

Mestrado em Engenharia Informática
Estágio
Relatório Final

Enhancing chat bots using machine learning

Tiago Rodrigues
tmrodr@student.dei.uc.pt

Orientador(DEI):
Professor Doutor Hugo Oliveira

Orientador(WIT):
Ricardo Loureiro
Data: 23-01-2018



FCTUC DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Mestrado em Engenharia Informática
Estágio
Relatório Final

Enhancing chat bots using machine learning

Tiago Rodrigues
tmrodr@student.dei.uc.pt

Orientador(DEI):
Professor Doutor Hugo Oliveira

Orientador(WIT):
Ricardo Loureiro

Júri Arguente:
Professor Doutor Henrique Madeira

Júri Vogal:
Professor Doutor Carlos Nuno Laranjeiro

Data: 23-01-2018



FCTUC DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Agradecimento

Gostaria de agradecer ao meu orientador Dr. Hugo Oliveira, pela paciência e pelos conselhos dados durante este processo.

O meu agradecimento ainda ao Dr. Henrique Madeira e ao Dr. Carlos Nuno Laranjeiro, pelas críticas construtivas que ajudaram a moldar este trabalho e a torná-lo mais claro.

À WIT-Software, em especial nas pessoas de Ricardo Loureiro e Pedro Andrade, pela oportunidade de estagiar na área de sistemas inteligentes e num projeto tão relevante e atual.

Aos colegas na WIT-Software com quem troquei ideias, e que certamente tiveram um impacto no trabalho final apresentado.

Ao Joel Bernardo, pela revisão deste documento.

À minha família, por todo o apoio dado ao longo deste percurso.

E a Deus, pela sua Graça.

A todos muito obrigado.

Tiago Rodrigues

Resumo

A importância de um bom atendimento ao cliente implica por parte das grandes empresas na área das telecomunicações a subcontratação de *call centers*, o que se traduz numa grande complexidade ao nível da gestão de recursos e informação, com um custo associado elevado.

A tipologia de perguntas dos consumidores realizadas aos *call centers* segue aproximadamente o princípio de Pareto, onde um pequeno número de perguntas corresponde a uma grande quantidade de casos. Neste trabalho, exploramos a possibilidade de criar um sistema híbrido Humano/Máquina que seja capaz de responder de forma autónoma e automática às perguntas mais frequentes, bem como às perguntas mais exigentes, recorrendo sempre que necessário à ajuda do assistente humano. Isto permite reduzir de forma drástica a carga de assistência do *call center*, e aliviar focos de frustração entre cliente e assistente durante as sessões de apoio.

Com a possibilidade de aprender através da sua utilização frequente, o sistema desenvolvido será ainda capaz de responder a um número cada vez maior de perguntas.

Para atingirmos este objetivo, exploramos vários modelos de aprendizagem e vários conjuntos de *features* para tentarmos encontrar o modelo que melhor se adequa às necessidades das empresas.

Dada a natureza híbrida da solução com os modelos desenvolvidos é possível responder a 80% das questões mantendo a taxa de erro nos 0%

Com este sistema, damos resposta a duas necessidades do mercado. Primeiramente, oferecemos ao utilizador uma forma de ter as suas questões respondidas em formato de texto, algo que os estudos de mercado apontam como preferencial e como tendência atual. Em segundo lugar, apresentamos às empresas um plano de assistência mais económico e com gestão menos complexa.

Keywords: Call Center , Apoio ao cliente, Learning Model, Machine Learning, Similaridade Semântica, SVM, NLP

Abstract

A good customer support is essential for big companies in the telecommunications sector and it frequently entails difficulties subcontracting call centers, specifically issues managing human resources and great volumes of information, all of which end up surging up the costs.

The type of question asked to the call center follows the Pareto principle, where a small subset of questions appears in a big number of the calls. In this work, we try to explore the possibility of using a hybrid system, as Human/Machine. This would be able to answer frequent questions automatically, as well as difficult questions with the help of the human expertise, thus relieving the workload on the call center.

With the possibility to learn from its frequent usage, the system that's developed will be able to have a greater capacity at answering an ever increasing number of questions.

To achieve this goal, we explore some learning models and try different sets of features, in order to find a suitable model that can respond to the companies' needs. Taking into account the hybrid nature of the solution, following the models developed, it's possible to answer to about 80% of the questions, and still have an error percentage of 0%.

With this system, we tackle two big needs of the market. First by providing the user with a way to have his questions answered in a textual manner, something a market study has shown the users are increasingly demanding and that is on trend. And secondly, by presenting companies with a less expensive and less complex support plan.

Keywords: Call Center ,Customer support, Learning Model, Machine Learning, Semantic similarity, SVM, NLP

Conteúdo

1	Introdução	17
1.1	Motivação	17
1.2	Enquadramento	18
1.3	Abordagem	19
1.4	Contribuições	19
1.5	Estrutura do documento	19
2	Conceitos Teóricos	21
2.1	Natural Language Processing	21
2.1.1	Corpus	22
2.1.2	Tokenization	22
2.1.3	Stemming e Lemmatization	22
2.1.4	Part-of-speech tagging	23
2.1.5	Word vectors	23
2.2	Supervised Machine Learning	23
2.2.1	Neural Network	23
2.2.2	SVM	24
2.3	Similaridade semântica	25
2.3.1	SemEval	25
3	Estado da Arte	27
3.1	Competidores	27
3.1.1	agent.ai	27
3.1.2	Homegrown Software	28
3.2	Similaridade semântica	28
3.2.1	Natural Language Processing	28
3.2.2	Learning Models	28

4	Procedimento experimental	31
4.1	Datasets	31
4.1.1	SEMEVAL	31
4.1.2	Vodafone FAQs Dataset	32
4.1.3	Quora Dataset	32
4.1.4	Stanford Natural Language Inference	33
4.2	Seleção da NLP toolkit	33
4.3	Modelos de similaridade semântica	34
4.3.1	Prova de Conceito	34
4.3.2	Modelo 1	35
4.3.3	Modelo 2	35
4.3.4	Modelo 3	36
4.3.5	Modelo 4	36
4.4	Ambiente de apoio ao desenvolvimento	36
4.5	Teste dos modelos	36
5	Modelos	37
5.1	Prova de Conceito	37
5.1.1	Dataset	37
5.1.2	Features	38
5.1.3	SVM Classifier	41
5.2	Modelo 1	42
5.2.1	Features	42
5.2.2	Treino do modelo	45
5.3	Modelo 2	45
5.3.1	Treino do Modelo	46
5.4	Modelo 3	47
5.5	Modelo 4	47
6	Ambiente de suporte ao desenvolvimento	49
6.1	Funcionamento	49
6.1.1	Estrutura da informação	49
6.1.2	Integração na WIT Bot Platform	52
6.2	Tecnologias	55
7	Resultados	57
7.1	Prova de conceito	58
7.2	Modelo 1	58
7.3	Modelo 2	60

7.4	Modelo 3	61
7.5	Modelo 4	62
8	Conclusão	63
	Apêndice A Resultados SemEval 2016 Task 3 Subtask B	67

Lista de Tabelas

4.1	Comparação das funcionalidades oferecidas pelas <i>toolkits</i> em comparação	34
4.2	Resultados da ferramenta de avaliação de performance	34
7.1	Resultados Modelo 1	59
7.2	Resultados Modelo 2	61
7.3	Resultados Modelo 3	62
7.4	Resultados Modelo 4	62

Lista de Figuras

2.1	Demonstração da classificação em classes gramaticais	23
2.2	Demonstração da distância máxima de separação	24
6.1	Sistema a devolver a resposta que mais se aproxima a pergunta efetuada	50
6.2	Formulário de gestão de perguntas e resposta	51
6.3	Formulário de gestão de constraints	52
6.4	Página de gestão da configuração activa	53
7.1	Resultados retornados quando é feita a pergunta "I lost my user- name"	59
7.2	Resultados do avaliador do SemEval 2016 Task B	60
7.3	Resultados do avaliador do SemEval 2016 Task B	61

Lista de acrónimos

- GloVe Global Vectors for Word Representation
- MWC Mobile World Congress
- NLP Natural Language Processing
- PLN Processamento de Linguagem Natural
- POC Proof Of Concept
- POS Part-Of-Speech
- SEMEVAL (International Workshop on) Semantic Evaluation
- SNLI Stanford Natural Language Inference
- SVM Support Vector Machine
- XGB or XGboost eXtreme Gradient Boost

Capítulo 1

Introdução

– O recente investimento das grandes multinacionais no mercado das assistentes pessoais mostra que, e cria uma abertura por parte dos clientes para interagir com a informação de forma inovadora. Os utilizadores, e por sua vez, clientes, estão cada vez mais exigentes. A rapidez no acesso à informação, assim como a relevância da mesma, são aspetos determinantes para decidirem contratar o suporte.

Sabendo que os utilizadores estão cada vez mais familiarizados com *bots* e o seu funcionamento, isso constitui uma excelente oportunidade para dar um passo na direção da resolução de um problema de eficiência de operação e custos no mercado das telecomunicações (TELCO).

1.1 Motivação

O apoio ao cliente é uma área fundamental para a fidelização dos clientes, daí ser vital para as empresas fornecerem este serviço. A maneira mais comum de Apoio ao cliente na área das telecomunicações é a utilização de *call centers* para lidar com as questões dos clientes, mas esta solução tem os seus custos associados. Exige despesas operacionais, de recursos humanos, e limitações recorrentes da necessidade de funcionamento constante.

Quando questionados sobre se preferiam suporte em formato de texto em vez de chamadas telefónicas, 64% dos clientes referiram que o utilizariam se estivesse disponível[11]. Isto permitiria reduzir o número de chamadas no *call center*. Mas para esta redução de chamadas no *call center* poder ser traduzida numa redução de custos, a solução alternativa teria de ser necessariamente mais barata. Para tal, o nosso objetivo será automatizar a resposta aos pedidos de

assistência via texto o mais possível, sem prejudicar a qualidade da assistência oferecida ao cliente.

1.2 Enquadramento

Este relatório foi elaborado no contexto do mestrado em Engenharia Informática na área de sistemas inteligentes, durante o estágio inserido no programa curricular. O estágio teve lugar na WIT Software, S.A. com sede em Taveiro.

O trabalho enquadra-se numa solução de suporte ao cliente inteligente da WIT que permitirá reduzir os custos associados a esta atividade para empresas prestadoras de serviços.

O meu objetivo foi implementar um sistema que fosse capaz de interpretar uma pergunta do cliente, e procurar entre as respostas que está habilitado a fornecer qual a que mais se adequa à pergunta realizada, (com base no cálculo do índice de confiança associado a cada resposta) a este sistema é chamado de *knowledgebase*.

Existe um sistema que verifica se a confiança associada fica aquém de um determinado limiar. Se esse cenário se verificar, então existe um mecanismo que tratará de reconhecer essa discrepância e de reencaminhar a conversação para um operador.

Neste caso, existem vários níveis de ação que o operador pode tomar.

- Pode verificar que a resposta que o sistema sugeriu está de fato certa e validá-la.
- Pode selecionar entre aquelas que o sistema não escolheu manualmente, caso a resposta sugerida esteja errada. Contudo a resposta correta existe no sistema.
- Pode ter de inserir a resposta manualmente caso a pergunta ainda não esteja no sistema

Em qualquer um dos casos, quando um operador intervém, é gerada informação adicional que pode ser usada pelo sistema para o treinar e enriquecer os dados. Se a resposta já estiver no sistema, então é adicionada a pergunta tal como foi escrita pelo cliente como mais uma forma de apresentar aquela resposta. Se ainda não estiver no sistema, o operador poderá adicionar a resposta, assim como um conjunto de perguntas que a devem desencadear.

1.3 Abordagem

Durante o decorrer do estágio foi necessário decidir como usar processamento de linguagem natural para melhorar a experiência com *chat bots*, em coordenação com os responsáveis foi decidido que a maneira mais óbvia de abordar o problema seria fazer uso de métodos de similaridade semântica (*semantic similarity*), ou seja, métodos automáticos que têm como objectivo quantificar a proximidade entre o significado de dois fragmentos de texto. Com esta abordagem é possível fazer correspondência uma pergunta feita pelo cliente com todas as perguntas já inseridas no sistema. A expectativa é que uma resposta correta para uma pergunta, também será válida para uma pergunta semanticamente equivalente.

A semelhança semântica utiliza ferramentas de processamento de linguagem para extrair características das frases e modelos de aprendizagem, com o objetivo de criar um modelo que permita classificar se duas frases são semanticamente equivalentes.

1.4 Contribuições

Durante o estágio foi possível desenvolver modelos que forma usados em POC (*Proof of concept*) demonstrados em feiras como a Mobile World Congress (MWC) de Barcelona e de Shangai. Entre as principais contribuições deste trabalho destacam-se um sistema integrado no eco-sistema de bots da WIT que melhore a performance com o tempo e a utilização. Temos a noção que o nossa solução vai errar. Porém, um dos objetivos deste trabalho é criar uma solução capaz de errar com o menos prejuízo possível para o cliente. Pretendemos por isso treinar o sistema para que este esteja *skewed* e não responder quando a certeza é baixa, e nem tentar a solução do *best guess*. Por isso a métrica não estará exclusivamente focado em maximizar o número de respostas corretas, mas principalmente em melhorar a razão entre respostas corretas e o número de falsos positivos.

1.5 Estrutura do documento

O documento está estruturado da seguinte forma

- No capítulo 2 é são introduzidos os conceitos importantes para a compreensão deste trabalho.

- No capítulo 3 é dividido em duas partes numa delas é apresentada a concorrência enquanto na segunda parte é olhado para o estado da arte no referente a processamento de linguagem natural e também no que diz respeito a aprendizagem computacional.
- O capítulo 4 aborda as experiências realizadas mas num ponto de vista introdutório porque estas depois são aprofundadas do capítulo 5 ao 7
- O capítulo 5 detalha o processo de criação dos modelos.
- O capítulo 6 apresenta o sistema envolvente dos modelos.
- O capítulo 7 apresenta os resultados e algumas considerações aos mesmos.

Capítulo 2

Conceitos Teóricos

Este capítulo tem como objectivo introduzir temas específicos ao trabalho e que lhe são estruturais. Por outras palavras são apresentados conceitos de termos que ajudam na compreensão do tema, nomeadamente, Processamento de Linguagem Natural (PLN) e Aprendizagem Computacional (Machine Learning, ML).

A partir de um pedaço de texto é possível extrair informação utilizando o processamento de linguagem natural. Este trabalho assenta, por isso, em dois grandes pressupostos: conseguir extrair informação, e conseguir utilizar a informação extraída. Ter informação por si só não resolve o nosso problema. Torna-se então imperativo tratar e compreender os dados. No nosso caso pretendemos aprender e mudar o comportamento do sistema com a chegada de nova informação, sendo necessários modelos de aprendizagem que nos permitam fazer isso.

2.1 Natural Language Processing

O Processamento de linguagem natural é um conjunto de técnicas que podem ser utilizadas para conseguir extrair informação do texto. Existem diversas técnicas para extrair tipos de informações diferentes.

No contexto deste trabalho são utilizadas ferramentas de processamento de linguagem natural para explorar mais profundamente as frases sujeitas ao teste de semelhança semântica. Utilizamos funções como *tokenzation*, *lemmatization* e POS, entre outras. Cada uma delas está num estágio de maturação distinto.

Todas estas técnicas têm associadas pequenos erros na resposta, o que é problemático, uma vez que o erro é cumulativo quando usado num *pipeline* (o presente caso).

2.1.1 Corpus

Um *corpus* (plural *corpora*) é uma coleção de textos onde os modelos podem ser treinados. Dependendo do que está a ser treinado, pode ser anotado (*supervised learning*), ou pode não necessitar de anotações (*unsupervised learning*).

No caso específico desta solução, são utilizadas ferramentas como o spaCy, que inclui modelos pré-treinados para a execução das tarefas a que se propõe. Estes modelos são pré-treinados num corpus externo à solução da WIT. Para o processo de ensinar a semelhança semântica, são usados dois *datasets* constituídos por pares de frases. São também indicadas se as duas frases são ou não semelhantes.

2.1.2 Tokenization

Dado um documento composto por uma sequência de caracteres, *tokenization* é a tarefa de dividir o documento em partes chamadas *tokens*. [8, p. 22]. Existem casos óbvios de avaliar, como por exemplo:

He kicked the ball → He | kicked | the | ball

Contudo, existem casos onde é mais ambíguo qual deverá ser o resultado:

o'neill → o'neill | o' | neill | o | neill | oneill | neill

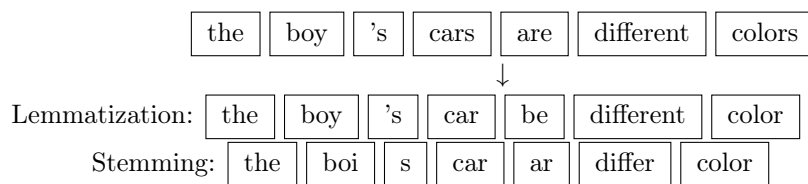
e como diferenciar o caso anterior do seguinte?

aren't → aren't | arent | are | n't | aren | t

2.1.3 Stemming e Lemmatization

Por razões gramaticais, documentos vão usar diferentes formas de uma palavra como *organize*, *organizes*, e *organizing*. Para além disso, existem famílias de derivações de palavras relacionadas com significado semelhante como *democracy*, *democratic* e *democratization*. O objetivo tanto de *stemming* como *lemmatization* é de fazer convergir todas as palavras relacionadas a um termo comum.

Stemming faz isto com heurísticas que removem os sufixos de derivação, embora o resultado final não seja obrigatoriamente uma palavra. Já o *lemmatization* é uma maneira mais cuidada de fazer o mesmo utilizando análise morfológica das palavras e dicionários. Normalmente, ao tentar retirar as terminações e retornar apenas a base ou *lemma*. [8, p. 32]



2.1.4 Part-of-speech tagging

Figura 2.1: Demonstração da classificação em classes gramaticais

the	boy	's	cars	are	different	colors
DT	NN	JJ	NNS	VBP	JJ	NNS

Part-of-speech tagging consiste na classificação de palavras como nomes, verbos, advérbios, ou algo mais específico. Pode-se especificar, por exemplo, o tempo verbal e a pessoa, ou outros, quando essa informação é aplicável. Na imagem 2.1 mostramos um exemplo da classificação gramatical para uma frase de exemplo.

2.1.5 Word vectors

Word vectors é a representação de uma palavra por um vetor a N dimensões, sujeito a um processo de treino não supervisionado. Como é treinado num *corpus* muito extenso, adquire algumas propriedades interessantes, como *clustering* de termos relacionados. Existem varios algoritmos para a criação destes vectores sendo os mais conhecidos GloVe[12] e WordVec[9]

2.2 Supervised Machine Learning

Learning models é o processo onde este trabalho está assente. A necessidade de aprender com exemplos em vez de com modelos *handcrafted* onde é necessário a criação de regras e de exceções, é o que torna este trabalho diferenciador. Para isso são criados modelos que com informação e treino podem aprender a classificar duas frases como semelhantes ou como diferentes.

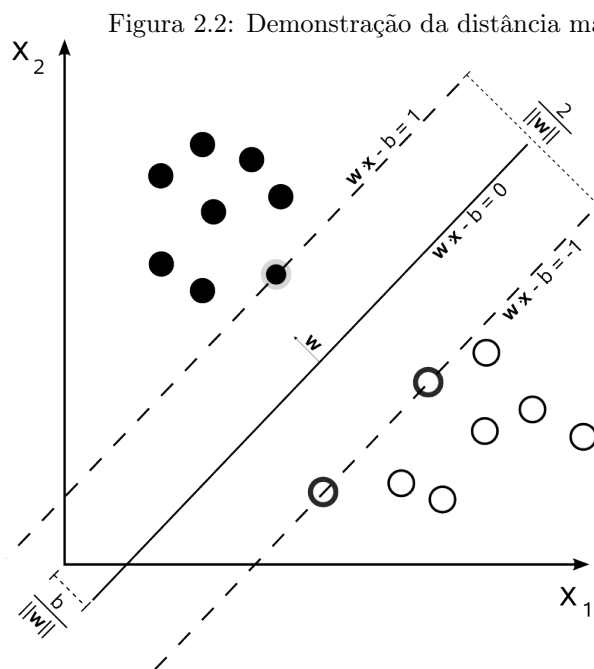
2.2.1 Neural Network

Redes neuronais consiste num conjunto de camadas onde cada camada está densamente conectada à camada seguinte. Cada rede neuronal tem uma camada

de entrada e uma camada de saída, estando separadas por um conjunto de camadas chamado "camadas escondidas". Cada conexão tem um peso associado. No processo de treino, esses pesos são alterados por algoritmos de *backpropagation* com o intuito de convergirem para valores que modelem para casos nunca vistos.

2.2.2 SVM

Support Vector Machine é uma abordagem que tenta maximizar a distância entre duas classes. Para isso usa o método da "estrada mais larga". Como podemos ver na figura 2.2, existem duas classes, e o objetivo neste caso a duas dimensões é encontrar a linha que separe as classes, e que maximize a distância ao ponto mais próximo de cada classe.



Fonte: https://commons.wikimedia.org/wiki/File:Svm_max_sep_hyperplane_with_margin.png (Retirado no dia 19 de Dezembro de 2017)

XGBOOST

Extreme Gradient Boosting é um processo supervisionado e iterativo para a construção de um classificador. Composto por vários classificadores mais simples. Utiliza o método de gradiente descendente para decidir qual o conjunto de classificadores simples que fazem parte da solução.

BI-LSTM

BI-LSTM é um processo supervisionado de aprendizagem. Para além dos resultados este destacou-se pela forma diferenciada das suas features. Enquanto os modelos anteriores as features têm de ser de tamanho fixo e numéricas a BI-LSTM é alimentado a partir dos *word vectors* de cada palavra na frase. Recebendo iterativamente um de cada vez e mudando o seu estado interno com a chegada de cada palavra.

2.3 Similaridade semântica

Semelhança semântica é a medida que quantifica a proximidade semântica entre dois pedaços de texto, e constitui por isso uma disciplina da análise semântica. Para isso recorre-se ao uso de técnicas de NLP e de *Machine learning* para a construção de modelos que permitam comparar se duas frases são ou não semelhantes.

2.3.1 SemEval

O SemEval é um workshop internacional em avaliação semântica, que todos os anos abre tarefas relacionadas com avaliação semântica para promover a investigação dentro da área. Várias conferências já ocorreram e várias tarefas já foram avaliadas mas no contexto deste relatório a tarefa 3B do SemEval 2016 [10] é a mais relevante. A tarefa é sobre modelação de semelhança semântica entre perguntas que é exatamente a abordagem que iremos seguir.

Capítulo 3

Estado da Arte

No estudo do estado da arte este capítulo centra-se em duas frentes por um lado podemos analisar que soluções completas existentes que fazem o que nos propomos realizar os quais são os competidores diretos. Por outro lado, podemos analisar o estado das técnicas utilizadas para nós desenvolvermos a solução.

3.1 Competidores

Quando analisamos o projeto como um todo em vez de nos centrarmos apenas nos módulos de similaridade semântica o numero de soluções existentes reduz drasticamente, por isso vale a pena não só analisar a concorrência existente mas também a possibilidade de empresas criarem o seu próprio software a partir das ferramentas que têm sido disponibilizadas

3.1.1 agent.ai

Agent.ai é uma solução que permite criar uma plataforma para apoio ao cliente com "*AI Automated Agent Responses*" e com "*AI Agent-Assisting Responses*", de fácil integração em todas as plataformas de *messaging*. Embora seja difícil avaliar quais as tecnologias utilizadas, é afirmado que aprende com a experiência, e por isso necessariamente algo que não funciona apenas *rules-based*. Tem um frontend apelativo e intuitivo.

No contexto do estágio foi impossível fazer a utilização do produto devido ao modelo de negócio da empresa em causa, no processo de pesquisa dos competidores.

3.1.2 Homegrown Software

Devido ao facto dos clientes a quem esta solução se dirige serem empresas tecnológicas, podemos ter de considerar a possibilidade de os clientes preferirem criar as suas próprias soluções em vez de recorrer á solução desenvolvida. Isto torna-se possível porque empresas como o Facebook estão a libertar em regime de código aberto soluções que fazem parte do trabalho difícil.

3.2 Similaridade semântica

A similaridade semântica, assim como outros campos na área do processamento da linguagem natural, tem evoluído muito com o ressurgimento de redes neuronais e com o surgimento de *Deep Neural Network*. Isto cria necessariamente uma dicotomia do estado da arte no momento do início da tese e do seu estado atual. Enquanto os melhores modelos durante o SemEval 2016 faziam recurso a tecnologias como as descritas no modelo 1[7], durante o estágio foi possível observar que os resultados obtidos pela utilização de *word embeddings* como o gloVe[12] ultrapassaram os métodos mais tradicionais, dando um novo fôlego a toda a área de processamento de linguagem natural.[13]

3.2.1 Natural Language Processing

Para o processamento de linguagem natural é utilizado o spaCy, uma ferramenta com um compromisso interessante entre precisão e eficiência, sem prescindir da facilidade de uso[5]. Foi escolhido depois de comparado com openNLP e NLTK. Se a escolha fosse feita no momento da escrita do relatório, consideraria adicionar à comparação Parsey McParseface, da Google. Mas por ser um produto muito recente, não houve possibilidade de análise no momento da comparação.

Por ser um *toolkit* bastante completo, seleccionei o spaCy. Ele implementa todas as *features* necessárias, e apresenta uma implementação de algoritmos eficientes sobre todas as áreas, pode ser visto os resultados da comparação no capítulo 4.2 do presente relatório

3.2.2 Learning Models

Relativamente à semelhança semântica, as abordagens que conseguiram melhores resultados no SEMEVAL 2016 usavam SVM_{rank} como classificador[7]. Como o exercício que estamos a tentar resolver era equivalente à estratégia passou por tentar usar o mesmo classificado. Contudo testamos com sucesso

o XGBoost e apercebemos da melhoria dos resultados XGBoost por isso alteramos o modelo de aprendizagem. Como referido anteriormente também os modelos mais tradicionais foram ultrapassados por algoritmos como BI-LSTM recorrentes da nova revitalização.

Capítulo 4

Procedimento experimental

Neste capítulo descrevem-se as principais experiências que suportaram as decisões tomadas durante este trabalho e tentamos explicar os detalhes da implementação da nossa ferramenta que permite o teste da similaridade semântica. Este capítulo dá-nos a conhecer os detalhes dos *dataset* e da tomada de decisão relativa a ferramenta utilizada e introduz os temas discutidos nos capítulos seguintes dos modelos e dos resultados.

4.1 Datasets

Na generalidade dos projetos de Inteligência Artificial, a qualidade do projeto está altamente dependente da qualidade da informação disponível. Como este projeto não tem como objetivo automatizar um processo implementado e em funcionamento, não existe um histórico de utilização que possa ser utilizado como ferramenta de ensino para os modelos. Por este motivo, foi construído um *dataset* a partir da Vodafone FAQs. Este *dataset* é usado para a fase de teste. Foram utilizados outros dois *datasets*: um criado para um concurso da Quora¹, e outro designado *Stanford Natural Language Inference (SNLI)*[4]², que é o da Universidade de Standford. Este tem sido utilizado em investigação na área de NLP desde que foi criado.

4.1.1 SEMEVAL

Durante a fase de aquisição de conhecimento e de familiarização com a área foi usado o dataset do SEMEVAL. A vantagem da utilização deste dataset era

¹<https://www.kaggle.com/c/quora-question-pairs>

²<https://nlp.stanford.edu/projects/snli/>

a quantidade de literatura que tinha associada e a possibilidade de comparar os resultados com a literatura. Mais especificamente, foi utilizado o *dataset* do SEMEVAL 2016 task 3 [1], com especial foco na *subtask* B que consiste na semelhança entre perguntas, sendo que esta constitui a base do nosso trabalho. O *dataset* está estruturado em *post* (cada *post* é constituído por um título e por um corpo), e para cada um deles existem dez *posts* associados, juntamente com a nota referente à semelhança entre eles.

4.1.2 Vodafone FAQs Dataset

Com o objetivo de fazer provas de conceito dentro do âmbito e da área dos clientes a quem iria ser demonstrado, foi-me pedido que criasse um *dataset* a partir das Vodafone FAQs. Para isso selecionamos categorias específicas das Vodafone FAQ retiramos todas as questões e compilamos um Dataset dentro da área em que o projecto iria ser demonstrado, Este *dataset* simularia o funcionamento de um sistema já em operação, onde cada questão tem mais que uma maneira de fazer a pergunta. Num cenário de utilização real é possível que cada pergunta seja feita de várias maneiras, devido à natureza híbrida da solução. Foram atribuídas quatro maneiras possíveis de fazer a mesma pergunta, e uma adicional que não é carregada para a *knowledgebase*, mas que em vez disso é usada para teste. Este dataset foi criado apenas por duas pessoas o que significa que não é uma amostra representativa do conjunto de pessoas que irá fazer perguntas.

4.1.3 Quora Dataset

O *dataset* da Quora foi disponibilizado para um concurso da Kaggle. Consiste em pares de frases, e inclui a avaliação se estas têm ou não o mesmo significado. Este *corpus* é constituído por mais de 400 000 pares de frases marcadas como equivalentes ou como diferentes.

Este conjunto de frases é sobre os mais variados temas, e não se centra exclusivamente em telecomunicações.

Exemplo 1:

”What is the step by step guide to invest in share market in india”

”What is the step by step guide to invest in share market?”

Diferentes

Exemplo 2:

”How should I prepare for CA final law?”

”How one should know that he/she completely prepare for CA final exam?”

Semelhantes

4.1.4 Stanford Natural Language Inference

O *corpus* fornecido pela Universidade de *Stanford* classifica cada par de frases em três categorias diferentes: contradizentes, condizentes ou neutras. É considerado ainda um passo intermédio, por forma a que as condizentes passem a ser a classe positiva e todas as outras frases representem a classe negativa.

Exemplo 1

”A black race car starts up in front of a crowd of people. contradiction”

”A man is driving down a lonely road.”

Contradizentes

Exemplo 2

”An older and younger man smiling.”

”Two men are smiling and laughing at the cats playing on the floor”

Neutras

Exemplo 3

”A soccer game with multiple males playing.”

”Some men are playing a sport.”

Sinónimos

4.2 Seleção da NLP toolkit

Para escolher qual a ferramenta de NLP que seria usada durante o estágio foram submetidos três *toolkits*, (spaCy, NLTK[3], OpenNLP[2]) a teste. Estudando as ferramentas pareceu-me desde logo pela simplicidade de usar, pelo facto de ser um projeto melhor organizado e pelo facto de implementar mais funcionalidades, ver tabela 4.1, que a minha preferência iria recair sobre a *toolkit* spaCy, contudo era necessário analisar e perceber se o spaCy não ficava significativamente atrás das outras soluções existentes na questão da performance dos modelos. Foram testados *tokenization*, *lemmatization* e POS. Não foi testada a *dependency tree*, porque nem o NLTK nem o openNLP a implementam.

Foi construído um avaliador que dado um texto com *golden standard* incluído, utilizaria cada um dos *toolkits* anteriores para gerar uma solução a ser comparada com o *golden standard*. Para isso teve de se ter em conta um erro sistemático devido a implementações de standards diferentes das incluídas com o texto. Na prática, trata-se de um exercício de tradução para um standard comum a todos. Depois são criadas matrizes de confusão para cada um dos *toolkits*, bem como os totais de erro para cada uma das categorias. Na tabela

Tabela 4.1: Comparação das funcionalidades oferecidas pelas *toolkits* em comparação

	SpaCy	OpenNLP	NLTK
Tokkenization	Sim	Sim	Sim
Lemmatization	Sim	Sim	Sim
POS parser	Sim	Sim	Sim
Dependency parser	Sim	Não	Não
Entities Recognition	Sim	Sim	Sim
Co-reference	Não	Sim	Não
WordVec Integrado	Sim (gloVe)	Não	Não
Processamento paralelo integrado	Sim	Não	Não

Tabela 4.2: Resultados da ferramenta de avaliação de performance

	SpaCy	OpenNLP	NLTK
POS(Agrupado)	92.15%	92.14%	92.67%
POS(Específico)	87.12%	86.31%	88.83%
Lemmatization	97.29%	95.26%	95.78%
Tempo de execução(494513 palavras)	57.6 Seg	>25 Min	58.5 seg

4.2 é apresentado o sumário dos resultados, onde é possível de observar que o spaCy não é pior do que nenhuma das *toolkits* testadas e como oferecia mais funcionalidades foi escolhida.

4.3 Modelos de similaridade semântica

A *Knowledgebase* foi criada de maneira modelar. O modelo de comparação era uma peça central, contudo continuou a ser uma peça. Durante o estágio experimentaram-se vários modelos. Aqui é possível ver-se a motivação por detrás da criação de cada um deles, e no capítulo seguinte, é apresentado um plano detalhado sobre o funcionamento de cada um.

4.3.1 Prova de Conceito

A prova de conceito foi o único modelo que foi treinado e implementado para trabalhar com a informação do dataset do SemEval. Foi baseado na implementação de [7].

A solução é baseada numa abordagem onde ambos os *posts* são decompostos de maneira a depois poderem ser comparados entre eles. São decompostos em características facilmente comparáveis, como o número de verbos do *post*, ou quais os adjetivos em cada um deles.

Como dito anteriormente, depois da decomposição dos dois *posts* é necessário

fazer a comparação entre eles e gerar um vetor de features. Algumas delas são simples, como a diferença do número de verbos usados (fazendo referência ao exemplo dado no parágrafo anterior), mas outras são mais complexas (como por exemplo se existir um adjetivo em cada um e se pretender saber se estes são sinónimos). Para isso usamos *gloVe*, que tem como grande vantagem o facto da *cosine similarity* estar perto de 1 para sinónimos, e perto de 0 para palavras não relacionadas.

Como classificador foi utilizado SVM com *Kernel* polinomial, tendo sido treinado com o dataset fornecido pelo SemEval 2016 *task B*.

Os resultados deste modelo têm de ser por isso comparados com os da literatura em vez de serem comparados com os outros por mim implementados.

4.3.2 Modelo 1

O modelo 1 constitui a implementação do modelo que funcionou como prova de conceito. Com algumas pequenas alterações desde logo os dados deixam de ser *posts* e passam a ser perguntas diretas como tal a decomposição em título e corpo deixa de fazer sentido.

Outra das alterações deve-se ao facto de pretendermos um modelo com tempo de resposta aceitável. O modelo anterior apenas comparava um *post* com um conjunto de outros 10 *posts*, mas no caso de utilização previsto para a *knowledgebase* era necessário que o modelo comparasse cada pergunta com todas as questões na base de dados. Por isso foi reduzido o número de características que descrevem com a retirada da versão detalhada da categoria gramatical e a retirada do reconhecimento de entidades.

O XGBoost foi utilizado como classificador e treinado no dataset do Quora, porque este obteve melhores resultados do que o dataset de SNLI.

4.3.3 Modelo 2

O modelo 2 aparece devido à dificuldade em perceber o que fazer na situação de ser necessário comparar os vetores de duas frases quando ambas as frases têm mais de um elemento na categoria em questão. Por exemplo, se a frase A contiver os verbos "andar" e "ouvir" e a frase B contiver os verbos "caminhar" e "escutar", podemos fazer a média da semelhança da comparação (utilizando para isso *cosine similarity*) dos 4 pares possíveis mas dois deles servem só para diminuir a média. Como resolução para este problema no Modelo 1 para além da média das semelhanças adicionamos a média dos máximos nas duas direções e isso serviu de inspiração para resolver o *assignment problem* de todas as palavras

da frase A para todas as palavras da frase B

4.3.4 Modelo 3

Durante parte de investigação para implementar bi-directional LSTM, encontramos um modelo com o objectivo de representar cada frase como um vector, ou seja uma representação semântica de cada frase como um vector. Os dois vectores são combinados gerando um conjunto de features que pode ser utilizado como classificador.

Este modelo genérico de representação semântica foi treinado com o dataset do SNLI para a tarefa de semelhança semântica. Utiliza bi-directional LSTM com recurso a caracterização das palavras em vetores feito pelo gloVe.

4.3.5 Modelo 4

Devido à intuição dada pelo gloVe onde é possível testar se duas palavras são relacionadas olhado para a *cosine similarity* foi aplicada a mesma métrica entre os vetores integrantes no modelo 3.

4.4 Ambiente de apoio ao desenvolvimento

Durante o estágio foi criada a estrutura necessária para carregar e manter a informação, bem como para tornar esta informação acessível ao resto do ecossistema de bots da WIT. O tópico vai ser desenvolvido em maior detalhe no capítulo 6, onde são explicados os componentes, a arquitetura, e as tecnologias.

4.5 Teste dos modelos

Para avaliar a qualidade dos modelos foi criado um teste em que todos os modelos são colocados à prova. É possível encontrar o teste e os resultados para cada um dos modelos no capítulo 7.

Capítulo 5

Modelos

Neste capítulo encontra-se descrito o funcionamento dos modelos implementados e que chegaram a estar integrados com o resto do sistema. Aqui não se discutem meta-heurísticas, nem todos os classificadores testados para cada uma das maneiras de recolher features. É antes uma discussão de cada maneira de recolher features implementada com o melhor conjunto de heurísticas e de classificadores testado.

5.1 Prova de Conceito

Antes de se decidir se este problema seria resolvido com base em semelhança semântica foi-me pedido para testar a abordagem. Para isso implementei um modelo baseado em Franco-Salvador et al.[7]. Nesse modelo foi usado o *dataset* do Semeval utilizando a informação da semelhança entre *posts*.

5.1.1 Dataset

Como descrito anteriormente, o *dataset* é composto por um *post*, chamado *post* original, e de dez outros *posts* que podem ser ou não relacionados com o original, e a nota associada a essa comparação de acordo com os avaliadores do SEMEVAL. A este conjunto mais óbvio de comparações podemos adicionar mais alguns casos. Salienta-se que:

- Pode-se assumir que cada pergunta é igual a si própria. Ao longo deste trabalho, este tipo de comparações são chamadas de *self-referencing*.
- Também se pode assumir que cada *post* é diferente de todos os outros que não estão nos 10 do *dataset*. Este assumir de diferenças no contexto deste

trabalho é chamado de *assume-difference*

Cada modelo tem o seu processo de treino. Por esse motivo, podem beneficiar de maneiras diferentes da inclusão ou da exclusão de cada um deste grupo de comparações.

5.1.2 Features

Através da utilização do spaCy cada um dos *posts* é decomposto num conjunto de características que o descrevem. O spaCy permite decompor uma frase gramaticalmente, atribuindo categorias aos seus elementos: se é um nome, um verbo, etc. Ao todo são 18 categorias distintas (50 se pretendermos a versão mais detalhada). O spaCy também permite distinguir qual a função que uma palavra tem na frase (predicado, sujeito, num total de 60 categorias). Ainda é possível analisar se existem entidades no texto como cidades ou nomes de pessoas (num total de 20 tipos de entidade). Uma vez que nenhuma destas features é numérica e que cada conjunto de features apenas caracteriza uma frase, é impossível construir um classificador sendo por isso uma codificação intermédia da frase.

A partir da codificação intermédia, é feito o trabalho de tornar as features não numéricas em features numéricas. Para isso são usadas três técnicas diferentes.

A semelhança semântica é a comparação de duas perguntas (ver características da comparação). Por isso são calculadas as diferenças entre cada uma das características das duas perguntas.

Características de uma pergunta

Para cada *post* que entra no sistema são calculadas um conjunto de características para mais tarde ser possível comparar com outros *posts* (ver características da comparação).

Cada *post* começa por se dividir em 3 grupos, como o cabeçalho, o corpo, e a união destes dois.

$$Grupos = [Titulo, Corpo, Conjunto]$$

Seja $tokens(g)$ o conjunto de *tokens* do grupo, com $g \in Grupos$

O spaCy recolhe e retorna algumas características do texto. Com essa informação é possível definir um conjunto a que chamamos particularidades.

- POS - Corresponde a cada uma dos 18 POS possíveis de ser retornado pelo spaCy, não entrando na especificidade.
- TAG - Parecido com o POS mas mais específico existindo um total de 60.
- ENT - Conjunto das 18 entidades detetáveis pelo spaCy.
- DEP - Conjunto das 50 dependências que o spaCy consegue detetar.

$$Particularidades = [POS \cup TAG \cup ENT \cup DEP]$$

E definir ainda que um *token* t tem a particularidade p se o spaCy marcou o *token* t com a (POS/TAG/ENT/DEP) p

Existem 3 tipos de características de uma pergunta:

O primeiro é simplesmente o número de *tokens* que têm uma determinada particularidade.

$$contagem(t, p) = \begin{cases} 1, & \text{Se } t \text{ têm a particularidade de ser } p \\ 0, & \text{Nos outros casos} \end{cases}$$

$$T1 = \left[\sum_{t \in tokens(g)} contagem(t, p) : \forall p \in Particularidades \wedge \forall g \in Grupos \right]$$

O segundo é o conjunto de *lemmas* de todos os *tokens* que têm uma determinada particularidade.

$$lemmas(t, p) = \begin{cases} lemma(t), & \text{Se } t \text{ têm a particularidade de ser } p \\ \emptyset, & \text{Nos outros casos} \end{cases}$$

$$T2 = [lemmas(t, p) : \forall p \in particularidades \wedge \forall t \in tokens(g) \wedge \forall g \in Grupos]$$

O terceiro é o conjunto de vetores devolvidos pelo gloVe para todos os *tokens* que têm uma determinada particularidade.

$$vetor(t, p) = \begin{cases} gloVe\ vector(t), & \text{Se } t \text{ têm a particularidade de ser } p \\ \emptyset, & \text{Nos outros casos} \end{cases}$$

$$T3 = [\text{vetor}(t, p) : \forall p \in \text{Particularidades} \wedge \forall t \in \text{tokens}(g) \wedge \forall g \in \text{Grupos}]$$

Características da comparação

Dadas duas perguntas A e B, a semelhança entre a pergunta A e a pergunta B não é inerente nem a A nem a B, mas sim às diferenças entre elas.

Para isso é necessário calcular as diferenças entre A e B, referidas apenas por *features* daqui para a frente.

Para cada um dos tipos de características de uma pergunta (ver Características de uma pergunta) o método de cálculo é diferente.

Para as do primeiro tipo, as *features* são dadas pela diferença nas contagens.

$$F1 = [\text{abs}(C(A) - C(B)) : \forall C \in T1]$$

Para o segundo grupo é dado pelo tamanho da intersecção do conjunto de *lemmas* de A e do conjunto de *lemmas* de B.

$$F2 = [\text{lenght}(C(A) \cap C(B)) : \forall C \in T2]$$

Cada uma das características do terceiro grupo divide-se em 4 *features*.

A primeira é a média dos *cosine distance* entre cada vetor do conjunto da pergunta A para cada um dos vetores do conjunto da pergunta B.

$$F3_P1 = \left[\text{média} \left([\text{cosine_distance}(a, b) : \forall a \in C(A) \wedge \forall b \in C(B)] \right) : \forall C \in T3 \right]$$

A segunda é o máximo das *cosine distance* entre cada vetor do conjunto da pergunta A para cada um dos vetores do conjunto da pergunta B.

$$F3_P2 = \left[\text{máximo} \left([\text{cosine_distance}(a, b) : \forall a \in C(A) \wedge \forall b \in C(B)] \right) : \forall C \in T3 \right]$$

A terceira consiste na média dos máximos. Seja g_A um conjunto do tipo 3 da pergunta A e g_B o conjunto correspondente a g_A mas na pergunta B. Então temos a média dos *cosine distances* máximos a partir de cada elemento em g_A para todos em g_B

$$F3_P3 = \left[\text{média} \left(\left[\text{máximo} \left([\text{cosine_distance}(a, b) : \forall b \in C(B)] \right) : \forall a \in C(A) \right] \right) : \forall C \in T3 \right]$$

O quarto é semelhante ao terceiro, trocando a ordem das perguntas.

$$F3_P4 = \left[\text{m\u00e9dia} \left(\left[\text{m\u00e1ximo} \left([\text{cosine_distance}(b, a) : \forall a \in C(A)] : \forall b \in C(B) \right) \right] : \forall C \in T3 \right) \right]$$

Por fim temos que:

$$F3 = F3_P1 \cup F3_P2 \cup F3_P3 \cup F3_P4$$

e que:

$$Features = F1 \cup F2 \cup F3$$

5.1.3 SVM Classifier

SVM, ou as suas variantes, s\u00e3o os modelos mais utilizados em semelhan\u00e7a sem\u00e2ntica[7]. Existem tr\u00eas classes: equivalente, relevante, e n\u00e3o relevante, de acordo com a defini\u00e7\u00e3o do *dataset*.

Cada uma destas classifica\u00e7\u00f5es tem uma representa\u00e7\u00e3o bin\u00e1ria [(1,0,0), (0,1,0), (0,0,1)].

O processo de treino de um modelo SVM come\u00e7a por pegar nos dados existentes e tentar criar hiperplanos que dividam o espa\u00e7o de maneira a que as classes sejam separ\u00e1veis.

Cria-se assim um modelo que, se funcionasse perfeitamente para duas frases semelhantes, obteria (1,0,0). Contudo o modelo n\u00e3o \u00e9 perfeito, sendo o resultado esperado que 1 seja atenuado e os valores onde deveriam ser zeros sejam aumentados. E ser semelhante pressup\u00f5e tamb\u00e9m v\u00e1rios n\u00edveis de semelhan\u00e7a. A ideia \u00e9 que se \u00e9 "mais semelhante", ent\u00e3o tamb\u00e9m dever\u00e1 ter menos ru\u00eddo. Por isso foi criada uma f\u00f3rmula que d\u00e1 o *score* da compara\u00e7\u00e3o.

$$x = SVM(PerguntaA, PerguntaB)$$

$$score = x[0] \times 100 + x[1] \times 50 - 100 \times x[2]$$

Feature Selection e Feature reduction

Foi utilizado o coeficiente de *Pearson* para a sele\u00e7\u00e3o de *features*, sendo que a uma primeira vista o melhor resultado \u00e9 obtido pela utiliza\u00e7\u00e3o de todas as *features*.

Contudo existem outros modelos que podem ser testados, para saber como afetam a precis\u00e3o do modelo, estando tamb\u00e9m esta tarefa no trabalho a desenvolver.

5.2 Modelo 1

Foi implementado o algoritmo da prova de conceito com algumas adaptações para o novo problema, onde já não existem *posts*, mas antes frases. Na readaptação do problema a solução também foi simplificada, desde logo com a retirada das particularidades TAG e ENT. Contudo alguns dos problemas presentes na prova de conceito continuaram a persistir neste modelo, como o número de features (que permanece elevado), e a informação diminuta em cada uma.

5.2.1 Features

O conjunto de features que faz parte da caracterização de cada uma das frases é um *subset* do utilizado na prova de conceito. A informação foi reduzida devido a não trazer nenhuma mais valia e por tornar lenta a pesquisa de sistema. Isto porque para cada frase nova é necessário transformar as duas codificações intermédias no conjunto das features.

Para tentar detalhar o modelo que testa a similaridade semântica é necessário explicar vários aspetos diferentes como que dados são utilizados, as *features* utilizadas, os modelos de aprendizagem utilizados bem como a maneira que tudo isto se integra.

Treino

O treino foi feito com o dataset da Quora. Composto por pares de frases, e a anotação se as frases têm ou não o mesmo significado.

Foi testado o dataset do SNLI mas os resultados obtidos no teste eram mais baixos.

Querying

Quando uma pergunta é feita ao sistema, esta nova pergunta vai ser comparada com todas as perguntas no sistema. Estas estão agrupadas por grupos de questões que respondem à mesma pergunta, e por isso só é necessário que o modelo faça *match* com uma. Devido a este comportamento existem vantagens em favorecer o erro do lado dos falsos negativos.

Features

Através da utilização do spaCy cada uma das perguntas é decomposta num conjunto de características que a descrevem. Mas como referido anteriormente,

a semelhança semântica é a comparação de duas perguntas (ver características da comparação) e por isso são calculadas as diferenças entre cada uma das características das duas perguntas.

Características de uma pergunta

Para cada pergunta que entra no sistema são calculadas um conjunto de características para mais tarde ser possível comparar com outras perguntas (ver características da comparação).

Seja $tokens(f)$ o conjunto de $tokens$ de cada frase

O spaCy recolhe e retorna algumas características da frase. Com essa informação é possível definir um conjunto a que chamamos particularidades.

- POS - Corresponde a cada uma dos 18 POS possíveis de ser retornado pelo spaCy, não entrando na especificidade.
- DEP - Conjunto das 50 dependências que o spaCy consegue detetar.

$$Particularidades = [POS \cup TAG]$$

E definir ainda que um $token$ t tem a particularidade p se o spaCy marcou o $token$ t com a (POS/DEP) p

Existem 3 tipos de características de uma pergunta:

O primeiro é simplesmente o número de $tokens$ que têm uma determinada particularidade.

$$contagem(t, p) = \begin{cases} 1, & \text{Se } t \text{ têm a particularidade de ser } p \\ 0, & \text{Nos outros casos} \end{cases}$$

$$T1 = \left[\sum_{t \in tokens(f)} contagem(t, p) : \forall p \in Particularidades \wedge \right]$$

O segundo é o conjunto de $lemmas$ de todos os $tokens$ que têm uma determinada particularidade.

$$lemmas(t, p) = \begin{cases} lemma(t), & \text{Se } t \text{ têm a particularidade de ser } p \\ \emptyset, & \text{Nos outros casos} \end{cases}$$

$$T2 = [lemmas(t, p) : \forall p \in particularidades \wedge \forall t \in tokens(f)]$$

O terceiro é o conjunto de vetores devolvidos pelo gloVe para todos os *tokens* que têm uma determinada particularidade.

$$vetor(t, p) = \begin{cases} \text{gloVe vector}(t), & \text{Se } t \text{ têm a particularidade de ser } p \\ \emptyset, & \text{Nos outros casos} \end{cases}$$

$$T3 = [vetor(t, p) : \forall p \in Particularidades \wedge \forall t \in tokens(f)]$$

Características da comparação

Dado duas perguntas A e B a semelhança entre a pergunta A e a pergunta B, não é inerente nem a A nem a B mas sim às diferenças entre elas.

E para isso é necessário calcular as diferenças entre A e B. Referidas apenas por *features* daqui para a frente.

Para cada um dos tipos de características de uma pergunta (ver Características de uma pergunta) o método de calculo é diferente.

Para as do primeiro tipo as *features* são dadas pela diferença nas contagens.

$$F1 = [abs(C(A) - C(B)) : \forall C \in T1]$$

Para o segundo grupo é dado pelo tamanho da intersecção do conjunto de *lemmas* de A e do conjunto de *lemmas* de B.

$$F2 = [length(C(A) \cap C(B)) : \forall C \in T2]$$

Cada uma das características do terceiro grupo divide-se em 4 *features*.

A primeira é a média dos *cosine distance* entre cada vetor do conjunto da pergunta A para cada um dos vetores do conjunto da pergunta B.

$$F3_P1 = \left[\text{média} \left([\text{cosine_distance}(a, b) : \forall a \in C(A) \wedge \forall b \in C(B)] \right) : \forall C \in T3 \right]$$

A segunda é o máximo das *cosine distance* entre cada vetor do conjunto da pergunta A para cada um dos vetores do conjunto da pergunta B.

$$F3_P2 = \left[\text{máximo} \left([\text{cosine_distance}(a, b) : \forall a \in C(A) \wedge \forall b \in C(B)] \right) : \forall C \in T3 \right]$$

A terceira consiste na média dos máximos. Seja gA um conjunto do tipo 3 da pergunta A e gB o conjunto correspondente a gA mas na pergunta B. Então temos a média dos *cosine distances* máximos a partir de cada elemento em gA para todos em gB

$$F3_P3 = \left[\text{média} \left(\left[\text{máximo} \left([\text{cosine_distance}(a, b) : \forall b \in C(B)] \right) : \forall a \in C(A) \right] \right) : \forall C \in T3 \right]$$

O quarto é semelhante ao terceiro trocando a ordem das perguntas.

$$F3_P4 = \left[\text{média} \left(\left[\text{máximo} \left([\text{cosine_distance}(b, a) : \forall a \in C(A)] \right) : \forall b \in C(B) \right] \right) : \forall C \in T3 \right]$$

Por fim temos que:

$$F3 = F3_P1 \cup F3_P2 \cup F3_P3 \cup F3_P4$$

e que:

$$Features = F1 \cup F2 \cup F3$$

5.2.2 Treino do modelo

Para modelo foi utilizado o *xgboost* treinado no *dataset* da quora. Este combinação mostrou apresentar melhores resultados do que quando comparado aos modelos que utilizavam SVM.

5.3 Modelo 2

Para solucionar o problema da grande partição de features do modelo anterior, foi realizado um novo modelo onde as features são mais cuidadas e acarretam consigo mais informação. Para isso em vez de subdividir em várias classes, como o modelo anterior, as palavras são agrupadas em unigramas, bigramas e 2skip1grams:

```

for  $i \in \text{token\_selector}(\text{FraserA})$  do
  | for  $j \in \text{token\_selector}(\text{FraserB})$  do
  | | list_similarity = (similarity_function(i,j), i, j);
  | end
end
sorted_similarity = sort(list_similarity);
used_token_A = False * len(token_selector(FraserA));
used_token_B = False * len(token_selector(FraserB));
total_similarity = 0;
for (similarity, i, j) in list_similarity do
  | if used_token_A == False AND used_token_B == False then
  | | used_token_A[i] = True ;
  | | used_token_B[j] = True ;
  | | total_similarity += similarity ;
  | end
  | if all(used_token_A) or all(used_token_B) then
  | | break;
  | end
end
total_similarity = average(total_similarity)

```

Algorithm 1: Heurística usada para resolução do Assignment problem

features

Partindo dos unigramas, bigramas e 2skip1grams de ambas as frases, são criadas as features utilizando a distância Levenshtein e *cosine distance* fazendo estas métricas é calculado a partir de todos os elementos de um conjunto para o conjunto correspondente na outra frase para isso e é necessário resolver o *assignment problem*. A resolução do *assignment problem* é trivialmente n^4 podendo ser resolvido em n^3 a solução usa uma heurística n^2 garantindo ser um *lowerbound*. Mesmo com a utilização de uma heurística neste modelo é possível notar que este modelo é mais lento que o anterior mesmo que gere muito menos *features*.

5.3.1 Treino do Modelo

O modelo foi treinado de maneira semelhante ao modelo 2. Utilizando também xgboost, continuando a ser treinado no dataset da quora.

5.4 Modelo 3

No decorrer do estágio, o Facebook Research Group criou e disponibilizou sob licença de Creative Commons uma ferramenta para o *encoding* de frases. Um modelo pré-treinado que utiliza *deep learning* para fazer um *encoder* genérico que representa o valor semântico de uma frase.[6]

A utilidade de um *encoder* genérico para uma frase é parecido com a utilidade do *gloVe* vectors para palavras. Cria uma codificação para o sentido de uma frase inteira num vetor, com mais de 16000 dimensões, no qual é possível fazer operações com vetores. Para treinar o *encoder*, o Facebook Reserach Group utilizou o *dataset* de *entilement* e este é treinado de modo a reconhecer se duas frases são iguais, contradizentes, ou neutrais. O modelo aprende utilizando *backpropagation* o *encoding* não é mais do que a seleção de uma das camadas de neurónios escondidos para a representação da frase.

O objetivo final era criar um *encoder* genérico passível de ser usado para treinar qualquer tarefa de linguagem semântica.

Utilizando o vetor como features de um classificador, é possível classificar os mais diversos problemas. No entanto, como parte do código disponibilizado está o classificador para semelhança semântica que foi utilizado para o a criação do *encoder* criado, podemos utilizar a mesma estrutura para testar se duas frases são semelhantes.

5.5 Modelo 4

Como foi referido o modelo 3 assemelha-se a um *gloVe* vector, mas para frases em vez de palavras. Por esse motivo experimentamos usar a mesma estratégia utilizada nos *gloVe* vectors e usar a *cosine similarity*.

À primeira vista os resultados não pareciam muito interessantes, embora pudessem bem ordenados. Como tal, apenas ao recentrar o modelo, os resultados tornaram-se melhores do que os obtidos pelo modelo anterior.

Capítulo 6

Ambiente de suporte ao desenvolvimento

Durante o estágio também foi criado o ambiente de suporte ao desenvolvimento, constituído por um gestor dos conteúdos que a *knowledgebase* está habilitada a responder, o gestor de configurações, o avaliador do modelo e ainda expunha a API para ser integrado na WIT *bot platform*.

6.1 Funcionamento

Para perceber o funcionamento do sistema é necessário perceber como está estruturada a sua informação e como este se integra com o sistema como um todo. O sistema permite gerir a informação carregada, ou seja, as questões, respostas (figura 6.2) e restrições (figura 6.3, ver subsecção seguinte). O sistema tem um módulo de avaliação de modelos que será abordado no capítulo dos resultados. O sistema expõem ainda uma API que permite que o sistema seja integrado com o resto da WIT BOT Platform

6.1.1 Estrutura da informação

O sistema gira em torno da similaridade de questões. Como tal, estas estão no centro da estrutura da informação e estão agrupadas pelo seu significado. Cada conjunto de perguntas tem associado uma ou várias respostas, se estas tiverem condições associadas. Para perceber como uma pergunta pode ter mais que uma resposta, podemos pensar na pergunta "quanto é que eu pago por minuto numa chamada para outra rede?", embora esta pergunta só tenha um significado associado a resposta depende do contexto do utilizador, e neste caso

Figura 6.1: Sistema a devolver a resposta que mais se aproxima a pergunta efetuada

Test model

Manage active FAQ

which size sim should I buy?







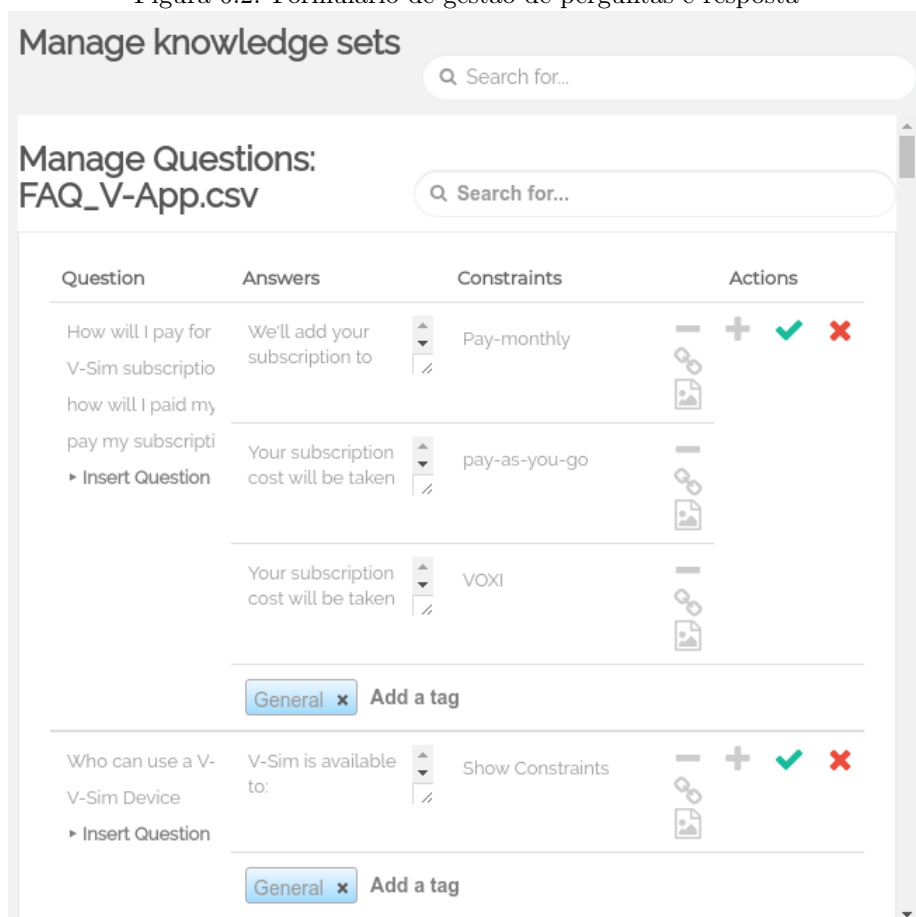
	ID	Sample Question	Confidence	Score
	1	What size SIM do I need?	1.00	17.12
	2	What type of SIM can be used to do a SIM swap?	0.00	-11.05
	3	Why is my new SIM or SIM swap not working?	0.00	-20.64
	4	What are premium rate services?	0.00	-22.29
	5	How can I unlock my SIM card?	0.00	-24.05
	6	What are bars?	0.00	-26.09

Figura 6.2: Formulário de gestão de perguntas e resposta



específico, o tarifário. Para fazer estas distinções existem "restrições" que têm associados várias "perfis". No caso anterior, a restrição poderia ser "tarifário" e o perfil a listagem de cada um dos tarifários disponibilizados pela rede. A cada resposta numa pergunta com condições associadas dizemos que existe um contexto para determinada resposta, e o mesmo é verdade para cada utilizador. O sistema faz depois o mapeamento entre os dois contextos e devolve a resposta que mais interessa para o cliente. Se o contexto do utilizador não for completamente definido, então é feita ao utilizador a pergunta associada à restrição. No exemplo anterior seria a pergunta "Qual é o seu tarifário?", e teria respostas clicáveis para cada um dos tarifários disponíveis.

Os conjuntos de perguntas podem depois ser agrupadas em *datasets* por assunto para que seja mais fácil a sua gestão.

Figura 6.3: Formulário de gestão de constraints

Manage Constraints

Name	Items	Desambiguation Question	Actions
PHONE_OS	<div style="background-color: #009688; color: white; padding: 2px 5px; display: inline-block; margin-bottom: 5px;">Android - android_id x</div> <div style="background-color: #009688; color: white; padding: 2px 5px; display: inline-block; margin-bottom: 5px;">iPhone - iphone_id x</div> <div style="margin-top: 5px;">add item</div>	What kind of device do you use?	✓ ✗
MOBILE_PLAN	<div style="background-color: #009688; color: white; padding: 2px 5px; display: inline-block; margin-bottom: 5px;">Pay-monthly - pay_monthly_id</div> <div style="background-color: #009688; color: white; padding: 2px 5px; display: inline-block; margin-bottom: 5px;">pay-as-you-go -</div> <div style="background-color: #009688; color: white; padding: 2px 5px; display: inline-block; margin-bottom: 5px;">VOXI - voxi_id x</div> <div style="margin-top: 5px;">add item</div>	What is your mobile plan?	✓ ✗
CALLS_USE	<div style="background-color: #009688; color: white; padding: 2px 5px; display: inline-block; margin-bottom: 5px;">Often - often_id x</div> <div style="background-color: #009688; color: white; padding: 2px 5px; display: inline-block; margin-bottom: 5px;">Mostly - mostly_id x</div> <div style="background-color: #009688; color: white; padding: 2px 5px; display: inline-block; margin-bottom: 5px;">Rarely - rarely_id x</div> <div style="margin-top: 5px;">add item</div>	How often do you make calls?	✓ ✗
YES_NO	<div style="background-color: #009688; color: white; padding: 2px 5px; display: inline-block; margin-bottom: 5px;">yes - yes_id x</div> <div style="background-color: #009688; color: white; padding: 2px 5px; display: inline-block; margin-bottom: 5px;">nop - nop_id x</div> <div style="background-color: #009688; color: white; padding: 2px 5px; display: inline-block; margin-bottom: 5px;">i-dont-know - i_dont_know_id x</div> <div style="margin-top: 5px;">add item</div>	Do you like pizza?	✓ ✗

6.1.2 Integração na WIT Bot Platform

A knowledgebase tem como função a disponibilização ao utilizador do conhecimento guardado. O utilizador adiciona um BOT em qualquer plataforma onde este esteja disponibilizado (como por exemplo *Facebook Messenger* ou similar), e comunica com este. Se o Bot não conseguir interpretar o que o utilizador inseriu, então a mensagem é reencaminhada para a *knowledgebase*. Na *knowledgebase* existem dois parâmetros para definir se uma resposta deve ou não ser enviada diretamente ao utilizador: a certeza e a pontuação (figura 6.4). Se estes dois parâmetros não estiverem acima do definido para uma determinada resposta, então a conversa é enviada para um assistente que poderá escolher do con-

Figura 6.4: Página de gestão da configuração activa

The screenshot shows a web interface for managing knowledge sets. At the top, there's a search bar labeled 'Search for...'. Below it, the title 'Manage Questions: FAQ_V-App.csv' is displayed with another search bar. The main content is a table with four columns: 'Question', 'Answers', 'Constraints', and 'Actions'. The first row shows a question about V-Sim subscription payment, with an answer 'We'll add your subscription to', a constraint 'Pay-monthly', and action buttons for minus, plus, checkmark, and cross. The second row shows a question about subscription cost, with an answer 'Your subscription cost will be taken', a constraint 'pay-as-you-go', and similar action buttons. Below the table, there are two tabs labeled 'General' and a button 'Add a tag'.

Question	Answers	Constraints	Actions
How will I pay for V-Sim subscription	We'll add your subscription to	Pay-monthly	- + ✓ ✗
how will I paid my pay my subscrip	Your subscription cost will be taken	pay-as-you-go	- + ✓ ✗
Who can use a V-Sim Device	V-Sim is available to:	Show Constraints	- + ✓ ✗

junto das respostas sugeridas, ou em alternativa, escrever a resposta. No caso de ser uma pergunta com condicionantes associadas e cujo o perfil do utilizador ainda não é conhecido, então o sistema envia ao BOT qual a pergunta que ele necessita de fazer ao utilizador primeiro para desambiguar, e depois saber qual é a resposta correta final para devolver ao utilizador.

Cálculo da pontuação

O cálculo da pontuação é um valor entre -100 e 100 que reflete a similaridade entre duas frases.

Seja q_1 uma questão e q_2 outra questão, mas esta pertencente a *knowledgebase*. Seja ainda $p[x]$ as probabilidades de pertença de cada classe dada pelo modelo quando pedido para avaliar a semelhança entre q_1 e q_2 . Com $p[x]$ trei-

nado para devolver

$$p[0] = 1, \text{ quando } q1 \text{ e } q2 \text{ são diferentes}$$

$$p[1] = 1, \text{ quando as frases são iguais}$$

Sendo certo que $p[0] + p[1] = 1$ mas os valores de $p[0]$ e de $p[1]$ flutuam dependendo da certeza do modelo.

Se quisermos ter a pontuação entre -100 e 100 temos:

$$\text{pontuação}(q1, q2) = -100 + p[1] * 200$$

Isto assume que uma variável chamada ruído na configuração do sistema está a zero, mas esta varia entre -100 e 100. Alterando o valor do ruído podemos pensar que se a primeira fórmula tivesse o valor definido igual ao valor de ruído, então passaria a ter 0. Isto é verdade para qualquer valor e trivialmente verdade para quando o ruído é 0. Para os outros valores de ruído é dada pela percentagem caminhada entre ruído e o extremo formalmente temos que:

$$E1 = ((-100 + p[1] * 200) - \text{ruído}) / (100 - \text{ruído})$$

$$E2 = ((-100 + p[1] * 200) - \text{ruído}) / (100 + \text{ruído})$$

$$\text{pontuação}(q1, q2) = \begin{cases} E1, & (-100 + p[1] * 200) < \text{ruído} \\ E2, & (-100 + p[1] * 200) \geq \text{ruído} \end{cases}$$

Existindo perguntas sinónimas agrupadas o grupo é representado pela pontuação mais elevada dentro do grupo.

Certeza

Enquanto que a pontuação é referente apenas a um par de perguntas, a certeza de uma pergunta está relacionada com o conjunto de todas as perguntas na *knowledgebase* que obtiveram a pontuação superior a 0.

Ou seja, se a pontuação de várias perguntas foi devolvida com um valor alto associado, então a certeza é baixa.

Mas se pelo contrário dada a comparação com uma pergunta, apenas uma pergunta tem a pontuação superior a 0, a certeza é 1. Formalmente temos que

Sejam $q1$ e $q2$ perguntas com $q2$ pertencente a *knowledgebase*

Seja ainda Q o conjunto de questões diferenciadas existentes na *knowledgebase* temos que:

$$\text{soma_pontua\~{c}o\~{e}s} = \sum_{q \in Q} \max(\text{pontua\~{c}o}(q1, q2), 0)^2$$

$$\text{certeza}(q1, q2) = \frac{\max(\text{pontua\~{c}o}(q1, q), 0)^2}{\text{soma_pontua\~{c}o\~{e}s}}$$

Todas as perguntas agrupadas pelo significado têm o mesmo valor de certeza,

6.2 Tecnologias

O sistema é constituído por um servidor HTTP (no caso, `cherryPy`¹) o qual fazia a resposta a pedidos API, e servia as páginas HTML do gestor de conteúdos, complementado com `Jinja2`² para HTML *rendering* e `jQuery`³ para carregamento dinâmico de páginas. Os dados eram guardados numa base de dados em `SQLite`⁴ que servia de SGBD usado como motor para o `SQLAlchemy`⁵. O `SQLAlchemy` serve como *mapper* de classes em programação orientada a objetos para tabelas num modelo relacional de uma base de dados. Nos modelos que utilizam SVM como classificadores, é utilizada a implementação do `scikit-learn`⁶ e os modelos que são implementados com eXtreme Gradient Boost utilizam a implementação `XGBoost`⁷.

¹<http://cherrypy.org/>

²<http://jinja.pocoo.org/>

³<https://jquery.com/>

⁴<https://sqlite.org/>

⁵<https://www.sqlalchemy.org/>

⁶<http://scikit-learn.org>

⁷<http://xgboost.readthedocs.io/en/latest/python/index.html>

Capítulo 7

Resultados

À exceção do modelo criado para prova de conceito, todos os modelos criados foram avaliados num teste construído por nós. O teste é dividido em três partes, duas delas triviais que todos os modelos passaram: "*ipsis verbis*" e "Falsos positivos". *Ipsis verbis* representa questões que estão na *knowledgebase* com a mesma grafia; falsos positivos é um teste com um conjunto de questões não relacionadas com o tema em discussão. O terceiro conjunto de perguntas questiona a *knowledgebase* com perguntas que esta sabe responder, mas que nunca viu. Para isto, no momento da criação do *dataset* *vodafone-faq*, cada pergunta no *dataset* era rescrita de 5 maneiras diferentes e quatro delas eram carregadas para a *knowledgebase*. Uma delas era usada para testar os modelos. Admito que quanto à criação do *dataset* da Vodafone FAQ, especialmente o que contém a pergunta de teste, este não foi feito da forma que mais me agrada. O *dataset* foi criado apenas por duas pessoas que trazem com elas os seus *bias* e a sua inerente subjetividade na maneira de pensar. Contudo continuo a considerar uma ferramenta válida para a comparação entre os modelos.

Existem 5 colunas de resultados:

- A primeira coluna *result*, tem o valor máximo se para cada par de perguntas em que a pontuação tem de ser -100, o modelo está abaixo de 0; e se para cada uma que tem de ser 100, a pontuação corresponde de facto a 100.
- A segunda coluna Top 1 caracteriza qual a percentagem de perguntas no teste que devolveu a pergunta correta com a pontuação mais alta.
- A terceira coluna Top 3 caracteriza qual a percentagem de perguntas que têm a resposta correta, dentro das três com a pontuação mais elevada.

- *Not wrong* é a percentagem de perguntas que tendo sido feitas ao bot, este não teria devolvido a resposta errada. O valor é máximo se não existir nenhuma pergunta errada no TOP1, cuja certeza e a pontuação estão acima dos limites definidos, despoletando assim uma resposta errada no bot.
- *Right* é a percentagem de perguntas tendo sido feitas ao bot e em que este teria devolvido a resposta certa, ou seja a pontuação e a certeza estão acima dos *thresholds* definidos

Existem ainda duas colunas de totais, em que numa delas os valores dos grupos não estão normalizados (ou seja cada pergunta vale o mesmo e por isso o teste vale mais se tiver mais perguntas). Em contrapartida no total de grupos normalizado cada teste vale $1/\#\text{numero de testes}$, nos casos apresentados cada teste vale um terço.

A execução do teste gera ainda um relatório detalhado. Na figura 7.1 podemos verificar que o modelo escolheu a opção correta com 38.68 de score e devolveu ainda um falso negativo com 11.10 como a confiança é proporcional ao quadrado do score as confianças são respetivamente 92% e 8%

Na figura 7.2 podemos analisar os resultados com cada uma das variantes da pergunta.

7.1 Prova de conceito

Resultados obtidos utilizando o avaliador do SemEval 2016 para a task3 sub-task B que valeria o 8 lugar com 0.7185 (ver figura 7.3) no ranking do SemEval. No anexo 1 encontra-se a tabela completa com os resultados. Não é possível comparar com o resto dos modelos por este ser o único a correr com dados do tipo *post*. A discrepância para a implementação do primeiro classificado está relacionada com o facto de não ter sido implementado completamente o modelo descrito. Antes, foram retiradas as *features* mais fáceis de programar para mostrar o modelo a funcionar e este cumpriu o seu papel de validar a ideia. Por isso pediram-me para me focar nos modelos no contexto da utilização do dataset da Vodafone FAQ.

7.2 Modelo 1

Os resultados obtidos pelo modelo 1 foram muito animadores e para este teste, foram durante muito tempo, os melhores resultados obtidos. Contudo o

Figura 7.1: Resultados retornados quando é feita a pergunta "I lost my username"

ID	Sample Question	Confidence	Score
Corret			
1	I've forgotten my Vodafone Protect username, what should I do?	0.92	38.68
False Positive			
1	I've forgotten my Vodafone Protect password, what should I do?	0.08	11.10
False Negative			
8	How can I use My Vodafone?		37.12
9	I dont remember my memorable word		100.00
10	Why is MyVodafone asking me for a security code?		50.00

Tabela 7.1: Resultados Modelo 1

Test Type	Result	Top 1	Top 3	Not Wrong	Right
ipsis verbis	100.00	100.00	100.00	100.00	100.00
False Positive	94.00	100.00	100.00	100.00	100.00
Different ways	67.07	93.44	98.36	96.72	68.85
Total grupos não normalizados	94.46	99.15	99.79	99.58	95.97
Total grupos normalizados	87.02	97.81	99.45	98.91	89.62

seu comportamento errático onde pequenas alterações na pergunta resultavam em altas variabilidades nos resultados não deixavam confortáveis quem tinha de ir para a rua vender o produto e fazer demonstrações com ele. É possível ver o alto nível de variabilidade reparando que o *Result* do teste *different ways* é o mais baixo de todos os modelos mesmo que este acerte mais que o Modelo 2 e

Figura 7.2: Resultados do avaliador do SemEval 2016 Task B

7		I lost my username	92.39
Corret			
ID	Sample Question	Confidence	Score
1	I've forgotten my Vodafone Protect username, what should I do?	0.92	38.68

Sentence	Score
I've forgotten my Vodafone Protect username, what should I do?	-19.29
I forgot my username	38.68
I have forgotten my username, what do I do now?	-9.13
What can I do to find my vodafone username?	-18.96
You'll need your Vodafone device and number to hand to get your username. If you don't know your number just call *#100# from your	-52.44

3. Por isso foi preterido pelo Modelo 2, embora em termos de casos resolvidos com sucesso, este não fosse um passo na direção certa.

7.3 Modelo 2

O modelo dois é claramente mais conservador e apresenta resultados mais baixos durante o teste. Mas o facto de ser mais previsível, permitia a quem vendia os produtos e tinha um guião do que mostrar nas apresentações, que ficasse mais descansado e confiante com os resultados. Os fracos resultados foram combatidos durante as apresentações com um aumento no número de

Figura 7.3: Resultados do avaliador do SemEval 2016 Task B

```

*** Official score (MAP for SYS): 0.7185

*****
*** Classification results ***
*****

Acc = 0.6920
P   = 0.7542
R   = 0.4159
F1  = 0.5361

*****
*** Detailed ranking results ***
*****

IR -- Score for the output of the IR system (baseline).
SYS -- Score for the output of the tested system.

          IR   SYS
MAP   : 0.7135 0.7185
AvgRec: 0.8611 0.8689
MRR   : 76.67 77.33

          IR   SYS          IR   SYS          IR   SYS          IR   SYS
REC-1@01: 70.00 72.00 ACC@01: 70.00 72.00 AC1@01:  0.81 0.84 AC2@01:  35 36
REC-1@02: 82.00 80.00 ACC@02: 69.00 68.00 AC1@02:  0.83 0.82 AC2@02:  69 68
REC-1@03: 82.00 82.00 ACC@03: 62.67 63.33 AC1@03:  0.82 0.83 AC2@03:  94 95
REC-1@04: 82.00 82.00 ACC@04: 57.00 58.50 AC1@04:  0.80 0.82 AC2@04: 114 117
REC-1@05: 82.00 82.00 ACC@05: 54.40 55.60 AC1@05:  0.80 0.82 AC2@05: 136 139
REC-1@06: 86.00 86.00 ACC@06: 52.00 52.33 AC1@06:  0.83 0.83 AC2@06: 156 157
REC-1@07: 86.00 86.00 ACC@07: 49.71 49.71 AC1@07:  0.87 0.87 AC2@07: 174 174
REC-1@08: 86.00 86.00 ACC@08: 46.50 47.00 AC1@08:  0.90 0.91 AC2@08: 186 188
REC-1@09: 86.00 86.00 ACC@09: 44.67 44.89 AC1@09:  0.95 0.95 AC2@09: 201 202
REC-1@10: 86.00 86.00 ACC@10: 42.80 42.80 AC1@10:  1.00 1.00 AC2@10: 214 214

REC-1 - percentage of questions with at least 1 correct answer in the top @X positions
         (useful for tasks where questions have at most one correct answer)
ACC - accuracy, i.e., number of correct answers retrieved at rank @X normalized by the
      rank and the total number of questions
AC1 - the number of correct answers at @X normalized by the number of maximum possible
      answers (perfect re-ranker)
AC2 - the absolute number of correct answers at @X

```

Tabela 7.2: Resultados Modelo 2

Test Type	Result	Top 1	Top 3	Not Wrong	Right
ipsis verbis	100.00	100.00	100.00	100.00	100.00
False Positive	100.00	100.00	100.00	100.00	100.00
Different ways	75.59	81.97	96.72	96.72	32.79
Total grupos não normalizados	96.84	97.66	99.58	99.58	91.30
Total grupos normalizados	91.86	93.99	98.91	98.91	77.60

formas de fazer as perguntas, para as perguntas que faziam parte do show case.

7.4 Modelo 3

O modelo 3 esteve implementado durante um curto período de tempo porque a partir da criação deste, foi desenvolvido o modelo 4 (que teve melhores resultados em toda a linha). No entanto este representou um passo na direção

Tabela 7.3: Resultados Modelo 3

Test Type	Result	Top 1	Top 3	Not Wrong	Right
ipsis verbis	100.00	100.00	100.00	100.00	100.00
False Positive	100.00	100.00	100.00	100.00	100.00
Different ways	78.13	88.52	98.36	95.08	44.26
Total grupos não normalizados	97.17	98.51	99.79	99.36	92.78
Total grupos normalizados	92.71	96.17	99.45	98.36	81.42

certa depois do Modelo 2, porque é um modelo que resolve mais questões sem os problemas existentes no Modelo 1.

7.5 Modelo 4

Tabela 7.4: Resultados Modelo 4

Test Type	Result	Top 1	Top 3	Not Wrong	Right
ipsis verbis	100.00	100.00	100.00	100.00	100.00
False Positive	100.00	100.00	100.00	100.00	100.00
Different ways	88.56	95.08	100.00	100.00	80.33
Total grupos não normalizados	98.52	99.36	100.00	100.00	97.45
Total grupos normalizados	96.19	98.36	100.00	100.00	93.44

Os resultados obtidos no Modelo 4 foram melhores em toda a linha, em comparação com qualquer outro modelo. Não só tinha a confiança de quem o tinha de apresentar em público, mas também obtinha bons resultados e respondia a grande parte dos pedidos. À data do fim do estágio, este era o modelo implementado e em utilização.

Capítulo 8

Conclusão

Este trabalho teve como objetivo principal a criação de um modelo que fosse capaz de responder a um conjunto de perguntas do chamado *tier 1*, e que fosse capaz de o fazer reconhecendo as suas limitações. Reconhece ainda a necessidade de intervenção do operador humano sempre que necessário. Consideramos que o objetivo foi completado com sucesso.

A utilização da similaridade semântica para a construção da *knowledgebase* revelou-se frutífera para a prossecução dos objetivos.

A existência de uma grande recetividade da parte dos operadores às varias iterações das provas de conceito mostra a atratividade da tecnologia e das funcionalidades para o mercado. Mostra também que o caminho da arquitetura híbrida é um compromisso interessante para as empresas que continuam a ter o cliente em mente, e que não querem prescindir de um apoio autónomo e personalizável ao mesmo tempo.

Fazendo fé nos valores obtidos pelo teste, a redução em 80% numa fase embrionária do sistema indica que é possível reduzir o custo do atendimento ao cliente.

Considera-se que os resultados podem melhorar com a criação de datasets de treino a partir da utilização dos modelos, aproximando assim o *dataset* de treino com o contexto dos casos de utilização.

O aparecimento das redes BI-LSTM impacta significativamente toda a área de NLP.

Bibliografia

- [1] P. L. AlessandroMoschitti, J. Glass, and B. Randeree. Semeval-2016 task 3: Community question answering. , 2016.
- [2] Apache Software Foundation. Apache OpenNLP - Download. , 2011. URL <http://incubator.apache.org/opennlp/download.cgi>.
- [3] S. Bird, E. Klein, and E. Loper. *Natural language processing with Python*. "O'Reilly Media, Inc.", 2009.
- [4] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2015.
- [5] J. D. Choi, J. Tetreault, and A. Stent. It depends: Dependency parser comparison using a web-based evaluation tool. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL*, pages 26–31, 2015.
- [6] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes. Supervised learning of universal sentence representations from natural language inference data. *CoRR*, abs/1705.02364, 2017. URL <http://arxiv.org/abs/1705.02364>.
- [7] M. Franco-Salvador, S. Kar, T. Solorio, and P. Rosso. Uh-prhlt at semeval-2016 task 3: Combining lexical and semantic-based features for community question answering. *Proceedings of SemEval*, 16:814–821, 2016.
- [8] C. D. Manning, P. Raghavan, H. Schütze, et al. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

- [9] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013. URL <http://arxiv.org/abs/1301.3781>.
- [10] P. Nakov, L. Màrquez, A. Moschitti, W. Magdy, H. Mubarak, A. A. Freihat, J. Glass, and B. Randeree. SemEval-2016 task 3: Community question answering. In *Proceedings of the 10th International Workshop on Semantic Evaluation, SemEval '16*, San Diego, California, June 2016. Association for Computational Linguistics.
- [11] OneReach. 2014 report high demand for text messaging, Janeiro 2017. URL <https://onereach.com/resources/high-demand-for-text-message-2014-report>.
- [12] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014. URL <http://www.aclweb.org/anthology/D14-1162>.
- [13] J. Tian, Z. Zhou, M. Lan, and Y. Wu. 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. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 191–197, 2017.

Apêndice A

Resultados SemEval 2016

Task 3 Subtask B

	Submission	MAP	AvgRec	MRR	P	R	F1	Acc
	UH-PRHLT-contrastive2	77.33	90.84	83.93	63.57	70.39	66.80	76.71
1	UH-PRHLT-primary	76.70₁	90.31₄	83.02₄	63.53₇	69.53₃	66.39₃	76.57₄
	UH-PRHLT-contrastive1	76.56	90.22	83.02	62.74	70.82	66.53	76.29
	Kelp-contrastive1	76.28	91.33	82.71	63.83	77.25	69.90	77.86
	Kelp-contrastive2	76.27	91.44	84.10	64.06	77.25	70.04	78.00
	SLS-contrastive1	76.17	90.55	85.48	74.39	52.36	61.46	78.14
	SLS-contrastive2	76.09	90.14	84.21	77.21	45.06	56.91	77.29
2	ConvKN-primary	76.02₂	90.70₂	84.64₁	68.58₃	66.52₆	67.54₂	78.71₃
3	Kelp-primary	75.83₃	91.02₁	82.71₆	66.79₄	75.97₂	71.08₁	79.43₁
	ConvKN-contrastive1	75.57	89.64	83.57	63.77	72.53	67.87	77.14
4	SLS-primary	75.55₄	90.65₃	84.64₁	76.33₂	55.36₉	64.18₆	79.43₁
	SUper_team-contrastive1	75.17	88.84	83.66	63.25	63.52	63.38	75.57
5	ICL00-primary	75.11₅	89.33₅	83.02₄	33.29₁₁	100.00₁	49.95₉	33.29₁₁
	ICL00-contrastive1	74.89	89.08	82.71	33.29	100.00	49.95	33.29
6	SUper_team-primary	74.82₆	88.54₇	83.66₃	63.64₆	57.08₈	60.18₇	74.86₇
	ICL00-contrastive2	74.05	89.11	82.79	33.29	100.00	49.95	33.29
7	ECNU-primary	73.92₇	89.07₆	81.48₇	100.00₁	18.03₁₁	30.55₁₁	72.71₉
	ECNU-contrastive1	73.25	88.55	80.81	100.00	18.03	30.55	72.71
	ECNU-contrastive2	71.62	86.55	80.88	54.61	71.24	61.82	70.71
8	ITNLP-AiKF-primary	71.43₈	87.31₈	81.28₈	62.75₉	68.67₄	65.57₄	76.00₆
9	UniMelb-primary	70.20₉	86.21₉	78.58₁₁	63.96₅	54.08₁₀	58.60₈	74.57₈
10	overfitting-primary	69.68₁₀	85.10₁₀	80.18₉	63.20₈	67.81₅	65.42₅	76.14₅
	QAIIT-contrastive1	69.24	85.24	80.30	38.99	66.09	49.04	54.29
11	QAIIT-primary	69.04₁₁	84.53₁₁	79.55₁₀	39.53₁₀	64.81₇	49.11₁₀	55.29₁₀
	QAIIT-contrastive2	46.23	68.07	48.92	36.25	51.50	42.55	53.71
	Baseline 1 (IR)	74.75	88.30	83.79	—	—	—	—
	Baseline 2 (random)	46.98	67.92	50.96	32.58	73.82	45.20	40.43
	Baseline 3 (all ‘true’)	—	—	—	33.29	100.00	49.95	33.29
	Baseline 4 (all ‘false’)	—	—	—	—	—	—	66.71

Table 2: **Subtask B, English (Question-Question Similarity)**: results for all submissions. The first column shows the rank of the primary runs with respect to the official MAP score. The second column contains the team’s name and its submission type (primary vs. contrastive). The following columns show the results for the primary, and then for other, unofficial evaluation measures. The subindices show the rank of the primary runs with respect to the evaluation measure in the respective column.