

Faculdade de Ciências e Tecnologia  
Departamento de Engenharia Informática

# Exploração de técnicas para a Resposta Automática a Perguntas por Agentes Conversacionais

José Pedro Pessoa dos Santos

Dissertação no contexto do Mestrado em Engenharia Informática, Especialização em Sistemas Inteligentes orientada pelos Professores Hugo Oliveira e Ana Alves, e apresentada à Faculdade de Ciências e Tecnologia / Departamento de Engenharia Informática.

Setembro 2019

1 2  9 0

UNIVERSIDADE D  
COIMBRA



---

## Resumo

Esta tese introduz o tema da Resposta Automática a Perguntas por Agentes Conversacionais, cujo interesse tem vindo a crescer ao longo dos últimos anos devido à importância que estes têm na criação de uma relação entre os utilizadores e os produtos que utilizam, por exemplo, através de assistentes pessoais, websites de apoio ao cliente, entre outros.

Numa primeira fase foram desenvolvidos modelos para o cálculo da Similaridade Semântica Textual em português com o objetivo de os utilizar para fazer o mapeamento de perguntas feitas por um utilizador e as respetivas respostas. Estes modelos recorrem ao cálculo de características textuais entre pares de frases que são utilizadas para treinar algoritmos de aprendizagem computacional supervisionada que lhes atribuem um único valor de similaridade. A avaliação dos modelos foi feita com recurso à coleção da tarefa ASSIN de 2016, e, apesar de não terem atingido a performance do atual estado-da-arte, o seu desempenho está em par com os das melhores equipas participantes.

Numa segunda fase, o modelo desenvolvido com melhor desempenho foi integrado num agente de diálogo de domínio específico. Este é capaz de identificar perguntas fora do seu domínio e, assim, responder com base num conjunto de frases de legendas de filmes, o que torna a conversa mais natural. A avaliação foi feita com recurso a variações das perguntas da base de conhecimento do agente, que permitiram quantificar o número de respostas corretas num cenário de diálogo mais próximo da realidade. Os resultados obtidos são promissores e substancialmente melhores do que os das baselines criadas para comparação.

## Palavras-Chave

Processamento de Linguagem Natural, Similaridade Semântica Textual, Resposta Automática a Perguntas, Agentes Conversacionais, Chatbots, Aprendizagem Computacional.



---

## Abstract

This thesis introduces the subject of Question Answering by Conversational Agents, whose interest has been rising over the past few years due to their importance on creating a relationship between consumers and the products they use, for example, through personal assistants, customer support websites, among others.

In a first phase, a set of models to compute the Semantic Textual Similarity between sentences in Portuguese were developed with the aim of mapping questions from a user and their corresponding responses. These models required the extraction of textual features between pairs of sentences in order to train a variety of machine learning algorithms that can assign them a single value of similarity. The evaluation of these models resorted to the ASSIN 2016 task collection, and, although they did not reach state-of-the-art performance, it was up to the results obtained by the best participating teams.

In a second phase, the model with the best performance was integrated into a domain-specific dialogue agent as its search engine. This agent is capable of identifying out-of-domain interactions and respond to them using a set of movie subtitles, which make the conversation feel more natural. In order to test how well the agent performed, a set of variations of the questions in the agent's knowledge-base were created. These allowed to quantify the number of correct responses in a more realistic conversational environment. The results were promising and substantially superior to the baselines developed.

## Keywords

Natural Language Processing, Semantic Textual Similarity, Question Answering, Conversational Agents, Chatbots, Machine Learning



---

## Agradecimentos

Começo por agradecer aos meus orientadores, os Professores Hugo Oliveira e Ana Alves, pela excelente supervisão que fizeram do meu trabalho e disponibilidade que sempre demonstraram.

Aos meus colegas e amigos por todos os momentos que passámos juntos.

Por fim, agradecer aos meus pais, irmã e madrinha pelo apoio incondicional que me deram ao longo destes cinco anos, sem o qual a concretização desta tese nunca teria sido possível. A eles devo quem sou hoje, e, por isso, para eles vai o meu maior obrigado.

Este trabalho foi financiado pela iniciativa Nacional em Competências Digitais da FCT, Portugal INCoDe 2030, no âmbito do projeto de demonstração “Agente Inteligente para Atendimento no Balcão do Empreendedor” (AIA).

*José Santos*  
*Coimbra, Setembro 2019*





# Conteúdo

<b>Glossário</b>	<b>xi</b>
<b>Acrónimos</b>	<b>xiii</b>
<b>1 Introdução</b>	<b>1</b>
1.1 Motivação . . . . .	2
1.2 Objetivos . . . . .	3
1.3 Abordagem . . . . .	3
1.4 Contribuições . . . . .	4
1.5 Estrutura do documento . . . . .	4
<b>2 Estado da Arte</b>	<b>7</b>
2.1 Resposta automática a perguntas . . . . .	7
2.2 Agentes de diálogo e Chatbots . . . . .	8
2.2.1 Agentes de diálogo orientados a uma tarefa . . . . .	9
2.2.2 Chatbots . . . . .	10
2.3 Similaridade Semântica Textual . . . . .	11
2.3.1 Métodos para o pré-processamento do texto . . . . .	13
2.3.2 Métodos de processamento para a extração de caraterísticas . . . . .	14
2.3.3 Algoritmos de aprendizagem . . . . .	17
2.3.4 Caraterísticas textuais . . . . .	24
2.3.5 Ferramentas utilizadas . . . . .	28
2.3.6 Visão Integradora . . . . .	32
<b>3 Cálculo da Similaridade Semântica Textual em Português</b>	<b>33</b>
3.1 Caraterísticas aplicadas aos modelos . . . . .	33
3.1.1 Caraterísticas lexicais . . . . .	34
3.1.2 Caraterísticas sintáticas . . . . .	35
3.1.3 Caraterísticas semânticas . . . . .	36
3.1.4 Caraterísticas distribucionais . . . . .	37
3.2 Seleção de Caraterísticas . . . . .	38
3.3 Modelos para o cálculo da SST . . . . .	39
3.3.1 Modelos baseados em Vetores de Palavras . . . . .	40
3.3.2 Modelos baseados na aprendizagem de Algoritmos de Regressão . . . . .	40
3.4 Avaliação dos modelos de SST . . . . .	42
<b>4 Agente Conversacional baseado em Similaridade Semântica Textual</b>	<b>47</b>
4.1 Integração do modelo SST num Agente de Diálogo . . . . .	47
4.2 Avaliação do desempenho do Agente de Diálogo . . . . .	48
4.3 Identificação e Resposta a Perguntas Fora-do-Domínio . . . . .	53
<b>5 Conclusão</b>	<b>57</b>

5.1 Trabalho futuro . . . . .	58
<b>Bibliografia</b>	<b>59</b>

# Glossário

**Baseline** Termo utilizado para definir um conjunto de critérios base utilizados como controlo em projetos de investigação, que servem para avaliar o progresso do trabalho desenvolvido. 5, 39, 40, 47, 51–54, 57

**Corpus** Conjunto de documentos ou informações relativos a um determinado tema ou domínio. 10, 11, 17, 28, 31, 32, 37, 53, 54

**Overfitting** Termo utilizado para descrever um sistema que se adaptou demasiado bem a um conjunto de dados e que por esse motivo pode ter dificuldade em ajustar-se a um novo conjunto de dados. 17, 20, 38



# Acrónimos

**ASSIN** Avaliação de Similaridade Semântica e Inferência Textual. xvii, 3, 11, 12, 33, 39–42, 45, 58

**BCL** Bases de Conhecimento Lexical. 24, 28, 29, 31, 36, 37

**BDE** Portal do Balcão do Empreendedor. 2, 4, 48, 52

**EM** Entidades Mencionadas. 8, 15, 24, 34, 36

**ILN** Interface em Linguagem Natural. 3

**PLN** Processamento de Linguagem Natural. 1, 2, 4, 9, 28, 29, 33

**SST** Similaridade Semântica Textual. iii, ix, xvii, 1–4, 7, 11–14, 17, 21, 24, 28, 32–35, 39–43, 45, 47, 48, 50, 51, 53, 54, 57, 58



# Lista de Figuras

2.1	Exemplo de utilização da análise de dependências para uma frase em inglês.	16
2.2	Exemplo de utilização do alinhamento de frases entre uma frase em inglês e francês. . . . .	16
2.3	Exemplo de utilização de uma função de <i>kernel</i> numa MVS (Kim, 2013). . .	18
2.4	Exemplo de aplicação da MVS num problema de regressão. . . . .	19
2.5	Arquitetura de um modelo baseado em <i>Random Forest</i> . <b>Fonte:</b> (Verikas et al., 2016) . . . . .	22
2.6	Exemplo da utilização de uma janela de tamanho 2 para a obtenção de exemplos de treino. . . . .	26
2.7	Exemplo da arquitetura de uma rede neuronal utilizada para treinar um modelo baseado em Skip-Gram. . . . .	27





# Lista de Tabelas

2.1	Exemplos de frases de treino da tarefa ASSIN para português de Portugal. . .	12
2.2	Desempenho das equipas nas diferentes tarefas. . . . .	13
2.3	Pré-processamento e Algoritmos de aprendizagem utilizados pelas equipas nas diferentes tarefas. . . . .	23
2.4	Caraterísticas textuais extraídas e ferramentas utilizadas pelas equipas nas diferentes tarefas. . . . .	31
3.1	Caraterísticas textuais mais relevantes de acordo com os resultados do processo de seleção. . . . .	43
3.2	Desempenho Máximo e Médio dos modelos desenvolvidos com recurso ao 1º conjunto de caraterísticas quando testados nos datasets em PTPT e PTBR. . . . .	43
3.3	Desempenho dos modelos quando testados nos datasets em PTPT e PTBR sem seleção de caraterísticas. . . . .	44
3.4	Desempenho dos modelos quando testados nos datasets em PTPT e PTBR com seleção de caraterísticas. . . . .	44
4.1	Exemplos de variações das perguntas originais usadas para testar o agente. . .	49
4.2	Desempenho do agente de diálogo quando testado em todas as variações das FAQs, utilizando o modelo SST selecionado. . . . .	50
4.3	Desempenho do agente de diálogo quando testado em todas as variações das FAQs, utilizando o modelo SST selecionado sobre uma pré-seleção de 30 possíveis correspondências corretas. . . . .	50
4.4	Desempenho do agente de diálogo quando testado em todas as variações das FAQs, utilizando um novo modelo SST ao qual foram removidas as caraterísticas textuais baseadas no modelo <i>FastText</i> . . . . .	51
4.5	Desempenho da baseline Whoosh quando testado em todas as variações das FAQs. . . . .	52
4.6	Desempenho da baseline Chatterbot quando testado em todas as variações das FAQs. . . . .	53
4.7	Desempenho do agente de diálogo quando testado em todas as variações das FAQs por limiar de similaridade, utilizando o modelo SST selecionado. . . .	54



# Capítulo 1

## Introdução

A interação entre humanos e computadores tem sido objeto de estudo há várias décadas, com vista a melhorar a forma como ela se estabelece. De entre as mais diversas formas de interação, a língua, falada ou escrita, constitui a forma mais simples e natural de humanos interagirem. Diariamente, os seres humanos utilizam-na como forma de comunicação entre si, e a língua é parte da sua identidade desde o momento em que aprendem a falar. Falar e escrever são, talvez por isso, das atividades mais básicas que qualquer pessoa aprende a efetuar. Esta seria também uma forma natural de interação com computadores. Contudo, algo que é relativamente simples para os humanos, revela-se uma tarefa substancialmente mais complexa para as máquinas.

A língua humana, também chamada de Linguagem Natural (LN) por oposição às linguagens formais, apresenta determinadas características que dificultam o seu processamento por uma máquina, das quais se pode destacar a variabilidade linguística – palavras (sinonímia) ou frases (paráfrase) diferentes que apresentam o mesmo significado/sentido; ou a ambiguidade a vários níveis, incluindo a homonímia – palavras com a mesma grafia e pronúncia que apresentam um significado diferente, de acordo com o contexto em que estão inseridas (e.g. manga: fruta/parte de uma peça de roupa); a homografia – palavras com a mesma grafia, mas pronúncia diferente (impossível de detetar em texto) que apresentam significado diferente (e.g. olho: órgão da visão/1<sup>a</sup> pessoa do singular do verbo olhar); registos de linguagem (e.g. formal e informal), domínio do documento (e.g. académico, artístico) ou figuras de estilo (e.g. metáfora, eufemismo, hipérbole).

Na procura de extrair conhecimento de um texto e de o manipular de forma computacionalmente eficiente, surgiu uma área da computação designada por Processamento de Linguagem Natural (PLN), que lida com a manipulação da linguagem a vários níveis, tais como o morfológico, sintático e semântico.

O foco desta tese incidirá sobre a análise semântica do texto, mais concretamente no que diz respeito à similaridade entre pedaços textuais escritos em português. Neste contexto surge um dos principais objetivos deste projeto que será a exploração de modelos para o cálculo da Similaridade Semântica Textual (SST), que consiste na atribuição de um valor de similaridade entre pares de frases, com base no sentido das mesmas e não apenas nas palavras que as compõem e respetiva grafia. Assumindo que existem já pares de frases com o respetivo valor de similaridade anotado, como é o caso para português (Fonseca et al., 2016), chegar até uma função para o cálculo da SST pode passar pela aprendizagem computacional supervisionada. Mais propriamente, é extraído um conjunto de características a partir dos pares, e considerado o seu valor de similaridade, que são depois usados como dados de entrada para um algoritmo de aprendizagem. No caso desta

tese, os modelos desenvolvidos serão ainda aplicados à associação automática de interações em linguagem natural com perguntas de uma base de conhecimento, para as quais já existem respostas que poderão assim ser recuperadas de uma forma mais natural.

Este trabalho está a ser realizado no contexto do projeto de investigação AIA – Agente Inteligente para Atendimento no Balcão do Empreendedor, onde se pretende desenvolver mecanismos que permitam interagir através de linguagem natural escrita com o Portal do Balcão do Empreendedor (BDE)<sup>1</sup>, um portal que constitui um ponto de acesso aos serviços digitais relacionados com o exercício de atividade económica em Portugal. O BDE disponibiliza mais de um milhar de serviços, descritos ao longo de diversas páginas, onde se encontram ainda perguntas já respondidas, as chamadas *Frequently Asked Questions* (FAQs).

A estrutura deste capítulo é feita da seguinte forma: na secção 1.1 será abordada a motivação para o desenvolvimento deste tese, na secção 1.2 os objetivos que se pretendem atingir, na secção 1.3 a abordagem a seguir de forma a atingir os objetivos apresentados, na secção 1.4 serão apresentadas as contribuições esperadas no final do desenvolvimento deste trabalho e na secção 1.5 serão descritos cada um dos capítulos que compõem este documento.

## 1.1 Motivação

A Similaridade Semântica Textual surge como uma área importante para a melhoria da capacidade de compreensão de texto em linguagem natural pelos computadores. O crescente destaque da mesma pode ser comprovado pela organização de diversas tarefas, quer para o inglês, quer, mais recentemente, para o português (Fonseca et al., 2016), em competições destinadas a avaliar sistemas que realizam tarefas de PLN.

As aplicações da similaridade semântica em sistemas presentes no dia-a-dia são diversas, podendo ser utilizada em sistemas de pesquisa de informação em grandes coleções de documentos, como a Web, agentes conversacionais, tais como chatbots, cuja utilização como ferramenta de apoio ao consumidor em sites de comércio eletrónico tem sido cada vez maior, e assistentes virtuais, como a *Siri* ou *Google Assistant*.

É neste contexto que se torna importante a extração de características textuais e técnicas para as explorar, com vista à criação de modelos que possam ser utilizados em aplicações de diferentes domínios e possam ajudar a melhorar o estado-da-arte.

Modelos baseados em técnicas para o cálculo da SST apresentam características que os tornam adequados à utilização em agentes de diálogo baseados em FAQs, uma vez que permitem atribuir um valor de similaridade entre pedaços textuais, o que os torna úteis como mecanismos de mapeamento entre a pergunta do utilizador e a resposta do agente. Para além disso, permitem identificar perguntas que não estejam dentro do domínio do agente, caso o processo de mapeamento retorne valores de similaridade abaixo de um determinado limite. Isto permite criar mecanismos que lidem com este tipo de interações e tornem a conversa do agente mais "humana", o que é importante tendo em conta que o mesmo pode ser utilizado, por exemplo, para agilizar a interação dos utilizadores com o BDE.

---

<sup>1</sup><https://www.portaldocidadao.pt/balcao-do-empendedor>

## 1.2 Objetivos

Os principais objetivos deste trabalho são:

1. Uniformizar ao máximo a extração de características textuais de texto em português, idealmente numa única pipeline e única linguagem de programação.
2. Analisar a relevância das características textuais já usadas noutros sistemas de cálculo da SST e tirar conclusões acerca da sua importância para o cálculo da mesma, com a finalidade de simplificar o processo de extração de características.
3. Desenvolver modelos para o cálculo da SST com base na experimentação, quer de diferentes características textuais, quer de diferentes algoritmos de aprendizagem.
4. Desenvolver um agente / Interface em Linguagem Natural (ILN) que utilize o melhor modelo de SST para mapear interações do utilizador com perguntas da sua base de conhecimento.

## 1.3 Abordagem

De forma a atingir os objetivos acima mencionados foram explorados diversos sistemas utilizados no cálculo da SST e analisadas as suas abordagens, de forma a poder inferir qual a melhor forma de abordar o problema. Um desses sistemas foi o ASAPP 2.0 (Alves et al., 2018) desenvolvido no CISUC<sup>2</sup> que obteve os melhores resultados no cálculo da similaridade semântica em português, baseando-se num grande conjunto de características, extraídas com recurso a diferentes ferramentas e recursos.

Após esta análise, iniciou-se o processo de desenvolvimento dos modelos de SST. Estes baseiam-se na aprendizagem supervisionada de algoritmos de regressão com base num conjunto de características textuais extraídas dos pares de frases do conjunto de dados de treino. Estes fazem parte dos datasets de treino da coleção da tarefa ASSIN (Fonseca et al., 2016) e são compostos por pares de frases em português com similaridade anotada.

A análise do desempenho das características textuais extraídas e dos algoritmos de aprendizagem utilizados nos modelos desenvolvidos para o cálculo da SST foi efetuada em paralelo. Sempre que se pretendia testar um determinado algoritmo eram aplicados três métodos distintos para a seleção automática de características. O método com maior ganho de desempenho para o modelo, com base numa métrica apresentada na secção 3.2, era escolhido, bem com as características por este selecionadas. Os modelos desenvolvidos eram posteriormente avaliados com recurso aos datasets de teste da coleção ASSIN.

Por último, o modelo que apresentou o melhor desempenho foi integrado num agente de diálogo aplicado a listas de FAQs. Estas listas serviram como base de conhecimento do agente, e foram utilizadas para fazer o mapeamento com a interação do utilizador. Dado que esta base de conhecimento era composta por pares de frases no formato Pergunta-Resposta, a similaridade era calculada entre as perguntas existentes nessa base e a interação do utilizador, e a resposta à pergunta que apresentasse o valor de similaridade mais alto era devolvida. Sempre que o valor de similaridade mais elevado se encontrasse abaixo de um limite pré-estabelecido, a interação do utilizador era considerada fora-do-domínio do agente, o que fazia com que este tentasse dar uma resposta com base num conjunto de

---

<sup>2</sup><https://www.cisuc.uc.pt>

legendas de filmes. Este mecanismo foi implementado com recurso a um sistema para a recuperação de informação e permitiu tornar o diálogo com o agente mais natural. O agente foi ainda avaliado com recurso a variações das perguntas existentes na sua base de conhecimento e os resultados comparados com outros dois sistemas.

O desenvolvimento do agente, desde a extração de características ao treino dos algoritmos de aprendizagem, foi feito em Python devido à existência de diversas bibliotecas externas, tais como, o NLTK, gensim e scikit-learn, que agilizam a implementação de programas que lidam com PLN e aprendizagem computacional.

## 1.4 Contribuições

As principais contribuições alcançadas com o desenvolvimento deste trabalho podem ser resumidas nos seguintes pontos:

- Novos modelos de regressão baseados em Python para o cálculo da SST em português, baseados em diferentes algoritmos de aprendizagem e características textuais.
- Estudo das características mais relevantes para o cálculo da SST em português.
- Agente conversacional com base em técnicas de Similaridade Semântica Textual, capaz de identificar e lidar com perguntas fora do domínio específico para o qual foi desenvolvido.

## 1.5 Estrutura do documento

Este documento está organizado nos seguintes capítulos:

No **Capítulo 2** será apresentado o estado da arte para a Resposta Automática a Perguntas e as duas abordagens mais comuns para resolver esse problema, através de sistemas baseados na extração de informação e sistemas baseados em conhecimento. Serão também introduzidos os dois principais tipos de agente de diálogo, nomeadamente, Chatbots e Agentes direcionados para tarefas. Por último, será analisado o estado da arte para a SST, desde as características que se costumam extrair do texto aos algoritmos de aprendizagem mais utilizados na criação de modelos para o cálculo da similaridade. Esta análise será feita com base na análise de sistemas que participaram em competições, quer para o português, quer para o inglês, que avaliavam a capacidade dos sistemas calcularem a similaridade semântica entre um conjunto de frases.

No **Capítulo 3** será abordado o processo de desenvolvimento dos modelos para o cálculo da SST. Este processo iniciou-se com a extração de características textuais utilizadas para treinar algoritmos de regressão capazes de representar a similaridade entre pedaços textuais através de funções. Em simultâneo com o treino dos modelos, foi também efetuada a seleção das características textuais, que, de acordo com os algoritmos utilizados eram mais relevantes. Por último, foi efetuada a avaliação dos modelos desenvolvidos com o objetivo de selecionar o melhor e de o integrar num agente conversacional.

No **Capítulo 4** será apresentada a forma como o modelo selecionado foi aplicado a um agente de diálogo cujo objetivo será dar resposta a um conjunto de FAQs no domínio do BDE. Será abordado o mecanismo adotado pelo agente para lidar com perguntas fora do domínio para o qual foi desenvolvido. Por fim, será apresentada a estratégia escolhida

para avaliar o desempenho do agente, bem como este se compara com três outras baselines: duas desenvolvidas em Python e uma em Java.

No **Capítulo 5** serão apresentadas as principais conclusões, debatidas as contribuições desta tese e descrito o trabalho a desenvolver no futuro.





## Capítulo 2

# Estado da Arte

Neste capítulo será abordado o estado da arte no que diz respeito aos métodos utilizados para a criação de modelos para o cálculo da SST, englobando técnicas de pré-processamento de dados, características textuais extraídas, algoritmos de aprendizagem e ferramentas utilizadas. Será também explorado o estado da arte dos agentes de diálogo e chatbots, bem como da resposta automática a perguntas.

### 2.1 Resposta automática a perguntas

A resposta automática a perguntas surge como uma forma de permitir que os humanos possam interagir com computadores em linguagem natural com o objetivo de obterem respostas a perguntas que queiram colocar. Durante a década de 60 surgiram dois dos maiores paradigmas para lidar com este tipo de problemas: **sistemas baseados em extração de informação**, que utilizam coleções de texto não estruturadas para fazer a pesquisa da resposta à pergunta do utilizador, e **sistemas baseados em conhecimento**, que convertem a pergunta do utilizador numa *query* formal utilizada numa base de conhecimento estruturada. Apesar do objetivo em comum de cada um destes, a forma como abordam o problema é bastante distinta, em parte devido ao formato dos dados utilizados para a procura de informação.

Os **sistemas baseados em extração de informação** recorrem a grandes quantidades de dados textuais, por exemplo, existentes em coleções como a Wikipédia<sup>1</sup>, para fazer o mapeamento entre a pergunta do utilizador e uma possível resposta que possa existir nesses mesmos conjuntos de dados. Este processo engloba várias etapas sequenciais, desde o momento em que a pergunta é colocada ao sistema até ao momento em que a resposta é devolvida. Numa primeira fase é efetuado o processamento da pergunta, de forma a identificar o objetivo da mesma e qual o tipo de resposta esperado. Por exemplo, para a pergunta "Qual é o distrito de Portugal com maior população?", o objetivo será identificar o distrito em Portugal com maior população e a resposta esperada será o nome desse distrito. Com base na identificação dos componentes da pergunta, é depois efetuada uma pesquisa num conjunto de dados, de forma a extrair documentos que possam conter a resposta. Esta pesquisa pode ser enriquecida com recurso a palavras relacionadas com o domínio da pergunta ou sinónimos dos termos que a constituem. Posteriormente, o mapeamento entre o conteúdo da pergunta e a informação presente nos dados de procura pode ser feito com recurso, por exemplo, a modelos para o cálculo da Similaridade Semântica Textual, como os

---

<sup>1</sup><https://www.wikipedia.org>

discutidos na secção 2.3; cosseno entre os respetivos vetores de TF-IDF (ver secção 2.3.4); sobreposição de n-gramas (ver secção 2.3.2), entre outros. Os documentos extraídos são de seguida ordenados de acordo com a sua relevância, através da identificação, por exemplo, de palavras-chave e entidades reconhecidas presentes quer na pergunta quer em pedaços de texto desses mesmos documentos. Por último é efetuada a extração de pedaços textuais que vão de encontro aos critérios da questão colocada, através de uma técnica designada por *span labeling*, que consiste na identificação de uma possível resposta em pedaços de texto na vizinhança dos identificados nos documentos que permitiram fazer o mapeamento com a pergunta. Nos exemplos abaixo demonstra-se a utilização desta técnica para encontrar a resposta às perguntas "Quem é o Presidente dos Estados Unidos da América?" e "Quem criou a personagem Sherlock Holmes?". A resposta às perguntas encontra-se sublinhada e os pedaços textuais que permitiram fazer o mapeamento da pergunta com as frases onde poderia ser encontrada a resposta a negrito.

"Quem é o **Presidente dos Estados Unidos** da América?"

Donald John Trump é um empresário, personalidade televisiva e político americano, sendo atualmente o **45.º presidente dos Estados Unidos**.

"Quem criou a personagem **Sherlock Holmes**?"

**Sherlock Holmes** é um personagem de ficção da literatura britânica criado pelo médico e escritor Sir Arthur Conan Doyle.

Os **sistemas baseados em conhecimento** fazem o mapeamento da pergunta do utilizador numa representação lógica da mesma, através da identificação dos elementos da frase que permitem identificar o seu objetivo. Esta identificação pode ser feita, por exemplo, com recurso ao reconhecimento de Entidades Mencionadas e identificação de funções sintáticas, o que é importante visto que normalmente a finalidade da pergunta é indicada pelo verbo. A pergunta introduzida é, com base nos elementos identificados, transformada numa representação lógica, por exemplo, a pergunta "Quando nasceu Albert Einstein?" poderia ser representada por *ano-de-nascimento (Albert Einstein, ?x)*, onde *x* representa o ano que se pretende descobrir. Esta representação lógica é utilizada para a construção de uma *query* para a pesquisa numa base de dados. O SQL e SPARQL são duas das linguagens mais utilizadas para este efeito, permitindo a primeira realizar a pesquisa numa base de dados relacional, e a segunda numa base de conhecimento em RDF (*Resource Description Framework*), como, por exemplo, uma ontologia. As ontologias são modelos de dados que representam um conjunto de conceitos pertencentes a um ou mais domínios e que estabelecem relações de inferência entre os mesmos. A resposta à pergunta será então o resultado da *query* efetuada.

## 2.2 Agentes de diálogo e Chatbots

Os agentes diálogo são programas desenvolvidos com o objetivo de simular a capacidade de diálogo dos humanos, quer seja através de texto, voz ou ambos. Estes agentes devem ser capazes de estabelecer interações que vão muito para além da simples resposta a perguntas. Alguns são capazes de controlar aparelhos eletrónicos numa casa, ajudar na navegação e até criar jogos do tipo quiz, nos quais os utilizadores participam através da voz. Estes agentes podem ser divididos, de acordo com a finalidade para a qual são desenvolvidos, em dois grupos (Jurafsky e Martin, 2009): **Agentes de diálogo orientados a uma tarefa** e **Chatbots**.

### 2.2.1 Agentes de diálogo orientados a uma tarefa

Os agentes de diálogo orientados a uma tarefa são sistemas desenvolvidos com o objetivo de desempenhar uma tarefa solicitada pelo utilizador (e.g. ligar as luzes de uma determinada zona da casa, obter informações de trânsito de forma a fornecer o caminho mais rápido para um determinado destino), não sendo geralmente capazes de estabelecer uma conversa longa e/ou complexa. Alguns exemplos são as assistentes pessoais presentes em grande parte dos smartphones, computadores e colunas inteligentes, tais como, Google Assistant, Siri, Alexa e Cortana. Apesar de, como mencionado anteriormente, a generalidade destes sistemas não ser muito eficaz no que toca a estabelecer um longo diálogo, agentes como a Google Assistant e Alexa são já capazes de adaptar as suas respostas com base em interações prévias com o utilizador. Na atualidade, estes sistemas são desenvolvidos com base numa estrutura de conhecimento que define a abrangência de domínios a que o agente deve dar resposta. Dentro de cada domínio são definidos conjuntos de intenções que especificam cada uma das tarefas que o utilizador pode solicitar, sendo que cada uma dessas intenções é ainda composta por um conjunto de variáveis que são preenchidas de acordo com a informação extraída do diálogo com o utilizador. Dependendo da sua complexidade, as intenções podem estar sujeitas a estruturas de controlo responsáveis por assegurar que o agente possui toda a informação necessária para que a intenção seja concretizada com sucesso. Por exemplo, dada uma solicitação para fazer a compra de um bilhete de avião sem qualquer informação adicional, a estrutura de controlo é responsável por garantir que o agente, através do diálogo com o utilizador, consegue obter informação relativamente ao local de origem e destino da viagem, a data de partida e, se necessário, volta, e por fim confirmar se todas as informações estão corretas, de forma a garantir que a ação a efetuar está de acordo com o desejado pelo utilizador. O PLN é aplicado a este processo com o objetivo de permitir identificar o domínio da interação do utilizador, mapear a intenção a que a pergunta deve ser associada e extrair a informação necessária para preencher as variáveis dessa mesma intenção. Algumas das técnicas que podem ser utilizadas para esse efeito são o Reconhecimento de Entidades Mencionadas (ver secção 2.3.2), a análise de dependências sintáticas (ver secção 2.3.4) e a expansão do vocabulário da pergunta através de sinónimos, entre outras. De seguida apresenta-se um exemplo do mapeamento que um agente deste tipo teria de fazer para dar resposta a uma interação no domínio das viagens aéreas:

*Mostra-me viagens de avião no sábado entre Lisboa e Paris.*

o agente deve ser capaz de extrair a seguinte informação:

**DOMÍNIO:** VIAGENS-AÉREAS

**INTENÇÃO:** MOSTRAR-VOOS

**CIDADE-ORIGEM:** Lisboa

**DATA-ORIGEM:** Sábado

**CIDADE-DESTINO:** Paris

### 2.2.2 Chatbots

Ao contrário dos agentes mencionados acima, os chatbots são desenvolvidos com o objetivo de imitar o diálogo humano ao longo de uma conversa longa. Estes sistemas utilizam interações prévias com o utilizador para responder à interação atual, de forma a que a conversa tenha uma sequência lógica e pareça o mais natural possível. É neste aspeto que se verifica a principal diferença entre os chatbots e os agentes direcionados para uma tarefa, visto que os últimos, geralmente, após desempenharem uma tarefa para a qual tenham sido solicitados, não são capazes de utilizar essa informação em futuras interações com o utilizador. Este tipo de agentes pode ser dividido, de acordo com a sua arquitetura, em duas classes:

- **Sistemas baseados em regras:** utilizam um conjunto predefinido de regras como forma de produzir a resposta a dar a uma pergunta colocada pelo utilizador. Cada regra é composta por um padrão e por uma transformação. Sempre que ocorra uma interação do utilizador, o sistema procura fazer o mapeamento entre esta e um dos padrões pré-definidos. Caso este mapeamento seja efetuado com sucesso, o agente aplica a respetiva transformação. Esta consiste num modelo de resposta com secções por preencher. Secções essas que são completadas com informação extraída da interação inicial, tal como, nomes, verbos, locais, entre outros, o que permite tornar a resposta mais pessoal e manter uma sequência lógica de diálogo. Na impossibilidade de fazer o mapeamento, o sistema apresenta uma interação genérica, tal como, *Não percebi o que disse, pode repetir*. Devido à complexidade de gerar um conjunto de regras com elevada abrangência, este tipo de sistemas é essencialmente aplicado a domínios específicos, tal como, a psicologia.

Um dos mais importantes sistemas deste tipo foi o ELIZA (Weizenbaum, 1966), que tinha o objetivo de simular o diálogo de um psicólogo que utilize uma abordagem de terapia não direcionada. Este agente baseava-se no reconhecimento e ranking de palavras-chave na afirmação do utilizador como forma de estabelecer uma relação entre essas mesmas palavras e uma determinada regra. Regra essa que transformava a frase introduzida numa resposta coerente posteriormente devolvida ao utilizador. Caso não fosse detetada nenhuma palavra-chave, o sistema utilizava uma resposta predefinida. Em 1972, foi desenvolvido um outro sistema, PARRY (Colby et al., 1972), também aplicado à área da psicologia, que para além da utilização de expressões regulares, possuía um modelo que definia o seu estado mental e que era responsável por influenciar as respostas do mesmo.

- **Sistemas baseados em corpus:** utilizam um conjunto de dados, geralmente de conversas entre humanos, provenientes de diálogos de filmes, interações em redes sociais, tais como, Twitter<sup>2</sup>, Reddit<sup>3</sup> e Facebook<sup>4</sup>, e plataformas de comunicação como forma de extrair e identificar a melhor resposta a dar tendo em conta a interação do utilizador. Geralmente, estes agentes não fazem qualquer modelação da resposta encontrada no conjunto de dados utilizado, de forma a adaptá-la ao contexto da conversa, preocupando-se apenas com a coerência desta com a interação anterior do utilizador. Estes sistemas podem ser divididos em dois tipos, de acordo com a sua arquitetura:

---

<sup>2</sup><https://twitter.com/>

<sup>3</sup><https://www.reddit.com>

<sup>4</sup><https://www.facebook.com>

- **Sistemas baseados em Recuperação de Informação** - utilizam corpus de diálogo proveniente de legendas de filmes e/ou interações em redes sociais para fazer o mapeamento entre a frase do utilizador e a frase mais semelhante existente no corpus. Para tal, podem aplicar, por exemplo, técnicas de SST. A resposta pode ser dada de duas formas, retornando a interação mais semelhante à do utilizador, ou a interação a seguir a essa. Esta última estratégia baseia-se no facto do corpus de pesquisa utilizado apresentar uma conversa estruturada de forma lógica, e, por consequência, ao mapear uma interação semelhante nesse corpus, uma possível resposta apropriada será a seguinte.
- **Sistemas Sequence-to-sequence (*SEQ2SEQ*)** - utilizam algoritmos de aprendizagem computacional supervisionada treinados num corpus de diálogo com o objetivo de aprenderem um modelo capaz de transformar a interação do utilizador numa resposta coerente. A aprendizagem do modelo pode ser dividida em duas etapas: codificação, na qual cada uma das frases do corpus de treino é convertida num vetor, e descodificação, na qual esse mesmo vetor é utilizado para gerar um conjunto de vetores de possíveis respostas. O vetor inicial e os vetores gerados são utilizados para treinar um modelo que procura encontrar a relação mais coerente entre estes, de acordo com a função objetivo.

### 2.3 Similaridade Semântica Textual

A Similaridade Semântica Textual é uma área de investigação no âmbito do Processamento de Linguagem Natural que se dedica à exploração de técnicas para o cálculo de proximidades no significado entre palavras, frases e documentos. O interesse por este tópico tem crescido nos últimos anos, o que se verifica pela organização de tarefas destinadas à avaliação de sistemas que calculam a similaridade semântica entre frases e que, para além de fomentarem a discussão do tema, permitem avançar o seu estado-da-arte. Estas tarefas são compostas por um ou mais datasets, que consistem num conjunto de pares de frases com similaridade anotada. Parte dos datasets são disponibilizados pela organização das competições com as anotações para que as equipas possam treinar os seus modelos, e outra parte é disponibilizada, posteriormente, sem as anotações para o teste dos modelos. Os resultados obtidos pelas equipas nos dados de teste são enviados à organização para a avaliação dos mesmos. O SemEval é uma tarefa de avaliação conjunta destinada à avaliação de sistemas focados em semântica computacional, na qual algumas das tarefas mencionadas anteriormente se inserem. No que diz respeito à SST, estas tarefas tiveram lugar entre 2012 (Agirre et al., 2012) e 2017 (Cer et al., 2017) e tiveram o inglês como foco principal, sendo que o espanhol e o árabe surgiram nos últimos anos em tarefas multilingues. Para o português surgiu em 2016 a tarefa ASSIN (Avaliação de Similaridade Semântica e Inferência textual) inserida na conferência PROPOR.

Os sistemas que participaram nas competições acima mencionadas apresentam um variado número de abordagens, assentes sobretudo na aprendizagem computacional, que podem ir da existência ou não de pré-processamento do texto, seleção das características do texto a explorar e combinação das mesmas, para além de algoritmos de aprendizagem a utilizar.

De forma a fazer um levantamento das características dos sistemas que obtiveram um melhor desempenho, quer no SemEval, quer no ASSIN, foram recolhidas informações sobre os mesmos no que diz respeito ao pré-processamento do texto, características utilizadas, algoritmos de aprendizagem, recursos externos e desempenho conseguido. Os dados obtidos dizem respeito às duas melhores equipas concorrentes do SemEval, em cada ano, e às

três melhores equipas concorrentes do ASSIN, em todos os anos em que estas tarefas se realizaram e para os quais foi possível obter essa informação. Os dados encontram-se sumarizados nas tabelas comparativas 2.2, na qual são apresentados os melhores resultados de cada equipa; 2.3, na qual são apresentadas as técnicas de processamento utilizadas e os algoritmos de aprendizagem; e 2.4, na qual se mostram o tipo de características extraídas e as ferramentas utilizadas. A divisão horizontal nestas tabelas faz a separação entre as equipas que participaram no SemEval (em cima) e as equipas que participaram no ASSIN (em baixo). A dupla divisão separa o modelo estado-da-arte para o cálculo da SST em português dos modelos que participaram no ASSIN, que pela sua relevância é também apresentado nestas tabelas.

As equipas participantes nas tarefas mencionadas podiam apresentar mais do que um modelo na competição, pelo que a informação apresentada representa o leque de opções que optaram por utilizar. Isto não significa que as tenham utilizado todas no sistema que obteve o valor de desempenho apresentado. Algumas das tarefas englobavam a análise de diversos datasets, pelo que o valor de desempenho apresentado, nos casos em que isso se verifica, corresponde ao desempenho médio do melhor sistema desenvolvido pela equipa.

A avaliação dos modelos é tipicamente feita utilizando a correlação de Pearson (daqui em diante  $\rho$ , entre 0, nada correlacionados, e 1, totalmente correlacionados) entre os valores de similaridade calculados entre pares de frases de uma coleção pelo sistema,  $x$ , e aqueles anotados por júris humanos,  $y$ , atribuindo um valor de correlação entre os dois conjuntos, de acordo com a fórmula 2.1. Os valores de similaridade anotados variam entre 0, as duas frases são completamente diferentes, e 5, as duas frases são equivalentes e apresentam o mesmo significado, nas tarefas do SemEval, e entre 1, as duas frases são completamente diferentes, e 5, as duas frases apresentam o mesmo significado, na tarefa ASSIN. Na tabela 2.1 são apresentados, como exemplo, pares de frases extraídos do conjunto de dados de treino da tarefa ASSIN e o respetivo valor de similaridade atribuído por um júri humano.

$$\rho_{xy} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2 (y_i - \bar{y})^2}} \quad (2.1)$$

onde:

$n$  = tamanho dos conjuntos

$x_i, y_i$  = valores individuais de similaridade calculados,  $x_i$ , e anotados  $y_i$

$\bar{x}$  =  $\frac{1}{n} \sum_{i=1}^n x_i$ , média do conjunto  $x$ , análogo para  $\bar{y}$

Par de frases	Similaridade
t: Espero que corra tudo bem e que conquistem um título. h: Quem pensa que isto está a correr muito bem à esquerda, devia pensar duas vezes.	1.0
t: Bruno Alves faz 34 anos no dia 27 de Novembro. h: Bruno Alves chegará ao Europeu de 2016 com 34 anos e mais de 80 internacionalizações.	3.25
t: Mourinho é quem melhor pode gerir o problema, pois conhece a situação e a condição dos jogadores. h: Mourinho é a melhor pessoa para lidar com isto porque ele conhece o problema e as condições dos jogadores.	5.0

Tabela 2.1: Exemplos de frases de treino da tarefa ASSIN para português de Portugal.

O melhor resultado obtido na avaliação ASSIN foi da equipa INESC-ID (Fialho et al., 2016) que obteve um  $\rho = 0.73$ . Dado que os datasets da competição foram tornados públicos, algumas equipas continuaram a trabalhar no problema e a publicar novos resultados. Uma dessas equipas conseguiu desenvolver um modelo, ASAPP 2.0 (Alves et al., 2018), que é neste momento o estado-da-arte para o cálculo da SST em português, com um  $\rho$

= 0.75. Para o inglês não existe um único modelo que possa ser considerado o estado-da-arte, uma vez que todas as avaliações do SemEval já realizadas aplicaram variações nas tarefas para o cálculo da SST ao nível dos dados de treino e teste utilizados. Por esse motivo o máximo que se pode afirmar é que os melhores resultados para o cálculo da SST em inglês variam entre  $\rho = 0.619$  (Zarrella et al., 2015) e  $\rho = 0.852$  (Tian et al., 2017).

Nome da equipa	Desempenho
UKP (Bär et al., 2012)	$\rho = 0.824$
TakeLab (Šarić et al., 2012)	$\rho = 0.814$
UMBC (Han et al., 2013)	$\rho = 0.618$
ECNU (2014) (Marelli et al., 2014)	$\rho = 0.828$
MITRE (Zarrella et al., 2015)	$\rho = 0.619$
RTM-DCU (Biçici, 2015)	$\rho = 0.570$
DLS@CU (Sultan, Bethard e Sumner, 2015)	$\rho = 0.802$
ExB Themis (Hänig, Remus e De La Puente, 2015)	$\rho = 0.794$
ECNU (2017) (Tian et al., 2017)	$\rho = 0.852$
BIT (Wu et al., 2017)	$\rho = 0.840$
ASAPP 1.0 (Alves, Rodrigues e Oliveira, 2016)	$\rho = 0.680$
Solo Queue (Hartmann, 2016)	$\rho = 0.700$
INESC-ID (Fialho et al., 2016)	$\rho = 0.730$
ASAPP 2.0 (Alves et al., 2018)	$\rho = 0.750$

Tabela 2.2: Desempenho das equipas nas diferentes tarefas.

Apresentam-se de seguida, em maior detalhe, os métodos utilizados pelas equipas para o pré-processamento (ver secção 2.3.1) e processamento do texto para extração de características (ver secção 2.3.2), os algoritmos utilizados para treinar modelos para o cálculo da SST (ver secção 2.3.3), as características utilizadas para treinar esses modelos (ver secção 2.3.4) e, por fim, as ferramentas e recursos utilizados em todo este processo (ver secção 2.3.5). As tabelas comparativas 2.3 e 2.4 encontram-se no final das secções 2.3.3 e 2.3.5, respetivamente, e permitem identificar os métodos de processamento, características textuais, algoritmos de aprendizagem e ferramentas utilizadas por cada uma das equipas.

### 2.3.1 Métodos para o pré-processamento do texto

O pré-processamento do texto consiste num conjunto de técnicas destinadas ao tratamento do texto, antes deste ser utilizado para a extração de características e aprendizagem de modelos de similaridade. Estas técnicas são geralmente aplicadas antes da utilização dos métodos de processamento do texto para a extração de características. De seguida explicam-se algumas das técnicas utilizadas pelas equipas nas tarefas de avaliação de SST.

#### Tokenização

A tokenização consiste na divisão de uma frase ou texto nos átomos que os constituem, tais como palavras, números, pontuação, entre outros. Por exemplo, a frase:

"O José não tem visto séries."

é composta pelos seguintes átomos:

[O] [José] [não] [tem] [visto] [séries] [.]

## Normalização do texto

A Normalização do texto não representa uma técnica em específico, mas um conjunto de transformações que podem ser aplicadas ao texto para o modificar, tais como:

- Correção de erros de ortografia;
- Conversão das palavras para letra minúscula;
- Remoção de caracteres não-alfanuméricos;
- Expansão de abreviações (e.g. tv transformado em televisão);
- Substituição de contrações pela forma extensa (e.g n't e 'm substituídos por not e am, respetivamente, no caso do inglês).

### 2.3.2 Métodos de processamento para a extração de características

Os métodos de processamento do texto para extração de características permitem manipular os dados de forma a que destes possam ser extraídas características textuais para treinar os algoritmos para o cálculo da SST. Estas técnicas permitem não só aplicar pequenas alterações às palavras de forma a uniformizar o texto, por exemplo, através da conversão de todas as palavras para a respetiva entrada no dicionário, como também extrair informação relativa à função gramatical e análise sintática dos elementos constituintes da frase. O desenvolvimento de métodos para o pré-processamento e processamento do texto não representa um objetivo desta tese, pelo que irão ser utilizadas ferramentas já disponíveis para esse efeito, algumas das quais apresentadas na secção 2.3.5.

#### N-gramas

N-grama é uma sequência contínua de  $n$  elementos de uma determinada amostra textual. Estes elementos podem ser, por exemplo, palavras, caracteres ou sílabas. De acordo com o seu tamanho, os n-gramas podem ser designados por unigramas, quando compostos por apenas um elemento, bigramas, quando compostos por dois elementos, trigramas, quando compostos por três elementos, e assim sucessivamente. De seguida apresenta-se um exemplo no qual uma frase é transformada nos respetivos n-gramas de caracteres e palavras de tamanho 1, 2 e 3.

"O José não tem visto séries"

↓

**Unigramas de caracteres:** {O, J, o, s, é, n, ã, o, t, e, m, v, i, s, t, o, s, é, r, i, e, s}

**Bigramas de caracteres:** {Jo, os, sé, nã, ão, te, em, vi, is, st, to, sé, ér, ri, ie, es}

**Trigramas de caracteres:** {Jos, osé, não, tem, vis, ist, sto, sér, éri, rie, ies}

**Unigramas de palavras:** {(O), (José), (não), (tem), (visto), (séries)}

**Bigramas de palavras:** {(O, José), (José, não), (não, tem), (tem, visto), (visto, séries)}

**Trigramas de palavras:** {(O, José, não), (José, não, tem), (não, tem, visto), (tem, visto, séries)}



## Lematização

A Lematização consiste na obtenção da forma não-flexionada das palavras, com o objetivo de obter a palavra conforme a sua entrada no dicionário. Por exemplo, as seguintes palavras seriam transformadas como a seguir se demonstra:

gatos → gato

escrevendo → escrever

fomos → ir

## Stemming

Stemming é uma técnica que consiste na remoção da parte final das palavras/afixos. O resultado da sua utilização pode por vezes não ser uma palavra válida, no entanto, pode ser uma alternativa à lematização, por precisar de menos informação linguística. Alguns exemplos da sua utilização são:

quilométricos → quilométr

fomos → fom

irmos → irm

## Remoção de stopwords

Consiste na remoção de palavras muito frequentes mas que não acrescentam muito ao significado do texto. Aqui incluem-se normalmente palavras funcionais, como artigos, advérbios e preposições, entre outras.

## Reconhecimento de Entidades Mencionadas (EM)

Esta tarefa consiste na identificação de Entidades Mencionadas num pedaço textual. Estas consistem, por exemplo, em nomes de pessoas, locais, organizações e acontecimentos. A capitalização das palavras, por geralmente identificar nomes próprios, é importante para o reconhecimento destas entidades, pelo que, caso se pretenda converter o texto para letras minúsculas, esta técnica deve ser aplicada antes. Apresenta-se de seguida um exemplo da utilização desta técnica é:

O Cristiano Ronaldo<sub>Pessoa</sub> foi jogar para a Juventus<sub>Organização</sub> em Itália<sub>Localização</sub>.

## Part-of-speech (POS) tagging

Esta tarefa consiste na identificação da função gramatical dos elementos constituintes da frase com base na sua definição e no contexto em que aparecem. As *tags* correspondem às diferentes funções gramáticas que cada palavra pode ter, sendo algumas das mais comuns, Nome, Verbo, Adjetivo, Advérbio e Artigo. O contexto em que as palavras aparecem é importante na medida em que uma palavra pode, dependendo do seu contexto/posição na frase, apresentar diferentes funções gramaticais/tags. De seguida, apresenta-se um exemplo da utilização da técnica de *POS tagging* numa frase:

O<sub>Artigo</sub> José<sub>Nome</sub> não<sub>Advérbio</sub> fez<sub>Verbo</sub> as<sub>Artigo</sub> compras<sub>Nome</sub>.

## Análise de dependências

A análise de dependências é uma técnica que permite analisar a estrutura gramatical de uma frase e estabelecer relações de dependência entre os elementos que a constituem. A figura 2.1 apresenta a utilização desta técnica para uma frase em inglês.

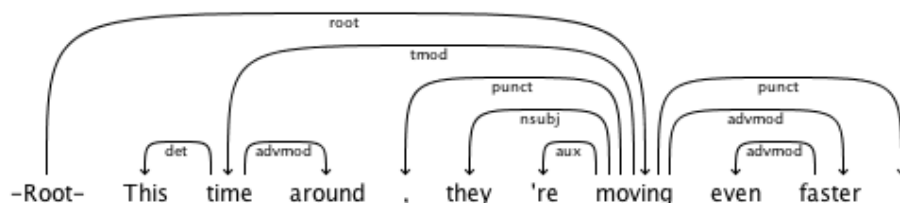


Figura 2.1: Exemplo de utilização da análise de dependências para uma frase em inglês.<sup>5</sup>

A análise de dependências da figura 2.1 é feita com base no manual de dependências anotadas de Stanford (De Marneffe e Manning, 2008). Apresenta-se, a seguir, a legenda da figura atrás referida com base nesse manual:

- **root** – palavra a partir da qual se estabelecem as relações gramaticais.
- **tmod** – modificador temporal.
- **det** – determinante.
- **advmod** – advérbio.
- **nsubj** – objeto nominal.
- **aux** – verbo auxiliar.

## Alinhamento de frases

Consiste no mapeamento de átomos lexicais entre frases, muito utilizado em tarefas que envolvam o cálculo de similaridade semântica entre frases de línguas diferentes. Na figura 2.2 demonstra-se o mapeamento entre as palavras de uma frase em inglês e as suas equivalentes na frase em francês.

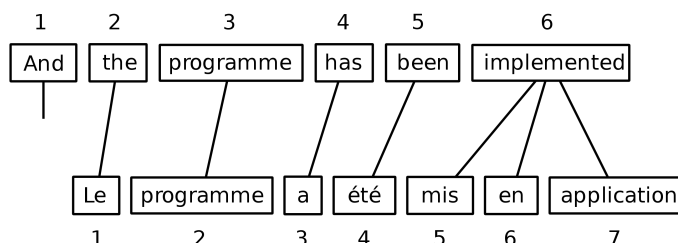


Figura 2.2: Exemplo de utilização do alinhamento de frases entre uma frase em inglês e francês.<sup>6</sup>

<sup>5</sup><https://nlp.stanford.edu/software/ndep.html>

<sup>6</sup>Redienss - Own work, <https://commons.wikimedia.org/w/index.php?curid=8978251>

### 2.3.3 Algoritmos de aprendizagem

A obtenção dos melhores resultados para o cálculo da similaridade semântica recorre normalmente à criação de modelos que têm por base algoritmos de aprendizagem treinados com um determinado corpus e que permitem a atribuição de um valor de similaridade entre frases. A aprendizagem dos modelos pode ser dividida em aprendizagem supervisionada, na qual um conjunto de dados de treino anotados é utilizado para treinar o algoritmo e gerar um modelo geral que consiga lidar corretamente com todos os dados de entrada possíveis, e aprendizagem não-supervisionada, na qual não são disponibilizados dados de treino anotados e o algoritmo procura identificar similaridades entre os dados de entrada de forma a categorizá-los.

Em aprendizagem computacional, as tarefas podem ser divididas em dois grupos principais: regressão e classificação. Em tarefas de classificação um modelo é treinado com o objetivo de categorizar os dados de entrada, com recurso às características dos mesmos, estabelecendo uma divisão em classes. Este tipo de modelos é utilizado em problemas cuja solução seja um valor discreto, por exemplo, a previsão do resultado de um jogo de futebol, cujo resultado pode ser apenas ganhar, perder ou empate. Os modelos de regressão são utilizados em problemas nos quais o resultado da previsão do sistema é de domínio contínuo. Podem ser utilizados em sistemas para estimar o desempenho financeiro de uma determinada empresa ou previsão do preço das casas no mercado imobiliário (Liu, Zhang e Wu, 2006).

No cálculo da similaridade semântica são maioritariamente utilizados modelos de regressão, uma vez que os valores de similaridade entre frases são de domínio contínuo. Estes modelos recebem como dados de entrada vetores com as características textuais calculadas para cada par de frases e atribuem-lhes, de acordo com a função de previsão calculada durante o treino do modelo, um valor de similaridade.

Os modelos podem depois ser avaliados através da correlação de Pearson.

De seguida apresentam-se os algoritmos utilizados pelas equipas participantes nas avaliações mencionadas na secção 2.3. Os algoritmos apresentados utilizados para desenvolver o nosso modelo SST encontram-se identificados por ●. Para além dos algoritmos de aprendizagem utilizados pelas equipas foram também utilizados quatro outros algoritmos (identificados por ★) seleccionados devido à similaridade com os algoritmos utilizados no ASAPP 2.0. O facto de serem similares e não os mesmos deve-se à impossibilidade de encontrar uma implementação em Python de alguns desses algoritmos.

#### Regressão Linear ●

É um algoritmo de aprendizagem computacional supervisionada através do qual se procura estimar um função do tipo  $y = mx + b$  capaz de representar os dados de treino do modelo, minimizando a distância entre o valor real desses mesmos dados e o valor da sua projecção na função. Passa, assim, a ser possível calcular o valor esperado para novos dados de entrada, com base na função calculada. Este método tem a vantagem de ser relativamente fácil de implementar em diversas linguagens de programação devido à sua baixa complexidade, é contudo propenso a *overfitting* e, como o próprio nome indica, não consegue estabelecer relações entre a variável dependente e independente para relações não-lineares.

## Máquinas de Vetores de Suporte (Cortes e Vapnik, 1995) •

As Máquinas de Vetores de Suporte (MVS, Support Vector Machines (SVM), em inglês) é um algoritmo de aprendizagem supervisionada utilizado em tarefas de classificação e regressão. Quando aplicado a problemas de classificação, este algoritmo permite efetuar a divisão dos dados em classes, com base nas suas características, através do cálculo de um hiperplano linear que garanta a máxima separação entre as mesmas. A máxima separação das classes é calculada com base na distância entre os designados *vetores de suporte*, que correspondem às instâncias que delimitam a margem de separação entre as classes. Alguns conjuntos de dados não permitem uma separação linear entre as classes, o que requer a utilização de funções designadas por *funções de kernel* que permitem fazer o mapeamento dos dados numa dimensão diferente da original que garanta que a divisão das classes possa ser feita através de um hiperplano linear. De seguida procede-se ao processo de mapeamento inverso, no qual os dados são novamente mapeados na sua dimensão original. A figura 2.3 ilustra a utilização de uma *função de kernel* para separar duas classes de dados (*azul e vermelha*), que seriam inseparáveis através de um hiperplano linear em duas dimensões (imagem da direita). Ao transformar os pontos originais para três dimensões passa a ser possível efetuar essa separação, como ilustra o hiperplano a verde, sendo por último necessário projetar os dados, juntamente com o plano de separação calculado, na sua dimensão original.

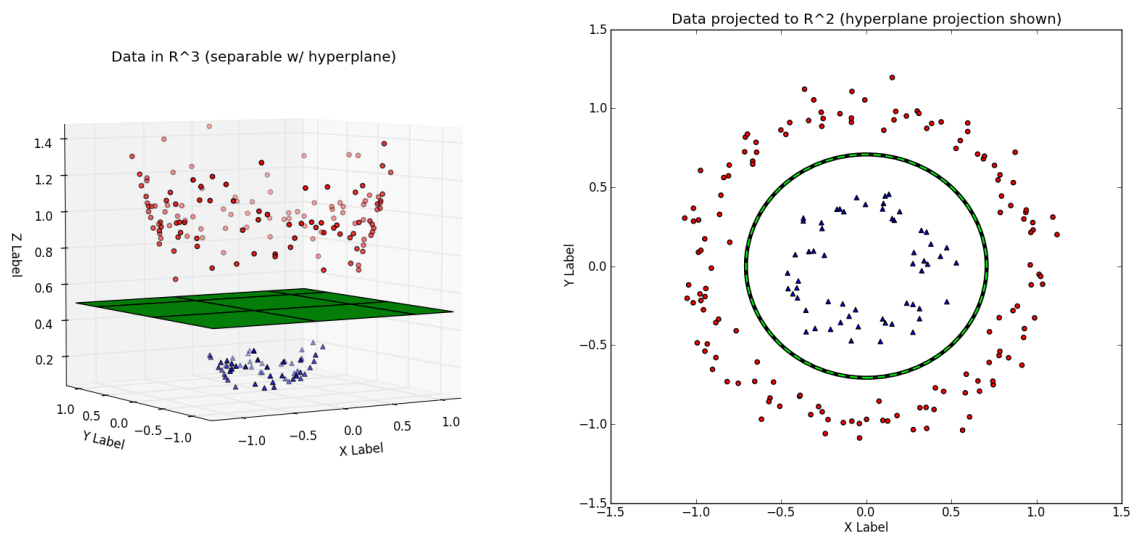


Figura 2.3: Exemplo de utilização de uma função de *kernel* numa MVS (Kim, 2013).

Relativamente aos problemas de regressão, os vetores de suporte são utilizados para delimitar a área máxima, designada por  $\epsilon$ -tubo, dentro da qual as instâncias dos dados de treino se podem encontrar sem que haja penalização do modelo. Tal como acontece em problemas de classificação, é possível aplicar *funções de kernel* caso não se consiga calcular a função de regressão na dimensão original dos dados. A figura 2.4 ilustra a utilização do algoritmo de SVR (*Support Vector Regressor*, em inglês), o qual recorre a sete vetores de suporte para definir o  $\epsilon$ -tubo e na qual duas instâncias são responsáveis por penalizar o desempenho do modelo.

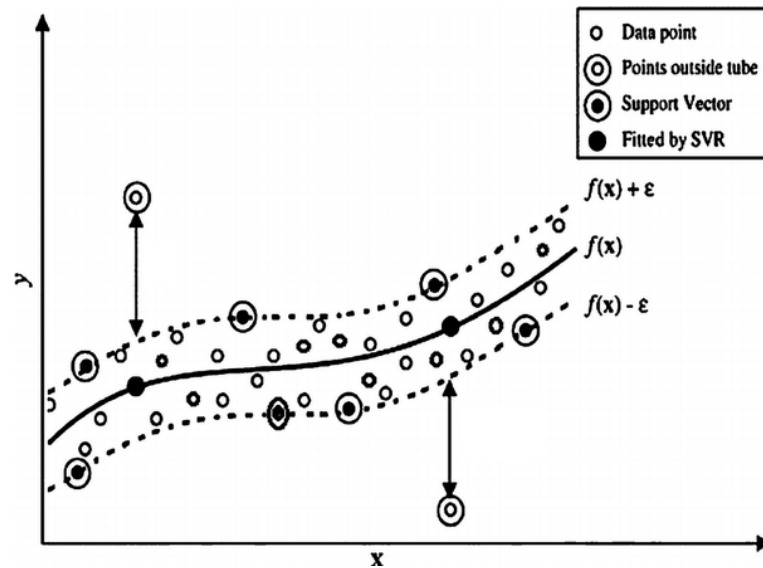


Figura 2.4: Exemplo de aplicação da MVS num problema de regressão.

### K-Vizinhos Mais Próximos (Altman, 1992)

O algoritmo *K-Vizinhos Mais Próximos* (em inglês, K-Nearest Neighbors) é frequentemente utilizado para a classificação de instâncias com base na proximidade das suas características, isto é, um novo elemento é classificado de acordo com a classificação previamente atribuída a outras instâncias cujas características sejam semelhantes. A proximidade entre as instâncias é calculada com base numa métrica, como, por exemplo, a distância Euclidiana (ver fórmula em 2.2).

Quando aplicado a problemas de regressão, procura-se estimar o valor a atribuir a uma nova instância com base nas instâncias de treino. Um dos métodos para calcular este valor é a média ponderada com base numa métrica de distância pré-definida entre os *K-Vizinhos mais próximos* e a nova instância.

$$\sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (2.2)$$

### Referential Translation Machine (Biçici e Way, 2014)

Referential Translation Machine (RTM) é um modelo computacional cujo objetivo é identificar mecanismos de tradução entre dois conjuntos de dados referentes ao mesmo domínio. Para esse efeito, recorre a um algoritmo designado por *Machine Translation Performance Prediction* (MTPP) (Biçici, Groves e Genabith, 2013) que através da extração de características textuais independentes da língua entre os dados de treino e os dados de teste atribui uma medida de qualidade entre a instância original e a instância traduzida. Dado que as características extraídas não dependem dos idiomas utilizados na tradução, a métrica de qualidade atribuída pelo MTPP a instâncias da mesma língua pode ser considerada um indicador da similaridade semântica entre o par.

### Alinhamento de Palavras/Pedaços Textuais

O Alinhamento de Palavras consiste num modelo para o cálculo da similaridade semântica entre frases, que utiliza a proporção de palavras alinhadas, de acordo com o seu significado, entre um par de frases como métrica de similaridade. Para identificar palavras semanticamente semelhantes, o modelo utiliza duas estratégias: a utilização de uma Base de Dados de Paráfrases (Ganitkevitch, Van Durme e Callison-Burch, 2013), como forma de encontrar palavras com significado próximo e a análise de dependências, com o objetivo de ter em conta o contexto em que essas mesmas palavras se inserem. As palavras são depois alinhadas com base na similaridade semântica e contextual existente entre o par de palavras de cada uma das frases. Para duas frases  $S^{(1)}$  e  $S^{(2)}$ , o valor de similaridade é calculado segundo a fórmula 2.3.

$$sst(S^{(1)}, S^{(2)}) = \frac{n_c^a(S^{(1)}) + n_c^a(S^{(2)})}{n_c(S^{(1)}) + n_c(S^{(2)})} \quad (2.3)$$

onde:

$$n_c(S^{(i)}) = \text{número de palavras em cada uma das frases, } S^{(i)}$$

$$n_c^a(S^{(i)}) = \text{número de palavras alinhadas em cada uma das frases, } S^{(i)}$$

### Regressão baseada em Processos Gaussianos (MacKay, 1998) •

A regressão baseada em Processos Gaussianos é um algoritmo que calcula a distribuição probabilística dos parâmetros de todas as funções admissíveis para descrever os dados de treino. Estes parâmetros são ajustados, ao longo do processo de treino, à medida que vão sendo consideradas novas instâncias dos dados. Por fim, de forma a obter uma única função capaz de descrever os dados e permitir fazer previsões, é calculada a média entre todas as funções.

### Regressão através de Árvores de Decisão ★

Uma Árvore de Decisão é um algoritmo de aprendizagem supervisionada que pode ser aplicado a problemas de classificação e regressão. Apresenta uma estrutura em forma de árvore, na qual os nós representam as características das instâncias de treino do modelo, as ligações entre os nós os pesos relativos aos valores ou intervalos de valores que o nó de origem pode tomar e que representam o processo de decisão, e as folhas que, no caso de problemas de classificação, correspondem às classes das instâncias, e no caso de problemas de regressão, correspondem à média dos valores esperados pelas instâncias. A construção de uma árvore de regressão utiliza a variância de cada uma das características como método para definir qual deve ser a ordem pela qual estas são usadas. Características com menor variância são utilizadas nos primeiros nós da árvore. Como critério de paragem da construção da árvore pode ser utilizado o número mínimo de instâncias que a decisão a tomar abrange. Uma das grandes vantagens da utilização deste algoritmo é a facilidade com que os seus resultados conseguem ser interpretados devido à forma como estes são apresentados, contudo é um algoritmo propenso a overfitting e instável, uma vez que pequenas alterações nos dados de treino podem resultar numa alteração profunda da estrutura da árvore. Para mitigar este último problema pode-se recorrer à construção de várias árvores de decisão através de técnicas como *Bagging* e *Boosting*, as quais se explicam com maior detalhe de seguida.

## Bagging e Boosting

Uma abordagem utilizada em aprendizagem computacional para tentar melhorar o resultado obtido por um determinado modelo construído com base num único algoritmo de aprendizagem consiste na construção de um modelo com base num grupo de diferentes algoritmos ou modelos do mesmo algoritmo, também designado por *ensemble*. Este combina o resultado dos diversos algoritmos ou modelos do mesmo algoritmo que o constituem, de forma a que o resultado apresentado pelo conjunto seja apenas um. Os *ensembles* podem ser aplicados quer a problemas de classificação, quer de regressão. Um caso particular deste mecanismo diz respeito a métodos que apenas utilizam modelos do mesmo algoritmo para construir o *ensemble*. Estes encontram-se divididos em dois principais grupos, *Bagging* e *Boosting*, de acordo com a arquitetura utilizada por cada um deles. Esta diverge quer na forma como o resultado final é calculado, neste caso aplicado a problemas de regressão, quer na forma como os diversos modelos são produzidos. No caso do *Bagging*, os modelos são criados independentemente e treinados com recurso a uma amostragem aleatória dos dados de treino, que é alterada através da substituição de amostras a cada treino de um novo modelo. Os resultados são depois combinados através da média. No caso do *Boosting*, os modelos são criados iterativamente e de forma a especializarem-se. Isto é, sempre que um novo modelo é criado, este explora os dados de treino nos quais o anterior teve piores resultados. Os resultados são no fim combinados através da média ponderada de acordo com o desempenho de cada modelo. Na etapa de desenvolvimento dos modelos para o cálculo da SST foi utilizada uma implementação do algoritmo de *Bagging Regression* ★, tal como descrito acima, baseado em árvores de decisão. Foram também utilizados dois algoritmos de *Boosting*: o *AdaBoost Regression (Freund e Schapire, 1997)* ★ e o *Gradient Boosting (Friedman, 2001)* ●. A principal diferença entre estes dois algoritmos está no facto de o *AdaBoost* atribuir pesos às instâncias de treino, que são atualizados sempre que um novo modelo é criado. Desta forma é colocada uma maior importância sobre instâncias cujo resultado é calculado erradamente, visto ser-lhes atribuído um peso maior. Os modelos são depois combinados de acordo com o seu desempenho. No que diz respeito ao *Gradient Boosting*, sempre que um novo modelo é treinado apenas as instâncias cujo resultado foi mal calculado são utilizadas, com objetivo de que o modelo seguinte seja utilizado para diminuir o erro do modelo anterior. A contribuição de cada modelo para o modelo antecedente é calculada de forma a minimizar o erro deste último, através de um processo de *Gradient Descent*. Apresentam-se de seguida alguns algoritmos que utilizam uma destas duas abordagens:

### Random Forest (Ho, 1995) ●

*Random Forest* (RF) é um algoritmo de aprendizagem supervisionada utilizado em tarefas de classificação e regressão, que recorre à criação de diversas árvores de decisão com base no mesmo conjunto de dados de treino e no resultado obtido por essas mesmas árvores para atribuir uma instância de teste a uma determinada classe, no caso de uma tarefa de classificação, ou calcular o valor previsto para essa instância, no caso de uma tarefa de regressão. Este algoritmo funciona de forma muito semelhante ao de *Bagging Regression*, sendo que a principal diferença entre ambos está no facto de o *Random Forest* utilizar apenas um subconjunto aleatório de características sempre que é construída uma nova árvore de decisão, ao passo que o primeiro utiliza sempre todas as características. A imagem 2.5 ilustra a arquitetura de um modelo desenvolvido com base no *Random Forest*, no qual os resultados de  $n$  árvores de decisão são utilizados para calcular um único resultado,  $k$ .

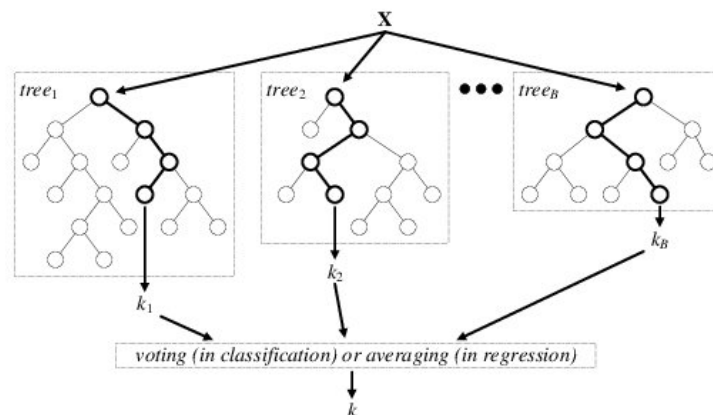


Figura 2.5: Arquitetura de um modelo baseado em *Random Forest*. **Fonte:** (Verikas et al., 2016)

### Random Subspace (Barandiaran, 1998)

*Random Subspace* é um algoritmo de *Bagging* cuja particularidade é utilizar apenas um subconjunto aleatório de características sempre que é construída uma nova árvore de decisão, ao mesmo tempo que utiliza todas as instâncias dos dados de treino. No algoritmo *Random Forest* quer as instâncias dos dados de treino, quer as características dos mesmos sofrem um processo de seleção aleatória, ao passo que no algoritmo de *Bagging* base, apenas as instâncias são submetidas a este processo.

### M5Rules (Holmes, Hall e Prank, 1999)

*M5Rules* é um algoritmo de *Boosting* aplicado principalmente a problemas de regressão, que se baseia na construção de árvores de decisão com base no algoritmo *M5* (Quinlan, 1992). A particularidade do algoritmo *M5* está no facto das suas folhas serem modelos de regressão linear calculados com base nas instâncias que fazem parte da sub-árvore que dá origem à folha, e não classes ou a média dos valores esperados para aquele intervalo de instâncias. A construção da árvore é efetuada com base no cálculo do desvio padrão dos valores esperados das instâncias que ainda não foram abrangidas por nenhuma regra. Caso o desvio padrão seja baixo, ou o número de instâncias seja reduzido é construída uma folha da árvore. O processo iterativo de construção de árvores do *M5Rules* descarta as instâncias abrangidas pela ramo da árvore atual com melhor desempenho sempre que é construída uma nova árvore.



Nome da equipa	Tokenização	POS tagging	Lematização	Reconhecimento de Entidades	Remoção de Stopwords	Stemming	Análise de dependências	Alinhamento de frases	Normalização do texto	Máquinas de Vetores de Suporte (MVS)	Random Forest (RF)	Regressão Linear	K-Vizinhos Mais Próximos	Referential Translation Machine (RTM)	Alinhamento de palavras/pedaços textuais	Gradient Boosting (GB)	M5Rules	Random Subspace	Regressão baseada em Processos Gaussianos
UKP	X		X	X								X							
TakeLab	X	X			X				X	X									
UMBC	X	X			X				X	X									
ECNU (2014)										X	X		X						
MITRE									X						X				
RTM-DCU														X					
DLS@CU															X				
ExB Themis	X	X	X		X				X	X									
ECNU (2017)	X	X	X		X		X			X	X					X			
BIT	X	X	X	X				X	X			X							
ASAPP 1.0	X	X	X	X														X	X
Solo Queue							X					X							
INESC-ID	X			X		X			X	X									
ASAPP 2.0	X	X	X	X	X	X	X										X	X	X

Tabela 2.3: Pré-processamento e Algoritmos de aprendizagem utilizados pelas equipas nas diferentes tarefas.

### 2.3.4 Caraterísticas textuais

Nesta secção são abordadas as caraterísticas textuais utilizadas pelas equipas participantes nas tarefas de SST na criação dos seus modelos. Estas caraterísticas podem ser divididas em quatro categorias principais: lexicais, sintáticas, semânticas e distribucionais, tal como se pode verificar na tabela 2.4 onde é feito um resumo desta secção.

#### Caraterísticas lexicais

As caraterísticas lexicais exploram as especificidades dos elementos que compõem uma frase ao nível das palavras. Alguns exemplos deste tipo de caraterísticas são as métricas de similaridade de conjuntos que podem ser calculadas utilizando a distância de Jaccard, Sobreposição, Coeficiente de Dice e Similaridade através do cosseno (ver fórmulas 2.4, 2.6, 2.5, 2.7) entre o número de átomos, n-gramas, lemas e stems em comum, e palavras negativas e a sua diferença absoluta entre um par de frases (T e H). Um outro tipo de abordagem consiste no cálculo da sub-sequência comum mais longa.

$$Jaccard(T, H) = \frac{|T \cap H|}{|T \cup H|} \quad (2.4) \quad Dice(T, H) = \frac{|T \cap H|}{|T| + |H|} \quad (2.6)$$

$$Overlap(T, H) = \frac{|T \cap H|}{|\min(T, H)|} \quad (2.5) \quad Cos(T, H) = \frac{|T \cap H|}{\sqrt{|T|} \sqrt{|H|}} \quad (2.7)$$

#### Caraterísticas sintáticas

As caraterísticas sintáticas dizem respeito à análise da frase do ponto de vista da função que as palavras que a constituem desempenham, algo que pode ser obtido recorrendo a um POS tagger. Algumas caraterísticas que se podem extrair dizem respeito ao número de nomes, verbos, advérbios e preposições, entre outros, em comum entre um par de frases, e à sua diferença absoluta. Pode também ser aplicada uma métrica de distância aos triplos que representam as dependências sintáticas entre um par de frases.

#### Caraterísticas semânticas

As caraterísticas semânticas exploram o significado dos elementos que compõem a frase. Algumas das métricas que podem ser utilizadas para a obtenção deste tipo de caraterísticas utilizam o cálculo da diferença absoluta dos diferentes tipos de Entidades Mencionadas entre pares de frases e/ou a expansão do vocabulário através da utilização de Bases de Conhecimento Lexical (BCL) e/ou Thesaurus, explorando assim relações entre palavras, tais como a sinonímia – palavras que apresentam significado semelhante (e.g. preto e negro), antonímia – palavras que apresentam significado oposto (e.g. quente e frio), hiperonímia – relação semântica na qual uma palavra assume superioridade hierárquica devido à sua maior abrangência de sentido (e.g. fruta é hiperónimo de maçã, banana ou laranja), hiponímia – corresponde à relação hierárquica inversa da hiperonímia (e.g. cão, gato ou elefante são hipónimos de animal), e meronímia – relação semântica entre palavras, na qual uma indica a parte (merónimo) relativamente à outra, que indica o todo (holónimo) (e.g. dedo é um merónimo de pé), que podem ser utilizadas para calcular o coeficiente de cada tipo de relação existente entre as palavras de cada uma das frases do par.

## Caraterísticas distribucionais

As caraterísticas distribucionais correspondem a uma abordagem assente na utilização de um conjunto de técnicas para a representação do significado das palavras tendo em conta o contexto no qual estas são utilizadas. O estudo deste tipo de relações entre palavras surgiu na década de 50 através do trabalho de alguns linguistas (Joos, 1950; Harris, 1954; Firth, 1957), que verificaram que palavras com significados relacionados tendem a aparecer em contextos semelhantes na frase, isto é, próximo do mesmo conjunto de palavras. A representação do significado das palavras em determinado documento é feita com recurso a vetores numéricos que representam a sua distribuição no texto.

A similaridade entre os vetores pode ser calculada com recurso ao produto vetorial normalizado (ver fórmula 2.8), também designado pelo cosseno do ângulo entre os vetores. Valores do cosseno próximos de 1 indicam uma grande similaridade entre os vetores, ao passo que valores de cosseno próximos de 0 indicam uma grande dissimilaridade entre os mesmos. A normalização dos vetores é utilizada como forma de não beneficiar vetores mais longos.

$$\text{cosine}(\vec{v}, \vec{w}) = \frac{\vec{v} \cdot \vec{w}}{|\vec{v}| |\vec{w}|} \quad (2.8)$$

De seguida serão abordados com maior detalhe os algoritmos mais utilizados para a criação destes vetores.

## Term Frequency-Inverse Document Frequency (TF-IDF)

O **algoritmo TF-IDF** é utilizado na análise da informação de documentos e surgiu como forma de resolver o problema resultante de utilizar apenas o número de ocorrências de um termo no texto (TF), uma vez que este não tinha em consideração a existência de palavras funcionais, tais como, artigos e preposições, que apresentavam uma elevada frequência, mas que são pouco discriminadoras no que diz respeito ao conteúdo. Este algoritmo resulta do produto de dois termos:

1. O primeiro representa a frequência de um termo,  $t$ , num documento,  $d$ . O cálculo desta frequência é feito de acordo com a seguinte fórmula:

$$tf_{t,d} = \begin{cases} 1 + \log_{10} \text{count}(t, d) & \text{if } \text{count}(t, d) > 0 \\ 0 & \text{otherwise} \end{cases} \quad (2.9)$$

O  $\log_{10}$  é utilizado para balancear a importância atribuída a uma palavra que apareça muitas vezes no texto, visto esta não ser muito relevante para o significado da frase ou documento, devido à sua frequência elevada.

2. O segundo termo (IDF) é utilizado como forma de atribuir um maior peso a palavras que ocorram apenas em alguns documentos, visto estas serem mais discriminadoras. A fórmula para o cálculo deste termo é a seguinte:

$$idf_t = \log_{10} \left( \frac{N}{df_t} \right) \quad (2.10)$$

Onde  $N$  representa o número total de documentos na coleção e  $df_t$  o número de documentos no qual o termo  $t$  ocorre.

O peso de um termo,  $t$ , num determinado documento resulta assim da multiplicação da sua frequência com o seu idf:

$$w_{t,d} = t f_{t,d} \times idf_t \quad (2.11)$$

O principal problema deste tipo de vetores é o facto de serem esparsos, visto que cada palavra é representada por um vetor com a dimensão do número de documentos utilizados para calcular os valores de idf, independentemente de a palavra existir ou não nesses documentos. Os algoritmos apresentados de seguida vieram solucionar este problema através de abordagens que permitem a criação de vetores densos.

### Word2vec e Doc2vec

O Word2vec (Mikolov et al., 2013a) é um método utilizado para o cálculo da representação semântica de palavras através de vetores densos, aprendidos com base no processamento de grandes quantidades de texto. É capaz de encapsular relações entre as palavras, tais como, sinonímia ou antonímia e analogias (Mikolov et al., 2013b).

Os vetores são calculados através de uma rede neuronal simples, geralmente com 1 ou 2 camadas para além da camada de entrada e saída, podendo ser utilizados para o efeito dois algoritmos distintos: Continuous Bag-of-Words (CBOW) e Skip-Gram.

A principal diferença entre estes dois algoritmos consiste na sua abordagem, uma vez que o primeiro treina o modelo de forma a calcular a probabilidade de uma palavra aparecer na proximidade de um conjunto de outras palavras, enquanto o segundo calcula a probabilidade de um conjunto de palavras aparecerem na proximidade da palavra de entrada. O contexto é definido através de uma “janela” de palavras em torno da palavra para a qual se quer calcular o vetor (ver figura 2.6), sendo este parâmetro responsável pelo quão representativos os vetores são, uma vez que a utilização de uma janela mais pequena tende a abranger palavras que normalmente aparecem mais vezes juntas.

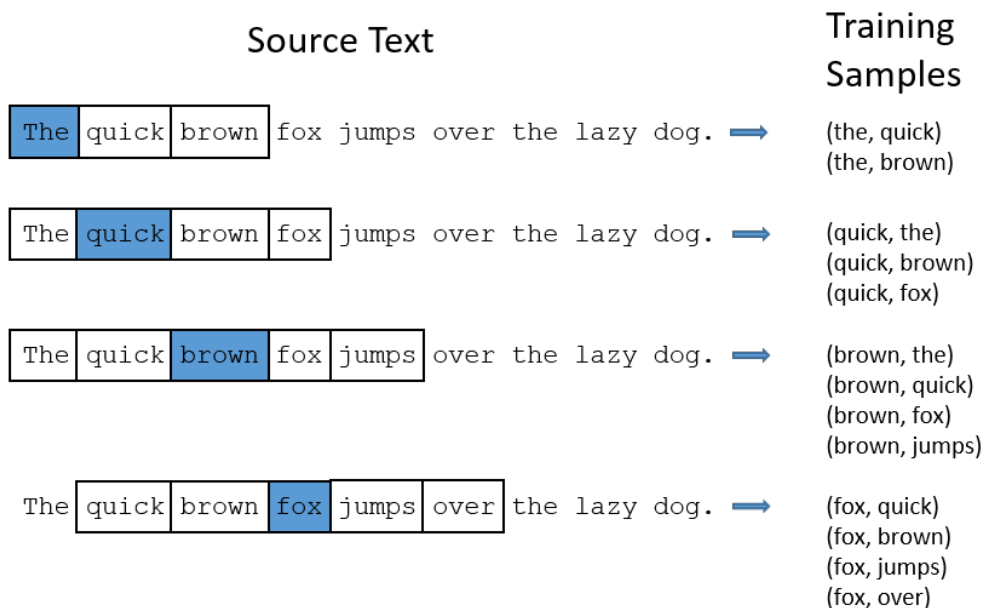


Figura 2.6: Exemplo da utilização de uma janela de tamanho 2 para a obtenção de exemplos de treino<sup>7</sup>.

Para que possam ser utilizadas pela rede neuronal, cada palavra é representada através de um vetor de dimensão igual ao número de palavras diferentes dos dados de treino. Dado que cada posição do vetor corresponde a uma palavra diferente, é possível identificar cada palavra colocando um 1 na posição correspondente e deixando todas as outras posições a 0. Este tipo de vetores são designados por *one-hot encoded vectors*, em inglês.

No final, os vetores das palavras correspondem aos vetores dos pesos da rede neuronal que foram sendo ajustados durante o treino, e não aos vetores resultantes na camada de saída, o que faz com que estes sejam substancialmente mais densos, tendo em conta que têm o tamanho correspondente ao número de pesos existentes na rede neuronal. O tamanho destes vetores deve ser ajustado de acordo com o tamanho dos dados de treino, de forma a minimizar a quantidade de ruído que os mesmos possam ter. De acordo com os autores do modelo (Mikolov et al., 2013a) este consegue obter um bom desempenho no cálculo dos vetores quando o tamanho dos mesmos é de 300 dimensões. Na figura 2.7 é apresentada uma possível arquitetura para uma rede neuronal utilizada para treinar um modelo baseado em Skip-Gram.

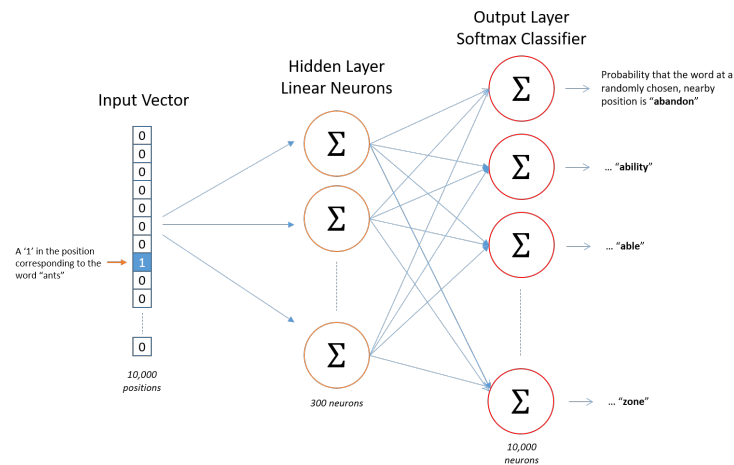


Figura 2.7: Exemplo da arquitetura de uma rede neuronal utilizada para treinar um modelo baseado em Skip-Gram<sup>8</sup>.

O Doc2vec (Le e Mikolov, 2014) é um método desenvolvido com base no Word2vec que permite calcular a representação vetorial de uma frase ou documento e não apenas de palavras individuais. Durante o treino do modelo, é calculado, para além dos vetores individuais de cada palavra, um vetor, designado por Paragraph ID, que encapsula o contexto dos dados de treino.

Uma vantagem da utilização deste tipo de métodos, Word2vec ou Doc2vec, é a possibilidade de estes poderem ser utilizados sem ser necessário fazer qualquer tipo de pré-processamento do texto, uma vez que os vetores são aprendidos apenas com base na ocorrência das palavras e suas vizinhas em grandes coleções de texto.

### FastText

O FastText (Bojanowski et al., 2017; Joulin et al., 2016) consiste numa variação do método Word2vec. A principal diferença está na forma como o modelo é treinado, uma vez que o FastText utiliza conjuntos de caracteres das palavras cujos vetores se pretendem obter para treinar o modelo, ao passo que o Word2vec utiliza as palavras como um todo.

<sup>7</sup><http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/>

<sup>8</sup><http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/>

Uma das vantagens da utilização do FastText está na sua capacidade em lidar com palavras fora do vocabulário de treino, precisamente por utilizar n-gramas de caracteres durante a criação dos vetores. É também capaz de gerar melhores vetores para palavras raras, uma vez que estas palavras podem partilhar conjuntos de caracteres com outras palavras que aparecerão mais vezes durante o processo de treino, o que não acontece no Word2vec.

### 2.3.5 Ferramentas utilizadas

Nesta secção são apresentados os diferentes recursos e ferramentas utilizadas pelas equipas participantes em tarefas de SST, quer ao nível da obtenção e expansão dos dados para a criação dos modelos de similaridade, por exemplo, através de wikis e bases conhecimento lexical, quer ao nível da utilização de bibliotecas de programação desenvolvidas para facilitar a implementação de soluções de PLN, tais como, o *NLTK* e o *spaCy*.

#### Recursos

Dizem respeito a um conjunto de ferramentas estáticas utilizadas para a obtenção e expansão dos dados e que permitem quer a extração de novas características textuais, quer a utilização de um maior número de instâncias para treinar os algoritmos de aprendizagem. Apresentam-se de seguida alguns exemplos deste tipo de ferramentas:

- **Bases de Conhecimento Lexical**

As Bases de Conhecimento Lexical são bases de dados que podem ser vistas como dicionários otimizados para a utilização por programas de computador. Elas podem agrupar palavras sinónimas em conjunto designados por synsets e estabelecer relações semânticas entre as mesmas. O recurso deste tipo mais amplamente usado para o inglês é o Princeton WordNet<sup>9</sup>, enquanto que para o português existem várias alternativas (Gonçalo Oliveira, 2018a).

Um *Thesaurus* é um tipo de base de conhecimento lexical cujo objetivo é agregar palavras com significado semelhante e apresentá-las num formato semelhante a um dicionário.

- **Wikis**

Wikis são coleções de documentos em hipertexto sobre diversos domínios criadas de forma colaborativa. Estes documentos são utilizados no PLN para aumentar a quantidade de dados disponíveis para treinar modelos de similaridade, extrair bases de conhecimento e/ou obter listas de entidades, entre outros. Um exemplo de maior sucesso deste tipo de coleções é a Wikipédia.

- **Corpus externo**

O corpus externo diz respeito a qualquer conjunto de dados de texto que tenham sido usados pelas equipas para treinar modelos para o cálculo da similaridade semântica, que não tenham sido fornecidos diretamente pelos organizadores da competição. Estes corpus podem ser textos obtidos através da Wikipédia, Reddit, Facebook ou Twitter, por exemplo.

---

<sup>9</sup><https://wordnet.princeton.edu>

## Toolkits

Toolkits são conjuntos de ferramentas que permitem fazer o processamento dos dados e a criação e treino de modelos para o cálculo da similaridade semântica textual. Estas podem ser divididas em **Cadeias de ferramentas para o PLN** e **Ferramentas para a aprendizagem automática**.

### Cadeias de ferramentas para o PLN

São ferramentas desenvolvidas para lidar especificamente com problemas relacionados com o PLN, incluindo um conjunto de técnicas para pré-processamento do texto e alguns algoritmos para a criação de modelos de cálculo de similaridade.

- ***NLTK***<sup>10</sup>

O *NLTK* (Bird, Klein e Loper, 2009) é uma plataforma em Python que serve de suporte ao desenvolvimento de programas que lidem com PLN. As ferramentas disponíveis incluem o acesso a Bases de Conhecimento Lexical, tais como, wordnets, e ainda classificadores, taggers, tokenizadores, entre outros.

- ***DKPro3***<sup>11</sup>

É uma coleção de ferramentas para o PLN construídas com base na framework Apache UIMA. Inclui diversos métodos que permitem lidar com o pré-processamento dos dados, tais como, taggers, tokenizadores e identificadores de entidades, bem como algoritmos para o cálculo da similaridade entre frases.

- ***Stanford CoreNLP***<sup>12</sup>

É um conjunto de ferramentas desenvolvidas em Java para o PLN. Inclui diversas ferramentas que podem ser utilizadas em tarefas, tais como, reconhecimento de entidades, POS tagging e análise de sentimentos.

- ***Apache OpenNLP toolkit***<sup>13</sup>

O Apache OpenNLP é uma biblioteca em Java que facilita a utilização de técnicas de aprendizagem computacional no PLN. Alguns dos métodos disponíveis permitem efetuar tarefas, tais como, tokenização, POS tagging e reconhecimento de entidades, entre outras.

- ***NLPPort***

O *NLPPort* (Rodrigues, Gonçalo Oliveira e Gomes, 2018) consiste num conjunto de ferramentas, que podem ser utilizadas em formato de pipeline, desenvolvidas em Java para o PLN em português.

- ***spaCy toolkit***<sup>14</sup>

O *spaCy* é uma biblioteca desenvolvida para Python composta por um conjunto de ferramentas para o PLN, incluindo métodos para o pré-processamento do texto, extração de características textuais e algoritmos de aprendizagem. Estas ferramentas estão disponíveis em diversas línguas, incluindo inglês e português.

<sup>10</sup><http://www.nltk.org>

<sup>11</sup><https://dkpro.github.io>

<sup>12</sup><https://stanfordnlp.github.io/CoreNLP/>

<sup>13</sup><https://opennlp.apache.org>

<sup>14</sup><https://spacy.io>

## Ferramentas para a aprendizagem automática

São um conjunto de ferramentas mais abrangente que disponibilizam diversos métodos para a criação de modelos de aprendizagem computacional.

- ***Weka toolkit***<sup>15</sup>

O *Weka toolkit* (Witten et al., 2016) é uma coleção em Java de algoritmos de aprendizagem computacional orientada para tarefas de exploração e análise de dados. Esta coleção inclui ferramentas para manipulação e visualização de dados, tarefas de classificação, regressão e clustering, entre outras, que podem ser utilizadas através de uma interface gráfica.

- ***LibSVM***<sup>16</sup>

O *LibSVM* (Chang e Lin, 2011) é um conjunto de algoritmos para classificação, regressão e estimação de distribuição baseados em MVS.

- ***scikit-learn toolkit***<sup>17</sup>

O *scikit-learn* (Pedregosa et al., 2011) é uma biblioteca desenvolvida para o Python que disponibiliza um conjunto de ferramentas e algoritmos que podem ser utilizados em tarefas de pré-processamento, classificação e regressão de dados, por exemplo.

---

<sup>15</sup><https://www.cs.waikato.ac.nz/ml/weka/>

<sup>16</sup><https://www.csie.ntu.edu.tw/~cjlin/libsvm/>

<sup>17</sup><https://scikit-learn.org/stable/>



Nome da equipa	Caraterísticas Lexicais	Caraterísticas Sintáticas	Caraterísticas Semânticas	Caraterísticas Distribucionais	Bases de Conhecimento Lexical	Wikis	Thesaurus	Weka toolkit	NLTK	DKPro3	LibSVM	Corpus externo	Bases de dados de paráfrases (PPDB)	Stanford Core NLP	scikit-learn toolkit	Apache OpenNLP toolkit	NLPPort	spaCy toolkit
UKP	X		X		X	X	X	X		X								
TakeLab	X	X	X	X	X				X		X						X	
UMBC	X			X					X		X	X						
ECNU (2014)		X	X	X	X							X						
MITRE		X	X	X								X						
RTM-DCU	X	X																
DLS@CU				X									X					
ExB Themis	X			X	X						X			X		X		
ECNU (2017)	X	X		X										X	X			
BIT				X	X				X			X						
ASAPP 1.0	X	X	X		X	X	X	X									X	
Solo Queue				X		X	X											
INESC-ID	X		X	X								X			X			
ASAPP 2.0	X	X	X	X	X	X	X	X									X	X

Tabela 2.4: Caraterísticas textuais extraídas e ferramentas utilizadas pelas equipas nas diferentes tarefas.

### 2.3.6 Visão Integradora

Com base na análise do estado-da-arte efetuada ao longo deste capítulo foi possível perceber de que forma o desenvolvimento deste projeto de tese deveria ser abordado.

Ao longo dos próximos capítulos irão ser exploradas as etapas de desenvolvimento de um agente de diálogo que capaz de responder a perguntas do domínio do BDE e identificar perguntas fora-do-domínio, de forma a utilizar um corpus de legendas de filmes para colmatar a sua falta de conhecimento. Isto permitirá fazer com que o diálogo do agente não sofra interrupções e seja o mais natural possível. Tendo em conta isto, pode-se concluir que o agente terá de seguir uma das abordagens utilizadas para desenvolvimento de chat-bots, visto que, apesar de ser desenvolvido para atuar num domínio específico, uma das características que se pretende que o mesmo tenha é a capacidade de manter uma conversa. Para efetuar o mapeamento entre a pergunta do utilizador e a resposta do agente irá ser utilizado um modelo para o cálculo da SST. Este modelo será utilizado para calcular a similaridade da pergunta do utilizador com as perguntas do BDE e a resposta àquela que tiver maior similaridade será a devolvida pelo agente. A opção por um modelo para o cálculo da SST foi feita com base na natureza dos dados do BDE. Estes irão constituir a base de conhecimento do agente e encontram-se estruturados num formato Pergunta-Resposta acessível através de documentos de texto. Por esse motivo o agente será baseado num sistema de Recuperação de Informação, dado que não é necessário treinar um sistema *SEQ2SEQ* para a transformação da pergunta numa resposta, visto estas estarem à partida disponíveis, e ser impraticável escrever regras capazes de cobrir a imprevisibilidade de interações do utilizador, tanto dentro como fora do domínio.

Dado o bom desempenho dos modelos criados pelas equipas para o cálculo da SST, os modelos desenvolvidos serão baseados no treino de algoritmos de regressão com base na extração do maior número possível de características textuais entre pares de frases. Contudo, será efetuada uma análise das características mais relevantes para a SST.

O processo de desenvolvimento dos modelos, desde a extração de características ao treino e avaliação dos mesmos, será apresentado no Capítulo 3. No Capítulo 4 será apresentada a construção do agente de diálogo e posterior avaliação do seu desempenho.

## Capítulo 3

# Cálculo da Similaridade Semântica Textual em Português

Tendo em conta a forte componente experimental associada ao desenvolvimento de modelos para o cálculo da SST, apresentam-se neste capítulo todos os passos desse processo, bem como a avaliação e posterior seleção do modelo a integrar num agente de diálogo.

O Python foi a linguagem de programação escolhida para o desenvolvimento quer dos modelos para o cálculo da SST descritos neste capítulo, quer do agente de diálogo (ver secção 4), uma vez que esta linguagem permite uma implementação e atualização relativamente rápida, quando comparada com outras linguagens de programação, destes modelos, uma vez que possui um vasto conjunto de bibliotecas externas para a aprendizagem computacional e o PLN.

### 3.1 Caraterísticas aplicadas aos modelos

Os modelos desenvolvidos para o cálculo da SST têm por base a aprendizagem supervisionada de diferentes algoritmos de regressão. Para tal, foi necessário realizar a extração de um conjunto de caraterísticas capazes de representar a relação de proximidade entre o significado de pedaços textuais. A metodologia utilizada pressupôs, tendo em conta o método escolhido para a avaliação do desempenho dos modelos, a extração de caraterísticas entre pares de frases. Estas caraterísticas permitem que os algoritmos de aprendizagem desenvolvam, com base num conjunto de dados de treino, um modelo de regressão capaz, não só de representar a similaridade semântica entre esses mesmos dados, mas também geral o suficiente para atribuir corretamente um valor de similaridade a novos conjuntos de dados. Os dados utilizados para treinar e testar os modelos foram disponibilizados no âmbito da avaliação ASSIN e são compostos por quatro datasets: um dataset de treino e um de teste em português de Portugal, e um dataset de treino e um de teste em português do Brasil. Como forma de aumentar o conjunto de dados de treino, os datasets de ambas as variantes da língua portuguesa foram utilizados em conjunto, visto que, de acordo com outros trabalhos aplicados à tarefa ASSIN (Alves et al., 2018) isto permitiu obter resultados iguais ou superiores nos dados de teste.

Um conjunto de bibliotecas externas foi utilizado para efetuar o pré-processamento do texto antes da extração de caraterísticas. Alguns métodos destas bibliotecas foram aplicados a todo o corpus utilizado, quer na etapa de treino, quer de teste dos modelos,

nomeadamente: *gensim*<sup>1</sup> (Řehůřek e Sojka, 2010), para a remoção de todos os caracteres não-alfanuméricos, de controlo e múltiplos espaços em branco; *NLTK*, para a remoção de stopwords; *num2words* para a conversão de números para a sua representação por extenso. O texto foi por último convertido para letra minúscula. Algumas bibliotecas foram utilizadas para auxiliar a extração de algumas das características, sendo estas: *NLTK*, para a criação de n-gramas de palavras e caracteres utilizados na extração de características lexicais; *NLPPort*<sup>2</sup> (Ferreira, Gonçalo Oliveira e Rodrigues, 2019), uma biblioteca que permite a utilização em Python das ferramentas do NLPPort, e que foi utilizada para a obtenção das tags de POS necessárias para a extração de características sintáticas, Reconhecimento de Entidades Mencionadas (único método dos apresentados que foi aplicado antes da conversão do texto para minúsculas dada a importância das letras maiúsculas no processo) para a extração de características semânticas, e lematização do texto, quer para o cálculo das relações semânticas entre palavras, quer para a utilização do modelo de vetores de palavras, PT-LKB, descrito na secção à frente sobre Características distribucionais; *gensim*, que em conjunto com o *scikit-learn* permitiram a extração e utilização das características distribucionais.

Os modelos apresentados neste capítulo foram inspirados pelo sistema ASAPP 2.0, uma vez que este representa o estado da arte para o cálculo da SST em português, como já referido.

Ao todo foram extraídas **71 características textuais** diferentes, para cada par, sendo que através de um processo de seleção (ver secção 3.2) foi possível reduzir este número sem perda de performance dos modelos. As características textuais extraídas podem ser divididas em características lexicais, sintáticas, semânticas e distribucionais, que se apresentam, em maior detalhe, nas seguintes secções.

### 3.1.1 Características lexicais

Para o cálculo das características lexicais cada frase do par foi dividida em n-gramas de palavras de dimensão  $n = 1$ ,  $n = 2$  e  $n = 3$ , bem como n-gramas de caracteres de tamanho  $n = 2$ ,  $n = 3$  e  $n = 4$ . Sobre as frases convertidas em n-gramas foram aplicadas três métricas para o cálculo da similaridade entre conjuntos, sendo estas o Coeficiente de Jaccard, Sobreposição e Coeficiente de Dice. Estas métricas foram sempre aplicadas nas situações em que ambas as frases do par se encontravam divididas em n-gramas do mesmo tipo (palavras ou caracteres) e dimensão, o que permitiu calcular um total de **18 características** (três por cada tipo e dimensão de n-gramas).

Para cada par de frases a comparar foram também criados dois conjuntos compostos pelo agrupamento dos n-gramas que compunham essas mesmas frases. Um desses conjuntos era composto por n-gramas de caracteres de tamanho  $n = 2$ ,  $n = 3$  e  $n = 4$  e o outro por n-gramas de palavras de tamanho  $n = 1$  e  $n = 2$ . As frases foram, com recurso aos conjuntos de n-gramas, convertidas em vetores binários, de acordo com a existência (1) ou não (0) do n-grama do conjunto na frase a ser convertida. Por último, a similaridade entre as frases foi calculada com recurso ao cosseno entre os vetores, o que representou a extração de mais **duas características** lexicais, uma por cada um dos conjuntos de n-gramas. Apresenta-se de seguida um exemplo no qual um par de frases (**t** e **h**) foi convertido no respetivo vetor binário, de acordo com o último método apresentado acima, e sobre o qual foi calculada a similaridade através do cosseno. Apesar do exemplo apresentado recorrer a n-gramas de caracteres, o método funciona da mesma forma para n-gramas de palavras.

<sup>1</sup><https://radimrehurek.com/gensim/>

<sup>2</sup><https://github.com/jdportugal/NLPPort>

**t:** braga bateu slovan liberec 1 0

↓

**N-gramas de caracteres de tamanho 2, 3 e 4 em t:** {lov, lib, ib, erec, br, teu, slov, ra, er, bate, re, ba, ov, van, ber, eu, ova, bra, rec, te, va, ate, aga, libe, at, ovan, ere, li, rag, lova, lo, brag, ga, slo, ec, raga, an, bat, be, iber, sl, ateu, ibe, bere, ag}

**h:** braga venceu slovan liberec 1 0

↓

**N-gramas de caracteres de tamanho 2, 3 e 4 em h:** {lov, lib, ib, erec, br, slov, ceu, ra, er, re, ov, van, ber, eu, ova, bra, nce, rec, ve, va, ence, nceu, aga, nc, libe, ovan, ere, li, rag, lova, lo, brag, ga, slo, ec, raga, ce, an, venc, enc, en, be, iber, sl, ven, ibe, bere, ag}

**Reunião dos n-gramas de caracteres em t e h:** {ib, br, ra, er, bate, re, ov, rec, nce, va, ve, nceu, nc, ovan, li, lo, brag, slo, ce, an, venc, iber, ibe, ag, lov, lib, erec, teu, slov, ceu, ba, van, ber, eu, ova, bra, te, ence, ate, libe, at, ere, rag, lova, ga, raga, ec, bat, enc, en, be, sl, ven, ateu, bere, aga}

**Vetor binário de t:** [1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1]

**Vetor binário de h:** [1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1]

**Cosseno entre os vetores t e h:** 0.79611

No total foram utilizadas **20 características lexicais** para o treino do modelo de SST.

### 3.1.2 Características sintáticas

À semelhança do processo que foi utilizado para calcular as características lexicais, no cálculo das características sintáticas todas as frases foram convertidas nos respectivos vetores de POS-tags. Sobre estes vetores foi calculada a diferença absoluta, para cada tag existente na união dos dois vetores, entre as tags presentes em cada uma das frases do par. No total foram utilizadas 24 tags diferentes, o que se traduziu em **24 características** sintáticas, as quais se enumeram abaixo:

- *n* - nome
- *ec* - prefixo
- *adj* - adjetivo
- *in* - interjeição
- *adv* - advérbio
- *nprop* - nome próprio
- *art* - artigo
- *num* - numeral
- *conj-c* - conjunção coordenativa
- *pden* - palavra denotativa
- *conj-s* - conjunção subordinativa
- *prep* - preposição

- *pro-ks* - pronome conectivo subordinativo
- *proadj* - pronome adjetivo
- *pron-det* - pronome determinativo
- *pron-indp* - pronome independente
- *propess* - pronome pessoal
- *prosub* - pronome substantivo
- *punc* - pontuação
- *v* - verbo
- *v-ger* - verbo no gerúndio
- *v-inf* - verbo no infinitivo
- *v-pcp* - verbo no particípio passado
- *vaux* - verbo auxiliar

Em adição às características extraídas através de POS-tagging, foi também efetuada a análise de dependências sintáticas com recurso ao *spaCy*. De cada frase foram extraídos triplos compostos por duas palavras e a sua relação de dependência. O tipo de relação não foi considerado, mas sim a existência das mesmas relações nas frases a comparar. Apesar disso, relações com tokens relativos a sinais de pontuação não foram considerados. Sobre os triplos que compunham cada frase do par foi calculado o Coeficiente de Jaccard. Esta característica juntamente com as 24 anteriores resultaram num total de **25 características sintáticas** calculadas.

### 3.1.3 Características semânticas

De forma a extrair as relações semânticas entre as palavras em cada par de frases, foi utilizado um conjunto de triplos, na forma *palavra<sub>1</sub> RELAÇÃO-SEMÂNTICA palavra<sub>2</sub>*. Estes triplos foram obtidos com recurso a dez Bases de Conhecimento Lexical para o português (Gonçalo Oliveira, 2018a), contudo, apenas foram utilizadas relações que existissem em pelo menos três destas BCL.

Foram extraídas **quatro características** semânticas, que dizem respeito ao número de relações existentes entre as palavras de cada par de frases. As relações utilizadas foram: (i) sinonímia; (ii) hiperonímia/hiponímia; (iii) antonímia; (iv) qualquer outra relação existente no conjunto de triplos.

Para além das relações semânticas, foram também utilizadas as Entidades Mencionadas, devido à importância que estas têm na compreensão de texto. As características extraídas consistiram na diferença absoluta entre o número de entidades de cada tipo identificadas entre o par de frases. Foram utilizadas dez categorias diferentes de EM, sendo estas: *Abstração, Acontecimento, Coisa, Local, Obra, Organização, Pessoa, Tempo, Valor* e *Outro*. Dado que cada categoria representou a extração de uma característica semântica, este método permitiu a extração de **10 características**. Para além destas, foi também calculada a diferença absoluta entre o número total, independentemente do tipo, de EM entre as frases, o que resultou numa característica adicional.

No total foram utilizadas **15 características semânticas** para treinar o modelo.

### 3.1.4 Características distribucionais

Algumas das características distribucionais utilizadas têm por base o cálculo da matriz de TF-IDF do corpus. Esta permitiu transformar cada frase num vetor, através da substituição de cada palavra pelo respetivo valor na matriz, aplicar a média desses vetores, de forma a obter um único vetor para representar a frase, e, por último, calcular o cosseno entre os vetores das frases que se pretendiam comparar. Este método permitiu a extração de **uma característica**.

Dada a importância dos modelos de vetores de palavras (em inglês, word embeddings) para o cálculo da similaridade semântica, foram explorados cinco modelos pré-treinados para o português, nomeadamente:

- Vetores de palavras **NILC**<sup>3</sup> (Hartmann et al., 2017), que oferecem uma grande variedade de modelos pré-treinados para o português, desenvolvidos com recurso a um corpus de grande dimensão. Dos modelos disponíveis, foram selecionados o *Word2vec-CBOW* e o *Glove* (Pennington, Socher e Manning, 2014), ambos com vetores de dimensão 300.
- Vetores de palavras **fastText.cc** (Bojanowski et al., 2016), disponibilizam modelos para 157 línguas diferentes, treinados com recurso ao *Common Crawl*<sup>4</sup> e à Wikipédia e utilizando o algoritmo de *fastText*. Para a utilização no modelo descrito neste documento, apenas os vetores de palavras que dizem respeito à língua portuguesa foram selecionados.
- Vetores de palavras **PT-LKB** (Gonçalo Oliveira, 2018b), recorrem a uma abordagem diferente para a aprendizagem do modelo, uma vez que utilizam o método *node2vec* (Grover e Leskovec, 2016) sobre as já referidas BCL para calcular os vetores de palavras, em vez de recorrerem a um corpus de treino. Os vetores selecionados têm dimensão 64, uma vez que apresentaram os melhores resultados em testes de cálculo da similaridade entre palavras (ver Gonçalo Oliveira, 2018b).
- Vetores de palavras **ConceptNet Numberbatch** (Speer, Chin e Havasi, 2017), treinados com recurso a uma generalização da técnica de *retrofitting* (Faruqui et al., 2015), que permite melhorar a capacidade de representação de palavras através de vetores com recurso à base de conhecimento *ConceptNet*, na qual são representadas relações entre palavras de acordo com a sua utilização comum. Visto que estes vetores são multilingue, foram extraídos apenas os vetores que diziam respeito a palavras em português, sendo que os mesmos têm dimensão 300.

Cada um dos cinco modelos mencionados permitiu extrair duas características distribucionais, num total de **dez características**, após a transformação das frases em vetores. Nestes, cada palavra era representada pelo respetivo vetor no modelo utilizado. A forma como o vetor da frase foi calculado corresponde à principal diferença entre as duas características extraídas. Na primeira, o vetor da frase foi calculado exclusivamente através da média dos vetores das palavras que a constituíam; na segunda, o vetor da frase foi calculado com recurso à média ponderada entre os vetores das palavras do modelo e o seu valor na matriz de TF-IDF.

Em ambos os casos, a similaridade entre cada par de frases foi calculada com recurso ao cosseno entre os respetivos vetores. No total foram extraídas **11 características distribucionais** para treinar o modelo.

<sup>3</sup><http://nilc.icmc.usp.br/embeddings>

<sup>4</sup><https://commoncrawl.org>

## 3.2 Seleção de Caraterísticas

A seleção de caraterísticas é um processo através do qual é feita uma escolha, com base num conjunto de critérios pré-definidos, das caraterísticas mais relevantes para o modelo desenvolvido, sem que isso implique uma perda de performance. Os principais benefícios desta técnica são:

1. Eliminação de caraterísticas não relevantes ou que introduzam ruído, o que pode levar a uma melhoria do desempenho do modelo.
2. Maior rapidez no treino do modelo, visto que existe uma redução do número de caraterísticas a calcular e a utilizar com os algoritmos de aprendizagem.
3. Prevenção de overfitting, uma vez que é feita uma redução do número de caraterísticas específicas do conjunto de dados de treino.

O processo de seleção de caraterísticas envolveu a utilização de três métodos supervisionados diferentes, cuja implementação está disponível no *scikit-learn*. Foram esses métodos:

- **Estatísticas Univariável** (*Univariate Statistics*, em inglês) - calcula individualmente para cada caraterística a sua relação estatística com os resultados esperados. Isto significa que caraterísticas que dependam de outras para serem informativas são descartadas. Como não implica a construção de nenhum modelo este teste é relativamente rápido de aplicar. A função de desempenho utilizada é a *f\_regression*, que calcula a correlação entre o resultado do modelo obtido com a utilização de cada uma das caraterísticas e o resultado esperado. A função *SelectPercentile* permite definir a percentagem de caraterísticas a escolher, de acordo com o seu desempenho. Como não existia conhecimento prévio sobre qual o número de caraterísticas que permitiam obter melhores resultados, os testes foram feitos com um conjunto de limites, sendo este [25, 35, 45, 50, 55, 65, 75].
- **Seleção baseada na construção de um Modelo** (*Model-Based Feature Selection*, em inglês) - utiliza um algoritmo de aprendizagem supervisionada para criar um modelo para calcular a importância de cada caraterística. O algoritmo utilizado não precisa de ser o mesmo do modelo que se pretende desenvolver, contudo deve incorporar uma métrica que permita fazer a avaliação das caraterísticas. No *scikit-learn*, modelos baseados em árvores de decisão contêm um atributo designado por *feature\_importances\_* que codifica a importância de cada caraterística, e que por esse motivo levou a que o algoritmo *Random Forest* fosse o escolhido. Para a seleção das caraterísticas foi utilizado o método *SelectFromModel*, que para além de permitir definir o algoritmo que se pretende utilizar para construir o modelo, permite também controlar o número de caraterísticas a serem selecionadas. Apesar deste número não poder ser definido em percentagem, utilizaram-se limites que correspondessem ao mesmo número de caraterísticas selecionadas pelo método *SelectPercentile*. Uma das principais vantagens deste método relativamente ao anterior é que este considera todas as caraterísticas de uma só vez, o que permite capturar interações entre estas, caso existam.
- **Seleção Iterativa** (*Iterative Feature Selection*, em inglês) - a seleção iterativa é semelhante ao método de seleção com base na construção de um Modelo, sendo que a principal diferença está no facto de este construir vários modelos para a análise da importância das caraterísticas. Em cada iteração deste método é construído um novo



modelo através de um processo designado por RFE (*Recursive Feature Elimination*, em inglês), que consiste na remoção da característica menos relevante para o modelo anterior, antes da construção do seguinte. Este processo é repetido até que se atinja um critério de paragem, que corresponderá ao número de características que se pretendem seleccionar. À semelhança do que acontece no método de seleção anterior, o algoritmo utilizado para a construção do modelo deve incorporar uma métrica que permita fazer a avaliação das características, e por esse motivo optou-se novamente por usar o *Random Forest*. Os critérios de paragem do algoritmo são definidos pelo número específico de características que se pretendem escolher, pelo que se utilizaram os números correspondentes aos limites definidos nos outros dois mecanismos de seleção.

Sempre que um novo modelo era treinado, cada um dos métodos mencionados era utilizado individualmente com o objetivo de obter o conjunto de características textuais que, de acordo com o método utilizado, permitia obter um melhor desempenho do modelo. Este desempenho foi medido com recurso ao coeficiente de determinação ( $R^2$ ) entre os valores de similaridade calculados pelo modelo através da função de regressão obtida com recurso aos algoritmos de aprendizagem e os valores de similaridade anotados pelo júri humano. Os valores de desempenho obtidos por cada método foram comparados, não só entre si, mas também com o desempenho obtido sem seleção de características. Caso um dos métodos permitisse atingir uma maior performance que a utilização de todas as características, a sua seleção de características era a escolhida para ser utilizada com o modelo, caso contrário eram utilizadas todas as características.

Após esta etapa, todos os modelos foram avaliados em cada um dos datasets de teste da coleção ASSIN.

### 3.3 Modelos para o cálculo da SST

Nesta secção são apresentados dois tipos de modelos para o cálculo da SST: **Modelos baseados em Vetores de Palavras**, que utilizam uma abordagem simplista baseada no cálculo do cosseno entre os vetores dos pares das frases a comparar; **Modelos baseados na aprendizagem de Algoritmos de Regressão**, que apresentam uma abordagem mais sofisticada, apoiada na aprendizagem de modelos de regressão com base nas características textuais, apresentadas na secção 3.1, dos pares de frases de treino da coleção ASSIN. Uma das diferenças entre estes dois tipos de modelos está no facto de o modelo baseado em vetores de palavras, pelo facto de não requerer a aprendizagem de um algoritmo de regressão, ser aplicado diretamente ao conjunto de dados de teste utilizados para avaliação. Os três principais objetivos do desenvolvimento dos modelos apresentados de seguida foram:

- Desempenho superior ao das baselines disponibilizadas na tarefa ASSIN. Estas baseiam-se exclusivamente na sobreposição dos tokens do par de frases a comparar e apresentam um resultado de  $\rho = 0.64$ , para o dataset de teste PTPT, e  $\rho = 0.63$ , para o dataset de teste PTBR.
- Desempenho equiparável ao estado-da-arte, obtido pelo modelo ASAPP 2.0 com um  $\rho = 0.75$ .
- Integração do modelo com melhor desempenho num agente de diálogo que mapeia interações do utilizador com perguntas na sua base de conhecimento.

### 3.3.1 Modelos baseados em Vetores de Palavras

Numa primeira abordagem ao problema do cálculo da SST entre pares de frases foram desenvolvidos dois modelos, com recurso ao *gensim*, exclusivamente baseados na utilização de vetores de palavras. Visto que estes não necessitam da aprendizagem de um algoritmo de regressão, nem da extração de características textuais para o seu treino, a sua implementação foi relativamente rápida. O primeiro modelo desenvolvido foi baseado no *FastText* e, para tal, foi utilizado um conjunto já treinado de vetores em português baseado no algoritmo *CBOW*, de dimensão 300, com uma janela de tamanho 5 e n-gramas também de tamanho 5. Estes vetores estão disponíveis gratuitamente no site *FastText.cc*<sup>5</sup>, que reúne uma coleção de vetores baseados em *FastText* para 157 línguas diferentes (Grave et al., 2018). O segundo modelo foi baseado no *Word2vec* e, à semelhança do modelo baseado em *FastText*, também utiliza um conjunto pré-treinado de vetores de dimensão 300 em português, neste caso baseados no algoritmo *Skip-Gram*. Estes vetores encontram-se disponíveis gratuitamente no site do *NILC*. O cálculo da similaridade entre pares de frases com recurso a modelos baseados em vetores de palavras foi feito através do cosseno entre os vetores que representam as frases. Estes vetores foram obtidos através da média dos vetores das palavras que constituem as frases, sendo que estes últimos estão disponíveis nos modelos pré-treinados de vetores de palavras. Os vetores das frases só foram calculados após uma etapa de pré-processamento composta pela remoção de espaços em branco, quebras de linha e caracteres não-alfanuméricos, tokenização das frases e conversão do texto para letra minúscula. No total foram efetuados 32 testes, 16 por cada um dos modelos, nos quais se experimentou remover stopwords, converter números para extenso e multiplicar cada um dos vetores de palavras que compõem a frase pelo respetivo valor na matriz de TF-IDF antes de calcular a sua média. Os melhores resultados foram de  $\rho = 0.54$ , para o dataset de teste PTPT, e  $\rho = 0.56$ , para o dataset de teste PTBR, obtidos por um modelo baseado em *Word2Vec*, sem remoção de stopwords, sem conversão de numerais e sem a multiplicação de vetores. Este resultado ficou muito abaixo quer do estado-da-arte, quer da baseline disponibilizada pela tarefa ASSIN, pelo que se considerou que não podia ser aceitável utilizar este modelo no agente de diálogo. Percebeu-se também que um sistema baseado exclusivamente em vetores de palavras não iria conseguir atingir os resultados pretendidos, pelo que se decidiu adotar uma abordagem assente na aprendizagem supervisionada de modelos de regressão.

### 3.3.2 Modelos baseados na aprendizagem de Algoritmos de Regressão

Os modelos baseados na aprendizagem de algoritmos de regressão, ao contrário do que acontece com os modelos *Word2vec* e *FastText*, baseiam-se na extração de características dos dados de treino para treinar algoritmos de aprendizagem capazes de representar as instâncias através de uma função. Isto permite fazer a previsão do valor de similaridade semântica entre pares de frases com características nunca antes “vistas” pelos modelos. Os algoritmos de regressão utilizados foram: Linear (L); *Random Forest* (RFR); Máquinas de Vetores de Suporte (SVR); *Gradient Boosting* (GBR); Árvores de Decisão (DTR); Processos Gaussianos (GP); *AdaBoost* (AR); *Bagging* (BR).

É importante referir que a extração de características para o treino dos modelos foi realizada de forma incremental, o que significa que nem todos os modelos foram treinados com as 71 características apresentadas na secção 3.1. No total, o conjunto inicial de características foi incrementado três vezes. Estes incrementos, descritos com maior detalhe

<sup>5</sup><https://fasttext.cc/docs/en/crawl-vectors.html>

no último parágrafo desta secção, permitiram dividir o processo de desenvolvimento dos modelos em duas etapas:

- 1ª Etapa
  1. Extração de um conjunto inicial de características textuais dos datasets de treino da coleção ASSIN, com base na facilidade e rapidez do processo.
  2. Treino de vários modelos, utilizando os diferentes algoritmos de aprendizagem mencionados, com base nas características textuais extraídas.
  3. Avaliação dos modelos treinados nos datasets de teste da coleção ASSIN.
  4. Seleção dos três algoritmos com melhor desempenho para utilização nos testes seguintes.
- 2ª Etapa
  1. Extração de um novo conjunto de características para o desenvolvimento de novos modelos.
  2. Treino dos novos modelos para o cálculo da SST, utilizando apenas os algoritmos selecionados na etapa anterior.
  3. Avaliação dos modelos nos datasets de teste da coleção ASSIN.
  4. Repetir os passos anteriores (apenas para a 2ª Etapa), caso o valor de Pearson obtido estagnasse e/ou se encontrasse abaixo do valor do estado-da-arte.

O **Conjunto Inicial** de características textuais extraídas para cada par era composto por 27 características, as quais incluíam: Coeficiente de Jaccard, Sobreposição e Coeficiente de Dice para todas as dimensões de n-gramas de caracteres e palavras referidos (18); Relações semânticas (4); Cosseno entre os vetores da Matriz de TF-IDF(1); Cosseno entre os vetores de palavras *Word2vec* e *FastText*, com e sem a multiplicação dos vetores pelos respetivos valores na Matriz de TF-IDF (4).

O **Segundo Conjunto** era composto por todas as características do primeiro, às quais se adicionaram o cosseno dos vetores de palavras *Glove* e *PTLKB*, com e sem a multiplicação dos vetores pelos respetivos valores na Matriz de TF-IDF (4), perfazendo um total de 31 características.

O **Terceiro Conjunto** era, mais uma vez, composto por todas as características dos dois anteriores, ao qual foram adicionadas 11 características distribucionais relativas à extração das Entidades Mencionadas e, pela primeira vez, foram adicionadas características sintáticas relacionadas com a utilização das tags de POS (24). Este conjunto era assim composto por 66 características textuais.

O **Quarto e Último Conjunto** era composto pelas características de todos os anteriores, com a adição da análise de dependências sintáticas (1), o cosseno entre os vetores de palavras *Numberbatch*, com e sem a multiplicação dos vetores pelos respetivos valores na Matriz de TF-IDF (2) e o cosseno entre os vetores binários de conjuntos de n-gramas de caracteres e palavras (2), perfazendo o total de 71 características textuais.

### 3.4 Avaliação dos modelos de SST

Os modelos desenvolvidos foram avaliados com recurso à coleção ASSIN. Como referido anteriormente (ver secção 2.3), a avaliação é feita através do cálculo da correlação de Pearson entre os valores de similaridade atribuídos pelos modelos desenvolvidos aos pares de frases dos datasets de teste da coleção e os valores anotados por um júri. Os testes foram efetuados separadamente quer no dataset em PTPT, quer no dataset em PTBR.

No total foram efetuados **92 testes** distribuídos da seguinte forma pelos quatro conjuntos de características:

- **1º Conjunto (44 testes):**

- Quatro testes (um por cada dataset de teste, com e sem seleção de características) por cada um dos oito algoritmos de regressão adotados (ver secção 3.3.2), num total de **32 testes**. Os três modelos com melhor desempenho foram os baseados nos algoritmos MVS ( $\rho = 0.72$ ), *Random Forest* ( $\rho = 0.71$ ) e *Gradient Boosting* ( $\rho = 0.72$ ), todos sem aplicar seleção de características. Uma vez que não traria qualquer benefício continuar a efetuar testes com todos os modelos, optou-se por utilizar somente os três com melhor desempenho nos testes seguintes.
- Quatro testes (um por cada dataset de teste, com e sem seleção de características) por cada um dos três algoritmos de regressão selecionados, nos quais se estudou o impacto da remoção de stopwords no cálculo das características distribucionais que recorrem aos vetores de palavras, num total de **12 testes**. Verificou-se que esta remoção não tem qualquer impacto no desempenho dos modelos, pelo que não se voltou a aplicar a mesma.

- **2º Conjunto (24 testes):**

- Quatro testes (um por cada dataset de teste, com e sem seleção de características) por cada um dos três algoritmos de regressão selecionados, nos quais se utilizaram apenas as características extraídas com recurso aos vetores de palavras *Glove*. Total de **12 testes**.
- Quatro testes (um por cada dataset de teste, com e sem seleção de características) por cada um dos três algoritmos de regressão selecionados, nos quais se utilizaram todas as características do conjunto. Total de **12 testes**.

- **3º Conjunto (12 testes):**

- Quatro testes (um por cada dataset de teste, com e sem seleção de características) por cada um dos três algoritmos de regressão selecionados. Total de **12 testes**.

- **4º Conjunto (12 testes):**

- Quatro testes (um por cada dataset de teste, com e sem seleção de características) por cada um dos três algoritmos de regressão selecionados. Total de **12 testes**.

Dos 46 testes aos quais foi aplicada seleção de características, existiram 19 casos em que o desempenho do modelo testado não beneficiou desta seleção. Através dos 27 testes nos quais o desempenho do modelo melhorou foi possível inferir quais as características textuais mais relevantes para o cálculo da SST. A relevância foi calculada com base no número de vezes que as características foram selecionadas em função do número de testes nos quais foram utilizadas e o desempenho do modelo melhorou com a seleção de características. Apresentam-se na tabela 3.1 as características textuais mais importantes para o cálculo da SST, organizadas pela respetiva ordem de relevância (em percentagem):

Caraterística	Relevância
Cosseno entre os Vetores binários de n-gramas de caracteres	100%
Cosseno entre Vetores Word2vec x Valores na matriz TF-IDF	92%
Cosseno entre Vetores FastText x Valores na matriz TF-IDF	92%
Cosseno entre vetores da Matriz de TF-IDF	92%
Cosseno entre Vetores FastText	85%
Coefficiente de Dice sobre n-gramas de caracteres de tamanho 3	81%
Coefficiente de Dice sobre n-gramas de caracteres de tamanho 2	77%
Coefficiente de Jaccard sobre n-gramas de caracteres de tamanho 2	74%
Coefficiente de Jaccard sobre n-gramas de caracteres de tamanho 3	74%

Tabela 3.1: Caraterísticas textuais mais relevantes de acordo com os resultados do processo de seleção.

Todas as caraterísticas textuais que não aparecem na tabela 3.1 foram selecionadas em menos de 50% dos testes, pelo que não podem ser consideradas suficientemente relevantes. As percentagens de seleção mais baixas foram obtidas pelas caraterísticas semânticas, que juntamente com a análise de dependências sintáticas, nunca foram selecionadas. À partida poderia parecer estranho as caraterísticas semânticas nunca terem sido selecionadas em modelos cujo objetivo é calcular a SST, contudo é importante referir que os vetores de palavras utilizados nas caraterísticas distribucionais conseguem encapsular várias tipos de relações semânticas (Mikolov et al., 2013b). A Sobreposição de n-gramas foi selecionada apenas em 18% dos testes exclusivamente nos n-gramas de palavras de tamanho 1. O conjunto de todas as tags de POS foi escolhido apenas em 11% dos casos. Devido aos baixos valores de seleção, pode-se concluir que este conjunto de caraterísticas textuais não apresenta relevância suficiente que justifique a sua integração em futuros modelos para o cálculo da SST em português.

A tabela 3.2 apresenta os resultados através dos quais foi feita a seleção dos modelos a utilizar após os testes com o 1º conjunto de caraterísticas. Para cada um dos datasets de teste é apresentado o valor médio e máximo de Pearson obtido por cada um dos modelos, independentemente dos testes terem sido realizados com ou sem seleção de caraterísticas. O valor máximo de Pearson foi utilizado como critério principal para a escolha dos modelos, dado um dos objetivos principais deste trabalho ser integrar o modelo com melhor desempenho num agente de diálogo, e a média como critério de desempate. Com base em ambos os critérios, foram escolhidos os modelos SVR, GBR e RFR, como referido anteriormente.

Modelo	PTPT		PTBR	
	$\rho$ - Médio	$\rho$ - Máximo	$\rho$ - Médio	$\rho$ - Máximo
L	0.69	0.71	0.70	0.70
SVR	<b>0.71</b>	<b>0.72</b>	0.71	0.71
GBR	<b>0.72</b>	<b>0.72</b>	0.71	0.71
RFR	<b>0.70</b>	<b>0.71</b>	0.70	0.70
DTR	0.51	0.52	0.51	0.51
GP	0.26	0.36	0.30	0.35
AR	0.70	0.70	0.69	0.69
BR	0.70	0.70	0.70	0.70

Tabela 3.2: Desempenho Máximo e Médio dos modelos desenvolvidos com recurso ao 1º conjunto de caraterísticas quando testados nos datasets em PTPT e PTBR.

Os testes efetuados com o segundo e terceiro conjuntos de características não permitiram melhorar o desempenho dos modelos, algo que só foi alcançado com o quarto conjunto. As tabelas 3.3 e 3.4 apresentam os resultados obtidos pelos modelos baseados nos três algoritmos selecionados, sem seleção de características e com seleção de características, respetivamente, quando treinados com todas as características textuais.

Modelo	PTPT		PTBR	
	$\rho$	MSE	$\rho$	MSE
SVR	0.63	0.75	0.64	0.46
GBR	0.72	0.65	0.71	0.38
RFR	0.72	0.64	0.71	0.38

Tabela 3.3: Desempenho dos modelos quando testados nos datasets em PTPT e PTBR sem seleção de características.

Modelo	PTPT		PTBR	
	$\rho$	MSE	$\rho$	MSE
SVR	<b>0.73</b>	<b>0.61</b>	0.72	0.36
GBR	0.73	0.62	0.72	0.37
RFR	0.72	0.63	0.71	0.38

Tabela 3.4: Desempenho dos modelos quando testados nos datasets em PTPT e PTBR com seleção de características.

Após a realização de todos os testes mencionados, o valor máximo de Pearson obtido foi de 0.73, pelos modelos baseados no SVR e GBR, quando testado no dataset PTPT com seleção de características. Numa tentativa de melhorar este resultado e atingir o valor do estado-da-arte foram adotadas duas estratégias: utilização de um *ensemble* dos três algoritmos com melhor desempenho; otimização dos parâmetros dos algoritmos utilizados, quer individualmente, quer no *ensemble*, uma vez que até ao momento apenas tinham sido utilizados os valores por defeito do *scikit-learn*.

O *ensemble* foi construída com base no método *VotingRegressor* do *scikit-learn*, que combina através da média os resultados calculados por todos os algoritmos individualmente. No que diz respeito à otimização de parâmetros, recorreu-se ao método *GridSearch*, também do *scikit-learn*, que permite definir uma lista em Python com os valores que se pretendem testar para cada um dos parâmetros configuráveis. Os valores dos parâmetros testados foram os seguintes:

**SVR** - *kernel*: [linear, polinomial, RBF, sigmoidal]

$C$ : [1.0, 10.0, 100.0]

**RFR** -  $n\_estimators$ : [10, 20, 100, 200]

O valor de  $C$  do SVR permite controlar a distância máxima entre os vetores de suporte, sendo que quanto mais alto este valor, maior poderá ser essa distância. Num cenário de regressão, como é o caso, isto implica que a tolerância ao erro seja maior, uma vez que instâncias fora do  $\epsilon$ -tubo possam vir a ficar dentro. O parâmetro  $n\_estimators$  do RFR controla o número de árvores utilizadas para criar a floresta.

A aplicação destas estratégias não permitiu melhorar o desempenho dos modelos, que obtiveram o mesmo resultado,  $\rho = 0.73$ , que os melhores modelos apresentados na tabela

3.4. Por esse motivo, foi calculado o erro quadrático médio (*Mean square error* (MSE), em inglês) entre os valores de similaridade calculados pelos modelos e os anotados pelo júri, tal como acontece com o cálculo da correlação de Pearson. Esta métrica, apresentada para cada modelo nas tabelas 3.3 e 3.4, foi utilizada como medida secundária na avaliação ASSIN e quanto menor for o seu valor, melhor é o desempenho do modelo. Com base nesta nova métrica foi possível observar que a performance do modelo SVR é ligeiramente superior à do GBR, o que fez com que este fosse escolhido para integrar o agente de diálogo. Este modelo recorre apenas a nove características textuais, sendo estas: Vetores de palavras Word2Vec e FastText com e sem a multiplicação de cada vetor pelo seu respetivo valor na matriz de TF-IDF; Matriz de TF-IDF; Coeficiente de Jaccard, Dice e Sobreposição entre n-gramas de palavras de tamanho 1; Cosseno entre os vetores binários calculados com recurso aos n-gramas de caracteres de tamanho 2, 3 e 4.

Apesar de não ter sido atingido o estado-da-arte no cálculo da SST, o modelo selecionado apresenta um desempenho a par com outros dos melhores modelos existentes para esse fim, nomeadamente ao nível dos melhores resultados oficiais na ASSIN, apresentados na tabela 2.2. A pequena diferença de desempenho entre o modelo desenvolvido e o ASAPP 2.0 poderá ser explicada pela forma como foram calculadas algumas das características textuais e pela utilização de algoritmos de aprendizagem diferentes, visto não ter sido possível utilizar exatamente os mesmos algoritmos, uma vez que a implementação do ASAPP 2.0 foi feita em Java.





## Capítulo 4

# Agente Conversacional baseado em Similaridade Semântica Textual

Neste capítulo é apresentado o processo de desenvolvimento do agente de diálogo. Este começa pela integração do modelo de SST, selecionado na secção 3.4, com a base de conhecimento do agente. De seguida, é feita a avaliação, com recurso à criação de variações das interações existentes nessa mesma base de conhecimento, do desempenho do agente relativamente a outras baselines. Por último, apresenta-se o mecanismo de identificação e resposta a perguntas fora-do-domínio, que permite tornar o diálogo com o sistema mais humano.

### 4.1 Integração do modelo SST num Agente de Diálogo

Uma das principais utilidades do desenvolvimento de um modelo para o cálculo da SST é a possibilidade de este poder ser integrado num agente de diálogo como método para a obtenção de respostas às interações do utilizador. Desta forma, o modelo selecionado na secção 3.4 foi utilizado para fazer o mapeamento entre pergunta-resposta num agente cujo principal objetivo é ser capaz de responder a perguntas relacionadas com o exercício da atividade económica em Portugal. Para o desenvolvimento deste agente foi utilizada uma lista com 379 FAQs sobre este domínio. Estas contêm 120 FAQs do *Guia de Aplicação do Regime Jurídico de Acesso e Exercício de Atividades de Comércio, Serviços e Restauração – RJACSR*, 56 da *Legislação do Alojamento Local*, e 203 do *Portal Empresa*<sup>1</sup>. A resposta do agente a uma interação do utilizador é obtida através do cálculo da SST entre essa mesma interação e todas as interações das FAQs existentes na base de conhecimento do agente, que neste caso serão 379. Visto que a base de conhecimento é criada com base nas FAQs, a informação é armazenada no formato Pergunta/Interação-Resposta. Por esse motivo, a resposta à interação do utilizador corresponde à resposta da interação na base de conhecimento com a qual o valor de similaridade calculado foi maior. Apresentam-se de seguida os mecanismos utilizados para fazer a avaliação do agente.

---

<sup>1</sup>Estes documentos foram obtidos através do portal do *Balcão do Empreendedor* (BDE), em Junho de 2018.

## 4.2 Avaliação do desempenho do Agente de Diálogo

De forma a testar a performance do modelo para o cálculo da SST no domínio do BDE, foi necessário aplicar uma avaliação mais sistemática. Tendo em conta que, na maioria dos casos, os utilizadores não vão fazer perguntas de forma exatamente igual àquelas que se encontram na base de conhecimento do agente, um conjunto de variações dessas perguntas foram criadas e utilizadas para o avaliar. No total foram criados quatro tipos de variações da seguinte forma:

- Produzidas manualmente no INESC por um conjunto de falantes nativos de português. Este conjunto é composto por apenas 75 variações no domínio do serviço *Empresa na Hora do Balcão do Empreendedor*. Algumas destas variações referem-se à mesma interação.
- Com recurso à API do Google Tradutor<sup>2</sup> foram criadas paráfrases das perguntas originais, através do seguinte conjunto de ações: tradução das perguntas em Português para Inglês e novamente para Português (VG1); aplicar novamente a tradução para Inglês, seguida da tradução para Português ao resultado anterior (VG2). Cada uma destas variantes é composta por 379 variações, uma por cada interação.
- Produzidas manualmente por um conjunto de alunos falantes nativos de português da Universidade de Coimbra. Estas incluem não só paráfrases das perguntas originais, mas também perguntas relacionadas, algumas das quais podem também incluir pequenos erros ortográficos (VUC). Este tipo de variações apenas foi criado para as FAQs do domínio do RJACSR e Balcão do Empreendedor. Este conjunto é composto por 451 variações, contudo estas dizem respeito a apenas 166 interações diferentes, o que significa que para algumas perguntas foram criadas várias variações.

A tabela 4.1 ilustra o dataset utilizado, com exemplos das diversas variações criadas para as perguntas originais (OG). É importante referir que devido ao método utilizado, as variações criadas com recurso ao Google Tradutor (VG1 e VG2) são lexicalmente e estruturalmente muito mais parecidas com as perguntas originais do que as criadas manualmente pelos falantes nativos de português (VIN e VUC). Como consequência adicional do método adotado, algumas variações VG1 e VG2 são iguais entre si. Por consequência de terem sido criadas manualmente por dois grupos diferentes de pessoas, não foi possível garantir a existência de uma variação VIN e VUC para cada interação na base de conhecimento do agente. Por esse motivo, nenhum dos exemplos da tabela contém todas as variações da mesma pergunta original.

---

<sup>2</sup><https://cloud.google.com/translate/docs/>

Variação	Texto
<b>OG</b>	<i>São permitidos animais em estabelecimentos de restauração ou bebidas?</i>
<b>VG1</b>	<i>São permitidos animais em restaurantes ou bebidas?</i>
<b>VG2</b>	<i>São permitidos animais em restaurantes ou bebidas?</i>
<b>VUC</b>	<i>Que animais podem entrar num bar ou restaurante?</i>
<b>OG</b>	<i>No alojamento local é obrigatória a certificação energética? Em que termos deve ser efetuada?</i>
<b>VG1</b>	<i>No alojamento local é obrigatório a certificação energética? Em que condições deveria ser feito?</i>
<b>VG2</b>	<i>A certificação energética é necessária em alojamento local? Em que condições deve ser feito?</i>
<b>VUC</b>	<i>Qual o procedimento para certificar energeticamente o meu alojamento local?</i>
<b>OG</b>	<i>Que tipos de sociedades podem ser constituídas nos balcões "Empresa na Hora"?</i>
<b>VIN</b>	<i>Podemos criar que tipos de sociedades nos balcões "Empresa na Hora"?</i>
<b>VG1</b>	<i>Que tipos de empresas podem ser configuradas nos contadores "Empresa no tempo"?</i>
<b>VG2</b>	<i>Quais tipos de empresas podem ser configuradas nos contadores "Empresa com base no tempo"?</i>
<b>OG</b>	<i>Qual a vantagem de indicar, desde logo, um Técnico Oficial de Contas?</i>
<b>VIN</b>	<i>Porque é que é vantajoso ter logo um contabilista?</i>
<b>VG1</b>	<i>Qual é a vantagem de indicar, primeiro, um contador?</i>
<b>VG2</b>	<i>Qual é a vantagem de primeiro indicar um contador?</i>

Tabela 4.1: Exemplos de variações das perguntas originais usadas para testar o agente.

A avaliação consistiu em calcular para cada variação criada o valor de similaridade entre esta e cada uma das 379 interações na base de conhecimento do agente. Considere-se a Variação<sub>X</sub> como exemplo do processo aplicado a todas as variações. Para esta, foram criados pares *Variação-Interação na Base de Conhecimento do Agente*, como se ilustra de seguida:

Variação<sub>X</sub> - Interação<sub>1</sub>

Variação<sub>X</sub> - Interação<sub>2</sub>

...

Variação<sub>X</sub> - Interação<sub>379</sub>

As características textuais de cada par foram extraídas e utilizadas com o modelo de regressão selecionado, de forma a calcular o respetivo valor de similaridade. Visto que cada variação corresponde exclusivamente a uma interação, sabe-se que, em cada teste, apenas um dos 379 pares era composto pela interação e pela respetiva variação. Caso este fosse o par com maior similaridade, o mapeamento e consequentemente a resposta eram considerados corretos. Nas situações em que este par não obtinha o maior valor de similaridade, verificou-se se este valor ficou no Top-3 ou Top-5 dos valores de similaridade de todos os pares, de forma a analisar o quão próximo o resultado ficou de ser a resposta correta. A tabela 4.2 mostra o número e proporção de respostas acertadas pelo agente, quer no total, quer por cada tipo de variação testado. Para além dos casos em que o mapeamento *Variação-Interação* foi feito corretamente (Top-1), apresentam-se ainda os resultados do Top-3 e Top-5. Estes resultados são apresentados de forma cumulativa, o que significa que o Top-5 incorpora os resultados do Top-3 e Top-1, e o Top-3 os do Top-1. Como esperado, devido à natureza de cada tipo de variação, os melhores resultados foram obtidos pelas VG1, com uma proporção de respostas certas de cerca de 83%, sendo que esta aumenta substancialmente se se considerar a presença da resposta correta no Top-3 (96%) ou no Top-5 (97%). Os resultados para as VG2 encontram-se ligeiramente abaixo daqueles obtidos para as VG1, possivelmente devido à variação adicional causada por mais uma etapa de tradução. As variações VUC obtiveram o pior resultado, com cerca de 60% de respostas corretas, valor que sobe para os 84% se se considerar o número de vezes que a resposta correta ficou no Top-5. A baixa performance do modelo para este conjunto de variações era esperada, visto que, como demonstra a tabela 4.1, as perguntas VUC eram mais “criativas”, embora também mais próximas de um cenário real. As variações VIN

obtiveram um resultado intermédio, com cerca de 78% de respostas corretas, o que pode ser explicado pelo facto de, apesar de terem sido criadas manualmente, não apresentarem tanta variabilidade como as VUC, mas apresentarem mais do que as VG1 e VG2.

Variação	Questões #	Top-1		Top-3		Top-5	
		#	%	#	%	#	%
OG	379	364	96.04	379	100.00	379	100.00
VIN	75	59	78.6(6)	68	90.6(6)	72	96.00
VG1	379	317	83.64	367	96.83	370	97.63
VG2	379	313	82.60	361	95.25	365	96.31
VUC	451	273	60.53	351	77.83	380	84.26
Total	1284	962	74.92	1147	89.33	1187	92.45

Tabela 4.2: Desempenho do agente de diálogo quando testado em todas as variações das FAQs, utilizando o modelo SST selecionado.

Após o teste inicial, procurou-se verificar se seria possível agilizar o tempo de resposta do agente. Para tal, foi desenvolvida uma etapa de pré-seleção que consistia em selecionar os 30 pares mais similares, apenas com recurso ao cosseno entre os vetores de palavras *FastText* de cada uma das frases do par. Sobre os pares selecionados era aplicado o método de cálculo de similaridade utilizado no teste anterior. Esta pré-seleção permitiu reduzir o tempo de execução do teste de aproximadamente 35 minutos para cerca de 6, o que seria de esperar visto que para cada variação testada passou a ser apenas necessário calcular as características textuais para 30 pares, em vez dos 379 iniciais. A tabela 4.3 apresenta os resultados obtidos para este teste. Considerando o Top-1, é possível observar que existiu um aumento não significativo do desempenho do agente nas variações VG1 e VG2. Contudo, a performance nas variações VIN sofreu uma ligeira quebra, uma vez que baixou de aproximadamente 78% para 76%, sendo que esta foi mais significativa nas variações VUC, que desceu de cerca de 60% para cerca de 55%. Esta descida pode ser explicada pelo facto de os vetores de palavras utilizados para fazer a pré-seleção, não serem capazes, por si só, de capturar de forma tão eficaz as relações semânticas entre frases com maior variabilidade lexical e estrutural. Isto significa que, em alguns casos, o par que corresponde à resposta correta pode nem ter sido selecionado durante a etapa de pré-seleção.

Variação	Questões #	Top-1		Top-3		Top-5	
		#	%	#	%	#	%
OG	379	358	94.46	379	100.00	379	100.00
VIN	75	57	76.00	69	92.00	71	94.6(6)
VG1	379	318	83.91	350	92.35	353	93.14
VG2	379	315	83.3(3)	346	91.29	349	92.08
VUC	451	250	55.43	301	66.74	311	68.96
Total	1284	940	73.21	1066	83.02	1084	84.42

Tabela 4.3: Desempenho do agente de diálogo quando testado em todas as variações das FAQs, utilizando o modelo SST selecionado sobre uma pré-seleção de 30 possíveis correspondências corretas.

Por último, foi realizado um teste no qual se utilizou um modelo SST que não recorreu às duas características baseadas nos vetores de palavras *FastText*. Isto implicou a criação de um novo modelo, que, face à remoção destas características durante o seu treino, se baseou num

conjunto de características substancialmente maior (24) que o modelo escolhido na secção 3.4 (9) para colmatar uma eventual perda de performance. O novo modelo era em tudo semelhante ao utilizado nos testes anteriores, exceto nas características utilizadas, as quais se enumeram de seguida: (i) todas as características do 1º conjunto de características (ver secção 3.3.2), à exceção do coeficiente de Jaccard e Sobreposição de n-gramas de palavras de tamanho 3, cosseno entre os vetores de palavras *PTLKB* e características baseados no modelo de *FastText* (21); (ii) análise de dependências sintáticas e cosseno entre os vetores binários de n-gramas de caracteres e palavras (3).

A motivação por detrás deste teste deve-se ao tamanho elevado do modelo *FastText*, que ocupa perto de 5 GB, cerca do dobro dos modelos *Word2vec* e *Glove*, o que torna a sua utilização computacionalmente mais exigente em termos de memória. Pelos resultados apresentados na tabela 4.4 observa-se que no geral a performance do agente não foi muito afetada pela remoção das duas características baseadas no *FastText*, uma vez que a diminuição de performance foi inferior a 1% quando comparada com a do modelo utilizado no teste inicial. Apesar disso, o facto deste modelo utilizar um maior número de características textuais fez com que os testes demorassem extremamente mais tempo que qualquer um dos modelos apresentados anteriormente, fixando-se esse tempo em aproximadamente 7 horas e 30 minutos. De forma a garantir que os tempos de execução apresentados nesta secção são o mais fidedignos possível, todos os testes foram efetuados individualmente na mesma máquina (MacBook Pro, 13 polegadas, 2016; Processador: 2,9 GHz Intel core i5; RAM: 8 GB 2133 MHz LPDDR3; Sistema Operativo: macOS Mojave 10.14.6), sendo os únicos processos em execução para além dos processos relacionados com as aplicações do próprio sistema operativo. A contagem do tempo foi iniciada com a primeira variação VIN e terminada com a última variação VUC. O tempo de execução das perguntas originais (OG) não foi considerado.

Este teste permitiu concluir que a não utilização do modelo de *FastText* para extração de características, apesar de reduzir a quantidade de memória necessária para utilizar o modelo, aumenta consideravelmente o tempo de resposta do mesmo, até a um ponto em que a remoção destas características deixa de ser uma alternativa viável.

Variação	Questões #	Top-1		Top-3		Top-5	
		#	%	#	%	#	%
OG	379	252	66.49	306	80.74	321	84.50
VIN	75	58	77.3(3)	67	89.3(3)	73	97.3(3)
VG1	379	332	87.60	371	97.89	373	98.42
VG2	379	311	82.28	359	94.72	365	96.31
VUC	451	257	56.98	349	77.38	366	81.15
Total	1284	958	74.61	1146	89.25	1177	91.6(6)

Tabela 4.4: Desempenho do agente de diálogo quando testado em todas as variações das FAQs, utilizando um novo modelo SST ao qual foram removidas as características textuais baseadas no modelo *FastText*.

Apesar de promissores, os resultados apresentados até ao momento nada dizem sobre a qualidade do agente relativamente a outros sistemas. Por esse motivo, foram desenvolvidas duas baselines em Python com recurso a duas bibliotecas externas, o *Whoosh*<sup>3</sup> e o *Chatterbot*<sup>4</sup>, as quais se apresentam de seguida em maior detalhe.

<sup>3</sup><https://whoosh.readthedocs.io>

<sup>4</sup><https://chatterbot.readthedocs.io/en/stable/index.html>

O *Whoosh* é uma biblioteca que disponibiliza mecanismos para a criação de motores de pesquisa. Esta biblioteca permite customizar a forma como é feita a indexação do texto, a pesquisa, o pré-processamento e armazenamento da informação, a função de score, entre outros parâmetros. Visto que o objetivo da utilização desta biblioteca foi criar uma baseline, procurou-se que esta fosse simples e relativamente rápida de implementar. O primeiro passo para a implementação deste sistema consistiu na criação do índice. Para tal, foi necessário definir os campos que o mesmo iria utilizar. Neste caso foram utilizados dois campos, um para a pergunta e outro para a resposta, dado que o objetivo era fazer a indexação do conjunto de FAQs do BDE. Foi também criado um analisador em português que permitia fazer o pré-processamento dos dados antes da sua indexação com base num conjunto de métodos que atuavam em sequência. Após alguns testes com os métodos disponíveis para customizar o analisador, optou-se por utilizar um método de *stemming*, seguido de um para remover acentos das palavras e, por fim, um para fazer a divisão do texto em n-gramas de caracteres de tamanho 2, 3 e 4, uma vez que estes foram os que permitiram obter o melhor desempenho do sistema. O segundo passo consistiu na definição do mecanismo de pesquisa. Este utilizou os parâmetros por defeito do *Whoosh*, à exceção de um, *FuzzyTerm*, que foi adicionado e que permite lidar com pequenos erros ortográficos, tais como, letras trocadas, na frase de pesquisa. A metodologia adotada para testar este sistema seguiu a mesma abordagem que os agentes baseados nos modelos de regressão: utilizar as variações das interações originais e contabilizar o número de vezes que o sistema consegue devolver a resposta correta. Os resultados obtidos por este sistema apresentam-se na tabela 4.5.

Variação	Questões #	Top-1	
		#	%
OG	379	378	99.74
VIN	75	0	0.00
VG1	379	29	7.65
VG2	379	21	5.54
VUC	451	47	10.42
Total	1284	97	7.55

Tabela 4.5: Desempenho da baseline Whoosh quando testado em todas as variações das FAQs.

Para além do *Whoosh* utilizou-se também o *Chatterbot*, que é uma biblioteca que permite o desenvolvimento de sistemas para a resposta automática a perguntas. Esta biblioteca apresenta uma grande variedade de módulos que podem ser integrados e que permitem definir, entre outras coisas, a forma como os dados podem ser armazenados, as métricas para o cálculo de similaridade entre a pergunta e a resposta e qual a resposta que deve ser escolhida, caso existiam várias opções disponíveis na base de dados do sistema para a pergunta com a qual foi feito o mapeamento. Esta biblioteca não é, contudo, direcionada para o processamento de linguagem natural, pelo que não apresenta métodos para tratamento dos dados. Contudo é flexível o suficiente para permitir a criação de novos módulos que permitem customizar o sistema por completo. Dado que o objetivo do sistema desenvolvido com base no *Chatterbot* é o de, tal como todos os agentes apresentados até ao momento, fazer o mapeamento entre uma interação do utilizador e a resposta correta, apenas foi necessário customizar o algoritmo utilizado para calcular a similaridade do par. Dos algoritmos disponíveis aquele que permitiu obter os melhores resultados foi a distância de *Levenshtein*, e por esse motivo foi o utilizado. A avaliação deste sistema adotou a mesma metodologia que os anteriores, estando os resultados obtidos apresentados na tabela 4.6.

Variação	Questões #	Top-1	
		#	%
OG	379	373	98.42
VIN	75	6	8.00
VG1	379	48	12.66
VG2	379	33	8.71
VUC	451	15	3.33
Total	1284	102	7.94

Tabela 4.6: Desempenho da baseline Chatterbot quando testado em todas as variações das FAQs.

A análise do desempenho das duas baselines apresentadas permite verificar a superioridade dos modelos baseados no cálculo da SST. Ainda que estas consigam calcular a resposta correta mais vezes que os modelos SST quando a interação de entrada é igual à existente na base de conhecimento, caso da Variação *OG*, assim que é introduzida variação lexical e estrutural nas perguntas colocadas, a performance de ambas as baselines diminui drasticamente.

### 4.3 Identificação e Resposta a Perguntas Fora-do-Domínio

Uma limitação comum de agentes de diálogo baseados na Recuperação de Informação é a dificuldade dos mesmos em lidar com interações fora do domínio para o qual foram desenvolvidos. Apesar deste não ser um requisito deste tipo de sistemas, é importante conseguir dar resposta a estas interações, de forma a tornar o diálogo do agente o mais humano possível. Uma possível solução seria recorrer a um pequeno conjunto de respostas-padrão para este tipo de casos, contudo este não deixaria de ser um fator limitador do agente. Por esse motivo, optou-se por recorrer a um corpus de legendas de filmes, uma vez que este agrega um grande conjunto de interações num cenário de comunicação humana.

O primeiro passo para lidar com interações fora-do-domínio passa por conseguir identificá-las. Para esse efeito, foi adotada uma abordagem simples com base na análise dos limiares (*thresholds*, em inglês) de SST. De acordo com o valor de similaridade mais alto calculado entre a interação e as FAQs na base de conhecimento do agente, considera-se fora-do-domínio a interação cujo valor de similaridade ficar abaixo de um determinado limiar  $\theta$  previamente estabelecido. De forma a encontrar um  $\theta$  adequado, foi analisado o seu impacto no dataset utilizado. A tabela 4.7 apresenta o número de respostas corretas para cada limiar de  $\theta$ , entre 1 (completamente diferente) e 5 (equivalente). Como apenas uma minoria das variações (duas para as VIN, uma para as VG2 e VUC) é mapeada de forma correta para um valor de similaridade inferior a 2.5, este foi o valor definido como limiar para definir perguntas fora-do-domínio.

Variação	Questões #	$\theta \geq 2$		$\theta \geq 2.5$		$\theta \geq 3$		$\theta \geq 3.5$		$\theta \geq 4$	
		#	%	#	%	#	%	#	%	#	%
OG	379	364	96.04	364	96.04	364	96.04	364	96.04	364	96.04
VIN	75	59	78.6(6)	57	76.00	54	72.00	46	61.3(3)	24	32.00
VG1	379	317	83.64	317	83.64	312	82.32	282	74.41	194	51.19
VG2	379	313	82.60	312	82.32	305	80.47	279	73.61	172	45.38
VUC	451	273	60.53	272	60.31	249	55.21	181	40.13	82	18.(18)
Total	1284	962	74.92	958	74.61	920	71.65	788	61.37	472	36.76

Tabela 4.7: Desempenho do agente de diálogo quando testado em todas as variações das FAQs por limiar de similaridade, utilizando o modelo SST selecionado.

Após a escolha de  $\theta$ , a resposta a interações do utilizador cuja similaridade com as perguntas do domínio seja inferior ao limiar definido é dada com recurso à secção em português do *Subtle* (Magarreiro, Coheur e Melour, 2014). Este é um corpus de legendas de filmes armazenadas num formato Pergunta-Resposta. Dado o tamanho deste corpus (três milhões de interações) calcular a similaridade, em tempo útil, entre a interação do utilizador e cada uma das frases que o compõem seria impossível. Por esse motivo, optou-se por fazer a indexação do *Subtle* com recurso ao *Whoosh*. Desta forma, sempre que uma interação fora-do-domínio é detetada, é efetuada uma pesquisa no índice de legendas pelas 30 interações mais semelhantes. Posteriormente, é calculada a similaridade semântica, com recurso ao modelo SST integrado no agente de diálogo, entre a interação fora-do-domínio utilizada para fazer a pesquisa e as 30 interações resultantes dessa pesquisa. Por último, é retornada a resposta à pergunta das legendas com a qual foi calculado o maior valor de similaridade. O fraco desempenho apresentado pelo *Whoosh* nos testes efetuados na secção 4.2 foi o que levou a que este fosse utilizado apenas para fazer a pré-seleção das 30 possíveis respostas e não para escolher a resposta final do agente. O índice de legendas e a função de pesquisa foram criados da mesma forma que a baseline *Whoosh*.

Encerra-se este capítulo com a apresentação de uma conversa com o agente de diálogo desenvolvido. As interações do utilizador são identificadas por **Eu:** e as do sistema por **Bot:**. A conversa começa com algumas interações (1-6) fora-do-domínio do agente, as quais são respondidas com recurso ao corpus de legendas. A partir daí (7-19), são colocadas algumas perguntas dentro do domínio do agente. Este responde sempre confirmando se a pergunta com a qual foi feito o mapeamento da interação do utilizador corresponde efetivamente ao que este queria perguntar. Em algumas situações (13, 14, 19), o agente sugere informação que também poderá ser do interesse do utilizador. Estas sugestões são feitas com base em perguntas cuja diferença absoluta de similaridade com a pergunta da resposta devolvida seja inferior a uma percentagem, parametrizável e que no contexto do agente apresentado foi de 10%, da diferença entre o valor máximo e mínimo de similaridade entre a interação do utilizador e as perguntas na base de conhecimento do agente. A conversa termina com mais algumas interações (20-27) fora-do-domínio. Em situações particulares (23), nas quais o agente não consegue encontrar qualquer resposta, é devolvida uma resposta por defeito que sugere ao utilizador para colocar a pergunta de outra forma. A utilização das legendas para lidar com as perguntas fora-do-domínio permite manter um diálogo contínuo e sem interrupções, com o prejuízo de que em diversas situações, tal como se pode observar nesta conversa, a resposta do agente seja completamente descontextualizada (21, 25) e, por vezes, até despropositada (6, 27).



1. **Eu:** Olá tudo bem?
2. **Bot:** Quem é?
3. **Eu:** Sou o José
4. **Bot:** Natalya.
5. **Eu:** Preciso da tua ajuda
6. **Bot:** Estás horrível.
7. **Eu:** É sobre o balcão do empreendedor
8. **Bot:** Se a sua pergunta foi: Para aceder ao Balcão do Empreendedor é preciso ter cartão de cidadão? R: Não é preciso ter cartão de cidadão pois é possível aceder utilizando a Chave Móvel Digital (CMD), um meio alternativo e voluntário de autenticação dos cidadãos nos portais e sítios na Internet da Administração Pública, pela associação do número de identificação civil a um único número de telemóvel e/ou a um único endereço de correio eletrónico.
9. **Eu:** existem empresas que não podem ser configuradas nos contadores empresa no horário
10. **Bot:** Se a sua pergunta foi: Há sociedades que não podem ser constituídas nos balcões "Empresa na Hora"? R: Sim, é o caso das sociedades cuja constituição careça de autorizações prévias para o efeito, das sociedades cujo capital social seja realizado por entradas em espécie, das sociedades anónimas europeias, dos agrupamentos complementares de empresas, das cooperativas e das sociedades civis.
11. **Eu:** o que são sociedades anónimas
12. **Bot:** Se a sua pergunta foi: O que são Sociedades Anónimas? R: Neste tipo de empresa, os sócios limitam a sua responsabilidade ao valor das ações por si subscritas. Assim, os credores sociais só se podem fazer pagar pelos bens sociais. O número mínimo de sócios é de cinco, embora possa ter um único sócio, desde que seja uma sociedade e não um indivíduo. A firma pode ser composta pelo nome (ou firma) de algum ou de todos os sócios, por uma denominação particular ou uma reunião dos dois. Em qualquer dos casos, tem que ser seguida do aditamento obrigatório "Sociedade Anónima" ou abreviado - "S.A.".
13. **Bot:** Também poderá estar interessado em: Exceptuando as sociedades unipessoais, as sociedades por quotas exigem um mínimo de dois sócios. A lei não admite sócios de indústria e todos têm que participar com dinheiro ou com bens avaliáveis em dinheiro. A firma pode ser composta pelo nome ou firma de algum ou de todos os sócios, por uma denominação particular ou uma reunião dos dois e, em qualquer dos casos, tem que ser seguida do aditamento obrigatório "Limitada" por extenso ou abreviado - "L.da".
14. **Bot:** Também poderá estar interessado em: As sociedades em comandita são de responsabilidade mista, uma vez que reúnem sócios cuja responsabilidade é limitada, que contribuem com o capital, e sócios de responsabilidade ilimitada e solidária entre si, que contribuem com bens ou serviços e assumem a gestão e a direcção efetiva da sociedade. Os comanditários e comanditados, respectivamente. Estas empresas devem adoptar uma firma composta pelo nome (completo ou abreviado) ou a firma de pelo menos um dos sócios de responsabilidade ilimitada. É obrigatório o aditamento "em Comandita" ou "& Comandita", para as sociedades em comandita simples e o aditamento obrigatório "em Comandita por Acções" ou "& Comandita por Acções", para as sociedades em comandita por acções.
15. **Eu:** Que medidas de segurança devo aplicar num alojamento local para mais de 10 hóspedes?
16. **Bot:** Se a sua pergunta foi: No caso dos alojamentos com capacidade igual ou inferior a 10 utentes, em termos de requisitos de segurança, qual o extintor e manta mais adequados ao meu alojamento local? R: De acordo com indicação da Autoridade Nacional de Protecção Civil (ANPC), entidade competente para assegurar o cumprimento do regime de segurança contra incêndios em edifícios (SCIE): O produto extintor a utilizar depende do "tipo de fogo", no entanto, o Pó Químico ABC (6kg) ou a Água Aditivada (5 Kg) poderão ser uma boa opção. Quanto à manta, ela deve ser dimensionada de acordo com o dispositivo de queima sobre o qual se pretenda atuar. A ANPC recomenda ainda que, em termos de prevenção, é importante que nestas unidades de alojamento sejam afixadas medidas de prevenção e instruções de segurança.
17. **Eu:** se tiver um estabelecimento com mais que 9 quartos e não conseguir fazer a alteração para empreendimento turístico, tenho de o fechar?
18. **Bot:** Se a sua pergunta foi: O que acontece aos estabelecimentos de alojamento local registados ao abrigo da legislação anterior que tenham mais de 9 quartos e não consigam por algum motivo fazer a sua alteração para empreendimento turístico? Têm de encerrar? R: Não, os limites previstos no artigo 11.º do Decreto-Lei n.º 128/2014, de 29 de agosto, apenas se aplicam aos estabelecimentos de alojamento local registados após a sua entrada em vigor.
19. **Bot:** Também poderá estar interessado em: A cessação da exploração do estabelecimento deve ser comunicada através do balcão único eletrónico no prazo máximo de 60 dias após a sua ocorrência.
20. **Eu:** Qual o sentido da vida?

21. **Bot:** Eu vou a esta sala. Ryko, vai àquela ali.
22. **Eu:** quanto é a raiz quadrada de pi?
23. **Bot:** Desculpe, não percebi, pode colocar a sua questão de outra forma?
24. **Eu:** Obrigado pela ajuda!
25. **Bot:** "Ainda não"? Adoras arriscar, não é?
26. **Eu:** Vou embora
27. **Bot:** Ótimo! Sai da minha vida! Quem precisa de você? Rua!

# Capítulo 5

## Conclusão

Os agentes conversacionais têm vindo a ganhar um papel cada vez mais importante no quotidiano da nossa sociedade, uma vez que permitem que os utilizadores interajam com as máquinas da forma que lhes é mais natural, a linguagem. Por esse motivo, cada vez mais surgem sistemas de apoio ao consumidor, como, por exemplo, chatbots e assistentes pessoais. Os modelos de Similaridade Semântica Textual, tal como demonstrado, podem ser utilizados como uma abordagem viável para o desenvolvimento destes agentes, dado que permitem fazer o mapeamento entre Pergunta-Resposta com base no significado do texto e ainda fazer a identificação de perguntas fora-do-domínio, permitindo criar mecanismos para lidar com as mesmas.

Esta tese pretende ser um contributo quer para o desenvolvimento de modelos para o cálculo da SST, quer para o desenvolvimento de agentes de diálogo, apresentando-se, de seguida, as maiores contribuições:

- Novos modelos de regressão baseados em Python para o cálculo da SST em português, baseados em diferentes algoritmos de aprendizagem e características textuais.
- Estudo das características mais relevantes para o cálculo da SST em português.
- Agente conversacional com base em técnicas de Similaridade Semântica Textual, capaz de identificar e lidar com perguntas fora do domínio específico para o qual foi desenvolvido.
- *Framework* para o desenvolvimento de agentes conversacionais, desde a extração de características à integração do modelo SST num agente de diálogo.

Os modelos desenvolvidos, apesar de não terem atingido o desempenho do estado-da-arte, encontram-se ao nível dos melhores modelos para o cálculo da SST em português (Fonseca et al., 2016). Com o desenvolvimento dos mesmos foi possível estudar a relevância que cada uma das características textuais extraídas tem no desempenho dos modelos, o que representa uma base sólida de conhecimento que permitirá agilizar o desenvolvimento de novos modelos no futuro. Ficou demonstrado que os modelos SST podem ser utilizadas como mecanismo de mapeamento entre Pergunta-Resposta em agentes conversacionais, conseguindo obter resultados substancialmente superiores aos das baselines utilizadas. Estes resultados foram medidos com recurso a um conjunto de variações das perguntas na base de conhecimento do agente, dado que a introdução de variabilidade lexical e estrutural aproximam os testes de um cenário de utilização real do agente, apesar de quanto maior essa variabilidade, mais afetado é o desempenho do agente. Como demonstrado,

os modelos SST são ainda eficazes na identificação de perguntas fora-do-domínio. A utilização do *Whoosh* para responder a este tipo de interações pode resultar em respostas descontextualizadas e desproporcionadas, contudo permite manter um diálogo mais fluído e sem interrupções.

## 5.1 Trabalho futuro

No futuro, tentar-se-á melhorar o desempenho quer do modelo SST, quer do agente de diálogo.

O primeiro foi treinado na coleção ASSIN, que é composta apenas por frases declarativas. Contudo, o facto de ter sido utilizado para calcular a similaridade entre perguntas poderá ter influenciado o seu desempenho. Existe algum trabalho efetuado no sentido de perceber o impacto disto (Rodrigues et al., 2018), no entanto, não existe nenhuma coleção em português, que seja do nosso conhecimento, composta por perguntas com similaridade semântica anotada, como a disponibilizada no âmbito de uma tarefa para o cálculo da SST do SemEval 2016 (Agirre et al., 2016). Dada a sua relevância, poderá ser interessante criar tal coleção.

A segunda edição da avaliação ASSIN irá decorrer ainda este ano, pelo que os modelos SST desenvolvidos com melhor desempenho serão utilizados para participar na mesma.

Para além da SST, um dos objetivos futuros será considerar o contexto da conversa para fazer o mapeamento entre a pergunta do utilizador e a resposta do agente de diálogo.

Apesar do conjunto de FAQs utilizadas estar relacionado com o exercício da atividade económica em Portugal, o agente desenvolvido poderá ser aplicado a outros domínios, desde que a base de conhecimento utilizada seja composta por listas de perguntas e respostas. Poderá também ser aplicado a diferentes domínios ao mesmo tempo, desde que seja criado um mecanismo para a identificação do domínio das interações do utilizador antes de procurar uma resposta. Este mecanismo poderá ser baseado apenas na similaridade da interação com as perguntas do domínio, contudo, para melhor escalabilidade, o ideal seria ter um classificador treinado para identificar o domínio com base em interações curtas.

# Bibliografia

- Agirre, Eneko, Mona Diab, Daniel Cer e Aitor Gonzalez-Agirre (2012). “Semeval-2012 task 6: A pilot on semantic textual similarity”. Em: *Proceedings of the First Joint Conference on Lexical and Computational Semantics- Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*. Association for Computational Linguistics, pp. 385–393.
- Agirre, Eneko, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau e Janyce Wiebe (2016). “SemEval-2016 Task 1: Semantic Textual Similarity, Monolingual and Cross-Lingual Evaluation”. Em: *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*. San Diego, California: ACL Press, pp. 497–511. DOI: 10.18653/v1/S16-1081. URL: <https://www.aclweb.org/anthology/S16-1081>.
- Altman, Naomi S (1992). “An introduction to kernel and nearest-neighbor nonparametric regression”. Em: *The American Statistician* 46.3, pp. 175–185.
- Alves, Ana, Hugo Gonalo Oliveira, Ricardo Rodrigues e Rui Encarnaao (2018). “ASAPP 2.0: Advancing the state-of-the-art of semantic textual similarity for Portuguese”. Em: *7th Symposium on Languages, Applications and Technologies (SLATE 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.
- Alves, Ana Oliveira, Ricardo Rodrigues e Hugo Gonalo Oliveira (2016). “ASAPP: alinhamento semântico automático de palavras aplicado ao português”. Em: *Linguamática* 8.2, pp. 43–58.
- Bär, Daniel, Chris Biemann, Iryna Gurevych e Torsten Zesch (2012). “Ukp: Computing semantic textual similarity by combining multiple content similarity measures”. Em: *Proceedings of the First Joint Conference on Lexical and Computational Semantics- Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*. Association for Computational Linguistics, pp. 435–440.
- Barandiaran, Iñigo (1998). “The random subspace method for constructing decision forests”. Em: *IEEE transactions on pattern analysis and machine intelligence* 20.8.
- Biçici, Ergun (2015). “RTM-DCU: Predicting semantic similarity with referential translation machines”. Em:
- Biçici, Ergun, Declan Groves e Josef van Genabith (2013). “Predicting sentence translation quality using extrinsic and language independent features”. Em: *Machine Translation* 27.3-4, pp. 171–192.
- Biçici, Ergun e Andy Way (2014). “RTM-DCU: Referential translation machines for semantic similarity”. Em: *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pp. 487–496.
- Bird, Steven, Ewan Klein e Edward Loper (2009). *Natural Language Processing with Python*. O’Reilly Media.
- Bojanowski, Piotr, Edouard Grave, Armand Joulin e Tomas Mikolov (2016). “Enriching Word Vectors with Subword Information”. Em: *arXiv preprint arXiv:1607.04606*.

- Bojanowski, Piotr, Edouard Grave, Armand Joulin e Tomas Mikolov (2017). “Enriching word vectors with subword information”. Em: *Transactions of the Association for Computational Linguistics* 5, pp. 135–146.
- Cer, Daniel, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio e Lucia Specia (2017). “SemEval-2017 Task 1: Semantic Textual Similarity-Multilingual and Cross-lingual Focused Evaluation”. Em: *arXiv preprint arXiv:1708.00055*.
- Chang, Chih-Chung e Chih-Jen Lin (2011). “LIBSVM: A library for support vector machines”. Em: *ACM Transactions on Intelligent Systems and Technology* 2 (3). Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>, 27:1–27:27.
- Colby, Kenneth Mark, Franklin Dennis Hilf, Sylvia Weber e Helena C Kraemer (1972). “Turing-like indistinguishability tests for the validation of a computer simulation of paranoid processes”. Em: *Artificial Intelligence* 3, pp. 199–221.
- Cortes, Corinna e Vladimir Vapnik (1995). “Support-vector networks”. Em: *Machine learning* 20.3, pp. 273–297.
- De Marneffe, Marie-Catherine e Christopher D Manning (2008). *Stanford typed dependencies manual*. Rel. téc. Technical report, Stanford University.
- Faruqui, Manaal, Jesse Dodge, Sujay K. Jauhar, Chris Dyer, Eduard Hovy e Noah A. Smith (2015). “Retrofitting Word Vectors to Semantic Lexicons”. Em: *Proceedings of NAACL*.
- Ferreira, João, Hugo Gonçalo Oliveira e Ricardo Rodrigues (2019). “Improving NLTK for Processing Portuguese”. Em: *Symposium on Languages, Applications and Technologies (SLATE 2019)*. In press.
- Fialho, Pedro, Ricardo Marques, Bruno Martins, Luisa Coheur e Paulo Quaresma (2016). “INESCID at ASSIN: Measuring semantic similarity and recognizing textual entailment”. Em: *Linguamática* 8.2, pp. 33–42.
- Firth, John Rupert (1957). *A Synopsis of Linguistic Theory 1930-1955 in Studies in Linguistic Analysis*, Philological Society.
- Fonseca, E, L Santos, Marcelo Criscuolo e S Aluisio (2016). “ASSIN: Avaliacao de similaridade semantica e inferencia textual”. Em: *Computational Processing of the Portuguese Language-12th International Conference, Tomar, Portugal*, pp. 13–15.
- Freund, Yoav e Robert E Schapire (1997). “A decision-theoretic generalization of on-line learning and an application to boosting”. Em: *Journal of computer and system sciences* 55.1, pp. 119–139.
- Friedman, Jerome H (2001). “Greedy function approximation: a gradient boosting machine”. Em: *Annals of statistics*, pp. 1189–1232.
- Ganitkevitch, Juri, Benjamin Van Durme e Chris Callison-Burch (2013). “PPDB: The paraphrase database”. Em: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 758–764.
- Gonçalo Oliveira, Hugo (2018a). “A Survey on Portuguese Lexical Knowledge Bases: Contents, Comparison and Combination”. Em: *Information* 9.2. ISSN: 2078-2489. DOI: 10.3390/info9020034. URL: <https://doi.org/10.3390/info9020034>.
- (2018b). “Learning Word Embeddings from Portuguese Lexical-Semantic Knowledge Bases”. Em: *Computational Processing of the Portuguese Language - 13th International Conference, PROPOR 2018, Canela, Brazil, September 24-26, 2018, Proceedings*. Vol. 11122. LNCS. Springer, pp. 265–271. DOI: 10.1007/978-3-319-99722-3\\_27. URL: [https://doi.org/10.1007/978-3-319-99722-3\\\_27](https://doi.org/10.1007/978-3-319-99722-3\_27).
- Grave, Edouard, Piotr Bojanowski, Prakhar Gupta, Armand Joulin e Tomas Mikolov (2018). “Learning Word Vectors for 157 Languages”. Em: *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

- Grover, Aditya e Jure Leskovec (2016). “node2vec: Scalable feature learning for networks”. Em: *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, pp. 855–864.
- Han, Lushan, Abhay L Kashyap, Tim Finin, James Mayfield e Jonathan Weese (2013). “UMBC\_EBIQUITY-CORE: semantic textual similarity systems”. Em: *Second Joint Conference on Lexical and Computational Semantics (\*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*. Vol. 1, pp. 44–52.
- Hänig, Christian, Robert Remus e Xose De La Puente (2015). “Exb themis: Extensive feature extraction from word alignments for semantic textual similarity”. Em: *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pp. 264–268.
- Harris, Zellig S (1954). “Distributional structure”. Em: *Word* 10.2-3, pp. 146–162.
- Hartmann, Nathan S., Erick R. Fonseca, Christopher D. Shulby, Marcos V. Treviso, Jéssica S. Rodrigues e Sandra M. Aluísio (2017). “Portuguese Word Embeddings: Evaluating on Word Analogies and Natural Language Tasks”. Em: *Proceedings the 11th Brazilian Symposium in Information and Human Language Technology*. STIL 2017.
- Hartmann, Nathan Siegle (2016). “Solo Queue at ASSIN: Mix of a traditional and an emerging approaches”. Em: *Linguamática* 8.2, pp. 59–64.
- Ho, Tin Kam (1995). “Random decision forests”. Em: *Proceedings of 3rd international conference on document analysis and recognition*. Vol. 1. IEEE, pp. 278–282.
- Holmes, Geoffrey, Mark Hall e Eibe Prank (1999). “Generating rule sets from model trees”. Em: *Australasian Joint Conference on Artificial Intelligence*. Springer, pp. 1–12.
- Joos, Martin (1950). “Description of language design”. Em: *The Journal of the Acoustical Society of America* 22.6, pp. 701–707.
- Joulin, Armand, Edouard Grave, Piotr Bojanowski e Tomas Mikolov (2016). “Bag of tricks for efficient text classification”. Em: *arXiv preprint arXiv:1607.01759*.
- Jurafsky, Daniel e James H. Martin (2009). *Speech and Language Processing (2Nd Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc. ISBN: 0131873210.
- Kim, Eric (2013). “Everything you wanted to know about the kernel trick”. Em: *URL: [http://www.eric-kim.net/eric-kim-net/posts/1/kernel\\_trick.html](http://www.eric-kim.net/eric-kim-net/posts/1/kernel_trick.html)*.
- Le, Quoc e Tomas Mikolov (2014). “Distributed representations of sentences and documents”. Em: *International Conference on Machine Learning*, pp. 1188–1196.
- Liu, Jian-Guo, Xiao-Li Zhang e Wei-Ping Wu (2006). “Application of fuzzy neural network for real estate prediction”. Em: *International Symposium on Neural Networks*. Springer, pp. 1187–1191.
- MacKay, David JC (1998). “Introduction to Gaussian processes”. Em: *NATO ASI Series F Computer and Systems Sciences* 168, pp. 133–166.
- Magarreiro, Daniel, Luísa Coheur e Francisco S. Melour (2014). “Using subtitles to deal with Out-of-Domain interactions”. Em: *Proceedings of 18th Workshop on the Semantics and Pragmatics of Dialogue (SemDial)*, pp. 98–106.
- Marelli, Marco, Luisa Bentivogli, Marco Baroni, Raffaella Bernardi, Stefano Menini e Roberto Zamparelli (2014). “Semeval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment”. Em: *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pp. 1–8.
- Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado e Jeff Dean (2013a). “Distributed representations of words and phrases and their compositionality”. Em: *Advances in neural information processing systems*, pp. 3111–3119.
- Mikolov, Tomas, Kai Chen, Greg Corrado e Jeffrey Dean (2013b). “Efficient estimation of word representations in vector space”. Em: *arXiv preprint arXiv:1301.3781*.

- Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot e E. Duchesnay (2011). “Scikit-learn: Machine Learning in Python”. Em: *Journal of Machine Learning Research* 12, pp. 2825–2830.
- Pennington, Jeffrey, Richard Socher e Christopher D. Manning (2014). “GloVe: Global Vectors for Word Representation”. Em: *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543. URL: <http://www.aclweb.org/anthology/D14-1162>.
- Quinlan, John R et al. (1992). “Learning with continuous classes”. Em: *5th Australian joint conference on artificial intelligence*. Vol. 92. World Scientific, pp. 343–348.
- Řehůřek, Radim e Petr Sojka (2010). “Software Framework for Topic Modelling with Large Corpora”. English. Em: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. <http://is.muni.cz/publication/884893/en>. Valletta, Malta: ELRA, pp. 45–50.
- Rodrigues, João, Chakaveh Saedi, António Branco e João Silva (2018). “Semantic Equivalence Detection: Are Interrogatives Harder than Declaratives?”. Em: *Proceedings of the 11th Language Resources and Evaluation Conference*. Miyazaki, Japan: ELRA. URL: <https://www.aclweb.org/anthology/L18-1513>.
- Rodrigues, Ricardo, Hugo Gonçalo Oliveira e Paulo Gomes (2018). “NLPPort: A Pipeline for Portuguese NLP (Short Paper)”. Em: *7th Symposium on Languages, Applications and Technologies (SLATE 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.
- Šarić, Frane, Goran Glavaš, Mladen Karan, Jan Šnajder e Bojana Dalbelo Bašić (2012). “Takelab: Systems for measuring semantic text similarity”. Em: *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*. Association for Computational Linguistics, pp. 441–448.
- Speer, Robyn, Joshua Chin e Catherine Havasi (2017). “ConceptNet 5.5: An Open Multilingual Graph of General Knowledge”. Em: pp. 4444–4451. URL: <http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14972>.
- Sultan, Md Arafat, Steven Bethard e Tamara Sumner (2015). “DLS @ CU: Sentence Similarity from Word Alignment and Semantic Vector Composition”. Em: *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pp. 148–153.
- Tian, Junfeng, Zhiheng Zhou, Man Lan e Yuanbin Wu (2017). “ECNU at SemEval-2017 Task 1: Leverage Kernel-based Traditional NLP features and Neural Networks to Build a Universal Model for Multilingual and Cross-lingual Semantic Textual Similarity”. Em: *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pp. 191–197.
- Verikas, Antanas, Evaldas Vaiciukynas, Adas Gelzinis, James Parker e M Charlotte Olsson (2016). “Electromyographic patterns during golf swing: Activation sequence profiling and prediction of shot effectiveness”. Em: *Sensors* 16.4, p. 592.
- Weizenbaum, Joseph (1966). “ELIZA—A Computer Program for the Study of Natural Language Communication Between Man and Machine”. Em: *Commun. ACM* 9.1, pp. 36–45. ISSN: 0001-0782. DOI: 10.1145/365153.365168. URL: <http://doi.acm.org/10.1145/365153.365168>.
- Witten, Ian H, Eibe Frank, Mark A Hall e Christopher J Pal (2016). *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann.
- Wu, Hao, Heyan Huang, Ping Jian, Yuhang Guo e Chao Su (2017). “BIT at SemEval-2017 Task 1: Using semantic information space to evaluate semantic textual similarity”. Em: *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pp. 77–84.



Zarrella, Guido, John Henderson, Elizabeth M Merkhofer e Laura Strickhart (2015). “MITRE: Seven systems for semantic similarity in tweets”. Em: *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pp. 12–17.