção de respostas está fundamentado na premissa de que deve haver forte relação semântica entre uma pergunta e sua resposta. Assim, dadas uma pergunta q e uma resposta candidata a , esses modelos são treinados para aprender uma função de similaridade $\text{sim}(q, a)$, que atribui um grau (*score*) de relação semântica ao par (q, a) .

Na tarefa de seleção de respostas (Definição 1, p.50), são medidos os graus de relação semântica entre uma pergunta q e todas suas respostas candidatas a_i . A medição é feita aos pares: $(q, a_1), (q, a_2), \dots, (q, a_n)$. A resposta candidata a_i para a qual se observar o maior valor de $\text{sim}(q, a_i)$, computado por modelo treinado em algum *dataset*, é selecionada como uma resposta correta para q .

Nesta subseção são descritos dois modelos de seleção de respostas baseados em CNNs. O modelo CNN-std, ilustrado na Figura 10, é um dos *baselines* implementados neste projeto. Trata-se de uma versão do modelo proposto por Severyn e Moschitti (2015), com algumas simplificações.

Figura 10 – Modelo CNN-std

Fonte: Elaborada pelo autor.

É possível observar na figura que dois extratores de *features* são usados no modelo CNN-std. Essa é uma configuração padrão na maior parte dos trabalhos da área. Porém, mais raramente, encontram-se modelos em que apenas um extrator é usado tanto para a pergunta quanto para a resposta candidata (FENG *et al.*, 2015), representadas na figura respectivamente pelas matrizes Q e A . Essas matrizes são análogas à matriz S , ilustrada na Figura 9.

No modelo CNN-std, tanto o funcionamento dos extratores de *features* quanto a organização das matrizes de entrada Q e A , são idênticos aos que foram descritos na Subseção 3.4.1, para o modelo de identificação de variantes de língua. Esse modelo, porém, admite o uso de vetores pré-treinados para iniciar a matriz W (matriz de *embeddings*, omitida na Figura 10).

Os vetores de *features* x_q e x_a , extraídos respectivamente de Q e A , são concatenados, formando o vetor x , que é então passado a uma camada completamente conectada, responsável por computar a saída do modelo $y \in \mathbb{R}$, que, diferentemente do modelo descrito na Subseção 3.4.1, representa, não um valor de probabilidade, mas o grau de relação semântica entre pergunta e resposta candidata. A saída y , da camada completamente conectada, é computada por:

$$y = \sigma(x^T \cdot w + b) \quad (3.15)$$

onde $w \in \mathbb{R}^{|x|}$ é um vetor de pesos, $b \in \mathbb{R}$ é um termo de *bias* e σ representa a função sigmoide, definida pela expressão:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

A exemplo do que foi feito por Wang, Mi e Ittycheriah (2016), a função sigmoide é usada para restringir o valor de y ao intervalo $[0, 1]$.

No modelo CNN-std, a camada completamente conectada é formada por um único neurônio. Trata-se, portanto, de um modelo de regressão, fato que fica evidente na [Equação 3.15](#).

Outro modelo de seleção de respostas, também baseado em CNNs, porém mais elaborado, é o CNN-LDC, proposto por [Wang, Mi e Ittycheriah \(2016\)](#), e ilustrado na [Figura 11](#). O A expressão LDC se refere à técnica de decomposição lexical¹⁶, empregada nesse trabalho. O CNN-LDC também foi implementado como um *baseline* neste projeto, exatamente como é descrito por seus autores. Na verdade, com exceção da decomposição lexical e do uso de operações de convolução em dois canais, o restante do modelo é idêntico ao modelo CNN-std, descrito anteriormente.

Figura 11 – Modelo CNN-LDC

Fonte: Elaborada pelo autor.

Os autores do modelo CNN-LDC ([WANG; MI; ITTYCHERIAH, 2016](#)) sugerem que, ao avaliar o grau de relação semântica, sejam consideradas tanto similaridades quanto dissimilaridades entre as duas frases envolvidas no processo. Para isso, propõem um método de decomposição lexical (decomposição de vetores de palavras) que divide cada matriz de entrada em dois canais: um positivo e um negativo, análogos aos canais de cor, RGB¹⁷, em uma imagem colorida. É interessante ressaltar que as matrizes decompostas afetam minimamente a configuração dos extratores de *features*, e não representam qualquer complexidade adicional para a arquitetura das CNNs. É necessário apenas que os filtros de convolução tenham o mesmo número de canais que as matrizes entrada. Algumas variações da técnica de decomposição lexical são apresentadas por [Wang, Mi e Ittycheriah \(2016\)](#). Descreve-se, a seguir, a variação que apresentou os melhores resultados naquele trabalho.

Considere-se que as matrizes de entrada Q e A sejam formadas por vetores linha $q_i, a_j \in \mathbb{R}^d$, que representam cada palavra da pergunta e da resposta, respectivamente,

¹⁶ *Lexical Decomposition and Composition (LDC)*

¹⁷ *Red, Green, Blue*

conforme:

$$Q = \begin{bmatrix} - & q_1 & - \\ - & q_2 & - \\ & \dots & \\ - & q_m & - \end{bmatrix}_{m \times d} \quad A = \begin{bmatrix} - & a_1 & - \\ - & a_2 & - \\ & \dots & \\ - & a_n & - \end{bmatrix}_{n \times d}$$

Define-se então uma matriz de similaridade¹⁸ $S_{m \times n}$, em que cada elemento $s_{i,j}$ é computado como a similaridade cosseno entre as palavras representadas pelos vetores q_i e a_j :

$$s_{i,j} = \text{sim}_{\cos}(q_i, a_j) = \frac{q_i \cdot a_j}{\|q_i\| \|a_j\|} \quad (3.16)$$

Para se decompor a matriz Q , por exemplo, é necessário computar, para cada palavra q_i , um vetor de correspondência semântica de q_i em A , definido por:

$$\hat{q}_i = \frac{\sum_{j=k-w}^{k+w} s_{i,j} a_j}{\sum_{j=k-w}^{k+w} s_{i,j}} \quad (3.17)$$

onde $k = \arg \max_j s_{i,j}$ e w é um parâmetro, cujo melhor valor verificado pelos autores do método foi $w = 3$. Verifica-se, pela [Equação 3.17](#), que o vetor $\hat{q}_i$ é uma combinação linear de vetores linha embutidos em na matriz A , compreendidos em uma janela de tamanho $2w + 1$, centrada no vetor a_k , mais próximo de q_i , de acordo com a similaridade cosseno.

A partir do vetor de correspondência $\hat{q}_i$, o vetor q_i é decomposto em um componente de similaridade q_i^+ , paralelo a $\hat{q}_i$, e um componente de dissimilaridade q_i^- , perpendicular a $\hat{q}_i$, de acordo com:

$$q_i^+ = \frac{q_i \cdot \hat{q}_i}{\hat{q}_i \cdot \hat{q}_i} \hat{q}_i \quad q_i^- = q_i - q_i^+ \quad (3.18)$$

Assim, a matriz Q é decomposta em dois canais, Q^+ e Q^- que servirão como entrada para os extratores de *features* no modelo CNN-LDC. A decomposição da matriz A é feita por um processo análogo.

Originalmente, o modelo CNN-LDC foi avaliado nos *datasets* WikiQA e TREC-QA, alcançando o estado da arte da tarefa de seleção de respostas ([ACL, 2017](#)). Em experimentos realizados neste trabalho ([Capítulo 6](#)), os resultados apresentados por esse modelo foram superiores aos resultados do modelo CNN-std. No entanto, por se tratar de um modelo que envolve operações elaboradas, assim como outros encontrados na literatura ([SANTOS et al., 2016](#)), as exigências do CNN-LDC em termos de recursos computacionais também foram maiores.

¹⁸ No artigo de [Wang, Mi e Ittycheriah \(2016\)](#) essa matriz é denotada por A .

3.5 Grafos textuais

Como foi discutido, no início deste capítulo, a escolha da representação textual pode influenciar diretamente o desempenho de aplicações de processamento de línguas naturais. Nesta seção, são discutidos trabalhos em que se adotam grafos, como uma maneira de representar relações entre unidades textuais, tais como palavras e frases.

Formalmente, um grafo pode ser definido como um par ordenado $G = (V, E)$, onde $V = \{v_1, v_2, \dots, v_n\}$ é um conjunto de vértices e $E \subseteq V \times V$ é um conjunto de arestas, tal que cada aresta $(a, b) \in E$ representa uma relação entre os vértices a e b . Considerando que n seja o número de vértices, um grafo pode ser representado por uma matriz de adjacência A , de tamanho $n \times n$, onde $A_{i,j} = 1$ se existir uma aresta do vértice v_i para v_j , e $A_{i,j} = 0$, em caso contrário. Um exemplo de grafo, com 4 vértices e 5 arestas, é apresentado na Figura 12.

Figura 12 – Exemplo de grafo

Fonte: Elaborada pelo autor.

Em um importante trabalho, [Mihalcea e Tarau \(2004\)](#) propuseram um modelo de classificação (*ranking*) não-supervisionado para o processamento de textos, baseado em grafos, chamado TextRank. Modelos de *ranking* são usados para avaliar a importância de um dado vértice dentro de um grafo. No TextRank, é empregado o conceito de votação (ou recomendação) entre vértices, popularizado pelo algoritmo PageRank ([BRIN; PAGE, 1998](#)). Nesse algoritmo, a importância de um vértice é computada por um processo iterativo, em função do número de arestas vindas de outros vértices. O modelo TextRank foi avaliado nas tarefas de (i) extração de palavras-chave e de (ii) seleção de sentenças, para geração de sumários extrativos, alcançando bom desempenho nos dois casos.

O tipo de relação capturada pelas arestas (relações semânticas, por exemplo) e o tamanho das unidades textuais (como palavras ou até frases inteiras) representadas pelos vértices do grafo vão depender da aplicação ([MIHALCEA; TARAU, 2004](#)). Na tarefa de extração de palavras-chave, por exemplo, os autores usaram um processo relativamente simples para identificar relações entre palavras. Os textos são *tokenizados* e filtrados por classes de palavras – apenas verbos e substantivos são deixados, por exemplo. Em seguida,

considera-se que dois vértices estão conectados se as respectivas palavras co-ocorrerem em uma janela de $N \in [2, 10]$ palavras. O grafo resultante é representado por uma matriz de adjacência. Após a execução do algoritmo de *ranking*, são extraídas as palavras-chave correspondentes aos vértices que receberam as maiores pontuações.

Um exemplo de grafo construído por esse processo, para a frase:

(3.19) A silagem de Capiáçu exige maior quantidade de concentrado na dieta.

é apresentado na [Figura 13](#). Neste exemplo, usou-se $N = 3$ palavras como o tamanho da janela de co-ocorrência.

Figura 13 – Grafo representando uma frase

Fonte: Elaborada pelo autor.

O conceito de prestígio (votação) entre vértices de um grafo também é empregado no modelo LexRank ([ERKAN; RADEV, 2004](#)) para compor sumários extrativos em um cenários multi-documento. Nesse modelo, múltiplos documentos são representados na forma de um grafo, em que os vértices correspondem às sentenças de todos os documentos. Medidas de centralidade – como o grau de um vértice (número de arestas) – são usadas para identificar as sentenças mais salientes, que são selecionadas para fazer parte do sumário.

Grafos textuais também podem ser usados como fonte de *features* para modelos de aprendizado de máquina, como fizeram [Rousseau, Kiagias e Vazirgiannis \(2015\)](#). Nesse trabalho, a tarefa de classificação de textos é abordada com um problema de classificação de grafos. Assim como [Mihalcea e Tarau \(2004\)](#), esses autores também usaram co-ocorrência de palavras, em uma janela de tamanho N , para estabelecer relações entre vértices. Um grafo é construído para cada texto, e uma ferramenta é usada para extrair *features* dessas estruturas, que então são usadas como entrada para um classificador SVM ([JOACHIMS, 1998](#)). Apenas as partes mais densas de cada grafo (maior concentração de arestas) são consideradas. Segundo os autores, essa estratégia permite reduzir o número de *features*

processadas pelo classificador, causando apenas mínimas perdas de desempenho em alguns casos específicos.

Uma alternativa à extração de *features* explícita é o uso de *kernels* sobre grafos. Essa estratégia – comum em áreas como a bioinformática (MENCHETTI; COSTA; FRASCONI, 2005) – consiste em adotar algoritmos de aprendizado de máquina baseados em *kernels*, que, de maneira informal, são funções capazes de capturar a semântica inerente às estruturas de um grafo (VISHWANATHAN *et al.*, 2010).

3.5.1 Redes convolutivas sobre grafos

O *framework* Patchy-SAN (NIEPERT; AHMED; KUTZKOV, 2016) estende conceitos de CNNs – originalmente aplicados a imagens – para grafos arbitrários, permitindo que modelos convolutivos sejam treinados diretamente sobre dados representados nessa forma. De maneira simples, trata-se de um método para extrair, de grafos, regiões localmente conectadas, análogas aos campos receptivos implementados pelas CNNs.

De acordo com os autores desse trabalho, uma imagem pode ser representada como um grafo em forma de grade (Figura 14), em que cada vértice corresponda a um pixel, e em que arestas indiquem pixels adjacentes na imagem. Nesse modelo, um campo receptivo é representado por um subgrafo de vizinhança, centrado em um vértice v e incluindo vértices v' para os quais exista um caminho $\langle v, \dots, v' \rangle$ – ou seja, uma sequência de vértices conectados entre v e v' .

Figura 14 – Imagem representada como grafo

Fonte: Adaptada de Niepert, Ahmed e Kutzkov (2016).

Nesse cenário, devido à organização espacial dos pixels em uma imagem, a ordem de criação dos campos receptivos (ordem em que são deslizados os filtros de convolução) pode ser determinada unicamente – da esquerda para a direita e de cima para baixo. Porém, para muitos tipos de grafos, não há uma relação de ordem pré-definida para os vértices. Nesses casos, dois problemas principais precisam ser resolvidos: (i) determinar a ordem em que os vértices serão selecionados para a criação dos subgrafos de vizinhança (campos receptivos) e (ii) normalizar subgrafos de vizinhança, isto é, definir um mapeamento único de uma representação em forma de grafo para um espaço vetorial, que possa servir como entrada para uma CNN. O *framework* Patchy-SAN foi proposto como uma solução para

esses problemas, e é apresentado na forma de um processo – ilustrado na [Figura 15](#) – composto por 3 etapas: Seleção de vértices, Agregação de vizinhança e Normalização (SAN).

Figura 15 – *Framework Patchy-SAN*

Fonte: Adaptada de [Niepert, Ahmed e Kutzkov \(2016\)](#).

Para impor uma relação de ordem aos vértices, tanto a etapa de seleção quanto a etapa de normalização dependem de um procedimento de rotulação de grafos. A rotulação consiste na aplicação de uma função $\ell: V \rightarrow S$ aos vértices de um grafo, onde V é o conjunto de vértices e S é um conjunto ordenado, tal como o conjunto dos números reais ou o conjunto dos números inteiros. Portanto, um procedimento de rotulação atribui rótulos a cada vértice de um grafo de entrada, de maneira que seja possível ordená-los.

Pode-se definir, por exemplo, um procedimento de rotulação baseado no grau de cada vértice – ou em alguma outra medida de centralidade ([NIEPERT; AHMED; KUTZ-KOV, 2016](#)). Nesse caso, $\ell(v)$ é definida como o número de arestas conectadas a v . Outro exemplo de procedimento de rotulação que pode ser usado com *framework* Patchy-SAN, segundo os autores, é o algoritmo PageRank, mencionado anteriormente nesta seção (3.5).

Uma vez que um procedimento de rotulação tenha sido executado para atribuir posições relativas similares (em relação à ordem) a vértices que tenham papel estrutural similar em grafos diferentes, inicia-se a etapa de seleção de vértices. Nessa etapa é gerada uma sequência de vértices, para os quais serão criados m campos receptivos. Primeiro, todos os vértices de um grafo são ordenados, de acordo com os respectivos rótulos. Em seguida, percorre-se essa sequência ordenada usando passos de tamanho s (*stride*), para selecionar os centros de cada subgrafo de vizinhança. Os autores indicam que também são possíveis outros métodos de seleção baseados em rotulação de grafos.

Na etapa seguinte, são construídos os subgrafos de vizinhança. Para cada vértice v , é executada uma busca em largura explorando vértices em ordem crescente de distância de v , até que sejam coletados k vértices, onde k é um tamanho fixo dos campos receptivos.

Finalmente, o subgrafo de vizinhança de cada vértice v é normalizado. A norma-

lização impõe uma ordem aos vértices do subgrafo, de maneira que sejam mapeados de um espaço de grafos não-ordenado para um espaço vetorial com ordem linear. A ideia é, novamente, recorrer a um procedimento de rotulação, para fazer com que dois vértices de subgrafos diferentes, que tenham papel estrutural similar, sejam atribuídos a posições similares no espaço vetorial de destino (NIEPERT; AHMED; KUTZKOV, 2016). Por exemplo, se a cada vértice v estiver associado um atributo escalar a_v , então o subgrafo de vizinhança será normalizado para um vetor de k elementos, conforme a distância de cada vértice até v . Logo, o atributo a_v , associado a v , sempre será o primeiro elemento do vetor, seguido por a_u , na segunda posição, se u for o vértice mais próximo de v , e assim por diante.

Se, para cada grafo G , forem extraídos m campos receptivos de tamanho k – considerando que a cada vértice esteja associado um atributo escalar –, o resultado do processo será uma matriz de tamanho $m \times k$, que poderá ser passada como entrada a uma CNN. A única restrição é que os filtros de convolução tenham a mesma largura k dos campos receptivos extraídos dos grafos, assim como ocorre com os filtros da CNN para processamento de textos, descrita na Subseção 3.4.1, que devem ter a mesma largura que os vetores de palavras.

O modelo SlimRank, proposto neste trabalho, foi desenvolvido sobre o *framework* Patchy-SAN. Em nosso modelo, propomos um procedimento de rotulação de grafos textuais baseado em uma medida de centralidade empregada por Traina Jr. *et al.* (2000) na construção da árvore métrica Slim-tree¹⁹.

A Slim-tree é uma estrutura de indexação de dados métricos, projetada explicitamente para reduzir a sobreposição dos raios de cobertura dos ramos da árvore. (Essa sobreposição afeta negativamente o desempenho de consultas). Uma das principais contribuições desse trabalho (TRAINA JR. *et al.*, 2000) é um algoritmo eficiente para o particionamento (*split*) de nós, baseado em árvores geradoras mínimas (MSTs)²⁰ (KRUSKAL, 1956). Uma definição de MST é apresentada a seguir.

Definição 2 (Árvore Geradora Mínima – MST). Seja $G = (V, E)$ um grafo, com um (custo) peso positivo w associado a cada aresta $e \in E$. Uma MST é um grafo acíclico (V, T) , onde $T \subseteq E$ é um subconjunto de arestas que conectam todos os vértices em V (grafo conectado), de maneira que a soma dos pesos das arestas seja mínima.

Além de resolver o problema de minimização do custo total em grafos, as MSTs também são estruturas convenientes em problemas de agrupamento (*clustering*). Para se obter n *clusters* de objetos, a partir de uma MST, por exemplo, basta remover as $n - 1$ arestas mais longas (maior peso) (KLEINBERG; TARDOS, 2005). Na Slim-tree, o parti-

¹⁹ Origem do nome SlimRank.

²⁰ Do inglês, *Minimum Spanning Tree* (MST)

cionamento de um nó envolve a construção de uma MST, que é dividida em dois *clusters* menores, contendo os elementos dos dois novos nós. Nesse ponto, um critério de centralidade é adotado para identificar o elemento representativo de cada novo nó (TRAINA JR. *et al.*, 2000).

Tanto a organização de dados em forma de uma MST quanto o critério de centralidade mencionado são particularmente úteis no cenário deste projeto. Por esse motivo, foram adotados como base do procedimento de rotulação de grafos, executado na etapa de seleção de vértices do modelo SlimRank.

3.6 Considerações finais

Acredita-se que a escolha de uma representação textual é fundamental para o sucesso de aplicações de processamento de línguas naturais. Mais ainda, é preciso que ambos, representação e modelo, possam ser combinados adequadamente. Modelos baseados em CNNs têm apresentado bons resultados na tarefa de seleção de respostas. Contudo, textos longos podem ser um problema para esses modelos, quando são representados pela concatenação de vetores de palavras, resultando em grandes matrizes cujo processamento exige grande quantidade de recursos computacionais. Grafos são uma interessante alternativa para a representação de textos, uma vez que podem capturar relações entre unidades textuais, tais como relações de coesão. Além disso, representações em forma de grafos tendem a ser mais concisas do que representações baseadas em sequências de palavras. Portanto, a combinação de grafos com CNNs parece ser um caminho natural para o processamento de textos longos e complexos que ocorrem em perguntas (e respostas) de consumidores.

O DATASET MILKQA

Apresenta-se, neste capítulo, o *dataset* MilkQA, criado neste projeto de pesquisa para treinamento e avaliação de modelos de seleção de respostas para perguntas de consumidores. Descreve-se o método de preparação dos dados, sua origem, estatísticas e características linguísticas do *dataset*, além de se apresentar uma definição para um dos objetos de estudo desta pesquisa: as perguntas de consumidores.

4.1 Considerações iniciais

Há vários *datasets* de QA disponíveis, alguns, grandes, chegam a conter até 100 mil perguntas. No entanto, quase todos esses *datasets* são formados por perguntas bastante simples, que, ou são formuladas artificialmente, ou são derivadas de consultas submetidas a mecanismos de busca da Internet ([Tabela 2](#)). Não se encontram, nesses *datasets*, perguntas complexas, como as perguntas de consumidores.

Há ainda trabalhos em que são usados arquivos brutos, isto é, que não passaram por um processo de seleção. Arquivos obtidos de *sites* de perguntas e respostas, como o AskUbuntu (ver [Subseção 2.3.3](#)), são um exemplo. Em arquivos originados de fóruns técnicos, como esse, os textos naturalmente contêm elementos estranhos como exemplos de código fonte e amostras de *logs* de erro. Mesmo que, em algumas situações, ignorar tais elementos não traga prejuízos, uma análise cautelosa revelará que, muitas vezes, eles também precisam ser considerados parte do texto. No entanto, dada sua natureza não-linguística, não é possível processá-los como se fossem trechos de língua natural. A situação é análoga a interpretar um texto que faz referência ao conteúdo de uma imagem ou de uma tabela.

Outros arquivos brutos, originados de fóruns não-técnicos, como o English, da família StackExchange ([Subseção 2.3.3](#)), não contêm elementos estranhos. Contudo, sem

um processo de seleção e análise, é muito difícil caracterizar seu conteúdo. Não é possível afirmar, por exemplo, que a maior parte das perguntas é feita por não-nativos ou que há determinada taxa de perguntas duplicadas. O que se deve esperar de um modelo treinado em dados cujas características principais são desconhecidas? Não se trata aqui de argumentar contra técnicas de aprendizado não-supervisionado. Mas para determinadas tarefas, em que se deseja treinar modelos computacionais capazes de imitar habilidades humanas, como reconhecer uma resposta correta ou identificar objetos em uma imagem, é fundamental que algumas propriedades dos dados sejam conhecidas e que possam se garantir. Principalmente, é fundamental que se encontrem no conjunto de dados exemplos dos fenômenos que se deseja aprender.

Tabela 2 – *Datasets* de seleção de respostas

Dataset	Origem das Perguntas	# Perguntas
SelQA	<i>crowdsourcing</i>	7.904
SQuAD	<i>crowdsourcing</i>	100K
WikiQA	usuários de mecanismos de busca	3.047
MS MARCO	usuários de mecanismos de busca	100K
TREC-QA	usuários de mecanismos de busca + editores	227
InsuranceQA	usuários (limite de uma frase)	16.889
Págico	formuladas por editores	153
MilkQA	emails de consumidores	2.657

Fonte: [Criscuolo et al. \(2017\)](#).

Nota: Esses *datasets* são descritos na [Subseção 2.4.1](#).

Um exemplo da importância de bons *datasets* para o treinamento de modelos computacionais é o ImageNet ([DENG et al., 2009](#)), um *dataset* organizado na forma de uma ontologia de imagens, montada sobre a estrutura da WordNet ([MILLER, 1995](#)). O ImageNet apresenta importantes propriedades que favorecem o desenvolvimento de modelos, como quantidade e diversificação de exemplos e qualidade das anotações. Diversos modelos capazes de identificar e localizar objetos em imagens foram treinados sobre esse que se tornou um importante *dataset* na área de visão computacional.

O *dataset* MilkQA, apresentado neste trabalho, ainda tem tamanho modesto quando comparado a outros *datasets* de QA ([Tabela 2](#)), mas apresenta duas importantes propriedades: variedade linguística e qualidade de conteúdo. Enquanto a variedade linguística vem da própria origem dos dados, a qualidade do conteúdo é resultado de um trabalho de seleção de exemplos, guiado por uma definição clara do objeto de pesquisa. Desde o início, o MilkQA foi construído para servir de base ao estudo das perguntas de consumidores.

4.1.1 Perguntas de consumidores: uma definição

É fato que as perguntas de consumidores se distinguem de outros tipos de pergunta abordados na área de QA, como se pode observar nos exemplos apresentados na Seção 2.5 (Quadros 8 e 9). Contudo, não se encontra na literatura uma definição clara para esse objeto de pesquisa, que possa ser usada para a compilação de um *dataset*.

A princípio, chama a atenção o tamanho dessas perguntas. Porém, em análises preliminares, esse critério não foi suficiente para sua distinção. Tampouco a presença de uma descrição de problema foi suficiente para distingui-las. Há diversos casos em que perguntas acompanhadas dessas descrições podem ser isoladas sem prejuízo algum, tornando-se semelhantes às perguntas normalmente encontradas em *datasets* de QA.

Depois de uma tentativa inicial, mal sucedida, de compilação de um *dataset* de perguntas de consumidores, ficou clara a necessidade de uma definição que pudesse servir de crivo em um processo de seleção de exemplos. Como resultado da análise de dados no *dataset* MilkQA, contrastando-os com dados de outros *datasets* de seleção de respostas, propomos a definição a seguir.

Definição 3 (Perguntas de Consumidores). Considera-se uma pergunta de consumidor, um segmento de texto que atenda a pelo menos um dos critérios: (a) contenha uma descrição de um problema, ou de um contexto, e uma ou mais sub-perguntas que dependam dessa descrição prévia para serem compreendidas completamente; ou (b) contenha duas ou mais sub-perguntas relacionadas, que reforcem uma a outra; de maneira que uma sub-pergunta isolada não tenha a mesma interpretação.

Pela definição proposta, a principal característica de uma pergunta de consumidor, não é a presença da descrição de um contexto, nem mesmo seu tamanho, apesar de serem usualmente mais longas. O que realmente caracteriza um segmento de texto como uma pergunta de consumidor são as relações de complementação de sentido entre frases usadas para expressar uma necessidade de informação.

Exemplos de perguntas de consumidores são apresentados na Seção 4.3.

4.2 Origem e preparação do *dataset*

Os dados do MilkQA foram originalmente coletados pela Empresa Brasileira de Pesquisa Agropecuária (Embrapa). Como todas as unidades de pesquisa da Embrapa, a Embrapa Gado de Leite¹ mantém um Serviço de Atendimento ao Cidadão (SAC), nesse caso, dedicado a atender qualquer cidadão interessado em assuntos relacionados à produção leiteira. O SAC recebe um grande número de mensagens de e-mail, que são todas

¹ Embrapa Gado de Leite: <<https://www.embrapa.br/gado-de-leite>>

arquivadas, com suas respectivas respostas, após receberem rótulos descrevendo seu conteúdo, tais como **Raças** e **Doenças**. Foram identificados 58 rótulos diferentes e, em média, são aplicados dois rótulos a cada mensagem².

O *dataset* MilkQA é derivado de um arquivo contendo aproximadamente 27 mil pares de mensagens (mensagens recebidas e respectivas respostas), coletadas entre os anos de 2003 e 2012 pelo SAC da Embrapa Gado de Leite. As perguntas foram formuladas por milhares de pessoas, de diferentes níveis de letramento e que atuam em diferentes ramos de atividade (veterinários, agrônomos, produtores de leite, estudantes, etc.), enquanto as respostas foram elaboradas por especialistas da Embrapa.

O SAC da Embrapa Gado de Leite funciona como um balcão de atendimento, recebendo diversos tipos de solicitações: de perguntas sobre gado leiteiro a pedidos de emprego. No entanto, apenas perguntas de consumidores foram consideradas para a criação do MilkQA. Assim, muitas mensagens contidas no arquivo inicial, que não representavam necessidades de informação sobre a atividade leiteira, foram descartadas. Um processo de seleção foi conduzido em duas fases. Primeiro, mensagens com baixo potencial de aproveitamento foram descartadas por um processo automático. Em seguida, as mensagens restantes passaram por um cuidadoso processo de seleção e limpeza manual.

A filtragem automática foi baseada em critérios estatísticos, determinados pelos rótulos das mensagens. Uma amostra aleatória do arquivo inicial, contendo 642 pares de mensagens, foi analisada manualmente para se determinar a taxa de aproveitamento de cada rótulo. Essa análise estatística revelou que mensagens marcadas com certos rótulos – como **Treinamento** e **Estágio**, por exemplo – eram raramente selecionadas para o *dataset*. Assim, mensagens marcadas com rótulos de baixo aproveitamento foram descartadas, reduzindo consideravelmente a quantidade de trabalho manual, a ser realizada por anotadores humanos. Para validar o processo de filtragem automática, realizou-se uma análise estatística em sentido inverso: uma amostra aleatória das mensagens descartadas foi revisada manualmente. Uma vez que poucas mensagens úteis haviam sido descartadas, e o processo foi considerado válido. Também descartadas cópias de mensagens, que haviam sido armazenadas mais de uma vez, provavelmente por alguma falha no processo de arquivamento. Após essa fase de pré-seleção automática, restaram em torno de 10 mil pares de mensagens.

Na segunda fase, as mensagens restantes foram selecionadas por três anotadores, que realizaram simultaneamente duas tarefas: ao mesmo tempo que rejeitavam mensagens que não atendiam aos critérios de seleção, os anotadores também anonimizaram e limparam as mensagens selecionadas para o *dataset*. O critério adotado para seleção

² A existência de múltiplos rótulos por mensagem pode fazer do MilkQA um *dataset* também interessante para o problema de classificação multi-rótulo (*multi-label classification* – MLC) (TSOUMAKAS; KATAKIS; VLAHAVAS, 2010).

de mensagens foi a aplicação da definição de *pergunta de consumidor* (Subseção 4.1.1). Para garantir a uniformidade das decisões tomadas pelos três anotadores, também foi elaborado um manual contendo diretrizes para o descarte e limpeza de mensagens. Os principais motivos para o descarte de mensagens, listados no manual, foram:

- indicações de livros, ou outras fontes, como única informação contida em uma resposta;
- mensagens solicitando serviços prestados pela Embrapa, como análise de carrapatos ou venda de mudas de capim;
- mensagens que fossem continuções de contatos anteriores, em que o contexto inicial da conversa tivesse sido perdido.

Por meio de orientações como essas, os anotadores puderam compartilhar uma visão comum a respeito de sua tarefa.

Figura 16 – Ferramenta de limpeza e seleção de mensagens

Fonte: Elaborada pelo autor.

Nota: Nomes de pessoas e informações de contato são fictícios.

Para facilitar esse trabalho manual foi desenvolvida a ferramenta ilustrada na Figura 16. A *questioned* (*Question Editor*), como foi chamada a ferramenta, foi desenvolvida para elevar a produtividade dos anotadores durante o processo de seleção e para

minimizar as chances de erro humano no processo de anonimização. Como pode ser visto na figura, a ferramenta carrega mensagens aos pares, permitindo que o anotador visualize e possa alterar simultaneamente uma pergunta e sua resposta.

Para a anonimização das mensagens, foram removidos nomes, informações de contato e outros dados particulares que pudessem identificar seus autores, como nomes e localização de propriedades rurais. A realização dessa tarefa é auxiliada pela ferramenta, onde expressões que possam facilitar a identificação desses elementos são exibidas em destaque. Também foram removidos dados não relacionados às perguntas, como propaganda e assinaturas corporativas, tipicamente encontradas em mensagens de e-mail. Alguns desses dados podem ser identificados por meio de expressões regulares e são automaticamente removidos pela ferramenta. Como resultado desse trabalho manual, foram obtidos pares de mensagens contendo apenas perguntas e respostas relacionadas à atividade de produção leiteira.

A versão atual do MilkQA contém 2.657 pares de mensagens anonimizadas. Para se obter esse total, aproximadamente metade dos 10 mil pares de mensagens, resultantes da fase de filtragem automática, foi examinada pelos anotadores, com taxa de aproveitamento em torno de 53%. Os rótulos aplicados com maior frequência às mensagens contidas na versão atual do *dataset* são mostrados no gráfico da [Figura 17](#). Pode-se verificar nesse gráfico, por exemplo, que o rótulo **Alimentação** foi aplicado a aproximadamente 35% das mensagens.

Figura 17 – Frequência dos rótulos no MilkQA

Fonte: Dados da pesquisa.

Nota: Rótulos com frequência inferior a 5% foram omitidos.

Pelos rótulos exibidos no gráfico, é possível ter uma visão geral do conteúdo das mensagens. A frequência dos rótulos **Forrageiras**, **Cana + Uréia**, **Capim Elefante**, **Pastagens** e **Silagem**, por exemplo, reforçam a interpretação de que há um grande número de mensagens relacionadas à alimentação animal. Outros rótulos, como **Plan. Propriedade**³ e **Manejo Rebanho**, sugerem que o aumento e a manutenção da produção de leite também são preocupações frequentes. As frequências de cada rótulo refletem claramente que as perguntas selecionadas foram motivadas pelo interesse dos consumidores em obter conhecimento sobre o domínio de gado leiteiro.

4.3 Perguntas de consumidores no MilkQA

Para destacar as características de perguntas encontradas no MilkQA, propomos um contraste entre exemplos de perguntas de consumidores, extraídas deste *dataset*, e exemplos de perguntas objetivas, extraídas de outros *datasets*. Os exemplos apresentados no [Quadro 11](#) foram extraídos do WikiQA e do InsuranceQA, e ilustram características comuns de perguntas encontradas nesses *datasets*. Normalmente, são perguntas representadas por frases interrogativas, que são curtas, diretas e completas. No [Quadro 12](#) são apresentadas perguntas extraídas do MilkQA. Trata-se de pequenos textos, formados por múltiplas frases, como tipicamente ocorre com perguntas de consumidores.

Quadro 11 – Perguntas objetivas (não-consumidor)

- (1) O que causa a destruição da camada de ozônio?
- (2) Qual é a taxa de mortes por raios?
- (3) Quais são os diferentes tipos de seguro residencial?

Fontes: Adaptadas dos *datasets* WikiQA ([YANG; YIH; MEEK, 2015](#)) (1, 2) e InsuranceQA ([FENG et al., 2015](#)) (3).

O tamanho médio de perguntas de consumidores normalmente é algumas vezes maior do que o tamanho de perguntas objetivas. Na [Tabela 3](#), são apresentadas estatísticas sobre o tamanho de perguntas encontradas no MilkQA comparadas com estatísticas de perguntas de outros dois *datasets* de seleção de respostas. Os tamanhos mínimo, médio e máximo, em número de palavras, são apresentados para cada *dataset*. As colunas 50-p e 99-p referem-se ao tamanho das perguntas no 50º e 99º percentis, respectivamente. Pode-se verificar na tabela, por exemplo, que 99% das perguntas no WikiQA contêm 16 palavras ou menos. São encontradas no MilkQA algumas perguntas bastante curtas. Essas são as responsáveis pelo pequeno número de palavras indicado na primeira coluna da

³ Planejamento da Propriedade

Quadro 12 – Perguntas de consumidor extraídas do MilkQA

- (1) Tenho algumas vacas leiteiras e percebi que tem umas que caíram bastante o leite. Fiz o teste c.m.t e constatei em alguns quartos de teto a presença de mastite. (a) *Nao sei qual remedio aplicar.* (b) *Melhor injetavel ou via teto?* (c) *Podem ajudar?*
- (2) Tenho um pequeno rebanho de leite. Comecei a notar o aparecimento de verrugas nas vacas, que comecam como nodulos e depois ficam espiculadas, atingindo um tamanho de ate 3 cm e caem. (a) *Solicito se possivel o diagnostico e o tratamento.* Atenciosamente.
- (3) Gostaria de obter informações sobre capim elefante. (a) *Rendimento por m2 para capineira adubada e irrigada?* Quantos cortes anuais nesta condições? (b) *Valores nutricionais?* (c) *Altura de corte?* Onde posso adquirir mudas ou sementes? etc.

Fonte: Dados da pesquisa.

Quadro 13 – Perguntas de consumidor sem contexto explícito

- (1) Bezerros podem comer aveia? É melhor misturar com algum mineral?
- (2) Posso aplicar ivermectina em vacas em lactação? (a) *É ruim para o bezerro?* (b) *E para seres humanos?*
- (3) (a) *alimentação adequada para maior produção de leite .* (b) *se possivel com a inclusao da cana.* gado de leite semi-confinado

Fonte: Dados da pesquisa.

tabela. Por outro lado, perguntas realmente longas, contendo centenas de palavras, também são encontradas no MilkQA. Nesses casos, sub-perguntas são encontradas em meio a verdadeiros relatos de problemas. Na [Tabela 4](#), são apresentadas estatísticas análogas para as respostas, que são ainda mais longas do que as perguntas.

Tabela 3 – Tamanhos de Perguntas: comparação de estatísticas

Dataset	Palavras por Pergunta				
	mínimo	média	50-p	99-p	máximo
WikiQA	2	7	7	16	23
InsuranceQA	2	7	7	14	57
MilkQA	4	57	46	217	681

Fonte: Dados da pesquisa.

Perguntas de consumidores são normalmente compostas por uma descrição de um

Tabela 4 – Tamanhos de Respostas: comparação de estatísticas

Dataset	Palavras por Resposta				
	mínimo	média	50-p	99-p	máximo
WikiQA	1	22	21	57	165
InsuranceQA	15	100	78	395	1.180
MilkQA	8	237	157	688	3.427

Fonte: Dados da pesquisa.

contexto, e por uma ou mais sub-perguntas. A maior parte das perguntas encontradas no MilkQA segue esse padrão, representado pelos exemplos no [Quadro 12](#). Pode-se verificar que o exemplo (1) contém três sub-perguntas, relacionadas ao problema de queda de produção de leite, causado por uma doença (mastite). Mais ainda, muitas sub-perguntas no MilkQA são apresentadas indiretamente, como a sub-pergunta (a), do exemplo (1), apresentada como uma frase afirmativa, que é equivalente à interrogação “*Qual remédio eu devo aplicar?*”. Outra construção indireta bastante usada no lugar de formas interrogativas é “*Eu gostaria de saber...*”.

Também ocorrem no MilkQA diversas perguntas onde um contexto não é fornecido explicitamente. Esses casos são representados pelos exemplos no [Quadro 13](#), em que cada uma das três perguntas é composta por frases inter-relacionadas e complementares. Ainda é possível observar nesse quadro mais uma característica desafiadora das perguntas de consumidores: as solicitações implícitas, representadas pela pergunta (3). Na sub-pergunta (3.a), particularmente, apenas o tema da pergunta é fornecido, ficando implícita a solicitação por informações.

Pelos exemplos nos Quadros [12](#) e [13](#), pode-se observar que, isoladas, raramente as sub-perguntas têm sentido completo. O sentido de uma sub-pergunta – como [Quadro 12\(1.b\)](#) e [Quadro 13\(2.a,3.a\)](#), por exemplo – normalmente depende de informações encontradas no contexto ou em outras sub-perguntas. Essa dependência ocorre por meio de mecanismos linguísticos de coesão textual, tais como a elipse.

De maneira informal, pode-se descrever uma elipse como algo que é omitido, mas que mesmo assim pode ser compreendido. Esse fenômeno ocorre em construções linguísticas cuja estrutura é tal que se possa pressupor algum item anterior, que deverá servir como fonte das informações que faltam ([HALLIDAY; HASAN, 1976](#)). A seguir, são apresentados casos de elipse, identificados nos exemplos de perguntas de consumidores apresentados anteriormente, nos Quadros [12](#) e [13](#). Em cada caso, os itens pressupostos são indicados entre [colchetes].

(4.1) • Altura de corte?

[Quadro 12\(3.c\)](#)

- Altura de corte [*do capim elefante*]?
- (4.2) • É ruim para o bezerro? Quadro 13(2.a)
- [*Aplicar ivermectina em vacas em lactação*] é ruim para o bezerro?
- (4.3) • E para seres humanos? Quadro 13(2.b)
- E [*aplicar ivermectina em vacas em lactação*] [*é ruim*] para seres humanos?

O processamento computacional de fenômenos linguísticos relacionados à coesão textual pode ser bastante desafiador. A abordagem direta da elipse, em particular, pode ser viável apenas em circunstâncias bastante específicas, como as que foram descritas na Seção 2.5, a respeito do método proposto por Kilicoglu, Fiszman e Demner-Fushman (2013). Além desse fenômeno, que talvez seja o mais desafiador, outros como a contiguidade semântica (KOCH, 1999) – ou “colocação”, segundo Halliday e Hasan (1976) – também podem ser observados nas perguntas contidas no MilkQA. A contiguidade semântica consiste no estabelecimento de uma relação entre duas frases por meio da ocorrência de termos do mesmo campo semântico. Um caso contiguidade pode ser observado no exemplo (2), no Quadro 13, em que o termo *bezerro* estabelece uma relação entre segunda frase e a primeira, na qual ocorrem os termos *vacas* e *lactação*.

Outros problemas comuns em perguntas do MilkQA são os erros ortográficos, gramaticais e o mau uso da pontuação, como se pode verificar nos exemplos. Essas características podem ter forte impacto sobre ferramentas como analisadores sintáticos (*parsers*). As respostas, por outro lado, geralmente são textos bem escritos e apresentam poucos problemas. Na realidade, a maioria das respostas são explicações técnicas detalhadas, que tendem a ser reutilizadas com pequenas adaptações, feitas para melhor atender às necessidades de informação representadas pelas perguntas.

4.4 Conjuntos de respostas candidatas

Originalmente, no arquivo do SAC Embrapa, que deu origem ao dataset MilkQA, cada pergunta está associada a apenas uma resposta. No entanto, para a tarefa de seleção de respostas, é necessário que haja um conjunto de respostas candidatas associado a cada pergunta. Uma alternativa para a formação desses conjuntos é extrair respostas candidatas do universo de todas as respostas no dataset. Essa estratégia foi empregada na compilação do dataset InsuranceQA (FENG *et al.*, 2015), onde, inicialmente, também havia apenas uma resposta associada a cada pergunta.

Contudo, durante o processo de seleção, foram identificados no dataset grupos de respostas quase idênticas, com pequenas diferenças entre si, geradas pela alteração de algumas frases. Esse fato se deve ao uso de modelos de resposta pelo SAC. Agrupamentos

de respostas similares podem causar problemas para a avaliação de modelos de seleção, caso uma resposta correta venha a conflitar com uma negativa semelhante, contida no mesmo conjunto de candidatas. Tal cenário é bastante provável, quando as respostas candidatas são extraídas aleatoriamente do universo do *dataset*. Para evitar esse problema, não foi permitido que houvesse mais de uma resposta do mesmo grupo em cada conjunto de candidatas no MilkQA. A implementação dessa política exigiu a identificação dos agrupamentos de respostas similares, ilustrada na Figura 18.

Figura 18 – Agrupamento de respostas similares

Fonte: Elaborada pelo autor.

O processo de agrupamento se inicia com a vetorização de todas as respostas contidas no *dataset*, que são representadas inicialmente por vetores tf-idf⁴ de 4701 elementos. Nessa fase, não foram empregadas técnicas como remoção de *stopwords* e lematização com o intuito de reduzir o tamanho do vocabulário. Em vez disso, um modelo autocodificador (*autoencoder*) (VINCENT *et al.*, 2008) foi treinado sobre os vetores tf-idf e, em seguida, usado para reduzir sua dimensionalidade, recodificando cada vetor com apenas 100 elementos. Um algoritmo de agrupamento por densidade (DBSCAN) (ESTER *et al.*, 1996), com parâmetros ajustados empiricamente, foi executado com o objetivo de se identificar agrupamentos coesos, formados por elementos próximos (quase idênticos). Por esse processo, foram identificados 97 agrupamentos contendo em média 6 respostas cada um. O maior e o menor agrupamento contêm 104 e 2 respostas, respectivamente, sendo que 75% dos agrupamentos contêm 4 respostas ou menos. Portanto, os grandes agrupamentos representam 25% do total.

As representações vetoriais reduzidas também foram usadas para na formação dos conjuntos de respostas candidatas. Cada conjunto contém $n = 50$ candidatas, incluindo uma resposta correta, que foi originalmente dada pelo SAC para a pergunta correspondente. As demais candidatas são as $n - 1$ vizinhas mais próximas da resposta correta, a

⁴ Tf-idf: *Term frequency-inverse document frequency*

partir de uma distância inicial, determinada empiricamente. Essa estratégia visa reproduzir situações desafiadoras, em que modelos de seleção de respostas precisam distinguir respostas similares. A relação de proximidade entre as respostas foi medida pela distância euclidiana, calculada a partir das representações vetoriais reduzidas, criadas no processo de agrupamento. Acreditamos que a escolha de 50 elementos para o tamanho do conjunto de candidatas seja compatível com as características do *dataset*. Não se verifica, na literatura, consenso ou regra para o estabelecimento desse número. Há trabalhos em que foram usadas 20 candidatas (SANTOS *et al.*, 2015; LEI *et al.*, 2016) e outros em que foram usados conjuntos muito maiores, contendo 500 candidatas (FENG *et al.*, 2015).

4.5 Considerações finais

O MilkQA é um importante resultado deste projeto de pesquisa e será disponibilizado como um recurso público no *site* do nosso grupo de pesquisas, o Núcleo Interinstitucional de Linguística Computacional (NILC)⁵. As características das perguntas de consumidores contidas neste *dataset*, inicialmente apresentado em Criscuolo *et al.* (2017), impõem diversos desafios a modelos computacionais. Alguns desses desafios são abordados nesse trabalho, e discutidos no Capítulo 6, onde são apresentados resultados de experimentos envolvendo o MilkQA.

⁵ O *dataset* MilkQA será disponibilizado em: <<http://www.nilc.icmc.usp.br/nilc/index.php/milkqa/>>

MODELO SLIMRANK

Neste capítulo, descreve-se o modelo SlimRank, proposto neste trabalho. Apresenta-se inicialmente a perspectiva de representação textual adotada no SlimRank, seguida de uma descrição detalhada do modelo.

5.1 Considerações iniciais

O modelo SlimRank foi idealizado para atender a dois requisitos fundamentais: (i) evitar o descarte arbitrário de informações e (ii) modelar textos como redes de relações semânticas.

Em diversos trabalhos ([YANG; YIH; MEEK, 2015](#); [LEI *et al.*, 2016](#); [YIN *et al.*, 2016](#), e outros), adota-se a prática de truncar textos de entrada com o propósito de poupar recursos computacionais. Essa prática é observada mesmo em trabalhos avaliados sobre *datasets* de sentenças curtas, como o WikiQA ([YANG; YIH; MEEK, 2015](#)). Em um *dataset* como o MilkQA, que contém textos mais longos, o impacto do truncamento tende a ser maior. Um dos objetivos do modelo SlimRank é criar representações sucintas dos textos de entrada, de maneira que todo conteúdo de um texto seja considerado, mas apenas as informações mais importantes sejam usadas para representá-lo. Pretende-se, assim, poupar recursos computacionais e, ao mesmo tempo, evitar o descarte de informações úteis.

Em modelos de CNNs, a representação de textos é tradicionalmente baseada na adjacência de palavras. Isto é, as matrizes de entrada são criadas pela concatenação de vetores correspondentes a palavras adjacentes no texto. Espera-se, assim, que os filtros de convolução aprendam a detectar padrões de adjacência entre palavras – como foi discutido na [Subseção 3.4.1](#). Considerando as características de textos encontrados no *dataset* MilkQA, nos quais são frequentes as relações de dependência entre elementos

textuais, propõe-se substituir a representação em nível sintático, baseada na adjacência de palavras, por uma representação em nível semântico, baseada em grafos, em que sejam capturadas relações semânticas presentes no texto. Espera-se que, nessa nova representação, os filtros de convolução passem a detectar relações de proximidade semântica entre elementos textuais.

Essa proposta é fundamentada nas definições de *texto* e *coesão*, apresentadas por Halliday e Hasan (1976), que também serviram de fundamentação para o uso de grafos no modelo TextRank (MIHALCEA; TARAU, 2004), discutido na Seção 3.5. De acordo com essas definições, um texto é uma unidade semântica de sentido, cujos elementos são relacionados entre si por meio de mecanismos de coesão. Por sua vez, “a coesão ocorre quando a interpretação de algum elemento no discurso depende da interpretação de outro elemento” (HALLIDAY; HASAN, 1976). Pode-se verificar que essas características são justamente as que foram observadas no *dataset* MilkQA, de acordo com o que foi discutido na Seção 4.3.

Atendendo a esses dois requisitos, propomos o modelo SlimRank, descrito a seguir.

5.2 Modelo SlimRank

Assim como ocorre em outros modelos de seleção de respostas, o SlimRank recebe como entrada um par formado por uma pergunta (Q) e uma resposta candidata (A), e gera como saída um valor numérico, representando o grau de relevância da resposta em relação à pergunta. O modelo SlimRank é uma combinação de uma abordagem de representação textual, baseada em grafos, com uma arquitetura de CNN, conforme é ilustrado na Figura 19.

Figura 19 – Modelo SlimRank

Fonte: Elaborada pelo autor.

Os desafios para a implementação desse modelo estão na definição de métodos

adequados para a construção dos grafos semânticos e, posteriormente, para a seleção de campos receptivos (subgrafos) e para a normalização desses campos em forma vetorial, de maneira que seja possível usá-los como entrada para modelos de CNNs. Nas subseções seguintes, apresentamos métodos propostos neste trabalho para a realização dessas tarefas.

5.2.1 Construção do grafo semântico

O modelo de construção de grafos semânticos descrito nesta seção é usado tanto para perguntas quanto para respostas, portanto, usa-se aqui o termo *texto*, sem distinção de tipo.

Seja $S = \{w_1, w_2, \dots, w_n\}$ o conjunto de palavras de um dado texto, para o qual será construído um grafo semântico. Cria-se então um grafo não-dirigido $G = (V, E)$, em que cada vértice $v_i \in V$ corresponde a uma palavra $w_i \in S$, e em que o peso de cada aresta $e = (v_i, v_j) \in E$ corresponde à distância entre as respectivas palavras w_i e w_j . O grafo G é representado por uma matriz de adjacência A , de tamanho $|S| \times |S|$, cujos elementos $a_{i,j}$ são definidos por:

$$a_{i,j} = d(w_i, w_j) \quad (5.1)$$

onde $d: W \times W \rightarrow \mathbb{R}^+$ é uma função de distância semântica entre duas palavras e o conjunto W é um vocabulário, com $S \subseteq W$.

Algumas propriedades devem ser satisfeitas para que uma função possa ser considerada uma função de distância (HARISPE *et al.*, 2015). Assim, a função d deve ser:

- i. não-negativa: $d(x, y) \geq 0$;
- ii. simétrica: $d(x, y) = d(y, x)$;
- iii. reflexiva: $d(x, x) = 0$ e $\forall y \mid y \neq x: d(x, x) < d(x, y)$.

Além disso, a distância entre dois pontos deve ser a mais curta considerando todos os caminhos possíveis. Em outras palavras, d também deve satisfazer a propriedade:

- iv. desigualdade triangular: $d(x, y) \leq d(x, z) + d(z, y)$.

Todas as palavras do texto são consideradas na construção do grafo. Não há remoção de *stopwords*, por exemplo. Em vez disso, a exemplo de Kenter e Rijke (2015) (ver Subseção 3.3.4), a saliência de cada palavra no cópulo (IDF) é considerada na definição da função de distância. Assim, definimos inicialmente a função $f: W \times W \rightarrow \mathbb{R}$ como:

$$f(w_1, w_2) = \text{idf}(w_1) \text{idf}(w_2) \text{sim}_{\cos}(v(w_1), v(w_2)) \quad (5.2)$$

onde $v(w_i)$ representa um vetor pré-treinado em algum corpus – por um modelo como o Word2vec, por exemplo – e $\text{idf}(w_i)$ é definido como:

$$\text{idf}(w_i) = \log \frac{N}{\text{df}(w_i)}$$

onde $\text{df}(w_i)$ representa o número de documentos no corpus em que ocorre a palavra w_i e N é o número total de documentos. Palavras w_i que não constem no vocabulário do conjunto de vetores pré-treinados são ignoradas.

A função f pode ser interpretada como uma versão da similaridade cosseno, ponderada pela saliência das palavras. Portanto, valores maiores de f indicam relações semânticas mais fortes entre as palavras. Precisamos definir uma função de distância, para ser usada na [Equação 5.1](#), que se comporte de maneira oposta: valores maiores devem indicar relações semânticas mais fracas. Assim, definimos finalmente a função de distância $d: W \times W \rightarrow \mathbb{R}^+$ (que apresenta as propriedades descritas anteriormente) como:

$$d(w_1, w_2) = \max_{t, u \in W} f(t, u) - f(w_1, w_2) \quad (5.3)$$

Por esta definição, para computar a distância entre duas palavras w_1 e w_2 , o valor da similaridade ponderada $f(w_1, w_2)$ é subtraído do maior valor de f verificado entre todos os possíveis pares de palavras (t, u) contidos no vocabulário W .

Assim, A é uma matriz simétrica, que representa um grafo completamente conectado em que o peso $a_{i,j}$ de cada aresta representa a distância entre as palavras $w_i, w_j \in S$, extraídas do texto de entrada. O grafo representado pela matriz A é usado como entrada para o processo de seleção e normalização de campos receptivos, descrito a seguir.

5.2.2 Seleção e normalização de campos receptivos

A seleção de campos receptivos começa com a criação de uma árvore geradora mínima (MST) a partir de um grafo G , representado por uma matriz de adjacência A – conforme as definições apresentadas na [Subseção 5.2.1](#). Apenas as ligações semânticas mais fortes (distâncias mais curtas) são preservadas na MST, que é criada pelo algoritmo de Kruskal ([Algoritmo 1](#))¹. Exemplos de uma pergunta encontrada no *dataset* MilkQA e da MST correspondente são apresentados, respectivamente, no [Quadro 14](#) e na [Figura 20](#).

É possível observar na [Figura 20](#) regiões mais densas, formadas por grupos (*clusters*) de vértices relacionados. Além disso, observa-se também que os conceitos mais salientes (representativos) no texto tendem a ficar no centro dos *clusters*, enquanto palavras menos significativas ocupam as extremidades do grafo. Esse comportamento se deve à maneira como foi definida a função de distância.

¹ Essa forma simplificada do algoritmo de Kruskal foi derivada das versões encontradas em [Cormen et al. \(2009\)](#) e [Kleinberg e Tardos \(2005\)](#).

Algoritmo 1 – Algoritmo de Kruskal**Requer:** que $G = (V, E)$ seja um grafo conectado

- 1: **procedimento** MST-KRUSKAL(G)
- 2: $T \leftarrow \emptyset$
- 3: ordene as arestas de $G.E$ por peso, em ordem crescente
- 4: **para** cada aresta $e \in G.E$, em ordem crescente de peso **faça**
- 5: **se** a introdução de e não criar um ciclo no grafo (V, T) **então**
- 6: $T \leftarrow T \cup \{e\}$
- 7: **fim se**
- 8: **fim para**
- 9: **retorna** T

Figura 20 – Exemplo de MST

Fonte: Elaborada pelo autor.

Comparando o conteúdo do texto no [Quadro 14](#) com sua representação na forma de MST, é possível verificar relações bastante interessantes que foram capturadas pela representação semântica. Alguns exemplos são: (comprei, hectare), (praga, região) e (galhos, metro). Porém, se destaca a relação (combater, praga) que liga corretamente o verbo “combater”, na última frase do texto, ao foco da pergunta (“praga”).

Em vermelho, na [Figura 20](#), estão os vértices que representam os campos receptivos. A seleção desses vértices representativos é feita do centro para as extremidades do grafo, explorando assim o fato de que os vértices mais salientes são encontrados no

Quadro 14 – Pergunta extraída do *dataset* MilkQA

Comprei um terreno de 30 hectares e está todo com uma praga chamado aqui na região de mixirico; uma planta de galhos de mais ou menos um metro com folhas verdes e solta um pó meio amarelado. Queria saber o que tenho que fazer para combater.

Fonte: Dados da pesquisa.

centro. O processo de seleção, apresentado no [Algoritmo 3](#), gera como saída uma lista de vértices representativos, na ordem em que foram selecionados. De maneira geral, o processo de seleção consiste em subdividir um grafo acíclico (uma MST) em componentes conectados – ou seja, subgrafos contendo arestas que permitam formar um caminho entre todos os possíveis pares de vértices – e então encontrar um vértice representativo para cada componente. Esse processo é controlado por uma estrutura de dados do tipo FIFO (*First In, First Out*), isto é, uma fila, representada no algoritmo por $\mathcal{Q}$. O primeiro componente conectado a ser considerado é a MST completa ([Algoritmo 3](#), linhas 5 e 7). Seguindo o algoritmo, para cada componente conectado, encontra-se um vértice representativo, por meio de uma chamada ao procedimento ENCONTRAR-REPRESENTATIVO ([Algoritmo 2](#)), que aplica o critério de centralidade empregado na estrutura de dados métricos Slim-tree ([TRAINA JR. et al., 2000](#)), que pode ser representado formalmente como:

$$v_{\text{repr}} = \arg \min_{v \in M.V} \max_{u \in M.V} d_c(v, u)$$

onde $d_c(v, u)$ representa a distância de v até u , computada como a soma das arestas que formam o caminho $(v, \dots, u)$.

Após a identificação de um vértice representativo, o componente atual é subdividido em componentes conectados menores, que são adicionados à fila $\mathcal{Q}$. A subdivisão ocorre pela remoção das arestas que ligam o vértice representativo, v_{repr} , aos seus N_1 -VIZINHOS – isto é, os vértices adjacentes. Para cada vértice adjacente u , identifica-se um componente conectado, por meio de uma busca em largura. Apenas componentes contendo um certo número mínimo, c_{min} , de vértices são considerados. O processo se repete até que não haja mais componentes conectados na fila $\mathcal{Q}$.

Algoritmo 2 – Identificação de vértice representativo em uma MST

Requer: que $M = (V, T)$ seja uma MST

- 1: **procedimento** ENCONTRAR-REPRESENTATIVO(M)
 - 2: encontre o vértice $v \in M.V$ tal que a maior distância de v até cada $u \in M.V$ seja mínima
 - 3: **retorna** v
-

Algoritmo 3 – Seleção de campos receptivos**Requer:** que $M = (V, T)$ seja uma MST e $c_{\min} \geq 3$

```

1: procedimento SELECIONAR-CAMPOS-RECEPTIVOS( $M, c_{\min}$ )
2:    $R \leftarrow \text{CRIAR-LISTA-VAZIA}()$ 
3:    $Q \leftarrow \text{CRIAR-FILA-VAZIA}()$ 
4:    $M' \leftarrow M$ 
5:   ADICIONAR-FILA( $Q, M'$ )
6:   enquanto não VAZIA?( $Q$ ) faça
7:      $C \leftarrow \text{TIRAR-FILA}(Q)$ 
8:      $v_{\text{repr}} \leftarrow \text{ENCONTRAR-REPRESENTATIVO}(C)$ 
9:     ADICIONAR-LISTA( $R, v_{\text{repr}}$ )
10:     $N \leftarrow N_1\text{-VIZINHOS}(v_{\text{repr}}, C)$ 
11:    para todo  $u \in N$  faça
12:      remover aresta  $(v_{\text{repr}}, u)$  de  $M'.T$ 
13:       $C \leftarrow \text{COMPONENTE-CONECTADO}(u, M')$ 
14:      se  $|C.V| \geq c_{\min}$  então
15:        ADICIONAR-FILA( $Q, C$ )
16:      fim se
17:    fim para
18:  fim enquanto
19:  retorna  $R$ 

```

Os campos receptivos são mapeados em vetores de tamanho fixo k , por meio do processo de normalização, apresentado no [Algoritmo 4](#). Cada campo receptivo é representado por um conjunto contendo o vértice representativo, v_{repr} , e seus vértices adjacentes. Cada vértice nesse conjunto é mapeado para um vetor de números inteiros, U , em ordem crescente de distância até v_{repr} . Isso implica em ter v_{repr} mapeado na primeira posição do vetor U , seguido pelo vértice mais próximo de v_{repr} na segunda posição, e assim por diante. Cada elemento do vetor representa o índice da palavra w_i no vocabulário W , representada pelo vértice v_i . Se o número de vértices em um campo receptivo for menor do que o tamanho k , estipulado para o vetor U , os elementos restantes são preenchidos com o índice da palavra fictícia PAD, incluída no vocabulário especialmente para essa finalidade. Por exemplo, a normalização do campo receptivo representado pelo vértice **praga**, na [Figura 20](#), com $k = 5$, resulta em um vetor U cujos elementos são os índices das palavras $\langle \text{praga}, \text{combater}, \text{região}, \text{PAD}, \text{PAD} \rangle$.

Esse processo de normalização é repetido para todos os campos receptivos de um grafo, transformando cada um em um vetor. Na seção seguinte, descreve-se como esses vetores são transformados em uma matriz que possa ser processada por extratores de *features* de CNNs.

Algoritmo 4 – Normalização de campos receptivos

Requer: que $M = (V, T)$ seja uma MST, que W seja um vocabulário e $k \geq 1$

- 1: **procedimento** NORMALIZAR-CAMPO-RECEPTIVO(v_{repr}, M, W, k)
- 2: seja U um vetor de k números inteiros, com índices $0..(k-1)$
- 3: $N \leftarrow N_1\text{-VIZINHOS}(v_{\text{repr}}, M)$
- 4: $C \leftarrow N \cup \{v_{\text{repr}}\}$
- 5: ordene os vértices $u \in C$ em ordem crescente de distância (peso) de u até v_{repr}
- 6: $i \leftarrow 0$
- 7: **para** cada vértice $u \in C$, em ordem crescente de distância até v_{repr} **faça**
- 8: **se** $i < k$ **então**
- 9: $w \leftarrow \text{WORD}(u)$
- 10: $U[i] \leftarrow \text{IDX-VOCAB}(w, W)$ $\triangleright v_{\text{repr}}$ ocupará a primeira posição, $U[0]$
- 11: **fim se**
- 12: $i \leftarrow i + 1$
- 13: **fim para**
- 14: **se** $|C| < k$ **então**
- 15: **para** $i \leftarrow |C|..(k-1)$ **faça**
- 16: $U[i] \leftarrow \text{IDX-VOCAB}(\text{PAD}, W)$
- 17: **fim para**
- 18: **fim se**
- 19: **retorna** U

5.2.3 Convolução com campos receptivos normalizados

Após passar pelos processos de seleção e de normalização de campos de receptivos, cada texto é transformado em uma lista de vetores contendo índices de palavras em um vocabulário W . Para servir de entrada a uma CNN, esses índices são mapeados em vetores de palavras (*word embeddings*), embutidos (*embedded*) em uma matriz E , de tamanho $(|W| + 1) \times d$, onde d é a dimensão dos vetores de palavras, conforme é ilustrado na Figura 21.

Ao mapear os índices em vetores de palavras, cada um dos vetores – de tamanho fixo k – que representam campos receptivos é transformado em uma matriz de tamanho $k \times d$. Essas matrizes, indicadas na Figura 21 por c_i , são concatenadas em uma matriz X , também de tamanho fixo $nk \times d$, sendo n um parâmetro do modelo que representa o número de campos receptivos comportados pela matriz X . Quando o número de campos receptivos extraídos de um texto é menor do que n , a matriz X é completada com vetores linha representando a palavra PAD.

O SlimRank requer duas matrizes de entrada (ver Figura 19) – uma para a pergunta e outra para a resposta candidata – que podem ter tamanhos diferentes. Em nossos experimentos, normalmente usamos uma quantidade de campos receptivos, n , maior para as respostas, uma vez que normalmente são textos maiores, que contêm mais informações.

Figura 21 – Convolução no modelo SlimRank

Fonte: Elaborada pelo autor.

Uma particularidade do modelo é que cada filtro de convolução f deve ter tamanho $k \times d$, coincidindo com o tamanho dos campos receptivos, e o tamanho do passo de convolução (*stride*) também deve ser k , para que o filtro cubra um campo receptivo inteiro a cada iteração. Como normalmente ocorre em CNNs, os valores computados pelas convoluções passam por uma função não-linear $g()$, tal como a tangente hiperbólica. Um processo completo de convolução resulta em um vetor de *features* x , de n elementos. É possível ter vários filtros de convolução, de maneira que se obtenham vários vetores de *features*. O número de filtros também é um parâmetro do modelo.

A matriz de vetores de palavras (*word embeddings*), E , pode ser iniciada com vetores pré-treinados ou com valores aleatórios. No segundo caso, os vetores de palavras são aprendidos na própria tarefa de seleção de resposta, assim como ocorre nos modelos apresentados por Collobert *et al.* (2011). A fase de geração das MSTs, contudo, requer vetores pré-treinados.

Quadro 15 – Parâmetros do modelo SlimRank

Parâmetro	Descrição
k	tamanho dos campos receptivos
n_q	número de campos receptivos por pergunta
n_a	número de campos receptivos por resposta
c_{qmin}	número mínimo de vértices em componentes conectados para perguntas
c_{amin}	número mínimo de vértices em componentes conectados para respostas

Fonte: Dados da pesquisa.

Uma vez que todos os textos no *dataset* tenham sido transformados em campos receptivos normalizados e que a convolução seja realizada conforme esta descrição, o restante do modelo é treinado como outros modelos baseados em CNNs. É possível, inclusive, criar variações do SlimRank, combinando-o com diferentes modelos de CNNs, como o CNN-std e o CNN-LDC, apresentados na [Subseção 3.4.2](#). Os parâmetros específicos do modelo SlimRank são sumarizados no [Quadro 15](#).

5.3 Considerações finais

Neste capítulo, foi descrito o modelo SlimRank. Partindo de uma visão geral do modelo, foram apresentados todos seus detalhes de construção e de funcionamento. O principal foco do SlimRank é uma mudança na perspectiva de representação textual normalmente empregada em CNNs, que visa evitar o descarte arbitrário de informação e explorar relações intra-textuais, por meio de grafos semânticos.

A seguir, no [Capítulo 6](#), são apresentados experimentos envolvendo o *dataset* MilkQA e o modelo SlimRank.

EXPERIMENTOS E DISCUSSÃO

Neste capítulo, são descritos experimentos realizados com o objetivo de avaliar o *dataset* MilkQA e o modelo de seleção de respostas SlimRank. Inicialmente, descrevem-se os métodos utilizados nos experimentos (Seção 6.1), seguidos pela apresentação dos objetivos e resultados de cada experimento (Seção 6.2 e Seção 6.3) e, finalmente, apresentam-se as discussões (Seção 6.4).

6.1 Preparação dos experimentos

Nesta seção, são descritos recursos e métodos empregados nos experimentos, que são descritos nas subseções seguintes.

6.1.1 Pré-processamento de textos

Os textos contidos no *dataset* MilkQA passam por uma etapa de pré-processamento que visa dar consistência aos vetores de palavras e também evitar o crescimento do vocabulário, devido a variações pouco significativas em alguns tipos de *tokens*, como números, por exemplo. Técnicas semelhantes às que são descritas a seguir foram empregadas por Hartmann *et al.* (2017).

Inicialmente, todos os caracteres numéricos são substituídos por zero, de maneira que o *token* 1994 seja substituído por 0000, por exemplo. Padrões correspondentes a endereços de e-mail, URLs e números de telefone são substituídos, respectivamente, por `_email_`, `_URL_` e `_telefone_`. Todos os sinais de pontuação são removidos, exceto hífen entre palavras, e os espaços em branco são normalizados, para que os *tokens* sejam separados por apenas um espaço em branco. Finalmente, *tokens* com menos de cinco ocorrências são substituídos por `_UNK_`.

6.1.2 Treinamento de vetores de palavras

Em todos os experimentos descritos neste capítulo foram usados vetores de palavras de 300 dimensões, treinados na arquitetura CBOW, do modelo Word2vec (MIKOLOV *et al.*, 2013a). Duas coleções de vetores foram usadas: (i) vetores pré-treinados em textos de domínio geral, disponibilizados no repositório NILC *Embeddings* (HARTMANN *et al.*, 2017), que são indicados nos resultados dos experimentos por “NILC” e (ii) vetores treinados no próprio domínio de aplicação, usando textos do *dataset* MilkQA e outros, descritos a seguir. Esses últimos são indicados nos resultados por “Embrapa”.

Para o treinamento de vetores de domínio, foram usados textos disponíveis publicamente no *site* da Embrapa, além de textos contidos no arquivo completo que deu origem ao *dataset* MilkQA¹.

Na etapa de coleta de textos, cogitou-se empregar um identificador de idiomas para excluir do *corpus* os textos em língua estrangeira, uma vez que há diversas publicações científicas da Embrapa também em inglês e espanhol. Contudo, apesar da realização de experimentos bem sucedidos com um classificador de idiomas – dos quais resultou uma publicação (CRISCUOLO; ALUÍSIO, 2017a) –, o uso de textos científicos acabou sendo desconsiderado pela dificuldade de coletá-los em quantidades significativas para o treinamento de vetores.

Os textos coletados passaram por uma etapa de pré-processamento, conforme a descrição apresentada na Subseção 6.1.1. Obteve-se, assim, um *corpus* contendo 14.240.909 *tokens*, empregado no treinamento de vetores Word2vec-CBOW de tamanho 300, como foi mencionado anteriormente.

Quadro 16 – Analogias resolvidas por vetores treinados no domínio

Pergunta	Resposta
leite:queijo $\equiv$ uva:X	vinho
vinho:tinto $\equiv$ queijo:X	mussarela
vacas:mosca-do-chifre $\equiv$ galinhas:X	piolho
farelo:soja $\equiv$ silagem:X	sorgo

Fonte: Dados da pesquisa.

Os vetores obtidos foram avaliados qualitativamente, por meio da tarefa de resolução de analogias (ver Subseção 3.3.1). Assim, foi possível verificar a capacidade dos vetores de capturar relações semânticas entre conceitos do domínio. Alguns exemplos de analogias resolvidas nesse processo de avaliação são apresentados no Quadro 16.

¹ Esses textos foram usados exclusivamente para o treinamento de vetores de palavras e, com exceção das perguntas e respostas contidas no MilkQA, não serão publicados neste projeto.

6.1.3 Modelos e baselines de seleção de respostas

Na avaliação do *dataset* MilkQA, foram empregados dois modelos de seleção de respostas baseados em CNNs e dois métodos *baseline*. Os dois *baselines* adotados – que ocorrem comumente na literatura – foram:

- i. WWM (*Weighted Word Matching*), que computa a soma da IDF de cada palavra contida em uma pergunta, que também ocorra em uma resposta candidata; e
- ii. WSE (*Weighted Sum of Embeddings*), que computa a similaridade cosseno entre uma pergunta e uma resposta candidata, representadas cada uma por um vetor, que é obtido pela soma dos vetores das palavras contidas no texto, multiplicados pela respectiva IDF.

Mais formalmente, no *baseline* WSE, um texto T é representado como um vetor u obtido por:

$$u = \sum_{w \in T} \text{idf}(w) v(w)$$

onde $v(w)$ representa um vetor correspondente à palavra w . O método *baseline* WSE foi implementado por [Feng et al. \(2015\)](#) e, de acordo com nossos experimentos, gera resultados ligeiramente melhores do que aqueles obtidos pela similaridade cosseno entre centroides (ver [Subseção 3.3.4](#)).

Ainda na avaliação do *dataset* MilkQA, foram usados dois modelos baseados em CNNs: CNN-std e CNN-LDC, ambos descritos detalhadamente na [Subseção 3.4.2](#). O modelo CNN-std representa uma arquitetura padrão de CNN para processamento de textos, enquanto o CNN-LDC ([WANG; MI; ITTYCHERIAH, 2016](#)) representa o estado da arte na tarefa de seleção de respostas. Também duas variações do modelo SlimRank foram consideradas nas avaliações: SR-std e SR-LDC, análogas aos modelos CNN-std e CNN-LDC. Após a etapa de convolução, que apresenta características específicas do SlimRank, o restante do modelo se comporta como outras CNNs encontradas na literatura, sendo possível, inclusive, a criação de variações, como essas que foram mencionadas. Para que os resultados obtidos possam ser comparados, esses quatro modelos baseados em CNNs (CNN-std, CNN-LDC, SR-std e SR-LDC) foram treinados com os mesmos hiperparâmetros e pelo mesmo processo, que é descrito a seguir. Detalhes específicos relacionados ao modelo SlimRank são indicados pontualmente.

Convém lembrar que, na tarefa de seleção de respostas, para cada pergunta há um conjunto de respostas candidatas, em que cada resposta é normalmente rotulada com os valores 0 ou 1, indicando uma resposta negativa (incorreta) ou positiva (correta), respectivamente. Uma vez que, neste trabalho, abordamos a seleção de respostas como um problema de regressão, os rótulos (*labels*) 0 e 1 são interpretados como os extremos do intervalo $[0, 1] \in \mathbb{R}$, que contém um grau (*score*) de relação semântica entre uma pergunta

e uma resposta candidata, a ser computado pelo modelo. A saída da camada de regressão é ajustada ao intervalo $[0,1]$ pela aplicação da função sigmoide. Todos os modelos são treinados por retropropagação (*backpropagation*), usando o otimizador Adam (KINGMA; BA, 2014) para minimizar o erro quadrático médio (MSE²), computado como

$$MSE = \frac{1}{N} \sum_{i=1}^N (label_i - score_i)^2$$

onde N é o número de pares (pergunta, resposta) em cada lote (*batch*) de treinamento.

Para a realização dos experimentos, os dados do *dataset* MilkQA foram particionados em três conjuntos, conforme é indicado na Tabela 5. A escolha dos tamanhos de cada partição foi feita de modo a manter a maior parte dos exemplos no conjunto de treinamento. Particularmente, a escolha de 50 exemplos para o conjunto de desenvolvimento visa evitar atrasos no processo de treinamento, uma vez que a avaliação desse conjunto ocorre diversas vezes. Acreditamos que esse tamanho represente um compromisso razoável entre a capacidade de estimar a eficiência do modelo durante o processo de treinamento e o tempo computacional consumido por essa atividade. Para cada uma das 50 perguntas há 50 respostas candidatas. Portanto, são avaliados 2.500 pares de (pergunta, resposta) no conjunto de desenvolvimento.

Tabela 5 – Partições do *dataset* MilkQA

Partição	# Perguntas
Treinamento	2.307
Desenvolvimento	50
Testes	300

Fonte: Dados da pesquisa.

No processo de treinamento, são usados mini-lotes (*mini-batches*) formados por n triplas do tipo (q, a^+, a^-) , onde q , a^+ e a^- representam, respectivamente, uma pergunta, uma resposta positiva e uma resposta negativa. Internamente, cada uma dessas triplas é transformada em dois pares: (q, a^+) e (q, a^-) . O emprego dessa estratégia – tanto no caso do *dataset* MilkQA quanto no caso do WikiQA, que também é usado nas avaliações – se deve ao fato de que *datasets* de seleção de respostas são naturalmente desbalanceados. No MilkQA, por exemplo, há apenas uma resposta positiva em cada conjunto de 50 candidatas – ou seja, há apenas 2% de candidatas positivas. A amostragem de respostas negativas é uma maneira de contornar esse problema: para a criação das triplas, são selecionadas aleatoriamente uma resposta positiva e uma negativa para cada pergunta q . Dessa maneira, explora-se toda a variedade de respostas contida no *dataset*, ao mesmo

² Do inglês, *Mean Squared Error* (MSE).

tempo em que se evita a indução de uma convergência prematura do modelo, devida ao desbalanceamento de dados.

O processo de treinamento é cíclico. Isto é, quando o conjunto de dados de treinamento chega ao fim, a seleção de lotes é reiniciada a partir do início. O treinamento para quando não houver melhora na MAP³ após duas avaliações consecutivas do conjunto de desenvolvimento (*early stopping*). A primeira avaliação do conjunto de desenvolvimento ocorre após 3000 iterações de treinamento (3000 mini-lotes) e uma nova avaliação é feita a cada 500 iterações. Para evitar o problema de *overfitting* aplica-se a técnica de regularização *dropout* (SRIVASTAVA *et al.*, 2014) na camada de regressão, com taxa de descarte de 20%. Isso implica em dizer que 80% dos neurônios de entrada são preservados.

Em todos os modelos, são usados 150 filtros de convolução. Nos modelos CNN-std e CNN-LDC, são 50 filtros de cada um dos tamanhos 1, 2 e 3 – análogos a unigramas, bigramas e trigramas – e nos modelos SR-std e SR-LDC são 150 filtros de tamanho k (tamanho dos campos receptivos, como requer o modelo). Em todos os filtros a não-linearidade é obtida pela função tangente hiperbólica.

Os modelos baseados em CNNs podem ser iniciados com vetores pré-treinados ou com vetores aleatórios, que são ajustados posteriormente na própria tarefa (COLLOBERT *et al.*, 2011). Quando são usados vetores pré-treinados, podem ocorrer no *dataset* palavras fora do vocabulário (OOV⁴) em que foram treinados os vetores. Os vetores correspondentes a essas palavras são iniciados aleatoriamente, segundo uma distribuição Normal com parâmetros média e desvio padrão estimados no próprio conjunto de vetores pré-treinados – técnica semelhante é usada por Severyn e Moschitti (2015). Nos casos em que os vetores são ajustados na própria tarefa, indicados por “random” nas tabelas de resultados, seus valores iniciais são extraídos aleatoriamente do intervalo $[-1, 1] \in \mathbb{R}$, segundo uma distribuição Uniforme.

O desempenho dos modelos de seleção de resposta é quantificado pelas medidas MAP, computada pela ferramenta `trec_eval`, e P@1, computada por uma ferramenta desenvolvida neste projeto (detalhes sobre avaliação na Subseção 2.4.2).

Com exceção dos *baselines*, todos os modelos foram implementados no *framework* TensorFlow (ABADI *et al.*, 2016) e treinados em uma GPU⁵ com 4 GiB de memória.

6.2 Avaliação do dataset MilkQA

Nesta seção, descrevem-se dois experimentos que visam avaliar a dificuldade de realização da tarefa de seleção de respostas no *dataset* MilkQA. No primeiro, é feita uma

³ Ver Subseção 2.4.2, sobre medidas de avaliação.

⁴ Do inglês, *Out Of Vocabulary (OOV) words*.

⁵ Do inglês, *Graphics Processing Unit (GPU)*

comparação entre o MilkQA e um *dataset* de perguntas curtas e objetivas, representado pelo WikiQA. No segundo experimento, o objetivo é verificar o melhor desempenho possível no MilkQA usando modelos de estado da arte e variando os vetores de palavras. Isto é, além dos *baselines*, deseja-se estabelecer um valor máximo, um teto, que possa servir como parâmetro para a avaliação de desempenho do modelo SlimRank. Nos dois experimentos, os lotes de treinamento são formados por uma tripla, o que implica na avaliação de dois pares (pergunta, resposta) a cada iteração. Os resultados do primeiro experimento – comparação entre MilkQA e WikiQA – são apresentados na [Tabela 6](#).

Tabela 6 – Desempenho da seleção de respostas nos *datasets* MilkQA e WikiQA

Modelo	MilkQA		WikiQA	
	MAP	P@1	MAP	P@1
WWM	0.3836	0.2467	0.5100	0.5062
WSE	0.2552	0.1733	0.5838	0.3951
CNN-std	0.5573	0.4100	0.5746	0.4135
CNN-LDC	0.6899	0.5700	0.6848	0.5485

Fonte: Dados da pesquisa.

Nesse primeiro experimento, foram avaliados os *baselines* WWM e WSE, e os modelos CNN-std e CNN-LDC. Vetores de domínio geral foram usados com ambos os *datasets*: vetores pré-treinados no corpus GoogleNews (ver [Subseção 3.3.3](#)) foram usados com o WikiQA, e vetores do repositório NILC *Embeddings* foram usados com o MilkQA.

É possível verificar, na [Tabela 6](#), que os resultados alcançados pelo *baseline* WWM no *dataset* MilkQA foram bastante inferiores aos alcançados no WikiQA. Esses resultados refletem as características de cada *dataset* e comprovam o que foi observado na fase de seleção de dados do MilkQA: ao contrário do que ocorre em *datasets* de perguntas objetivas, no MilkQA há pouca sobreposição de palavras entre perguntas e respostas.

Também é possível verificar que os modelos baseados em CNNs apresentam desempenho similar em ambos os *datasets*. Em particular, o modelo CNN-LDC é o que alcança os melhores resultados. Por outro lado, o *baseline* WSE, baseado em vetores de palavras, apresentou resultados bastante baixos no MilkQA com vetores de domínio geral.

No experimento seguinte, somente o *dataset* MilkQA foi avaliado. Novamente, foram considerados os mesmos *baselines*, e também os modelos CNN-std e CNN-LDC. Três alternativas de vetores de palavras foram consideradas:

- i. vetores treinados no próprio domínio (Embrapa);
- ii. vetores pré-treinados, de domínio geral (NILC); e
- iii. vetores aleatórios (random), aprendidos na própria tarefa.

Os resultados são apresentados na [Tabela 7](#), juntamente com a soma dos tempos computacionais referentes às etapas de treinamento e de avaliação dos modelos. Os tempos computacionais são apresentados em minutos (min) e horas (h). Os valores em minutos foram arredondados para o número inteiro mais próximo e os tempos omitidos não foram significativos (menos de um minuto).

Tabela 7 – Seleção de respostas no *dataset* MilkQA usando diferentes modelos e vetores

Modelo	Domínio	MAP	P@1	Tempo	
				(min)	(h)
WWM	n/a	0.3836	0.2467		
WSE	Embrapa	0.5233	0.3967		
CNN-std	Embrapa	0.5165	0.3867	6	0.1
CNN-LDC	Embrapa	0.5497	0.4167	378	6.3
WSE	NILC	0.2552	0.1733		
CNN-std	NILC	0.5573	0.4100	4	0.0
CNN-LDC	NILC	0.6899	0.5700	510	8.5
CNN-std	random	0.5697	0.4233	5	0.0
CNN-LDC	random	0.7023	0.5800	528	8.8

Fonte: Dados da pesquisa.

Como pode ser verificado pelos valores em destaque na [Tabela 7](#), o modelo CNN-LDC, iniciado com vetores aleatórios (CNN-LDC/random), foi o que alcançou os melhores resultados no *dataset* MilkQA, com MAP = 0.7023 e tempo computacional de 8,8 horas. Resultados próximos foram obtidos com a configuração CNN-LDC/NILC, com MAP = 0.6899 e tempo computacional de 8,5 horas. Ao contrário do que era esperado, os melhores resultados não foram alcançados com os vetores de domínio (Embrapa). Nesse cenário, tanto CNN-std quanto CNN-LDC apresentaram desempenho próximo ao do *baseline* WSE, principalmente quando se considera a medida P@1. Na realidade, a configuração CNN-std/Embrapa apresentou resultados ligeiramente inferiores àqueles obtidos com o WSE/Embrapa.

Em todos os experimentos descritos nesta seção, buscou-se minimizar o descarte de informação para que os melhores resultados pudessem ser alcançados com cada um dos modelos. Para o *dataset* MilkQA, foram estabelecidos os tamanhos máximos de 315 e 710 palavras para perguntas e respostas, respectivamente. Com esses pontos de corte, apenas 0,3% das perguntas e das respostas no *dataset* foram truncadas (*outliers*). Seguindo o trabalho em foi apresentado o modelo CNN-LDC ([WANG; MI; ITTYCHERIAH, 2016](#)), perguntas e respostas no *dataset* WikiQA foram limitadas a 20 e 40 palavras, respectivamente.

6.3 Avaliação do modelo SlimRank

Como foi mencionado no [Capítulo 5](#), o modelo SlimRank foi projetado para atender a dois requisitos: (i) evitar o descarte arbitrário de informação; e (ii) representar textos por meio de relações semânticas – em contraste com a representação por adjacência de palavras, comumente adotada em arquiteturas de CNNs. Os experimentos descritos nesta seção têm como objetivo avaliar o desempenho do modelo SlimRank, e também estabelecer um limite mínimo para a quantidade de dados de entrada necessária para que o desempenho do modelo possa ser considerado satisfatório. Para cumprir esses dois objetivos, foram realizadas diversas avaliações do modelo, variando a quantidade-limite de campos receptivos por pergunta (parâmetro n_q , [Quadro 15](#)) e por resposta (n_a). Para se determinar um conjunto de valores a ser considerado para esses limites, dividiu-se sistematicamente o *dataset* em estratos representados por percentis. Esses estratos, juntamente com as respectivas quantidades-limite, são apresentados na [Tabela 8](#).

Tabela 8 – Quantidades-limite de campos receptivos

Estrato	# Campos pergunta	Receptivos resposta
10-p	4	5
25-p	5	7
50-p	8	13
75-p	11	27
95-p	20	45

Fonte: Dados da pesquisa.

O estrato representado na [Tabela 8](#) por 10-p, por exemplo, indica que, com um limite de 4 e 5 campos receptivos, apenas 10% das perguntas e das respostas, respectivamente, serão representadas por todos os seus campos receptivos. Isto é, no estrato 10-p, 90% das perguntas e das respostas, que são as que contêm mais do que 4 e 5 campos receptivos, estarão limitadas a essas quantidades máximas, sendo que o restante de seus campos receptivos serão desprezados. Assim, quanto mais alto o estrato, maior o número de campos receptivos considerados. Em todas as situações, os campos mais relevantes são usados primeiro, ou seja, são usados aqueles que foram selecionados primeiro pelo algoritmo de seleção ([Algoritmo 3](#), p.107).

Foram consideradas nesses experimentos as variações SR-std e SR-LDC do modelo SlimRank, e novamente as três alternativas de vetores: Embrapa, NILC e random. Em todas as execuções, o tamanho máximo dos campos receptivos foi fixado em $k = 6$ palavras. Na etapa de extração de campos receptivos (criação das MSTs) foram usados vetores de 200 dimensões, treinados em textos do domínio (Embrapa) com a variante CBOW, do modelo Word2vec. Os valores dos parâmetros c_{qmin} e c_{amin} ([Quadro 15](#)) foram 3 e 5,

Tabela 9 – Desempenho do SlimRank no *dataset* MilkQA

Modelo	Domínio	MAP	P@1	Tempo (min) (h)		Estrato
SR-LDC	Embrapa	0.6084	0.4867	40	0.7	10-p
SR-std	Embrapa	0.5581	0.4300	12	0.2	
SR-LDC	NILC	0.4934	0.3367	41	0.7	
SR-std	NILC	0.5289	0.3933	12	0.2	
SR-LDC	random	0.4250	0.2433	44	0.7	
SR-std	random	0.4949	0.3433	12	0.2	
SR-LDC	Embrapa	0.5899	0.4667	42	0.7	25-p
SR-std	Embrapa	0.5763	0.4500	12	0.2	
SR-LDC	NILC	0.4958	0.3367	38	0.6	
SR-std	NILC	0.5148	0.3700	14	0.2	
SR-LDC	random	0.4790	0.3100	43	0.7	
SR-std	random	0.5089	0.3533	12	0.2	
SR-LDC	Embrapa	0.6403	0.5167	63	1.1	50-p
SR-std	Embrapa	0.5839	0.4667	12	0.2	
SR-LDC	NILC	0.5369	0.3833	55	0.9	
SR-std	NILC	0.5591	0.4267	13	0.2	
SR-LDC	random	0.5624	0.4067	58	1.0	
SR-std	random	0.5688	0.4400	13	0.2	
SR-LDC	Embrapa	0.6180	0.4833	106	1.8	75-p
SR-std	Embrapa	0.5910	0.4533	13	0.2	
SR-LDC	NILC	0.5939	0.4567	92	1.5	
SR-std	NILC	0.5701	0.4400	14	0.2	
SR-LDC	random	0.5613	0.4200	94	1.6	
SR-std	random	0.5572	0.4233	13	0.2	
SR-LDC	Embrapa	0.6510	0.5367	163	2.7	95-p
SR-std	Embrapa	0.5953	0.4733	14	0.2	
SR-LDC	NILC	0.5818	0.4300	162	2.7	
SR-std	NILC	0.5935	0.4567	14	0.2	
SR-LDC	random	0.5678	0.4100	182	3.0	
SR-std	random	0.5452	0.3900	14	0.2	

Fonte: Dados da pesquisa.

respectivamente. Os lotes de treinamento são formados por uma tripla, no caso da variação SR-LDC, e por dez triplas no caso da variação SR-std. Os resultados das avaliações são apresentados na [Tabela 9](#). Os tempos computacionais se referem à soma dos tempos de treinamento e de avaliação dos modelos, e de extração de campos receptivos (10 minutos, em todos os casos). Os resultados contidos na tabela são apresentados de forma gráfica na [Figura 22](#), onde linhas contínuas representam a variação SR-LDC, linhas tracejadas

Figura 22 – Desempenho do SlimRank no *dataset* MilkQA

Fonte: Dados da pesquisa.

representam a variação SR-std e a forma dos marcadores representa os diferentes tipos de vetores.

Observa-se, na [Tabela 9](#), que o melhor desempenho do modelo SlimRank no *dataset* MilkQA foi alcançado pela variação SR-LDC, com vetores pré-treinados no domínio da aplicação (Embrapa) e número de campos receptivos determinados pelo estrato 95-p (SR-LDC/Embrapa/95-p). Essa configuração atingiu o resultado de $MAP = 0.6510$, com tempo computacional de 2,7 horas. De fato, é possível verificar pelo gráfico na [Figura 22](#) que essa configuração (SR-LDC/Embrapa) foi a que obteve os melhores resultados com todos os números de campos receptivos (estratos), seguida pela configuração SR-std/Embrapa, na segunda posição. Em outras palavras, os modelos iniciados com vetores de domínio foram os que apresentaram os melhores resultados nas avaliações do SlimRank. Os modelos restantes, iniciados com vetores de domínio geral (NILC e random), apresentaram desempenho semelhante nos níveis 25-p, 50-p e 75-p, com significativa melhora do nível 25-p para o 50-p, variando pouco a partir desse nível.

A configuração SR-LDC/random/95-p foi a que consumiu o maior tempo com-

putacional, de 3,0 horas, próximo do tempo consumido pela melhor configuração (SR-LDC/Embrapa/95-p), porém, apresentando resultados bastante inferiores, como pode ser observado na [Tabela 9](#). Também fica claro pela tabela que os tempos computacionais consumidos pela variação SR-std são sempre próximos, com pequeno aumento proporcional ao volume de dados de entrada, independentemente do tipo de vetor utilizado.

6.4 Discussão

Nesta seção, apresentam-se algumas reflexões sobre os resultados dos experimentos descritos nas seções anteriores.

O desempenho alcançado pela configuração CNN-LDC/random ([Tabela 7](#)) é considerado, neste trabalho, o limite máximo para um modelo de estado da arte em nosso *dataset* MilkQA. O valor $\text{MAP} = 0.7023$ é, aparentemente, baixo, se for comparado ao atual estado da arte da tarefa de seleção de respostas, que está entre 0.77 e 0.80 ([Tabela 1](#)). Portanto, pode-se considerar, a princípio, que essa diferença seja devida ao tamanho do *dataset* MilkQA, que é relativamente pequeno para o treinamento de modelos neurais. Contudo, cabe lembrar que o atual *benchmark* considerado na classificação de estado da arte ([ACL, 2017](#)) é o *dataset* TREC-QA, que contém apenas 227 perguntas. Portanto, atribuímos o desempenho mais baixo no MilkQA não ao tamanho, mas à variedade de dados no *dataset* MilkQA, que contém 300 perguntas apenas no conjunto de testes.

Conforme foi destacado na [Seção 6.2](#), ao contrário do que era esperado, os modelos treinados com vetores de domínio (Embrapa) apresentaram resultados inferiores aos obtidos com vetores de domínio geral. Observa-se ainda que, naqueles experimentos ([Tabela 7](#)), os melhores resultados foram obtidos com vetores aleatórios (random). Acredita-se que os modelos treinados com vetores de domínio estivessem convergindo para uma região de mínimo local, uma vez que, mesmo deixando que o treinamento prosseguisse indefinidamente (desativação do *early stopping*), não foi observada melhora nos resultados. Também é possível observar na [Tabela 7](#) que tanto os modelos treinados com vetores de domínio geral (NILC) quanto os treinados com vetores aleatórios (random) apresentaram resultados próximos. Acreditamos que, no caso dos vetores NILC, a convergência para uma região de mínimo local tenha sido evitada pela ocorrência de palavras fora do vocabulário (OOV), que, assim como os vetores aleatórios, também são iniciadas aleatoriamente. Verificou-se que 3.149 de um total de 18.947 palavras contidas no vocabulário do *dataset* MilkQA não foram encontradas no vocabulário de treinamento dos vetores NILC. Acredita-se que essa fração de palavras OOV, de 16,6%, tenha influenciado a convergência do modelo. No caso dos vetores treinados no próprio domínio não ocorrem vetores aleatórios, uma vez que todas as palavras são encontradas no próprio vocabulário.

A hipótese de que o comportamento dos modelos é influenciado pela ocorrência

de vetores aleatórios é reforçada pelos resultados observados na [Tabela 9](#) e na [Figura 22](#), onde também é possível verificar o comportamento similar entre os modelos iniciados com vetores de domínio geral (NILC) e com vetores aleatórios (random). Nas avaliações do modelo SlimRank, ocorreu o oposto: os modelos iniciados com vetores treinados no próprio domínio (SR-LDC/Embrapa e SR-std/Embrapa) é que apresentaram os melhores resultados. Acredita-se que, no caso do SlimRank, esse fato seja devido à representação de textos por meio de relações semânticas, que foram melhor capturadas por vetores treinados no próprio domínio.

O principal objetivo dos experimentos descritos na [Seção 6.3](#) é verificar o melhor desempenho que pode ser obtido do modelo SlimRank, com a menor quantidade possível de campos receptivos. Verifica-se que o melhor desempenho foi obtido com a configuração SR-LDC/Embrapa/95-p ([Tabela 9](#)), com MAP = 0.6510 e tempo computacional de 2,7 horas. Esse resultado supera os *baselines* WWM e WSE ([Tabela 7](#)) com larga margem. Na verdade, a partir do estrato 50-p, todas as execuções do SlimRank superam os resultados do melhor *baseline* no *dataset* MilkQA (WSE/Embrapa).

Apesar de a configuração SR-LDC/Embrapa/95-p não ter superado os dois melhores modelos no MilkQA ([Tabela 7](#)) – CNN-LDC/random, com MAP = 0.7023 e CNN-LDC/NILC, com MAP = 0.6899 –, a diferença em relação ao desempenho obtido pela melhor configuração (CNN-random), pela medida P@1, foi de apenas 0.043 ($0.5800 - 0.5367 = 0.043$) consumindo, para isso, somente 30,8% ($163/528 = 0.308$) do tempo computacional. Considerando o segundo melhor desempenho do modelo SlimRank (SR-LDC/Embrapa/50-p), com MAP = 0.6403, o tempo computacional foi ainda menor, caindo de 163 para 63 minutos. Ou seja, para alcançar o resultado de MAP = 0.6510, a melhor configuração consumiu 2,6 vezes mais tempo do que a que ficou em segunda posição. Comparando a segunda melhor configuração nos experimentos com o SlimRank com a melhor configuração no MilkQA, a diferença em relação à medida P@1 foi de 0.063 ($0.5800 - 0.5167 = 0.063$), consumindo, para isso, apenas 11,9% do tempo computacional. É possível verificar ainda que o resultado da configuração SR-LDC/Embrapa/50-p supera todos os resultados obtidos nos experimentos realizados no MilkQA ([Tabela 7](#)), com exceção dos dois primeiros.

Com relação à quantidade mínima de campos receptivos, considerando-se o desempenho na tarefa de seleção de respostas e o tempo computacional necessário, conclui-se que as quantidades definidas pelo estrato 50-p (mediana) foram as que proporcionaram os melhores resultados. Assim, foram usados 8 campos receptivos para representar perguntas e 13 para representar respostas ([Tabela 8](#)). Na melhor configuração, por outro lado, foram usados 2,5 e 3,5 vezes mais campos receptivos para representar perguntas e respostas, respectivamente. Uma vez que essas quantidades não são diretamente refletidas no resultado obtido, conclui-se que os números de campos receptivos representados pelas

medianas são suficientes para o adequado funcionamento do modelo SlimRank.

6.5 Considerações finais

Neste capítulo, foram descritos experimentos envolvendo o *dataset* de perguntas de consumidores MilkQA e o modelo de seleção de respostas SlimRank, ambos criados neste projeto. Inicialmente, foi verificado o desempenho máximo obtido por modelos de estado da arte no *dataset* MilkQA. A partir desse marco, foi possível avaliar a eficiência do modelo SlimRank.

Concluímos, pelos resultados obtidos, que ambos os requisitos estabelecidos para o modelo SlimRank foram cumpridos. Acreditamos que a representação de textos por meio de relações semânticas permitiu que as características dos textos de entrada fossem capturadas pelo modelo e que o método de seleção de campos receptivos, baseado no critério de centralidade de *clusters*, permitiu que fossem priorizados os focos de informação mais relevantes de cada texto, evitando assim o descarte arbitrário de informação.

Finalmente, pudemos verificar que a combinação do modelo SlimRank com arquiteturas de CNN complexas, como a LDC, pode proporcionar a obtenção de bons resultados na tarefa de seleção de respostas, a um custo computacional significativamente menor do que aquele requerido pela arquitetura original.

CONCLUSÕES

Este trabalho é dedicado ao problema de seleção de respostas para perguntas de consumidores. Esse tipo de pergunta normalmente ocorre em fóruns e em Serviços de Atendimento ao Consumidor (SACs) como uma forma de se buscar informações sobre um determinado contexto ou problema. As perguntas de consumidores apresentam características próprias, que as tornam mais desafiadoras do que as frases interrogativas únicas e diretas, normalmente abordadas na área de QA. As perguntas de consumidores são compostas por múltiplas frases, costumam conter erros ortográficos, termos leigos, informações em excesso e interrogações subjetivas, que dificultam sua interpretação. Uma forte característica das perguntas de consumidores são as relações semânticas entre elementos do texto, que ocorrem na forma de fenômenos linguísticos como anáforas e elipses, cuja abordagem direta é ainda bastante desafiadora.

Neste trabalho, apresentamos um modelo de seleção de respostas para perguntas de consumidores, projetado para representar textos como redes de relações semânticas, de maneira que fenômenos linguísticos complexos sejam interpretados indiretamente. Também apresentamos um novo *dataset* desse tipo de pergunta no domínio da produção leiteira.

A seguir, são descritas as principais contribuições desta tese, as publicações produzidas no período de doutoramento e, finalmente, são apontadas limitações e trabalhos futuros.

7.1 Principais contribuições

As três principais contribuições desta tese são descritas a seguir.

7.1.1 Definição do objeto de pesquisa

Diversas características de perguntas de consumidores haviam sido apresentadas anteriormente em trabalhos pioneiros (ZHANG, 2010; KILICOGLU; FISZMAN; DEMNER-FUSHMAN, 2013). Contudo, não se encontra nesses trabalhos uma definição clara para esse objeto de pesquisa, que pudesse ser usada como crivo para a criação de um *dataset*, por exemplo.

Algumas características marcantes das perguntas de consumidores, como tamanho e presença de termos leigos, foram experimentadas inicialmente como crivo. No entanto, pode-se perceber rapidamente que essas não eram características suficientes para distingui-las de outros tipos de pergunta.

Assim, uma das importantes contribuições deste projeto é uma proposta de definição para o objeto de pesquisa *perguntas de consumidores* (Definição 3, p.91). O estabelecimento dessa definição foi fundamental para que pudéssemos selecionar, de maneira consistente, o conteúdo do *dataset* MilkQA. Acreditamos que essa contribuição possa ser uma referência para outros trabalhos que venham, no futuro, a abordar o mesmo objeto de pesquisa e possam, inclusive, avaliar a adequação da nossa proposta de definição em outros contextos.

7.1.2 Dataset MilkQA

Outra importante contribuição deste projeto de pesquisa é o *dataset* MilkQA, que deverá ser disponibilizado publicamente no *site* do NILC¹. Esse *dataset* foi criado a partir de dados coletados por uma renomada instituição pública de pesquisa agropecuária (Embrapa), e apresenta diversas características favoráveis à realização de estudos tanto linguísticos quanto computacionais.

A versão atual do *dataset* MilkQA contém 2.657 pares de perguntas e respostas, em língua portuguesa, selecionados e anonimizados manualmente. Sabe-se que existem *datasets* maiores, destinados à tarefa de seleção de respostas (ver Subseção 2.4.1). Porém, esses grandes *datasets* normalmente são criados por métodos artificiais, que não refletem, necessariamente, o uso natural da língua. As perguntas contidas no MilkQA, por outro lado, foram motivadas por situações reais, além de terem sido elaboradas por milhares de autores, de diferentes regiões do país e com diferentes níveis de letramento.

Além da tarefa de seleção de respostas, para a qual foi criado, as características do *dataset* MilkQA também podem beneficiar tarefas como a simplificação textual (ALUI-SIO; GASPERIN, 2010), na área de PLN, e a classificação multi-rótulo (TSOUMAKAS; KATAKIS; VLAHAVAS, 2010), na área de aprendizado de máquina.

¹ MilkQA: <<http://www.nilc.icmc.usp.br/nilc/index.php/milkqa/>>

7.1.3 Modelo SlimRank

A principal contribuição deste trabalho é o modelo SlimRank, que consiste em uma combinação de representação textual baseada em grafos semânticos com arquiteturas de redes convolutivas (CNNs). O modelo é bastante flexível, uma vez que, após a fase de convolução, não impõe qualquer restrição à arquitetura de CNN escolhida. Neste trabalho, foram realizados experimentos com duas arquiteturas de CNNs, de diferentes graus de complexidade e de eficiência.

O SlimRank foi projetado para resolver problemas típicos de perguntas de consumidores, relacionados ao tamanho e à subjetividade dos textos. O modelo foi avaliado no *dataset* MilkQA e comparado com *baselines* e arquiteturas de CNNs do estado da arte. Os resultados alcançados pelo SlimRank foram bastante superiores aos *baselines* e compatíveis com modelos do estado da arte, consumindo apenas uma fração dos recursos computacionais. As representações textuais expressivas e sucintas, geradas pelo SlimRank, proporcionaram a redução do uso de memória e da quantidade de cálculos requerida nas fases de treinamento e de avaliação do modelo. Essa redução foi refletida diretamente no tempo computacional medido nos experimentos.

As hipóteses de pesquisa (Seção 1.1) puderam ser comprovadas pelos resultados dos experimentos. As representações de textos como redes de relações semânticas se mostrou bastante eficiente e nos permitiu capturar relações de sentido que eram essenciais para a compreensão dos textos (ver exemplos na Figura 20) sem precisar abordar diretamente os mecanismos de realização de complexos fenômenos linguísticos. O descarte arbitrário de informação também foi evitado, uma vez que o texto é explorado como um todo na fase de construção dos grafos semânticos e que a seleção das porções mais expressivas obedece a um critério consistente e bem definido.

7.2 Publicações

A seguir, são descritas as publicações geradas durante o período de doutoramento.

1. Lean Semantic Representations of Dense Questions for the Task of Answer Selection. **Transactions of the Association for Computational Linguistics (TACL)**. (CRISCUOLO; ALUÍSIO, 2017b).

Esse artigo foi submetido a um importante periódico internacional da área de Linguística Computacional (TACL) e atualmente encontra-se em avaliação. Nesse trabalho são apresentadas as principais contribuições desta tese: a definição do objeto perguntas de consumidores, o *dataset* MilkQA e o modelo de seleção de respostas SlimRank.

2. MilkQA: a Dataset of Consumer Questions for the Task of Answer Selection. In: **Proceedings of the 6th Brazilian Conference on Intelligent Systems (BRACIS)**. (CRISCUOLO *et al.*, 2017).

Apresenta-se nesse trabalho o *dataset* MilkQA, criado neste projeto. Descrevem-se detalhadamente as características do *dataset*, a origem e o método de coleta dos dados e também experimentos que visam avaliar a dificuldade da tarefa de seleção de respostas nesse novo conjunto de dados. Também é destacada a utilidade do MilkQA para outras tarefas, como a de classificação multi-rótulo, na área de aprendizado de máquina. Esse trabalho foi apresentado na conferência BRACIS 2017.

3. Discriminating between similar languages with word-level convolutional neural networks. In: **Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)**. (CRISCUOLO; ALUÍSIO, 2017a).

Nesse trabalho, descrito em detalhes na [Subseção 3.4.1](#), descrevem-se experimentos envolvendo o uso de redes neurais convolutivas (CNNs) na tarefa de identificação de variantes de língua. Em certo momento deste projeto, foram realizados experimentos envolvendo o uso de um identificador de línguas para filtrar textos que seriam empregados no treinamento de vetores de palavras. Essa publicação é resultado desses experimentos.

4. Visão Geral da Avaliação de Similaridade Semântica e Inferência Textual. **Linguamática**. (FONSECA *et al.*, 2016).

Esse trabalho, publicado na Revista Linguamática, é resultado de uma colaboração com um projeto correlato, também desenvolvido no laboratório NILC. Descrevem-se, nesse artigo, a realização e os resultados de uma avaliação conjunta (*shared task*), proposta na conferência PROPOR² 2016, sobre as tarefas de avaliação de similaridade semântica e de inferência textual.

7.3 Trabalhos futuros

Nesta seção, são discutidas algumas limitações deste trabalho e também são indicados os trabalhos futuros.

Concluiu-se, por uma análise estatística, que o arquivo de onde foram obtidas as perguntas do *dataset* MilkQA teve apenas metade de seu potencial explorado até este momento. Seria possível, portanto, expandir o *dataset* ultrapassando o tamanho de 5 mil perguntas. O tamanho da versão atual foi limitado pela disponibilidade de trabalho humano e expandi-la é um caminho natural a ser seguido.

² *International Conference on the Computational Processing of Portuguese (PROPOR)*

A expressividade dos campos receptivos – elemento chave do modelo SlimRank – foi avaliada extrinsecamente, na tarefa de seleção de respostas. Porém, um método de avaliação intrínseca poderia ser útil em algumas situações. Na versão atual do modelo, um campo receptivo é composto por um vértice representativo e por seus N_1 vizinhos. Outra alternativa, por exemplo, seria considerar o caminho mais curto formado por k vértices a partir do representativo. Qual dos dois métodos geraria campos receptivos mais expressivos? O uso de tarefas simples (*toy tasks*) é proposto por [Weston et al. \(2015\)](#) para avaliar a eficiência de técnicas isoladas relacionadas à tarefa de QA. Um interessante trabalho futuro seria desenvolver tarefas desse tipo, que permitissem avaliar isoladamente a expressividade de campos receptivos.

O modelo SlimRank foi projetado para processar textos relativamente longos, em que sejam abundantes as relações semânticas entre os elementos textuais. Também seria interessante avaliar seu desempenho em textos curtos e objetivos. Haveria alguma vantagem em relação às representações tradicionais, baseadas na adjacência de palavras? Um bom ponto de partida seria avaliar o modelo no *dataset* InsuranceQA ([FENG et al., 2015](#)), em que as perguntas são frases curtas e objetivas, como na maior parte dos *datasets*, mas as respostas são explicações detalhadas, mais longas do que o usual.

O modelo mostrou bom desempenho na tarefa de avaliação de similaridade semântica (aplicada à tarefa de seleção de respostas). Seria interessante também avaliá-lo em outras tarefas, como, por exemplo, a classificação de textos. O método de representação textual poderia funcionar como um filtro redutor de ruído, proporcionando melhores resultados em textos longos?

Ainda considerando o tema avaliação, seria interessante conhecer o desempenho do modelo em um *dataset* de domínio geral, diferente do MilkQA e do InsuranceQA, mencionado anteriormente, ambos *datasets* de domínio específico.

Finalmente, seria interessante experimentar alternativas de construção do grafo semântico e também outras medidas de centralidade, para identificação dos vértices representativos. Um método de construção de grafos textuais que combina janelas de co-ocorrência com vetores de palavras é apresentado por [Santos et al. \(2017\)](#). Nesse mesmo trabalho, também são apresentados alguns critérios de centralidade, baseados em medidas de redes complexas.

REFERÊNCIAS

ABADI, M.; AGARWAL, A.; BARHAM, P.; BREVDO, E.; CHEN, Z.; CITRO, C.; CORRADO, G. S.; DAVIS, A.; DEAN, J.; DEVIN, M. *et al.* Tensorflow: Large-scale machine learning on heterogeneous distributed systems. **arXiv preprint arXiv:1603.04467**, 2016. Citado nas páginas 78 e 115.

ACL. **Question Answering (State of the art)**. 2017. Disponível em: <[https://aclweb.org/aclwiki/Question_Answering_\(State_of_the_art\)](https://aclweb.org/aclwiki/Question_Answering_(State_of_the_art))>. Acesso em: 23/09/2017. Citado nas páginas 56, 81 e 121.

AGHAEBRAHIMIAN, A. Quora question answer dataset. In: EKŠTEIN, K.; MATOUŠEK, V. (Ed.). **Text, Speech, and Dialogue: 20th International Conference, TSD 2017, Prague, Czech Republic, August 27-31, 2017, Proceedings**. Cham: Springer International Publishing, 2017. p. 66–73. ISBN 978-3-319-64206-2. Disponível em: <https://doi.org/10.1007/978-3-319-64206-2_8>. Citado na página 53.

AL-RFOU, R.; PEROZZI, B.; SKIENA, S. Polyglot: Distributed word representations for multilingual nlp. In: **Proceedings of the Seventeenth Conference on Computational Natural Language Learning (CoNLL)**. Sofia, Bulgaria: Association for Computational Linguistics, 2013. p. 183–192. Disponível em: <<http://www.aclweb.org/anthology/W13-3520>>. Citado na página 70.

ALUISIO, S.; GASPERIN, C. Fostering digital inclusion and accessibility: The porsimples project for simplification of portuguese texts. In: **Proceedings of the NAACL HLT 2010 Young Investigators Workshop on Computational Approaches to Languages of the Americas**. Association for Computational Linguistics, 2010. p. 46–53. Disponível em: <<http://aclanthology.coli.uni-saarland.de/pdf/W/W10/W10-1607.pdf>>. Citado na página 126.

AMARAL, C.; FIGUEIRA, H.; MARTINS, A.; MENDES, A.; MENDES, P.; PINTO, C. Priberam's question answering system for Portuguese. In: SPRINGER. **Workshop of the Cross-Language Evaluation Forum for European Languages**. [S.l.], 2005. p. 410–419. Citado nas páginas 43 e 44.

ANDROUTSOPOULOS, I.; RITCHIE, G.; THANISCH, P. Natural language interfaces to databases - an introduction. **Natural Language Engineering**, Cambridge University Press, v. 1, n. 1, p. 29–81, 1995. Citado na página 47.

ANTOL, S.; AGRAWAL, A.; LU, J.; MITCHELL, M.; BATRA, D.; ZITNICK, C. L.; PARIKH, D. VQA: Visual question answering. In: **The IEEE International Conference on Computer Vision (ICCV)**. [s.n.], 2015. Disponível em: <http://www.cv-foundation.org/openaccess/content_iccv_2015/html/Antol_VQA_Visual_Question_ICCV_2015_paper.html>. Citado na página 50.

BARONI, M.; DINU, G.; KRUSZEWSKI, G. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In: **Proceedings of the**

52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, 2014. p. 238–247. Disponível em: <<http://aclanthology.coli.uni-saarland.de/pdf/P/P14/P14-1023.pdf>>. Citado nas páginas 63 e 64.

BELINKOV, Y.; GLASS, J. A character-level convolutional neural network for distinguishing similar languages and dialects. In: **Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)**. The COLING 2016 Organizing Committee, 2016. p. 145–152. Disponível em: <<http://aclanthology.coli.uni-saarland.de/pdf/W/W16/W16-4819.pdf>>. Citado na página 76.

BELINKOV, Y.; MOHTARAMI, M.; CYPHERS, S.; GLASS, J. Vectorslu: A continuous word vector approach to answer selection in community question answering systems. In: **Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)**. Association for Computational Linguistics, 2015. p. 282–287. Disponível em: <<http://aclweb.org/anthology/S15-2048>>. Citado na página 49.

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. Citado nas páginas 63, 64, 65, 66 e 69.

BICK, E. **The Parsing System “Palavras”: Automatic Grammatical Analysis of Portuguese in a Constraint Grammar Framework**. [S.l.]: Aarhus Universitetsforlag, 2000. Citado na página 59.

BILOTTI, M.; NYBERG, E. Improving text retrieval precision and answer accuracy in question answering systems. In: **Coling 2008: Proceedings of the 2nd workshop on Information Retrieval for Question Answering**. Coling 2008 Organizing Committee, 2008. p. 1–8. Disponível em: <<http://aclweb.org/anthology/W08-1801>>. Citado na página 44.

BIZER, C.; HEATH, T.; BERNERS-LEE, T. Linked data - the story so far. **International Journal on Semantic Web and on Information Systems**, v. 5, n. 3, p. 1–22, 2009. Disponível em: <<https://doi.org/10.4018/jswis.2009081901>>. Citado na página 46.

BLACKBURN, P.; BOS, J. Representation and inference for natural language. **A first course in computational semantics**. CSLI, 2005. Citado na página 44.

BOS, J.; GUZZETTI, E.; CURRAN, J. R. The pronto qa system at trec 2007: Harvesting hyponyms, using nominalisation patterns, and computing answer cardinality. In: **TREC**. [S.l.: s.n.], 2007. v. 3, p. 2. Citado na página 42.

BRANCO, A.; RODRIGUES, L.; SILVA, J.; SILVEIRA, S. Xisquê: An online qa service for portuguese. In: SPRINGER. **PROPOR**. [S.l.], 2008. v. 8, p. 232–235. Citado nas páginas 42 e 44.

BRILL, E.; DUMAIS, S.; BANKO, M. An analysis of the askmsr question-answering system. In: **Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)**. Association for Computational Linguistics, 2002. p. 257–264. Disponível em: <<http://aclweb.org/anthology/W02-1033>>. Citado nas páginas 38, 41, 42 e 43.

- BRILL, E.; MOORE, R. C. An improved error model for noisy channel spelling correction. In: **Proceedings of the 38th Annual Meeting on Association for Computational Linguistics**. Stroudsburg, PA, USA: Association for Computational Linguistics, 2000. (ACL '00), p. 286–293. Disponível em: <<http://dx.doi.org/10.3115/1075218.1075255>>. Citado na página 59.
- BRIN, S.; PAGE, L. The anatomy of a large-scale hypertextual web search engine. **Computer Networks**, v. 30, n. 1-7, p. 107–117, 1998. Disponível em: <[https://doi.org/10.1016/S0169-7552\(98\)00110-X](https://doi.org/10.1016/S0169-7552(98)00110-X)>. Citado na página 82.
- CAMARGO, I. d.; BOULOS, P. **Geometria Analítica: um tratamento vetorial**. 3. ed. São Paulo: Prentice Hall, 2005. ISBN 978-85-87918-91-8. Citado na página 71.
- 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. Citado nas páginas 109 e 115.
- CORMEN, T. H.; LEISERSON, C. H.; RIVEST, R. L.; STEIN, C. **Introduction to Algorithms**. 3. ed. Cambridge, MA: MIT Press, 2009. ISBN 978-0262033848. Citado na página 104.
- COSTA, L. First evaluation of Esfinge – a question answering system for Portuguese. In: SPRINGER. **Workshop of the Cross-Language Evaluation Forum for European Languages**. 2004. p. 522–533. Disponível em: <<http://www.linguateca.pt/Esfinge/esfinge.html>>. Citado na página 41.
- CRISCUOLO, M.; ALUÍSIO, S. M. Discriminating between similar languages with word-level convolutional neural networks. In: **Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)**. Association for Computational Linguistics, 2017. p. 124–130. Disponível em: <<http://aclweb.org/anthology/W17-1215>>. Citado nas páginas 76, 78, 112 e 128.
- _____. Lean semantic representations of dense questions for the task of answer selection. **Transactions of the Association for Computational Linguistics (TACL)**, out. 2017. ISSN 2307-387X. (Artigo submetido para revisão). Citado na página 127.
- CRISCUOLO, M.; FONSECA, E. R.; ALUÍSIO, S. M.; SPERANÇA-CRISCUOLO, A. C. MilkQA: a dataset of consumer questions for the task of answer selection. In: **Proceedings of the 6th Brazilian Conference on Intelligent Systems (BRACIS)**. Uberlândia, Brazil: IEEE, 2017. v. 1, p. 354–359. ISBN 978-1-5386-2407-4. Citado nas páginas 90, 100 e 128.
- DAUMÉ III, H.; MARCU, D. Bayesian query-focused summarization. In: **Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics**. Stroudsburg, PA, USA: Association for Computational Linguistics, 2006. (ACL-44), p. 305–312. Disponível em: <<http://dx.doi.org/10.3115/1220175.1220214>>. Citado na página 37.
- 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. Citado na página 64.

- DENG, J.; DONG, W.; SOCHER, R.; LI, L.-J.; LI, K.; FEI-FEI, L. Imagenet: A large-scale hierarchical image database. In: **IEEE. Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on**. [S.l.], 2009. p. 248–255. Citado na página 90.
- ERKAN, G.; RADEV, D. R. Lexrank: Graph-based lexical centrality as salience in text summarization. **Journal of Artificial Intelligence Research**, v. 22, p. 457–479, 2004. Disponível em: <<http://www.jair.org/papers/paper1523.html>>. Citado na página 83.
- ESTER, M.; KRIEGEL, H.-P.; SANDER, J.; XU, X. *et al.* A density-based algorithm for discovering clusters in large spatial databases with noise. In: **Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD)**. AAAI Press, 1996. v. 96, n. 34, p. 226–231. Disponível em: <<https://ocs.aaai.org/Papers/KDD/1996/KDD96-037.pdf>>. Citado na página 99.
- FENG, M.; XIANG, B.; GLASS, M. R.; WANG, L.; ZHOU, B. Applying deep learning to answer selection: A study and an open task. In: **2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)**. [s.n.], 2015. p. 813–820. Disponível em: <<https://arxiv.org/abs/1508.01585>>. Citado nas páginas 50, 53, 54, 76, 79, 95, 98, 100, 113 e 129.
- FERRUCCI, D.; BROWN, E.; CHU-CARROLL, J.; FAN, J.; GONDEK, D.; KALYANPUR, A. A.; LALLY, A.; MURDOCK, J. W.; NYBERG, E.; PRAGER, J. *et al.* Building watson: An overview of the deepqa project. **AI Magazine**, v. 31, n. 3, p. 59–79, 2010. Disponível em: <<https://doi.org/10.1609/aimag.v31i3.2303>>. Citado na página 49.
- FONSECA, E. R.; SANTOS, L. B. dos; CRISCUOLO, M.; ALUÍSIO, S. M. Visão geral da avaliação de similaridade semântica e inferência textual. **Linguamática**, v. 8, n. 2, p. 3–13, dez. 2016. ISSN 1647–0818. Disponível em: <<http://linguamatica.com/index.php/linguamatica/article/view/v8n2-1>>. Citado na página 128.
- FORNER, P.; PEÑAS, A.; AGIRRE, E.; ALEGRIA, I.; FORĂSCU, C.; MOREAU, N.; OSENOVA, P.; PROKOPIDIS, P.; ROCHA, P.; SACALEANU, B.; SUTCLIFFE, R.; SANG, E. T. K. Overview of the clef 2008 multilingual question answering track. In: _____. **Evaluating Systems for Multilingual and Multimodal Information Access: 9th Workshop of the Cross-Language Evaluation Forum, CLEF 2008, Aarhus, Denmark, September 17-19, 2008, Revised Selected Papers**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. p. 262–295. ISBN 978-3-642-04447-2. Disponível em: <https://doi.org/10.1007/978-3-642-04447-2_34>. Citado na página 53.
- GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. [S.l.]: MIT Press, 2016. <<http://www.deeplearningbook.org>>. Citado nas páginas 31 e 47.
- GREEN, B. F.; WOLF, A. K.; CHOMSKY, C.; LAUGHERY, K. Baseball: An automatic question-answerer. In: **Papers Presented at the May 9-11, 1961, Western Joint IRE-AIEE-ACM Computer Conference**. New York, NY, USA: ACM, 1961. (IRE-AIEE-ACM '61 (Western)), p. 219–224. Disponível em: <<http://doi.acm.org/10.1145/1460690.1460714>>. Citado nas páginas 41 e 44.
- HALLIDAY, M. A. K.; HASAN, R. **Cohesion in English**. [S.l.]: Longman, 1976. ISBN 0582550416. Citado nas páginas 58, 97, 98 e 102.

- HARISPE, S.; RANWEZ, S.; JANAQI, S.; MONTMAIN, J. Semantic similarity from natural language and ontology analysis. **Synthesis Lectures on Human Language Technologies**, v. 8, n. 1, p. 1–254, 2015. Disponível em: <<https://doi.org/10.2200/S00639ED1V01Y201504HLT027>>. Citado nas páginas 31, 59, 62 e 103.
- HARRIS, Z. S. Distributional structure. **Word**, Taylor & Francis, v. 10, n. 2-3, p. 146–162, 1954. Citado na página 63.
- HARTMANN, N.; FONSECA, E.; SHULBY, C.; TREVISO, M.; RODRIGUES, J.; ALUISIO, S. Portuguese word embeddings: Evaluating on word analogies and natural language tasks. In: **Proceedings of the Symposium on Information and Human Language Technology (STIL)**. Uberlândia, Brazil: [s.n.], 2017. Aceito para publicação. Citado nas páginas 66, 68, 69, 70, 72, 111 e 112.
- HILDEBRANDT, W.; KATZ, B.; LIN, J. Answering definition questions with multiple knowledge sources. In: **Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004**. [s.n.], 2004. Disponível em: <<http://aclweb.org/anthology/N04-1007>>. Citado nas páginas 40 e 42.
- IYYER, M.; BOYD-GRABER, J.; CLAUDINO, L.; SOCHER, R.; III, H. D. A neural network for factoid question answering over paragraphs. In: **Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)**. Association for Computational Linguistics, 2014. p. 633–644. Disponível em: <<http://aclweb.org/anthology/D14-1070>>. Citado na página 49.
- JOACHIMS, T. Text categorization with support vector machines: Learning with many relevant features. In: NÉDELLEC, C.; ROUVEIROL, C. (Ed.). **Machine Learning: ECML-98: 10th European Conference on Machine Learning Chemnitz, Germany, April 21–23, 1998 Proceedings**. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998. p. 137–142. ISBN 978-3-540-69781-7. Disponível em: <<https://doi.org/10.1007/BFb0026683>>. Citado na página 83.
- JOULIN, A.; GRAVE, E.; BOJANOWSKI, P.; MIKOLOV, T. Bag of tricks for efficient text classification. **arXiv preprint arXiv:1607.01759**, 2016. Disponível em: <<https://arxiv.org/abs/1607.01759>>. Citado na página 69.
- JURAFSKY, D.; MARTIN, J. H. **Speech and Language Processing**. 2nd. ed. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2009. ISBN 0131873210. Citado na página 46.
- JURCZYK, T.; ZHAI, M.; CHOI, J. D. SelQA: A new benchmark for selection-based question answering. In: **Proceedings of the 28th International Conference on Tools with Artificial Intelligence**. San Jose, CA: [s.n.], 2016. (ICTAI'16). Disponível em: <<https://arxiv.org/abs/1606.08513>>. Citado nas páginas 51 e 52.
- KENTER, T.; RIJKE, M. de. Short text similarity with word embeddings. In: **Proceedings of the 24th ACM International on Conference on Information and Knowledge Management**. New York, NY, USA: ACM, 2015. (CIKM '15), p. 1411–1420. ISBN 978-1-4503-3794-6. Disponível em: <<http://doi.acm.org/10.1145/2806416.2806475>>. Citado nas páginas 71, 72 e 103.

- KILICOGLU, H.; FISZMAN, M.; DEMNER-FUSHMAN, D. Interpreting consumer health questions: The role of anaphora and ellipsis. In: **Proceedings of the 2013 Workshop on Biomedical Natural Language Processing**. Association for Computational Linguistics, 2013. p. 54–62. Disponível em: <<http://aclweb.org/anthology/W13-1907>>. Citado nas páginas 30, 57, 58, 59, 98 e 126.
- KIM, Y. Convolutional neural networks for sentence classification. In: **Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)**. Association for Computational Linguistics, 2014. p. 1746–1751. Disponível em: <<http://aclweb.org/anthology/D14-1181>>. Citado na página 76.
- KINGMA, D.; BA, J. Adam: A method for stochastic optimization. **arXiv preprint arXiv:1412.6980**, 2014. Citado nas páginas 78 e 114.
- KINGSBURY, P.; PALMER, M.; MARCUS, M. Adding semantic annotation to the Penn treebank. In: SAN DIEGO, CALIFORNIA. **Proceedings of the Human Language Technology Conference**. [S.l.], 2002. p. 252–256. Citado na página 44.
- KLEINBERG, J.; TARDOS, E. **Algorithm Design**. [S.l.]: Pearson Education, 2005. ISBN 0-321-29535-8. Citado nas páginas 86 e 104.
- KOCH, I. G. V. **A coesão Textual**. [S.l.]: Editora Contexto, 1999. Citado na página 98.
- KOLOMIYETS, O.; MOENS, M.-F. A survey on question answering technology from an information retrieval perspective. **Information Sciences**, v. 181, n. 24, p. 5412 – 5434, 2011. ISSN 0020-0255. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0020025511003860>>. Citado na página 44.
- KRUSKAL, J. B. On the shortest spanning subtree of a graph and the traveling salesman problem. **Proceedings of the American Mathematical Society**, JSTOR, v. 7, n. 1, p. 48–50, 1956. Citado na página 86.
- LE, Q.; MIKOLOV, T. Distributed representations of sentences and documents. In: XING, E. P.; JEBARA, T. (Ed.). **Proceedings of the 31st International Conference on Machine Learning**. Beijing, China: PMLR, 2014. (Proceedings of Machine Learning Research, 2), p. 1188–1196. Disponível em: <<http://proceedings.mlr.press/v32/le14.html>>. Citado na página 71.
- LECUN, Y.; BOSER, B.; DENKER, J. S.; HENDERSON, D.; HOWARD, R. E.; HUBBARD, W.; JACKEL, L. D. Backpropagation applied to handwritten zip code recognition. **Neural Computation**, v. 1, n. 4, p. 541–551, Winter 1989. Citado nas páginas 73 e 74.
- LECUN, Y.; BOSER, B.; DENKER, J. S.; HOWARD, R. E.; HABBARD, W.; JACKEL, L. D.; HENDERSON, D. Advances in neural information processing systems 2. In: TOURETZKY, D. S. (Ed.). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1990. p. 396–404. ISBN 1-55860-100-7. Disponível em: <<http://dl.acm.org/citation.cfm?id=109230.109279>>. Citado nas páginas 31, 47, 73 e 76.
- LEHMANN, J.; ISELE, R.; JAKOB, M.; JENTZSCH, A.; KONTOKOSTAS, D.; MENDES, P. N.; HELLMANN, S.; MORSEY, M.; KLEEF, P. van; AUER, S.; BIZER, C. DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. **Semantic Web Journal**, v. 6, n. 2, p. 167–195, 2015. Citado na página 46.

- LEI, T.; JOSHI, H.; BARZILAY, R.; JAAKKOLA, T.; TYMOSHENKO, K.; MOSCHITTI, A.; MÁRQUEZ, L. Semi-supervised question retrieval with gated convolutions. In: **Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies**. Association for Computational Linguistics, 2016. p. 1279–1289. Disponível em: <<http://aclweb.org/anthology/N16-1153>>. Citado nas páginas 32, 48, 100 e 101.
- LENCI, A. Distributional semantics in linguistic and cognitive research. **Italian Journal of Linguistics**, v. 20, n. 1, p. 1–31, 2008. Citado nas páginas 31 e 59.
- LI, X.; ROTH, D. Learning question classifiers. In: **Proceedings of the 19th International Conference on Computational Linguistics - Volume 1**. Stroudsburg, PA, USA: Association for Computational Linguistics, 2002. (COLING '02), p. 1–7. Disponível em: <<http://aclweb.org/anthology/C02-1150>>. Citado na página 38.
- LING, W.; DYER, C.; BLACK, A.; TRANCOSO, I. Two/too simple adaptations of word2vec for syntax problems. In: **Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies**. [S.l.]: Association for Computational Linguistics, 2015. Citado na página 69.
- LOPEZ, V.; FERNÁNDEZ, M.; MOTTA, E.; STIELER, N. Poweraqua: Supporting users in querying and exploring the semantic web. **Semantic Web**, IOS Press, Amsterdam, The Netherlands, The Netherlands, v. 3, n. 3, p. 249–265, ago. 2012. ISSN 1570-0844. Disponível em: <http://dx.doi.org/10.1007/978-3-319-00000-0_14>. Citado nas páginas 36 e 46.
- LOPEZ, V.; UNGER, C.; CIMIANO, P.; MOTTA, E. Evaluating question answering over linked data. **Web Semantics Science Services And Agents On The World Wide Web**, Elsevier, v. 21, p. 3–13, 2013. ISSN 1570-8268. Citado na página 46.
- LOPEZ, V.; UREN, V.; MOTTA, E.; PASIN, M. Aqualog: An ontology-driven question answering system for organizational semantic intranets. **Web Semantics: Science, Services and Agents on the World Wide Web**, v. 5, n. 2, p. 72 – 105, 2007. ISSN 1570-8268. Software Engineering and the Semantic Web. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S1570826807000145>>. Citado na página 41.
- MANNING, C. D.; RAGHAVAN, P.; SCHÜTZE, H. **Introduction to information retrieval**. Cambridge University Press, 2008. ISBN 978-0-521-86571-5. Disponível em: <<https://nlp.stanford.edu/IR-book/>>. Citado nas páginas 36 e 56.
- MENCHETTI, S.; COSTA, F.; FRASCONI, P. Weighted decomposition kernels. In: **Proceedings of the 22Nd International Conference on Machine Learning**. New York, NY, USA: ACM, 2005. (ICML '05), p. 585–592. ISBN 1-59593-180-5. Disponível em: <<http://doi.acm.org/10.1145/1102351.1102425>>. Citado na página 84.
- MIHALCEA, R.; TARAU, P. TextRNN: Bringing order into text. In: **Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP)**. [s.n.], 2004. Disponível em: <<http://aclanthology.coli.uni-saarland.de/pdf/W/W04/W04-3252.pdf>>. Citado nas páginas 82, 83 e 102.

- MIKOLOV, T.; CHEN, K.; CORRADO, G.; DEAN, J. Efficient estimation of word representations in vector space. **CoRR**, abs/1301.3781, 2013. Disponível em: <<http://arxiv.org/abs/1301.3781>>. Citado nas páginas 49, 66, 68, 69 e 112.
- MIKOLOV, T.; SUTSKEVER, I.; CHEN, K.; CORRADO, G. S.; DEAN, J. Distributed representations of words and phrases and their compositionality. In: BURGESS, C. J. C.; BOTTOU, L.; WELLING, M.; GHAHRAMANI, Z.; WEINBERGER, K. Q. (Ed.). **Advances in Neural Information Processing Systems 26**. Curran Associates, Inc., 2013. p. 3111–3119. Disponível em: <<http://arxiv.org/abs/1310.4546>>. Citado nas páginas 31, 64, 69 e 70.
- MIKOLOV, T.; YIH, W.-t.; ZWEIG, G. Linguistic regularities in continuous space word representations. In: **Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies**. Association for Computational Linguistics, 2013. p. 746–751. Disponível em: <<http://aclanthology.coli.uni-saarland.de/pdf/N/N13/N13-1090.pdf>>. Citado nas páginas 31, 61, 67 e 68.
- MILLER, G. A. Wordnet: A lexical database for english. **Commun. ACM**, ACM, New York, NY, USA, v. 38, n. 11, p. 39–41, nov. 1995. ISSN 0001-0782. Disponível em: <<http://doi.acm.org/10.1145/219717.219748>>. Citado nas páginas 43 e 90.
- MOLDOVAN, D. I.; HARABAGIU, S. M.; GIRJU, R.; MORARESCU, P.; LACATUSU, V. F.; NOVISCHI, A.; BADULESCU, A.; BOLOHAN, O. Lcc tools for question answering. In: **Proceedings of the TREC-2002 Conference**. [S.l.]: NIST, 2002. Citado na página 39.
- MOLLÁ, D.; VICEDO, J. L. Question answering in restricted domains: An overview. **Computational Linguistics**, MIT Press, v. 33, n. 1, p. 41–61, 2007. Citado na página 41.
- NAKOV, P.; HOOGEVEEN, D.; MÀRQUEZ, L.; MOSCHITTI, A.; MUBARAK, H.; BALDWIN, T.; VERSPOOR, K. SemEval-2017 task 3: Community question answering. In: **Proceedings of the 11th International Workshop on Semantic Evaluation**. Vancouver, Canada: Association for Computational Linguistics, 2017. (SemEval '17). Citado na página 49.
- NARAYANAN, S.; HARABAGIU, S. Question answering based on semantic structures. In: **Proceedings of the 20th International Conference on Computational Linguistics**. Stroudsburg, PA, USA: Association for Computational Linguistics, 2004. (COLING '04). Disponível em: <<https://doi.org/10.3115/1220355.1220455>>. Citado na página 43.
- NGUYEN, T.; ROSENBERG, M.; SONG, X.; GAO, J.; TIWARY, S.; MAJUMDER, R.; DENG, L. MS MARCO: A human generated machine reading comprehension dataset. In: **Proceedings of the 30th Conference on Neural Information Processing Systems (NIPS 2016)**. Barcelona, Spain: [s.n.], 2016. Disponível em: <<https://arxiv.org/abs/1611.09268v2>>. Citado na página 52.
- NIELSEN, M. A. **Neural Networks and Deep Learning**. Determination Press, 2015. Disponível em: <<http://neuralnetworksanddeeplearning.com/>>. Citado nas páginas 31, 47, 74 e 76.

- NIEPERT, M.; AHMED, M.; KUTZKOV, K. Learning convolutional neural networks for graphs. In: BALCAN, M. F.; WEINBERGER, K. Q. (Ed.). **Proceedings of The 33rd International Conference on Machine Learning**. New York, New York, USA: PMLR, 2016. (Proceedings of Machine Learning Research, v. 48), p. 2014–2023. Disponível em: <<http://proceedings.mlr.press/v48/niepert16.html>>. Citado nas páginas 33, 84, 85 e 86.
- NOVAIS, E.; TADEU, T.; PARABONI, I. Text generation for brazilian portuguese: the surface realization task. In: **Proceedings of the NAACL HLT 2010 Young Investigators Workshop on Computational Approaches to Languages of the Americas**. Association for Computational Linguistics, 2010. p. 125–131. Disponível em: <<http://aclweb.org/anthology/W10-1617>>. Citado na página 37.
- PEÑAS, A.; FORNER, P.; SUTCLIFFE, R.; RODRIGO, Á.; FORĂSCU, C.; ALEGRIA, I.; GIAMPICCOLO, D.; MOREAU, N.; OSENOVA, P. Overview of ResPubliQA 2009: question answering evaluation over European legislation. In: **Workshop of the Cross-Language Evaluation Forum for European Languages**. [S.l.]: Springer, 2009. p. 174–196. Citado na página 39.
- PENNINGTON, J.; SOCHER, R.; MANNING, C. D. Glove: Global vectors for word representation. In: **Empirical Methods in Natural Language Processing (EMNLP)**. [s.n.], 2014. p. 1532–1543. Disponível em: <<http://www.aclweb.org/anthology/D14-1162>>. Citado na página 70.
- QUARTERONI, S.; MANANDHAR, S. Designing an interactive open-domain question answering system. **Natural Language Engineering**, Cambridge University Press, v. 15, n. 1, p. 73–95, 2009. Citado na página 40.
- RAJPURKAR, P.; ZHANG, J.; LOPYREV, K.; LIANG, P. SQuAD: 100,000+ questions for machine comprehension of text. In: **Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing**. Association for Computational Linguistics, 2016. p. 2383–2392. Disponível em: <<http://aclweb.org/anthology/D16-1264>>. Citado na página 51.
- RAO, J.; HE, H.; LIN, J. Noise-contrastive estimation for answer selection with deep neural networks. In: ACM. **Proceedings of the 25th ACM International on Conference on Information and Knowledge Management**. [S.l.], 2016. p. 1913–1916. Citado nas páginas 30, 31 e 56.
- RODRIGUES, J.; BRANCO, A.; NEALE, S.; SILVA, J. Lx-dsemvectors: Distributional semantics models for portuguese. In: _____. **Computational Processing of the Portuguese Language: 12th International Conference, PROPOR 2016, Tomar, Portugal, July 13-15, 2016, Proceedings**. Cham: Springer International Publishing, 2016. p. 259–270. ISBN 978-3-319-41552-9. Disponível em: <https://doi.org/10.1007/978-3-319-41552-9_27>. Citado nas páginas 68 e 70.
- ROUSSEAU, F.; KIAGIAS, E.; VAZIRGIANNIS, M. Text categorization as a graph classification problem. In: **Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)**. Association for Computational Linguistics, 2015. p. 1702–1712. Disponível em: <<http://aclanthology.coli.uni-saarland.de/pdf/P/P15/P15-1164.pdf>>. Citado na página 83.

- SANTOS, C. dos; BARBOSA, L.; BOGDANOVA, D.; ZADROZNY, B. Learning hybrid representations to retrieve semantically equivalent questions. In: **Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)**. Association for Computational Linguistics, 2015. p. 694–699. Disponível em: <<http://aclweb.org/anthology/P15-2114>>. Citado nas páginas 30, 47, 73, 76 e 100.
- SANTOS, C. N. dos; TAN, M.; XIANG, B.; ZHOU, B. Attentive pooling networks. **CoRR**, abs/1602.03609, 2016. Disponível em: <<http://arxiv.org/abs/1602.03609>>. Citado nas páginas 31, 49, 50 e 81.
- SANTOS, D. Porquê o Págico? Razões para uma avaliação conjunta. **Linguamática**, v. 4, n. 1, p. 1–8, Abril 2012. ISSN 1647–0818. Disponível em: <<http://linguamatica.com/index.php/linguamatica/article/view/119/160>>. Citado na página 35.
- SANTOS, L.; JÚNIOR, E. A. C.; JR, O. O.; AMANCIO, D.; MANSUR, L.; ALUÍSIO, S. Enriching complex networks with word embeddings for detecting mild cognitive impairment from speech transcripts. In: **Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**. Association for Computational Linguistics, 2017. p. 1284–1296. Disponível em: <<http://aclanthology.coli.uni-saarland.de/pdf/P/P17/P17-1118.pdf>>. Citado na página 129.
- SARMENTO, L. Hunting answers with RAPOSA (FOX). In: **Working Notes for CLEF 2006 Workshop**. [s.n.], 2006. Disponível em: <<http://ceur-ws.org/Vol-1172/CLEF2006wn-QACLEF-Sarmiento2006.pdf>>. Citado nas páginas 41 e 43.
- SEBASTIANI, F. Machine learning in automated text categorization. **ACM Computing Surveys**, ACM, New York, NY, USA, v. 34, n. 1, p. 1–47, mar. 2002. ISSN 0360-0300. Disponível em: <<http://doi.acm.org/10.1145/505282.505283>>. Citado na página 62.
- SEVERYN, A.; MOSCHITTI, A. Learning to rank short text pairs with convolutional deep neural networks. In: **Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval**. New York, NY, USA: ACM, 2015. (SIGIR '15), p. 373–382. ISBN 978-1-4503-3621-5. Disponível em: <<http://doi.acm.org/10.1145/2766462.2767738>>. Citado nas páginas 30, 31, 49, 50, 73, 78 e 115.
- SIMMONS, R. F. Answering English questions by computer: A survey. **Commun. ACM**, ACM, New York, NY, USA, v. 8, n. 1, p. 53–70, jan. 1965. ISSN 0001-0782. Disponível em: <<http://doi.acm.org/10.1145/363707.363732>>. Citado na página 35.
- SIMÕES, A.; COSTA, L.; MOTA, C. Tirando o chapéu à Wikipédia: A coleção do Págico e o Cartola. **Linguamática**, v. 4, n. 1, p. 19–30, Abril 2012. ISSN 1647–0818. Disponível em: <<http://linguamatica.com/index.php/linguamatica/article/view/121/166>>. Citado nas páginas 36 e 53.
- SJAFRIZAL, Y.; BUDI, I.; ADRIANI, M.; ARTHUR, P. Question answering system using query expansion and heuristic features. In: **CLEF (Working Notes)**. [S.l.: s.n.], 2013. Citado nas páginas 42 e 43.

- SOCHER, R.; KARPATHY, A.; LE, Q. V.; MANNING, C. D.; NG, A. Y. Grounded compositional semantics for finding and describing images with sentences. **Transactions of the Association for Computational Linguistics**, v. 2, p. 207–218, 2014. Disponível em: <<https://www.transacl.org/ojs/index.php/tacl/article/view/325>>. Citado na página 50.
- SORICUT, R.; BRILL, E. Automatic question answering: Beyond the factoid. In: DUMAIS, D. M. S.; ROUKOS, S. (Ed.). **HLT-NAACL 2004: Main Proceedings**. Boston, Massachusetts, USA: Association for Computational Linguistics, 2004. p. 57–64. Disponível em: <<http://anthology.aclweb.org/N/N04/N04-1008.pdf>>. Citado na página 39.
- SPARCK-JONES, K. A statistical interpretation of term specificity and its application in retrieval. **Journal of Documentation**, v. 28, n. 1, p. 11–21, 1972. Disponível em: <<https://doi.org/10.1108/eb026526>>. Citado na página 72.
- SRIVASTAVA, N.; HINTON, G.; KRIZHEVSKY, A.; SUTSKEVER, I.; SALAKHUTDINOV, R. Dropout: A simple way to prevent neural networks from overfitting. **Journal of Machine Learning Research**, v. 15, p. 1929–1958, 2014. Disponível em: <<http://jmlr.org/papers/v15/srivastava14a.html>>. Citado nas páginas 78 e 115.
- TensorFlow. **Vector Representations of Words**. 2017. Disponível em: <https://www.tensorflow.org/tutorials/word2vec#motivation_why_learn_word_embeddings>. Acesso em: 23/08/2017. Citado na página 61.
- TRAINA JR., C.; TRAINA, A.; SEEGER, B.; FALOUTSOS, C. Slim-trees: High performance metric trees minimizing overlap between nodes. In: ZANIOLO, C.; LOCKEMANN, P. C.; SCHOLL, M. H.; GRUST, T. (Ed.). **Advances in Database Technology — EDBT 2000: 7th International Conference on Extending Database Technology Konstanz, Germany, March 27–31, 2000 Proceedings**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000. p. 51–65. ISBN 978-3-540-46439-6. Disponível em: <https://doi.org/10.1007/3-540-46439-5_4>. Citado nas páginas 33, 86, 87 e 106.
- TRAN, H. Q.; TRAN, V.; VU, T.; NGUYEN, M.; PHAM, S. B. Jaist: Combining multiple features for answer selection in community question answering. In: **Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)**. Association for Computational Linguistics, 2015. p. 215–219. Disponível em: <<http://aclweb.org/anthology/S15-2038>>. Citado na página 49.
- TSOUMAKAS, G.; KATAKIS, I.; VLAHAVAS, I. Mining multi-label data. **Data mining and knowledge discovery handbook**, Springer, p. 667–685, 2010. Citado nas páginas 92 e 126.
- TURNEY, P. D.; PANTEL, P. From frequency to meaning: Vector space models of semantics. **Journal of Artificial Intelligence Research**, v. 37, p. 141–188, 2010. Disponível em: <<http://jair.org/papers/paper2934.html>>. Citado nas páginas 62 e 63.
- VINCENT, P.; LAROCHELLE, H.; BENGIO, Y.; MANZAGOL, P.-A. Extracting and composing robust features with denoising autoencoders. In: **Proceedings of the 25th International Conference on Machine Learning**. New York, NY, USA: ACM, 2008. (ICML '08), p. 1096–1103. ISBN 978-1-60558-205-4. Disponível em: <<http://doi.acm.org/10.1145/1390156.1390294>>. Citado nas páginas 48 e 99.

- VISHWANATHAN, S. V. N.; SCHRAUDOLPH, N. N.; KONDOR, R.; BORGWARDT, K. M. Graph kernels. **Journal of Machine Learning Research**, v. 11, n. Apr, p. 1201–1242, 2010. Disponível em: <<http://www.jmlr.org/papers/v11/vishwanathan10a.html>>. Citado na página 84.
- VOORHEES, E. Overview of the TREC 2003 question answering track. In: **TREC**. [s.n.], 2003. v. 2003, p. 54–68. Disponível em: <<http://trec.nist.gov/pubs/trec12/papers/QA.OVERVIEW.pdf>>. Citado nas páginas 39 e 53.
- WANG, D. Answering contextual questions based on ontologies and question templates. **Frontiers of Computer Science in China**, v. 5, n. 4, p. 405–418, Dec 2011. ISSN 1673-7466. Disponível em: <<https://doi.org/10.1007/s11704-011-1031-9>>. Citado nas páginas 41 e 45.
- WANG, M.; SMITH, N. A.; MITAMURA, T. What is the Jeopardy model? a quasi-synchronous grammar for QA. In: **Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)**. Prague, Czech Republic: Association for Computational Linguistics, 2007. p. 22–32. Disponível em: <<http://www.aclweb.org/anthology/D/D07/D07-1003>>. Citado nas páginas 50, 51, 53 e 56.
- WANG, Z.; HAMZA, W.; FLORIAN, R. Bilateral multi-perspective matching for natural language sentences. **arXiv preprint arXiv:1702.03814**, 2017. Citado nas páginas 30 e 56.
- WANG, Z.; MI, H.; ITTYCHERIAH, A. Sentence similarity learning by lexical decomposition and composition. In: **Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers**. The COLING 2016 Organizing Committee, 2016. p. 1340–1349. Disponível em: <<http://aclweb.org/anthology/C16-1127>>. Citado nas páginas 30, 31, 49, 50, 56, 73, 76, 79, 80, 81, 113 e 117.
- WANTROBA, E. J.; ROMERO, R. A. F. An interactive question-answer system with dialogue for a receptionist avatar. In: **2015 12th Latin American Robotics Symposium and 2015 3rd Brazilian Symposium on Robotics (LARS-SBR)**. [S.l.: s.n.], 2015. p. 360–365. Citado nas páginas 41 e 46.
- WEBB, N.; WEBBER, B. Special issue on interactive question answering: Introduction. **Natural Language Engineering**, Cambridge University Press, v. 15, n. 1, p. 1–8, 2009. Citado na página 40.
- WESTON, J.; BORDES, A.; CHOPRA, S.; RUSH, A. M.; MERRIËNBOER, B. van; JOULIN, A.; MIKOLOV, T. Towards AI-complete question answering: A set of prerequisite toy tasks. **arXiv preprint arXiv:1502.05698**, 2015. Disponível em: <<https://arxiv.org/abs/1502.05698>>. Citado na página 129.
- WILKENS, R.; VILLAVICENCIO, A.; MULLER, D.; WIVES, L.; SILVA, F. da; LOH, S. COMUNICA: A question answering system for brazilian portuguese. In: **Proceedings of the 23rd International Conference on Computational Linguistics: Demonstrations**. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010. (COLING '10), p. 21–24. Disponível em: <<http://dl.acm.org/citation.cfm?id=1944284.1944290>>. Citado na página 46.

WOODS, W. A.; KAPLAN, R. M.; NASH-WEBBER, B. **The lunar sciences natural language information system**. [S.l.]: Bolt, Beranek and Newman, Inc., 1972. Relatório técnico. Citado nas páginas 41 e 45.

YANG, Y.; YIH, W.-t.; MEEK, C. WikiQA: A challenge dataset for open-domain question answering. In: **Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing**. Association for Computational Linguistics, 2015. p. 2013–2018. Disponível em: <<http://aclweb.org/anthology/D15-1237>>. Citado nas páginas 30, 32, 50, 51, 52, 95 e 101.

YIN, W.; SCHÜTZE, H.; XIANG, B.; ZHOU, B. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. **Transactions of the Association of Computational Linguistics – Volume 4, Issue 1**, p. 259–272, 2016. Disponível em: <<http://aclweb.org/anthology/Q16-1019>>. Citado na página 101.

ZHANG, Y. Contextualizing consumer health information searching: An analysis of questions in a social Q&A community. In: **Proceedings of the 1st ACM International Health Informatics Symposium**. New York, NY, USA: ACM, 2010. (IHI '10), p. 210–219. ISBN 978-1-4503-0030-8. Disponível em: <<http://doi.acm.org/10.1145/1882992.1883023>>. Citado nas páginas 30, 57, 59 e 126.