Universidade de Évora - Escola de Ciências e Tecnologia

Mestrado em Engenharia Informática

Dissertação

Classificação automática de eventos na linha de saúde SNS24

Rute Gomes Veladas

Orientador(es) | Paulo Miguel Quaresma

Évora 2021

Universidade de Évora - Escola de Ciências e Tecnologia

Mestrado em Engenharia Informática

Dissertação

Classificação automática de eventos na linha de saúde SNS24

Rute Gomes Veladas

Orientador(es) | Paulo Miguel Quaresma

Évora 2021

A dissertação foi objeto de apreciação e discussão pública pelo seguinte júri nomeado pelo Diretor da Escola de Ciências e Tecnologia:

Presidente	|	Irene Pimenta Rodrigues (Universidade de Évora)
Vogais	|	Paulo Miguel Quaresma (Universidade de Évora) (Orientador) Teresa Gonçalves (Universidade de Évora) (Arguente)

Évora 2021

Dedico esta dissertação à minha mãe

Agradecimentos

Esta dissertação representa o fechar de mais um capítulo na minha vida académica, capítulo este que me permitiu crescer como pessoa e que não teria sido possível sem a ajuda de todos os que contribuíram para o seu desenvolvimento direta ou indiretamente.

Gostaria de começar por agradecer ao professor Paulo Quaresma, orientador do presente trabalho, pelo tempo disponibilizado, paciência, e pelos conselhos e revisões literárias das diversas versões do documento. Agradeço também a todos os professores e investigadores envolvidos no projeto, pelas contribuições e diversas sugestões nas reuniões ao longo do ano.

Deixo também um agradecimento à Fundação pela Ciência e Tecnologia (FCT) pelo financiamento ao projeto Scout.AI (Referência DSAIPA/AI/0040/2019) e à Universidade de Évora (UE) pela bolsa de investigação inserida no contexto do mesmo.

Agradeço aos meus amigos próximos e ao meu namorado Daniel por todos os momentos de compreensão, ajuda e descontração. Foram longos os meses em que me ouviram falar sobre as fases boas e menos boas e me deixaram sempre mais motivada para continuar.

Finalmente, gostaria de agradecer à minha família, em especial à minha mãe, pelo apoio incondicional em todos os momentos, por celebrarem comigo cada pequena vitória ao longo do percurso, e por terem criado possibilidades para que pudesse entrar nesta aventura.

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Acrónimos

BERT *Bidirectional Encoder Representations from Transformers*

CharLM *Neural Character-level Language Modeling*

CSP Cuidados de Saúde Primários

DGS Direção Geral da Saúde

FCT Fundação pela Ciência e Tecnologia

IA Inteligência Artificial

LSTM *Long short-term memory*

ML *Machine Learning*

RBM Máquina de Boltzmann Restrita

RNA Rede Neural Artificial

SPMS Serviços Partilhados do Ministério da Saúde

SVM Máquinas de Vetores de Suporte

TAE Serviço de Triagem, Aconselhamento e Encaminhamento

TF-IDF *Term Frequency Inverse Document Frequency*

UE Universidade de Évora

Sumário

Nesta dissertação apresentamos uma nova ferramenta de suporte à decisão a ser implementada no Serviço de Triagem, Aconselhamento e Encaminhamento (TAE) do Centro de Contacto do Serviço Nacional de Saúde - SNS24. Atualmente a seleção do algoritmo clínico mais adequado a cada situação é efetuada manualmente pelo enfermeiro que atende a chamada. Esta seleção deve ser feita de entre um conjunto de 59 algoritmos clínicos, sendo que esta implementação vem responder à necessidade de reduzir a duração das chamadas recebidas pela linha e consequentemente maximizar o número de chamadas atendidas por unidade de tempo. Este será um modelo baseado em metodologias de Inteligência Artificial, com foco em abordagens de Aprendizagem automática e Processamento de língua natural. O modelo apresentado representa o modelo inicial que foi desenvolvido com um conjunto de dados com os registos de três meses de chamadas, equivalente a cerca de 270.000 registos, mas o modelo final será futuramente desenvolvido a partir de um conjunto de dados com cerca de 4 milhões de chamadas registadas ao longo de três anos pela linha de saúde. O modelo inicial permitiu atingir uma exatidão de 78,80% e medida-F de 78,45% para a classificação da classe do *top* 1, enquanto que a classificação para o *top* 3 e *top* 5 de classes atingiu valores de exatidão superiores a 90%.

Palavras chave: Inteligência Artificial, Processamento de Língua Natural, Aprendizagem automática

Abstract

Automatic event classification on the health phone line SNS24

In this dissertation we present a new decision support tool to be implemented in the Screening, Counseling and Referral Service (TAE) of the Contact Center of the National Health Service - SNS24. Currently, the selection of the most appropriate clinical algorithm for each situation is done manually by the nurse who answers the call. This selection must be made from a set of 59 clinical algorithms, and this implementation responds to the need to reduce the duration of calls received by the line and consequently maximize the number of calls answered per unit of time. This will be a model based on Artificial Intelligence methodologies, focusing on Machine Learning and Natural Language Processing approaches. The model presented represents the initial model that was developed with a set of data with the records of three months of calls, equivalent to about 270.000 records, but the final model will be developed in the future from a data set with about 4 million calls registered over three years by the health line. The initial model reached an accuracy of 78.80% and F-measure of 78.45% for the classification of the top 1 class, while the classification for the top 3 and top 5 classes reached values of accuracy greater than 90%.

Keywords: Artificial Inteligence, Natural Language Processing, Machine Learning

1

Introdução

1.1 Motivação

Ao longo dos anos tem vindo a crescer a popularidade e consequentemente a afluência de chamadas recebidas diariamente pelo Centro de Contacto do Serviço Nacional de Saúde - SNS 24. Este crescimento cria uma necessidade de melhorar o serviço de atendimento de forma a maximizar o número de chamadas atendidas por unidade de tempo. Tendo em conta esta necessidade foi criado o projeto SNS24 Scout.AI financiado pela Fundação pela Ciência e Tecnologia (FCT) e desenvolvido por uma equipa de investigação do Departamento de Informática da Universidade de Évora (UE) em conjunto com uma equipa de especialistas dos Serviços Partilhados do Ministério da Saúde (SPMS).

O Serviço de Triagem, Aconselhamento e Encaminhamento (TAE) é o serviço telefónico prestado pela linha do SNS 24. Em 2018 foram atendidas mais de 1 milhão de chamadas com a duração média de 7-8 minutos. Sendo de âmbito nacional, este é um serviço promotor da equidade de acesso aos cuidados de saúde.

O atendimento telefónico é efetuado por enfermeiros e obedece a algoritmos clínicos pré-definidos. A triagem é efetuada com base num determinado algoritmo clínico (de um conjunto de 59), sendo a escolha do algoritmo mais adequado da maior importância e relevância. O algoritmo clínico selecionado deve

garantir uma elevada segurança (não falhar na identificação de situações que precisam de contacto médico urgente) e elevada capacidade discriminatória (não enviar situações de baixo risco clínico para Urgência Hospitalar).

Os dois objetivos principais deste projeto são: criar uma ferramenta de suporte ao enfermeiro na seleção do algoritmo clínico mais adequado e posteriormente, através desta ferramenta, dar apoio à Direção Geral da Saúde (DGS) no processo de otimização do desenho dos algoritmos clínicos e respetivos encaminhamentos.

Esta dissertação foi desenvolvida no âmbito no primeiro objetivo do projeto. Esse sistema de apoio à decisão será desenvolvido recorrendo a metodologias de Inteligência Artificial (IA), com foco em abordagens de Aprendizagem Automática e Processamento de Linguagem Natural.

1.2 Objetivos

O objetivo desta dissertação é criar uma ferramenta de apoio à decisão que suporte o técnico que atende a chamada a selecionar o algoritmo clínico mais adequado a cada situação. Este objetivo será alcançado através da identificação dos algoritmos mais adequados para um determinado conjunto de sintomas, com ajuste para os diferentes atributos presentes no conjunto de dados como, por exemplo a idade. A metodologia de IA a aplicar será baseada num classificador construído com base em algoritmos de aprendizagem automática sobre um conjunto de dados anonimizados, obtidos a partir dos contactos efetuados para a linha de saúde SNS24 no período entre Janeiro e Março de 2018. Estes dados são constituídos por cerca de 270.000 chamadas.

1.3 Principais Contribuições

Após a criação do modelo final da ferramenta, o mesmo será implementado no serviço de TAE do SNS24, visando a sua funcionalidade como sistema de suporte à decisão, indicando em tempo real qual/quais os algoritmos mais prováveis a serem utilizados em cada caso específico. O sucesso desta tarefa poderá significar num aumento significativo no número de chamadas atendidas anualmente. Uma simples diminuição de 5% no tempo de atendimento (sem compromisso da qualidade), permitirá obter um ganho anual de 350.000 minutos, o que permite realizar mais 50.000 atendimentos do que atualmente. A avaliação deste impacto será efetuada por análises comparativas do número e percentagem de chamadas telefónicas em que há alteração do algoritmo inicialmente selecionado e da duração média das chamadas telefónicas.

A principal contribuição desta dissertação é então a criação de um classificador automático que prevê os algoritmos clínicos mais adequados a cada uma das situações.

1.4 Estrutura da dissertação

A dissertação encontra-se estruturada por capítulos. No capítulo 2 é apresentado o Estado da Arte relacionado com este trabalho, com foco em especial na Aprendizagem Automática e no Processamento de Linguagem Natural. De seguida o capítulo 3 contém uma descrição detalhada do conjunto de dados utilizado, incluindo a sua distribuição pelos conjuntos de treino, validação e teste, bem como a caracterização dos atributos que o compõem. O capítulo 4 define a metodologia seguida, que algoritmos e técnicas foram selecionados e a respetiva ordem pela qual foram seguidas as experiências. Os resultados e a sua discussão encontram-se detalhados no capítulo 5 seguidos, finalmente, pelo capítulo 6 que contém as conclusões finais e trabalho a desenvolver no futuro.

2

Conceitos

2.1 Aprendizagem Automática

A Aprendizagem Automática[1] pode ser considerada um subcampo da IA, uma vez que utiliza os seus métodos para auxiliar o processo de aprendizagem dos computadores que faz com que estes se comportem de forma mais inteligente, em vez de apenas armazenar e tratar dados como um sistema de bases de dados faz. A Aprendizagem Automática é inspirada numa grande variedade de áreas, entre estas a ciência da computação, estatística, biologia e psicologia.

Por outras palavras, a Aprendizagem Automática é uma forma de IA que permite a um sistema aprender a partir dos dados em vez de programação explícita [eJH18]. Usa uma variedade de algoritmos que aprendem iterativamente através dos dados para melhorar, descrever e prever resultados. À medida que os algoritmos recebem dados, passa a ser possível produzir modelos mais precisos baseados nesses dados [eJH18].

A Aprendizagem Automática também pode ser definida como o processo de resolver um problema prático ao recolher um conjunto de dados e algoritmicamente construir um modelo estatístico baseado nesses dados [Bur19].

[1] *Machine Learning* (ML)

Existem três tipos de aprendizagem automática principais que são a aprendizagem supervisionada, a aprendizagem não supervisionada e a aprendizagem por reforço. Na aprendizagem supervisionada o agente observa alguns exemplos de pares entrada-saída e aprende uma função que mapeia o caminho da entrada até à saída. Na aprendizagem não supervisionada o agente aprende padrões nos dados de entrada embora nenhuma informação explícita sobre a saída seja fornecida [RN09]. A aprendizagem por reforço consiste em aprender a realizar uma tarefa por meio de tentativa erro através de um sistema de recompensa como, por exemplo, um sistema de ganho e perda de pontos.

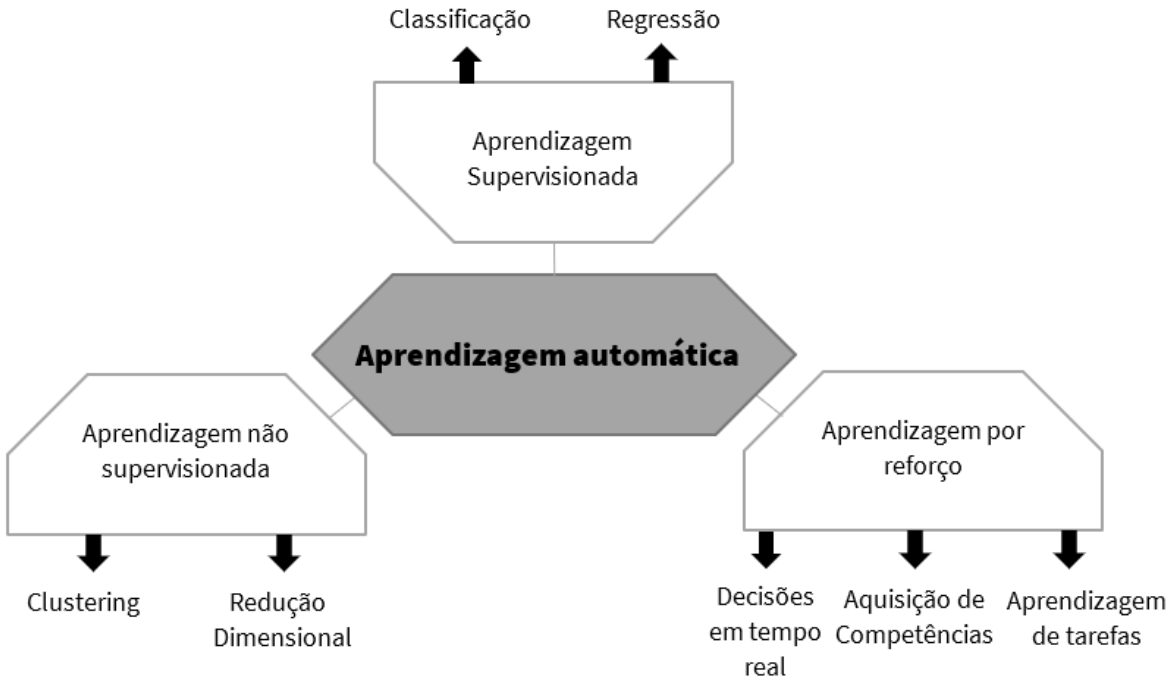

Figura 2.1: Tipos de Aprendizagem Automática

2.1.1 Aprendizagem supervisionada

A aprendizagem supervisionada é o ramo da aprendizagem automática mais amplamente estudado e investigado [Ham18]. Na aprendizagem supervisionada, a máquina aprende com um conjunto de treino composto por dados rotulados fornecidos por um perito externo que determina as ações corretas que o sistema deve realizar para cada exemplo [Ham18]. A tarefa do sistema é generalizar as suas respostas para agir corretamente em casos que não estão disponíveis nos exemplos de treino [Ham18].

Na aprendizagem supervisionada o primeiro passo consiste em lidar com o conjunto de dados que irá ser utilizado. É necessário fazer uma boa seleção de atributos e efetuar o pré-processamento do conjunto de dados de modo a eliminar possíveis valores em falta, ruído indesejado ou dados inconsistentes [ZZY03]. Várias técnicas foram introduzidas por diferentes investigadores para lidar com a questão dos dados em falta. Hodge e Austin [HA04] realizaram uma pesquisa de técnicas contemporâneas para detecção de ruído[2]. Karanjit e Shuchita [SU12] também discutiram diferentes métodos de detecção de *outliers* que são atualmente usados em diferentes métodos de Aprendizagem Automática.

Duas das tarefas principais de aprendizagem supervisionada são a classificação e a regressão.

[2] *outliers*

A classificação é o processo de encontrar um modelo que descreve e distingue classes de dados [HKP06]. O modelo é desenvolvido com base na análise de um conjunto de dados de treino. O modelo é usado para prever o rótulo da classe para os quais o rótulo da classe é desconhecido. Enquanto que a classificação prevê rótulos categóricos, a regressão modela funções de valor contínuo [HKP06]. Ou seja, a regressão é usada para prever valores de dados numéricos ausentes ou indisponíveis em vez de rótulos de classe. A análise de regressão é uma metodologia estatística usada com mais frequência para previsão numérica, embora também existam outros métodos [HKP06].

Alguns dos algoritmos de aprendizagem supervisionada mais conhecidos são: regressão logística, *Naïve Bayes*, árvores de decisão e Máquinas de Vetores de Suporte (SVM).

SVM's

SVM's são um algoritmo de aprendizagem supervisionada utilizado para tarefas de classificação e regressão. O SVM mapeia o vetor de entrada para um maior espaço dimensional onde é construído um hiperplano ótimo de separação que representa a margem máxima de separação entre duas classes [Vap95].

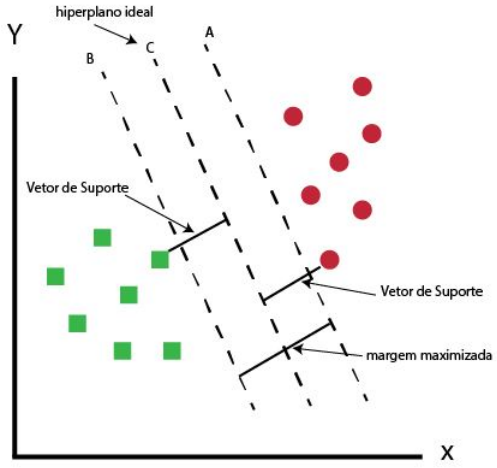

Figura 2.2: Representação do hiperplano ótimo de separação. Fonte: `https://www.codigofluente.com.br/aula-08-scikit-learn-maquina-de-vetores-de-suporte/`

A formulação mais simples do SVM é a linear, onde o hiperplano se encontra no espaço dos dados de entrada. Neste caso, o espaço de hipóteses é o conjunto de todos os hiperplanos da forma:

$$\mathrm{f}(x) = w \cdot x + b \tag{2.1}$$

onde w é o vetor de peso de X, e o parâmetro b é um escalar que costuma ser chamado de *bias*. A fórmula $w{\bullet}x$ representa o produto escalar de w e x. Para fazer com que o hiperplano classifique corretamente os dados de entrada é necessário satisfazer a seguinte restrição em que i corresponde a cada uma das entradas de dados [LSW14]:

$$y_i \left[(w \bullet x_i + b)\right] \geq 1, i = 1, 2, \ldots, l \tag{2.2}$$

Do ponto de vista geométrico, todo o espaço de entrada é dividido em duas partes por um hiperplano: uma com classe $+1$ e outra com -1. Obviamente o hiperplano aparece em forma de linha quando x pertence a

um espaço bidimensional e em forma de superfície quando x está num espaço tridimensional [LSW14].

Para dados separáveis linearmente, a SVM pode classificá-los diretamente em algumas classes dentro do espaço de entrada [LSW14]. Mas, na aplicação prática, a grande maioria dos dados definidos no espaço original não são separáveis, ou seja, a função linear não pode ser adotada para realizar a classificação correta. Para dados separáveis não linearmente, o SVM deve mapear os dados de entrada originais x (com mapeamento não linear) $(\phi : x \rightarrow f)$ noutro espaço de alta dimensão onde a margem máxima de classificação pode ser resolvida [LSW14].

A função núcleo pode servir como uma ponte da linearidade para a não linearidade para algoritmos que podem ser expressos em termos de produto escalar. No espaço de atributos, uma operação de produto escalar pode ser substituída diretamente pela função núcleo [LSW14]. Algumas funções núcleo são:

$$\text{Núcleo Linear: } K(X_i, X_j) = X_i \bullet X_j \tag{2.3}$$

$$\text{Núcleo Polinomial: } K(X_i, X_j) = (X_i \bullet X_j + 1)^h \tag{2.4}$$

$$\text{Núcleo RBF: } K(X_i, X_j) = \ell^{-\|X_i - X_j\|^2 / 2\sigma^2} \tag{2.5}$$

$$\text{Núcleo Sigmoide: } K(X_i, X_j) = \tanh(\kappa X_i \cdot X_j - \delta) \tag{2.6}$$

A função de núcleo linear é a função de núcleo mais simples fornecida pelo produto interno. A função de núcleo polinomial é adequada para problemas em que todos os dados de treino são normalizados [LSW14]. Na função núcleo da função de base radial gaussiana (rbf), o parâmetro σ tem um papel importante no desempenho desta função núcleo e deve ser cuidadosamente ajustado de acordo com o problema em questão [LSW14]. SVM com função de núcleo Sigmoide é equivalente a uma rede neuronal simples de 2 camadas chamada *Multilayer Perceptron* sem camada oculta [LSW14].

Alguns dos benefícios do uso de SVM's são a sua eficiência em espaços dimensionais elevados, o uso de um subconjunto de pontos de treino na função de decisão (chamados de vetores de suporte) que permite que também seja eficiente em termos de memória e é versátil porque disponibiliza diferentes funções do núcleo que podem ser especificadas para a função de decisão.

Random Forest

O *Random Forest* é um algoritmo de aprendizagem usado para classificação e regressão. Desenvolvido por Breiman [Bre01], o método combina uma abordagem de amostragem por *bagging* de Breiman [Bre96] com a seleção aleatória de características, introduzida independentemente por Ho [Tin95][Tin98] e Amit e Geman [AG97], a fim de construir uma coleção de árvores de decisão com a mesma distribuição.

O *bagging*, é um método que divide o conjunto de treino em vários subconjuntos [FGE14]. O Random Forest treina cada árvore com um desses subconjuntos, e, dessa forma, cada árvore é treinada com um subconjunto de dados diferente e prevê resultados de forma diferente. Cada árvore no conjunto atua como um classificador base para determinar a classe de um *input* não rotulado. Isso é feito através de votação por maioria, onde cada classificador lança um voto para a classe prevista, e a classe com mais votos é usada para classificar os dados [FGE14]. Na figura 2.3 é possível visualizar a estrutura de uma *Random Forest*.

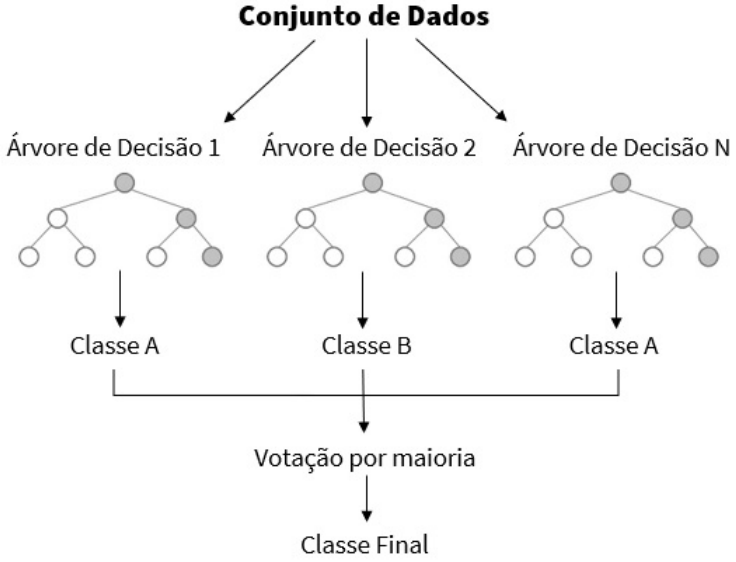

Figura 2.3: Estrutura de uma *Random Forest*

Redes Neuronais Artificiais

Recentemente, as Redes Neuronais Artificiais tornaram-se um modelo popular e útil para classificação, agrupamento e reconhecimento de padrões [AJO+18].

De acordo com Haykin [Hay09], uma Rede Neural Artificial (RNA) pode ser comparável a uma máquina produzida para funcionar da mesma forma que o cérebro humano executa uma determinada tarefa de interesse [AJO+18]. O elemento principal do cérebro humano é o design único da sua capacidade de processamento de informações. Constitui muitos "neurónios" interconectados complexos na forma de elementos que trabalham juntos para resolver problemas específicos diariamente. Um exemplo típico de função de rede neuronal é o cérebro humano que está conectado para enviar e receber sinais de ação humana [AJO+18]. Assim sendo, uma RNA é uma representação de um sistema de "neurónios" interconectados que podem computar um conjunto de valores através de uma função objetiva e produzir o valor de saída desejado [Sar15].

Uma rede neuronal é constituída por várias camadas, cada uma com um número de neurónios parametrizável. Um neurónio consiste numa função, normalmente uma função Sigmoide que usa f como entrada e devolve uma saída binária e um fator de peso e determina quanto esse neurónio é considerado para a saída da camada. Embora os detalhes possam variar entre as redes neuronais, a função $f(x_1, x_2, ..., x_n)$ costuma ser apenas uma soma ponderada:

$$f(x_1, x_2, ..., x_n) = w_1 \cdot x_1 + w_2 \cdot x_2 + ... + w_n \cdot x_n \tag{2.7}$$

Cada neurónio tem um vetor de peso $w = (w_1, w_2, ..., w_n)$, onde n é o número de entradas para esse neurónio. Essas entradas podem ser atributos de entrada "brutos" ou a saída de neurónios de uma camada anterior.

As redes neuronais artificiais supervisionadas aprendem ao receber pares de entrada rotulados, portanto é necessário ter conhecimento prévio do comportamento pretendido [Fer06]. Para cada entrada, o perito indica explicitamente se a resposta calculada é boa ou má. A resposta fornecida pela rede neuronal é

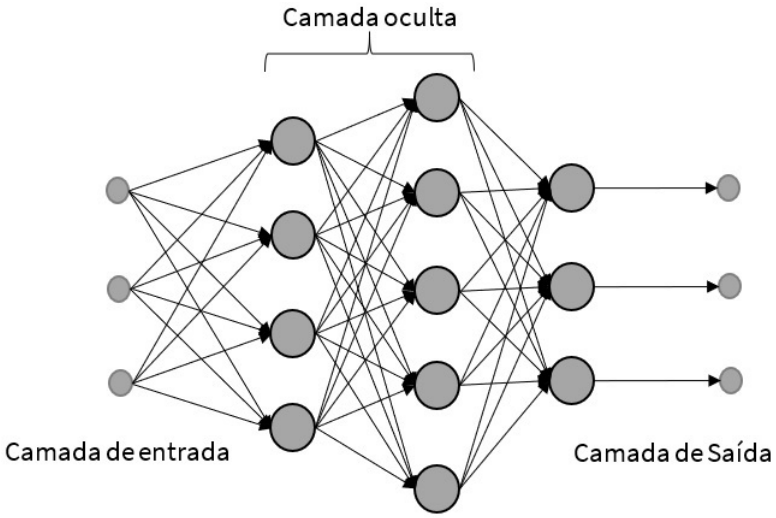

Figura 2.4: Representação de uma rede neuronal

comparada à resposta esperada. O erro verificado é informado à rede para que sejam feitos ajustes aos pesos com o objetivo de minimizar o erro e otimizar o desempenho [Fer06]. Este *fine-tuning* da rede continua até que o conjunto de pesos que minimiza a discrepância entre os valores de saída e os valores de saída pretendidos seja obtido [Fab19].

Uma rede neuronal profunda é definida como uma rede neuronal artificial com várias camadas ocultas de unidades entre as camadas de entrada e saída. Redes profundas também são mais complexas e exigentes em termos de computação [Sar15].

Alguns exemplos de redes neuronais artificiais são as redes neuronais *Long short-term memory* (LSTM) [HS97], utilizadas para reconhecimento de texto e de fala; redes neuronais *feedforward* (FF) [Saz06], utilizadas em reconhecimento de padrões, reconhecimento de caracteres escritos à mão, compressão de dados, entre outros e redes neuronais *Multilayer Perceptron* (MP) [MBPPM09], utilizadas em reconhecimento de fala e tradução de máquina.

2.1.2 Aprendizagem não supervisionada

A aprendizagem não supervisionada é usada quando não é possível obter conjuntos de dados previamente rotulados [Fab19]. Essa impossibilidade ocorre em situações onde não há conhecimento do meio ambiente, ou o custo para adquirir tal conhecimento é muito alto [Fab19]. É um tipo de aprendizagem em que apenas existem variáveis de entrada e que tem como objetivo aprender mais sobre os dados sem que estes estejam rotulados. É uma abordagem utilizada maioritariamente para agrupamento[3] e associação de dados. Rotular grandes quantias de dados é uma tarefa que demora muito tempo e por isso na ausência inicial de rótulos, uma abordagem não supervisionada consegue prever os resultados mais rapidamente do que uma abordagem supervisionada pois não perde esse tempo a rotular os dados.

[3] *clustering*

Clustering

O *clustering* é necessário para diversos fins em diferentes áreas da engenharia, ciência e tecnologia, humanidades, ciências médicas e na nossa vida diária. De acordo com Rokach [RM05], o *clustering* divide os padrões de dados em subconjuntos de forma a que padrões semelhantes sejam agrupados [SPG+17].

O *clustering* de dados é considerado mais complexo do que a classificação supervisionada, pois não há rótulo anexado aos padrões do agrupamento [SPG+17]. O rótulo fornecido no caso da classificação supervisionada torna-se uma pista para agrupar conjuntos de dados como um todo. Já no caso de agrupamento, torna-se difícil decidir a qual grupo um padrão pertencerá, na ausência de um rótulo [SPG+17].

É evidente que a similaridade é o fator central para que um objeto pertença a um *cluster* [SPG+17]. A abordagem mais comum para definir similaridade é defini-la como uma medida de distância entre os padrões, quanto menor a distância (por exemplo, distância euclidiana) entre os dois objetos, maior a similaridade [SPG+17].

Outras abordagens utilizadas no cálculo da similaridade para além da distância euclidiana são a distância coseno, a distância de Manhattan, distância de Jaccard, distância de Chebyshev e a distância de Minkowski [IPP16].

Existem diferentes abordagens de *clustering* associadas a diferentes técnicas [SPG+17]. Isto deve-se ao fato de que não existir uma definição precisa para a noção de *cluster* [SPG+17]. Estas abordagens dividem-se em *clustering* hierárquico e *clustering* parcial [FR98].

Nos métodos de *clustering* hierárquico, os *clusters* são formados pela divisão iterativa dos padrões usando uma abordagem de cima para baixo ou de baixo para cima. Existem duas formas de método hierárquico: agrupamento hierárquico aglomerativo e divisivo [SPG+17].

O *clustering* hierárquico aglomerativo segue a abordagem de baixo para cima, que constrói *clusters* começando com um único objeto e, em seguida, vai misturando esses *clusters* atómicos em *clusters* cada vez maiores, até que todos os objetos estejam finalmente num único *cluster* ou até que certas condições de terminação sejam satisfeitas [SPG+17]. O *clustering* hierárquico divisivo segue a abordagem de cima para baixo, que divide o *cluster* contendo todos os objetos em *clusters* menores, até que cada objeto forme um *cluster* por conta própria ou até que satisfaça certas condições de terminação [SPG+17]. Alguns algoritmos que usam *clustering* hierárquico são o BIRCH [ZRL96] e o CURE [GRS98].

O *clustering* parcial é oposto ao *clustering* hierárquico; aqui os dados são atribuídos a K *clusters* sem qualquer estrutura hierárquica, otimizando uma função critério [SPG+17]. O critério mais utilizado é a distância euclidiana, que encontra a distância mínima entre os pontos com cada um dos *clusters* disponíveis e atribui o ponto ao *cluster* [SPG+17]. Um dos algoritmos utilizados em *clustering* parcial é o K-means [Mac67].

Redes Neuronais Artificiais

Na aprendizagem não supervisionada, como o próprio nome indica, a RNA não está sob a supervisão de um perito [Fab19]. Em vez disso, são fornecidos conjuntos de dados não rotulados (contendo apenas os dados de entrada) e deixamos a RNA encontrar padrões nos dados e construir um novo modelo a partir deles. Nesta situação, a RNA aprende a categorizar os dados, explorando a distância entre os *clusters* dentro dela [Fab19].

As técnicas e algoritmos usados em redes neuronais artificiais não supervisionadas incluem máquinas boltz-

mann restritas e mapas auto-organizados.

Um mapa auto-organizado é uma rede neuronal de camada única com unidades definidas ao longo de uma grelha n-dimensional [Mil17]. A maioria das suas aplicações usa grelha bidimensional e retangular, embora muitas aplicações também usem grelhas hexagonais e alguns espaços de uma, três ou mais dimensões. Os mapas auto-organizados produzem imagens de projeção de baixa dimensão de distribuições de dados de alta dimensão, nas quais as relações de similaridade entre os objetos são preservadas [Mil17].

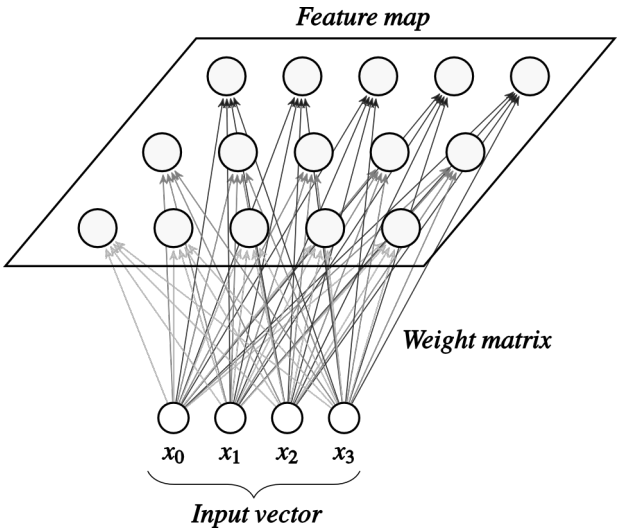

Figura 2.5: Representação de um mapa auto-organizado. Fonte: `http://gorayni.github.io/blog/2014/10/08/som.html`

A Máquina de Boltzmann Restrita (RBM) é uma rede neuronal de duas camadas de unidades estocásticas com conexões não direcionadas entre pares de unidades nas duas camadas. As duas camadas de nós são chamadas de nós visíveis e ocultos. Numa máquina de Boltzmann todas as unidades podem estar conectadas entre si, numa RBM não há conexões de nós visíveis para nós visíveis ou de nós ocultos para nós ocultos [US19]. Esta representa uma distribuição de probabilidade (sobre os estados das unidades visíveis) onde configurações de baixa energia têm maior probabilidade. A energia é determinada pelos pesos de conexão que são os parâmetros a serem aprendidos com os dados para construir um modelo [US19].

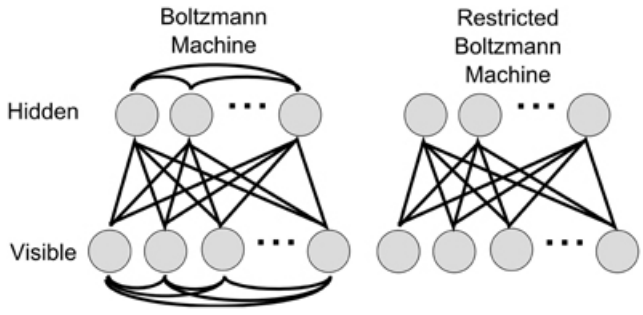

Figura 2.6: Representação de uma máquina de Boltzmann e de uma máquina de Boltzmann restrita. Fonte: https://www.researchgate.net/figure/Boltzmann-and-Restricted-Boltzmann-Machines-A-Boltzmann-machine-is-fully-connected_fig8_257649811

2.1.3 Aprendizagem por Reforço

Aprendizagem por reforço é definida pela aprendizagem por meio da interação com um ambiente. Ao longo desta interação diferentes ações vão sendo tomadas e são consecutivamente dados muitos erros e sucessos, esses erros devem ser identificados e aprendidos através de um mecanismo de recompensa.

O agente não é informado sobre a ação a ser executada [Ham18]. A aprendizagem por reforço é semelhante a processos naturais de aprendizagem onde um perito não está disponível e o processo de aprendizagem evolui com tentativa e erro, diferente da aprendizagem supervisionada, em que um agente precisa ser informado de qual é a ação correta para cada posição que encontra [Ham18]. Dada esta falta de um perito, um agente num jogo pode jogar sem falhas, com a exceção de um erro, e no final do jogo obter um único reforço que diz "Perdeu". O agente deve de alguma forma determinar qual movimento foi o erro e aprender através dele [RN09].

Os algoritmos de aprendizagem por reforço podem ser categorizados em: algoritmos baseados num modelo ou algoritmos sem modelo [Meh20]. Um dos algoritmos baseado em modelo é o I2A [WRR+18]. Atualmente os algoritmos mais utilizados são os algoritmos sem modelo, que se dividem em algoritmos baseados em valor[4] e algoritmos baseados em políticas. Os algoritmos A2C/A3C [MBM+16] são algoritmos baseados em políticas e o algoritmo DQN [MKS+13] é um algoritmo baseado em valor, enquanto que o algoritmo DDPG [LHP+15] é um algoritmo baseado em políticas e em valor.

2.2 Processamento de Linguagem Natural

Processamento de Linguagem Natural é uma área de investigação em Inteligência Artificial que se dedica ao processamento de linguagens naturais (idiomas) tais como o Inglês, Mandarim ou Português. Este processamento geralmente envolve transformar a linguagem em representações de dados que um computador possa usar para realizar a aprendizagem [LHH19].

O objetivo desta área de investigação é conseguir que os computadores realizem tarefas úteis utilizando linguagem humana, tais como habilitar a comunicação máquina-humano, melhorar a comunicação humano-humano ou simplesmente fazer processamento de texto ou fala.

2.2.1 Modelo de Linguagem

Modelação de linguagem é a tarefa de atribuir uma distribuição de probabilidade sobre sequências de palavras que corresponda à distribuição de uma língua como por exemplo o Português [Kal20]. Um modelo de linguagem é necessário para representar o texto de forma perceptível do ponto de vista da máquina e tem a capacidade de prever a probabilidade de ocorrência de uma palavra [Kal20].

A tarefa de atribuição de probabilidades sobre sequências de palavras é essencial para tarefas como o reconhecimento de fala, correção de erros gramaticais, correção ortográfica, tradução ou para sistemas de comunicação alternativos.

[4] *Q-learning*

2.2.2 Representação da Linguagem

Word N-grams

Um N-grama é uma sequência que contém até N elementos que foram extraídos de uma sequência desses elementos, normalmente uma frase. Geralmente os elementos de um N-grama podem ser letras, sílabas, palavras ou símbolos [LHH19].

Os N-gramas existem acima das palavras numa hierarquia natural: qualquer fluxo de linguagem pode ser dividido em n-gramas, da mesma forma que uma palavra pode ser segmentada em morfemas ou letras [SWB13]. N-gramas têm propriedades estatísticas semelhantes a outras unidades: cada n-grama tem uma probabilidade de ocorrer em qualquer ponto no tempo que pode ser estimado empiricamente, e essa probabilidade muda dependendo do contexto. A probabilidade de ocorrência de qualquer n-grama é geralmente estimada a partir da sua frequência de ocorrência num *corpus*, e quanto maior o *corpus*, mais precisa é a estimativa [SWB13].

Um *Word N-gram* é um N-grama composto por N palavras e são definidos como unigramas (compostos apenas por uma palavra), bigramas (compostos por sequências de duas palavras), trigramas (compostos por sequências de três palavras) e assim adiante como podemos observar na figura abaixo.

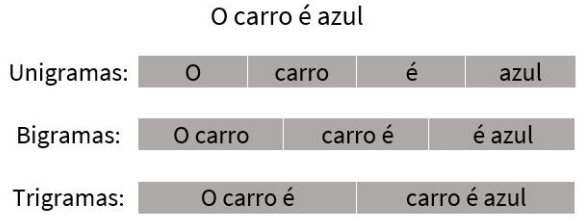

Figura 2.7: Exemplo da divisão de uma frase em N-Gramas

No cálculo de N-gramas, a probabilidade $P(w_1, ..., w_m)$ de observar uma frase $w_1, ..., w_m$ é aproximada por:[5]

$$P\left(w_1, \ldots, w_m\right) = \prod_{i=1}^{m} P\left(w_i \mid w_1, \ldots, w_{i-1}\right) \approx \prod_{i=1}^{m} P\left(w_i \mid w_{i-(n-1)}, \ldots, w_{i-1}\right) \tag{2.8}$$

Supõe-se que a probabilidade de observar a i-ésima palavra w_i na história de contexto das i - 1 palavras anteriores pode ser aproximada pela probabilidade de observá-la na história de contexto encurtada das n - 1 palavras anteriores. A probabilidade condicional pode ser calculada a partir de contagens de frequência do modelo de n-gramas:

$$P\left(w_i \mid w_{i-(n-1)}, \ldots, w_{i-1}\right) = \frac{\text{count}\left(w_{i-(n-1)}, \ldots, w_{i-1}, w_i\right)}{\text{count}\left(w_{i-(n-1)}, \ldots, w_{i-1}\right)} \tag{2.9}$$

Assim sendo, para bigramas e trigramas, a probabilidade da frase 'O carro dela é azul' é definida por:

P(O, carro, dela, é, azul) $\approx P$(O | $\langle s \rangle$)P(carro | O)P(dela | carro)P(é | dela)P(azul | é)P($\langle /s \rangle$ | azul)

P(O, carro, dela, é, azul) $\approx P$(O | $\langle s \rangle, \langle s \rangle$)$P$(carro | $\langle s \rangle, O$)P(dela | O, carro)P(é | carro, dela)P(azul

[5]Fonte: `https://en.wikipedia.org/wiki/Language_model`

| dela, é $)P(\langle/s\rangle$ | é, azul $)$

Sendo que $\langle s\rangle$ corresponde ao ínicio da frase e $\langle/s\rangle$ corresponde ao fim da frase.

TF-IDF Vectorizer O método *Term Frequency Inverse Document Frequency* (TF-IDF) tem sido amplamente utilizado nas áreas de recuperação de informação e mineração de texto para avaliar a importância de cada palavra numa coleção de frases [KG19]. Em particular, é usado para extrair as palavras principais (ou seja, palavras-chave) de frases e calcular graus de semelhança entre frases [KG19].

Essencialmente, o TF-IDF funciona determinando a frequência relativa de palavras num documento ou texto específico em comparação com a proporção inversa dessa palavra em todo o *corpus* do documento/texto. O cálculo do TF-IDF é então calculado através do produto do TF pelo IDF.

Para a frequência do termo (TF) existem várias possibilidades de cálculo, porém a mais simples consiste em calcular o número de ocorrências do termo no documento:[6]

$$\text{tf}(t,d) = f_{t,d} \tag{2.10}$$

A frequência inversa do documento (IDF) é a medida de quanta informação uma palavra fornece e é calculada através da seguinte fórmula:

$$\text{idf}(t,D) = \log \frac{N}{|\{d \in D : t \in d\}|} \tag{2.11}$$

em que:

N: Número total de documentos no *corpus*. N = |D|

|{d ∈D: t ∈d}|: Número de documentos em que o termo aparece

Como tal, o TF-IDF é calculado como:

$$\text{tfidf}(t,d,D) = \text{tf}(t,d) \cdot \text{idf}(t,D) \tag{2.12}$$

Intuitivamente, este cálculo determina a relevância de uma determinada palavra num determinado conjunto de frases. Palavras que são comuns num único ou num pequeno grupo de frases tendem a ter números TF-IDF mais altos do que palavras comuns, como artigos e preposições [JHS02].

Word embeddings

Um *word embedding* pode ser entendido como uma representação de uma palavra num espaço vetorial. Essa representação deve ser feita de forma que palavras com sentidos semelhantes estejam próximas umas das outras nesse mesmo espaço [LY17].

Uma das maiores vantagens de utilizar *word embeddings* é que estes oferecem uma forma prática de realizar transformações num conjunto de palavras, comparado com outras formas tradicionais de tratar textos, como a representação por BoW (*Bag of Words*) [ZJZ10].

Algumas das *frameworks* mais conhecidas atualmente para produzir *word embbedings* são o Word2Vec [MSC+13], GloVe [PSM14], ELMo [PNI+18], ULMFit [HR18], e, mais recentemente, o BERT [DCLT18]

[6] Fonte: `https://en.wikipedia.org/wiki/Tf-idf`

e o Flair [ABV18].

BERT (*Bidirectional Encoder Representations from Transformers*) O BERT usa modelos de linguagem mascarada para permitir representações bidirecionais profundas pré-treinadas. É o primeiro modelo de representação que atinge um alto desempenho num grande conjunto de tarefas a nível de frases e de *tokens*, ultrapassando muitas arquiteturas de tarefas específicas e que faz uso de uma afinação - fine-tuning - de alguns modelos de linguagem em algumas dessas tarefas [SQXH20].

Ao contrário dos *embeddings* estáticos convencionais, as representações do BERT são contextualizadas, ou seja, cada *token* de entrada é representado por um vetor dependente do seu contexto particular de ocorrência. Já em 2020, Constant e Mickus [MCPvD20] observaram que as representações de uma mesma palavra variam também dependendo da posição da frase em que esta ocorre [RKR20].

Ethayarajh [Eth19] mede o quão semelhantes os *embeddings* para palavras idênticas são em cada camada e descobre que as camadas de BERT posteriores produzem representações mais específicas ao contexto. Eles também descobriram que os *embeddings* de BERT ocupam um cone estreito no espaço vetorial e esse efeito aumenta das camadas inferiores para as superiores. Ou seja, duas palavras aleatórias terão, em média, uma similaridade de cosseno muito maior do que o esperado se os *embeddings* forem direcionalmente uniformes (isotrópicos) [RKR20].

O BERT é uma pilha de camadas de codificador do *Transformer* [VSP+17], que consistem em várias "cabeças", ou seja, redes neuronais totalmente conectadas aumentadas com um mecanismo de auto-atenção. Para cada *token* de entrada numa sequência, cada cabeça calcula os vetores de chave, valor e consulta, que são usados para criar uma representação ponderada. As saídas de todas as cabeças na mesma camada são combinadas e passam por uma camada totalmente conectada. Cada camada é envolvida com uma conexão de salto e a normalização da camada é aplicada depois dela [RKR20].

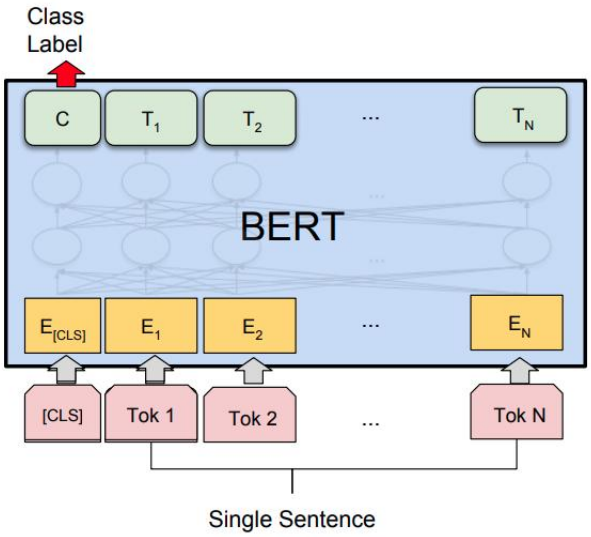

Figura 2.8: Arquitetura do BERT. Fonte: https://www.geeksforgeeks.org/sentiment-classification-using-bert/

O fluxo de trabalho convencional do BERT consiste em duas partes: pré-treino e *fine-tuning*. O pré-treino usa duas tarefas semi supervisionadas: modelação de linguagem mascarada (MLM, previsão de *tokens* de entrada mascarados aleatoriamente) e previsão da próxima frase (prevendo se duas frases de entrada são adjacentes uma à outra) [RKR20]. No *fine tuning* para aplicações *downstream*, uma ou mais camadas

totalmente conectadas são normalmente adicionadas no topo da camada final do codificador [RKR20].

Flair O Flair é um novo modelo de *embeddings* que leva em consideração a distribuição de sequências de caracteres em vez de apenas sequências de palavras, como é o caso dos modelos de linguagem Skip-gram e CBOW [SCS+19]. Em outras palavras, um modelo de Flair *Embedding* é um modelo a nível de palavra que lida não apenas com o contexto de palavras vizinhas, mas também com uma análise de nível de caracteres de palavras vizinhas [SCS+19].

O Flair representa um novo tipo de *embeddings* contextualizados a nível de caracter que pretende combinar os melhores atributos dos *embeddings* previamente mencionados: a capacidade de pré-treinar grandes conjuntos de dados não rotulados; capturar o significado da palavra no contexto e, portanto, produzir diferentes *embeddings* para palavras polissémicas dependendo do uso; e, modelar palavras e contexto fundamentalmente como sequências de caracteres, para lidar melhor com palavras raras e com erros ortográficos, bem como modelar estruturas de sub-palavra, como prefixos e sufixos [ABV18]. O Flair combina dois modelos, um treinado para frente e outro para trás[7] [ABV18].

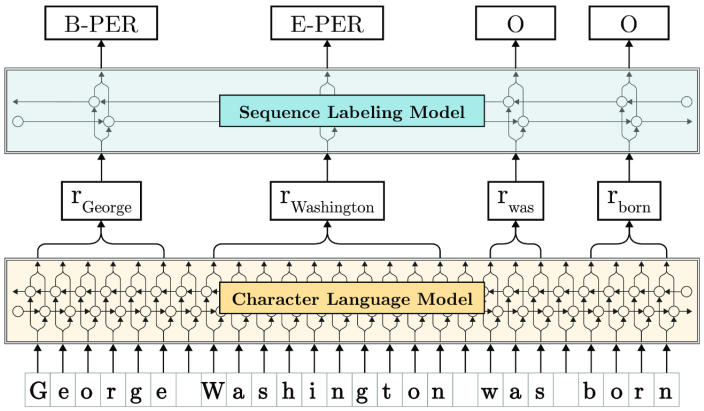

Figura 2.9: Arquitetura do Flair. Fonte: `https://alanakbik.github.io/flair.html`

Treinar um modelo de linguagem Flair consiste em capturar as características sintáticas e semânticas de uma dada linguagem natural de tal forma que qualquer sequência de caracteres será atribuída a um significado contextual [SCS+19]. Os *embeddings* são gerados através de uma rede neuronal chamada *Neural Character-level Language Modeling* (CharLM). O seu principal componente é uma rede LSTM que combina os estados ocultos de uma sequência nas direções para a frente e para trás, aprendendo representações a nível de palavra e de caracter de acordo com o contexto em que são encontradas para cada direção [SCS+19].

2.3 Métricas de avaliação

Para problemas de classificação, a avaliação da melhor solução durante a classificação é definida com base na matriz de confusão, como podemos observar na tabela 2.1 [HM15]. Cada linha representa a classe real, enquanto a coluna representa a classe prevista. A partir dessa matriz de confusão, tp e tn denotam o número de entradas positivas e negativas que estão classificadas corretamente. Enquanto isso, fp e fn denotam o número de entradas negativas e positivas classificadas incorretamente, respectivamente. A partir da matriz de confusão, várias métricas comummente usadas podem ser calculadas conforme é possível observar na tabela 2.2 para avaliar o desempenho do classificador com diferentes focos de avaliação [HM15].

[7] *forward* e *backward*

		Previsto	
		Classe A	**Classe B**
Real	**Classe A**	Verdadeiro Positivo (tp)	Falso Negativo (fn)
	Classe B	Falso Positivo (fp)	Verdadeiro Negativo (tn)

Tabela 2.1: Exemplo de uma matriz de confusão

A exatidão é a métrica de avaliação mais usada na prática para problemas de classificação binários ou multi-classe. Através da exatidão, a qualidade da solução produzida é avaliada com base na percentagem de previsões corretas sobre o número total de objetos de dados. A métrica complementar da exatidão é a taxa de erro que avalia a solução produzida através da sua percentagem de previsões incorretas [HM15].

Métrica	**Fórmula**	**Foco da avaliação**
Exatidão	$\frac{tp+tn}{tp+fp+tn+fn}$	A exatidão mede a percentagem de previsões corretas sobre o número total de entradas avaliadas.
Taxa de erro	$\frac{fp+fn}{tp+fp+tn+fn}$	A taxa de erro mede a percentagem de previsões incorretas sobre o número total de entradas avaliadas.
Precisão (p)	$\frac{tp}{tp+fp}$	A precisão é usada para medir a percentagem de classificações positivas corretas a partir do número total de classificações positivas nessa classe.
Cobertura (r)	$\frac{tp}{tp+fn}$	A cobertura é usado para medir a relação entre as observações positivas classificadas corretamente e o número total de observações nessa mesma classe.
Medida-F	$\frac{2*p*r}{p+r}$	A Medida-F representa a média harmónica entre os valores da precisão e cobertura.

Tabela 2.2: Métricas de avaliação. Fonte: [HM15]

Ambas estas métricas são fáceis de calcular e de entender, e a sua maior vantagem é serem aplicáveis para problemas multi-classe. Porém estas também apresentam limitações nos processos de avaliação. Uma das principais limitações da exatidão é que produz valores menos distintos e menos discrimináveis tendo em conta que vê os resultados como um todo e não classe a classe [HM15]. Consequentemente, leva a menos poder de discriminação na seleção e determinação do classificador ideal. Além disso, a exatidão também é menos favorável para entradas pertencentes a classes minoritárias [HM15].

A precisão e a cobertura são métricas inadequadas para avaliar e selecionar a solução ótima pelo facto de ambas se focarem numa única tarefa de avaliação (falsos positivos ou falsos negativos). Para ultrapassar estas limitações, a Medida-F calcula a média harmónica entre a precisão e a cobertura originando assim um resultado mais consistente e informativo [HM15].

2.3.1 Análise de Significância

Para complementar os valores das métricas de avaliação anteriormente referidas existem alguns testes estatísticos que podem ser realizados sobre os diferentes modelos para perceber se a diferença percentual nos seus resultados é ou não significativa tendo em conta o número total de entradas. Nesta secção apresentamos o Teste de McNemar pois é o teste que mais se adequa à abordagem seguida nesta dissertação.

Teste de McNemar

O teste de McNemar [McN47], é um teste estatístico não paramétrico para comparações emparelhadas que pode ser aplicado para comparar o desempenho de dois classificadores [Ras20].

Frequentemente, o teste de McNemar também é conhecido como "teste qui-quadrado" e é aplicado a dados pareados com base numa versão da tabela de contingência 2x2 que compara as previsões de dois modelos um para o outro [Ras20]. A diferença entre uma matriz de confusão e uma tabela de contingência é que as matrizes de confusão são utilizadas para descrever medidas de performance de um classificador, enquanto que a tabela de contingência é utilizada para descrever dados e relações entre dados. Na figura 2.10 temos a estrutura de uma tabela de contingência.

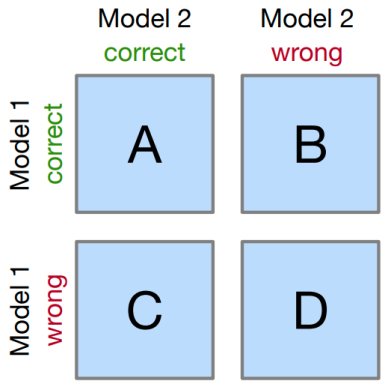

Figura 2.10: Estrutura da tabela de contingência no contexto do teste de McNemar Fonte: [Ras20]

No teste de McNemar, formulamos a hipótese nula de que as probabilidades $p(B)$ e $p(C)$ - onde B e C se referem às células da tabela de contingência introduzidas na figura 2.10 - são iguais, ou em termos simplificados: Nenhum dos dois modelos funciona melhor que o outro. Assim, podemos considerar a hipótese alternativa de que os desempenhos dos dois modelos não são iguais [Ras20].

A estatística do teste de McNemar ("qui-quadrado") pode ser calculada da seguinte forma:

$$\chi^2 = \frac{(|B - C| - 1)^2}{B + C} \tag{2.13}$$

Depois de definir um limite de significância, por exemplo, $\alpha = 0{,}05$, podemos calcular um valor p que, assumindo que a hipótese nula seja verdadeira, o valor p é a probabilidade de observar χ^2 maior ou igual a esse valor. Se o valor p for inferior ao nosso nível de significância escolhido, podemos rejeitar a hipótese nula de que o desempenho dos dois modelos é igual [Ras20].

3

Conjunto de Dados

Ter uma boa base de conhecimento acerca dos dados a utilizar é indispensável para qualquer tarefa. Neste capítulo é feita uma análise detalhada ao conjunto de dados que serviu de base no desenvolvimento da presente dissertação, que foi fornecido e devidamente anonimizado pelas entidades competentes, nomeadamente o SPMS.

O capítulo inicia com a descrição pormenorizada do conjunto de dados e das suas classes e termina com a caracterização dos atributos que o compõem.

3.1 Caracterização das classes

O conjunto de dados é composto por 269 663 entradas todas elas correspondentes a chamadas telefónicas recebidas pela linha de atendimento 24h do Serviço Nacional de Saúde no período entre Janeiro e Março de 2018. Cada uma destas entradas é composta por 18 atributos escritos em português pelo técnico que atendeu a respetiva chamada.

Estas chamadas estão divididas em 53 classes e podemos observar a sua distribuição na tabela 3.1.

Classe	Entradas	%
Tosse	37930	14,066%
Síndrome Gripal	34266	12,707%
Problemas por náuseas e vómitos	14453	5,360%
Dor abdominal	14382	5,333%
Problema da orofaringe	13503	5,007%
Rash	11418	4,234%
Problema de alteração de temperatura corporal	9285	3,443%
Dor torácica	8805	3,265%
Cefaleia	8333	3,090%
Problema Urinário	7992	2,964%
Diarreia	7909	2,933%
Problema do ouvido	7270	2,696%
Problema ocular	6988	2,591%
Problema nasal	5450	2,021%
Problema por tonturas	5397	2,001%
Problema da gravidez, puerpério	4690	1,739%
Problema por lombalgia	4672	1,733%
Problema na face	4509	1,672%
Problema de tensão arterial	4386	1,626%
Problema na cabeça e pescoço	4279	1,587%
Problema de saúde da mulher	4160	1,543%
Problema de integridade cutânea	4100	1,520%
Problemas inespecíficos	3830	1,420%
Problema de Ansiedade	3537	1,312%
Problema por ingestão de substâncias tóxicas	3204	1,188%
Problemas nos membros inferiores – Tornozelo Pé	2985	1,107%
Problemas nos membros inferiores - Anca	2554	0,947%
Problema respiratório	2390	0,886%
Problema de diabetes	2384	0,884%
Problema de obstipação	2222	0,824%
Problemas nos membros inferiores - Joelho	2164	0,802%
Problemas geriátricos	2162	0,802%
Problemas nos membros superiores – Ombro / Clavícula / Braço	2153	0,798%
Problema na criança que chora (0-1 ano)	1928	0,715%
Problema de alteração da cor das fezes	1871	0,694%
Problema nos membros superiores – Punho / Mão	1521	0,564%
Problema de saúde do homem	1445	0,536%
Problema de alergias	1360	0,504%
Problemas nos dedos	1320	0,489%
Problema de desmaio ou lipotimia	944	0,350%
Problema de depressão	876	0,325%
Problemas de reação a vacinação	844	0,313%
Emergência	726	0,269%
Problema da mama	703	0,261%
Problemas das queimaduras	702	0,260%
Problema por corpo estranho (Inalação, Aspiração)	657	0,244%
Problema na amamentação	367	0,136%

Problema de asma ou pieira	333	0,123%
Problemas no cotovelo	137	0,051%
Problemas por sarampo	98	0,036%
Problemas de adaptação em situação de crise	60	0,022%
NA	5	0,002%
Problemas por calor	4	0,001%
Total	**269663**	**100%**

Tabela 3.1: Divisão dos dados por classes

3.2 Caracterização dos Atributos

Um atributo é um campo do conjunto de dados que representa uma característica de um objeto. Neste conjunto de dados cada entrada é composta por 18 atributos, respetivamente: IDEncontro, Número SNS, Centro de Saúde, Data Início, Data Disposição Final, Data Nascimento, Idade, Género, Relação com utente, Intenção Inicial, Motivo Contacto, Último Algoritmo, Disposição final, Comentários, Unidade de Saúde Encaminhamento, Seguimento, Tipologia Interação, Canal. A descrição pormenorizada de cada um destes atributos pode ser observada na tabela 3.2.

Atributo	**Tipo**	**Descrição**	**Valores Possíveis**
IDEncontro	Numérico	ID da chamada	
Número SNS	Numérico	ID do utente	
Centro de Saúde	Categórico	Centro de saúde onde o utente está registado	
Data Início	Data	Data e hora em que a chamada foi recebida	
Data Disposição Final	Data	Data e hora do fim da chamada	
Data Nascimento	Data	Data de nascimento do utente	
Idade	Numérico	Idade do utente	
Género	Categórico	Género do utente	Feminino, Masculino
Relação com utente	Categórico	Relação da pessoa que efetuou a chamada com o utente	Próprio, Pai, Mãe, Conjugue, Filha, Filho, Avós, Outro
Intenção Inicial	Categórico	Intenção inicial do utente	Dirigir-se ao serviço de urgência, Efetuar autocuidados, Contatar 112, Procurar consulta, Procurar profissional de saúde, Outros
Motivo Contacto	Texto Livre	Motivo que levou o utente a efetuar a chamada	
Ultimo Algoritmo	Categórico	Algoritmo final selecionado pelo profissional que atendeu a chamada	
Disposição Final	Categórico	Resultado do algoritmo selecionado	Serviço de Urgência, Autocuidados, Observação medica em CSP, Transferência INEM, Transferência CIAV

Atributo	Tipo	Descrição	Valores Possíveis
Comentários	Texto Livre	Informações adicionais sobre a condição do utente	
Unidade Saúde Encaminhamento	Categórico	Estabelecimento de saúde para onde foi encaminhado	
Seguimento	Categórico	Seguimento do utente após a chamada	Sim , Não
Tipologia Interação	Categórico	Tipo de interação realizado	Triagem
Canal	Categórico	Canal através do qual foi feito o contacto	Telefónico

Tabela 3.2: Descrição dos atributos

Nas figuras seguintes é possível observar a distribuição dos dados pelos diferentes atributos categóricos descritos na tabela 3.2.

Através da figura 3.1 verifica-se que a faixa etária predominante, correspondente a cerca de 28% dos dados em utilização, é dos utentes com idades compreendidas entre os 18 e os 40 anos. A esta faixa etária seguem-se as crianças de idade compreendida entre 1 e 5 anos com cerca de 18% dos registos. Assumindo que a grande maioria das chamadas efetuadas para utentes entre 1 e 5 anos são feitas pelos respetivos pais, isto permite concluir que mais de 40% das chamadas foram efetuadas por indivíduos com idades compreendidas entre 18 e 40 anos.

Através da 3.3 é possível observar três grupos horários distintos na afluência de chamadas ao longo do dia, sendo estes entre as 23 e as 8h, entre as 8 e as 17h, e, entre as 17 e as 23h, sendo estas últimas consideradas as horas de pico de afluência de chamadas, e que foram utilizadas para o decorrer das experiências. Na figura 3.4 temos então a distribuição percentual das horas de pico que correspondem a cerca de 40% do volume de dados utilizado e esta elevada percentagem estará em grande parte dos casos relacionada com o fim do horário de expediente.

Verifica-se através da figura 3.5 que cerca de 48% das chamadas são efetuadas pelo próprio utente, e cerca de 30% são efetuadas pela mãe do utente, e através da figura 3.6 é possível observar que em 34,5% das chamadas recebidas a disposição final do algoritmo clínico resulta em autocuidados, enquanto que cerca de 32% resultam no encaminhamento para o serviço de urgência.

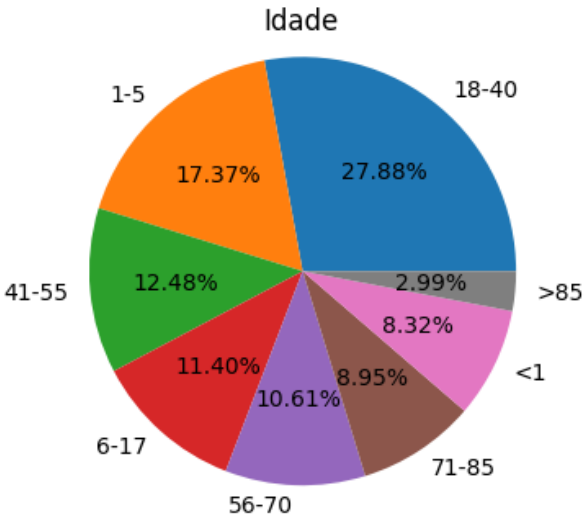

Figura 3.1: Distribuição da idade dos utentes por faixas etárias no conjunto de dados

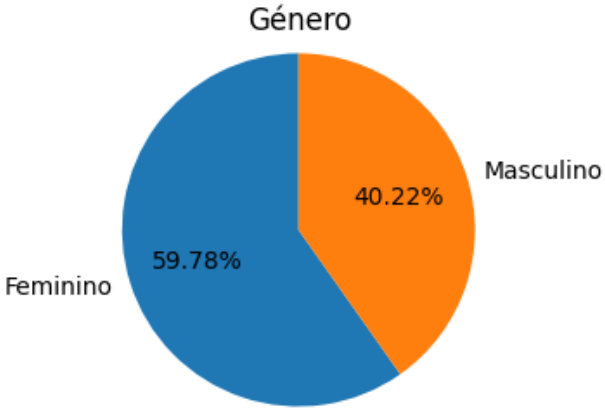

Figura 3.2: Distribuição do género dos utentes no conjunto de dados

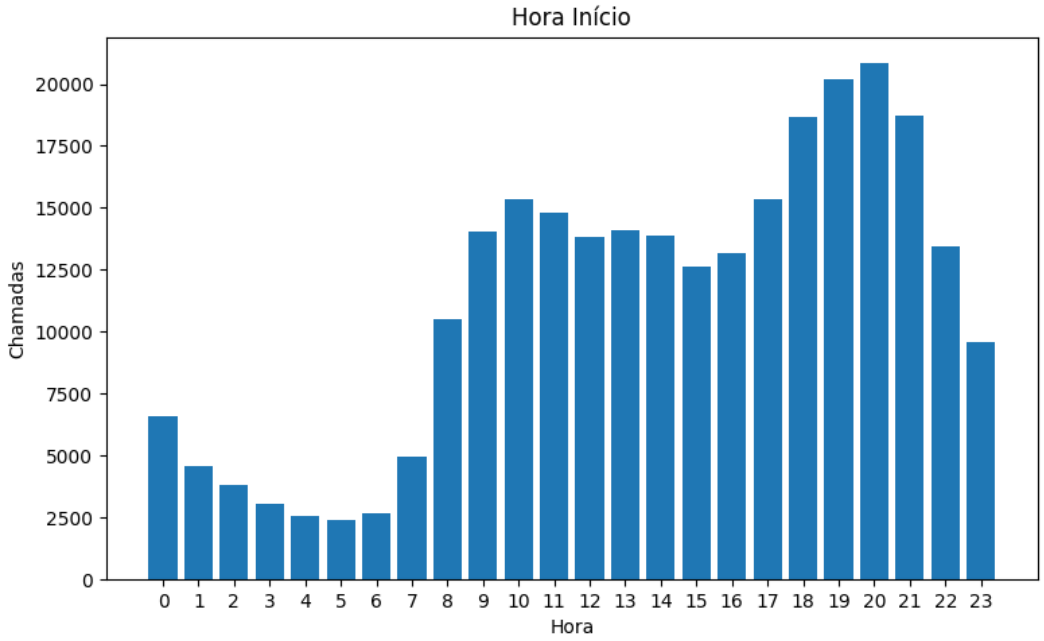

Figura 3.3: Distribuição das chamadas pela hora em que foram efetuadas

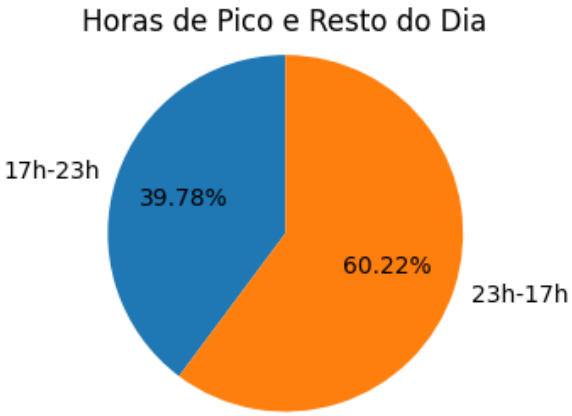

Figura 3.4: Distribuição percentual das chamadas dentro e fora das horas de pico

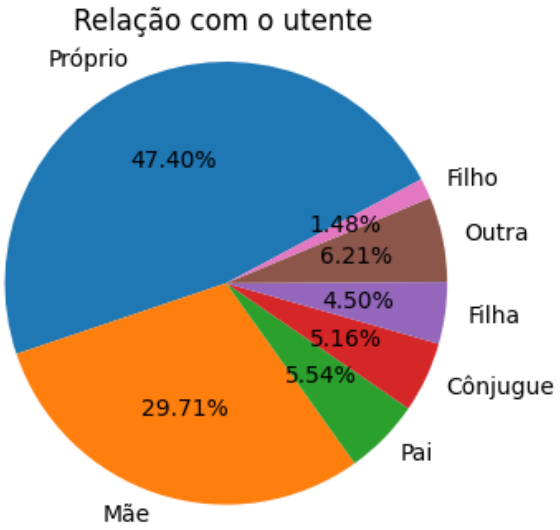

Figura 3.5: Distribuição percentual da relação entre a pessoa que efetua a chamada e o utente

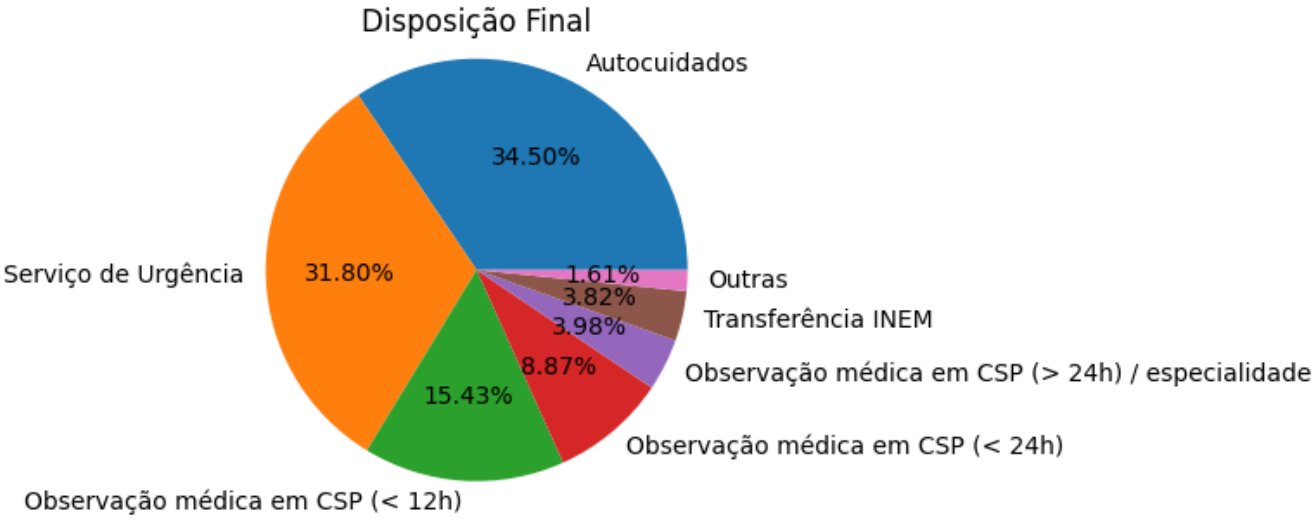

Figura 3.6: Distribuição percentual da disposição final obtida no conjunto de chamadas

4

Metodologia

Definir uma linha de trabalho para a realização de uma tarefa é um fator imprescindível para manter a organização e para evitar desvios do objetivo. A estrutura deste capítulo reflete os diferentes passos seguidos para a criação do modelo de classificação automática com foco nas chamadas telefónicas da linha SNS24.

O capítulo começa com a secção 4.1 correspondente à descrição das ferramentas utilizadas no decurso das experiências, seguida pela secção 4.2 que contém a seleção dos atributos considerados mais relevantes para esta tarefa de classificação. A seguir, a secção 4.3 define o pré-processamento realizado em cada um dos atributos definidos anteriormente. Na secção 4.4 é descrita a divisão estratificada feita ao conjunto de dados nos seus respetivos conjuntos de treino, validação e teste e as suas respetivas utilizações. A secção 4.5 descreve a ordem pela qual as experiências foram realizadas de acordo com as fases pelas quais foram divididas. O conjunto de abordagens de aprendizagem automática selecionadas, bem como as técnicas a aplicar são descritas na secção 4.6. Por último, na secção 4.7 é apresentada a forma como foram interpretados os resultados, e que métricas foram consideradas para a tomada de decisões.

4.1 Ferramentas utilizadas

Todas as experiências realizadas no desenvolvimento desta dissertação foram desenvolvidas na linguagem de programação Python na versão 3.7.9. Nas secções seguintes encontram-se as bibliotecas utilizadas para esse efeito.

4.1.1 Scikit-Learn

O Scikit-learn[1] é uma biblioteca de Python que integra uma ampla variedade de algoritmos de Aprendizagem Automática de última geração, supervisionados e não supervisionados, usando uma interface consistente e orientada a tarefas, permitindo assim uma comparação fácil de métodos para uma determinada abordagem [PVG+11].

Para a realização da tarefa alvo desta dissertação o módulo Scikit-learn (versão 0.23.2) foi utilizado para tarefas de classificação e pré-processamento de dados, com os métodos detalhados em cada uma das secções seguintes.

4.1.2 Transformers

O pacote Hugging Face Transformers[2] é uma biblioteca de Python imensamente popular que fornece modelos pré-treinados - entre estes vários modelos de BERT - que são úteis para uma variedade de tarefas de Processamento de Linguagem Natural. Esta biblioteca foi utilizada na sua versão 3.4.0 para criar word embeddings com um modelo pré-treinado para a língua portuguesa.

4.1.3 Flair

Flair[3] é também uma biblioteca de Python para tarefas de Processamento de Línguagem Natural com modelos pré-treinados disponíveis. Para as experiências selecionadas esta foi utilizada na versão 0.6.1 para criar word embeddings também com um modelo pré-treinado para a língua portuguesa.

4.2 Seleção de Atributos

A escolha dos atributos mais relevantes é uma parte essencial no desenvolvimento de qualquer tarefa de ML. O objetivo de um atributo é fornecer algum conhecimento sobre cada registo de dados. Geralmente, um atributo é um campo do conjunto de dados que possui um valor para cada entrada. Além disso, o tipo de atributo escolhido é um fator importante para a determinação dos algoritmos a utilizar, dado que os diferentes algoritmos lidam com diferentes tipos de dados. Os dois tipos principais são atributos categóricos e numéricos. Os atributos categóricos consistem num conjunto fixo de valores, enquanto que os numéricos são de natureza geralmente contínua e infinita.

O atributo considerado mais relevante foi o "Motivo de Contacto" que foi utilizado em todas as experiências e que permitiu fazer a seleção dos melhores resultados da primeira fase. Na segunda fase foram incremen-

[1] https://scikit-learn.org/stable/
[2] https://huggingface.co/transformers/v3.4.0/
[3] https://github.com/flairNLP/flair

tados gradualmente ao atributo "Motivo de Contacto" os atributos "Comentários", "Idade", "Género" e "Data Inicio". O atributo "Último Algoritmo" contém as classes para a classificação.

4.3 Pré-Processamento

Para efeitos de pré-processamento dos dados começou-se por remover qualquer linha com valores em falta nos atributos considerados de maior importância acima referidos. Após essa deteção de *missing values* cada um dos atributos não categóricos foi sujeito a uma categorização, como é o caso das idades dos utentes, que foram distríbuidas pelas seguintes faixas etárias: $<$1, 1-5, 6-17, 18-40, 41-55, 56-70, 71-85, $>$85; e da hora de início que foi distríbuida pelas seguintes distribuições horárias: Hora de Pico (17h-23h) ou resto do dia (23h-17h). A decisão dos grupos de faixas etárias e distribuições horárias selecionados será detalhada com os respetivos resultados no capítulo 5.

Após todos os atributos a utilizar estarem categorizados foram codificados para poderem ser utilizados pelos diferentes algoritmos selecionados e para essa finalidade foi utilizada a função *OneHotEncoder* da biblioteca scikit-learn que devolve um *array* numérico codificado de acordo com os valores únicos possíveis de cada atributo.

Os atributos de texto livre Motivo de Contacto e Comentários foram processados através das técnicas referidas nas próximas secções, nomeadamente através da utilização de *word n-grams* e *word embeddings*.

4.4 Divisão do Conjunto de Dados

O conjunto de dados foi dividido em três conjuntos, um conjunto de treino para induzir o classificador, um conjunto de validação e um conjunto de teste para avaliar o desempenho do classificador. Estes conjuntos foram divididos através do método train_test_split da biblioteca scikit-learn. Os dados foram divididos com uma distribuição de 64% treino, 16% validação e 20% teste. Como tal foi feita uma divisão estratificada de modo a manter o número de entradas por classe balanceado em todos os conjuntos como podemos observar na tabela 4.1. As classes "NA" e "Problemas por calor" foram removidas do conjunto de dados pelo seu número insuficiente de amostras.

Nesta tarefa de classificação o conjunto de validação foi utilizado na primeira e segunda fase de experiências, sendo que o conjunto de teste apenas foi utilizado na fase final.

Classes	**Treino**	**Validação**	**Teste**
Tosse	24275	6069	7586
Síndrome Gripal	21930	5483	6853
Problemas por náuseas e vómitos	9250	2312	2891
Dor abdominal	9205	2301	2876
Problema da orofaringe	8642	2160	2701
Rash	7307	1827	2284
Problema de alteração de temperatura corporal	5942	1486	1857
Dor torácica	5635	1409	1761
Cefaleia	5333	1333	1677
Problema Urinário	5115	1279	1598
Diarreia	5062	1265	1582
Problema do ouvido	4653	1163	1454
Problema ocular	4472	1118	1398

Classes	Treino	Validação	Teste
Problema nasal	3488	872	1090
Problema por tonturas	3454	864	1079
Problema da gravidez, puerpério	3002	750	938
Problema por lombalgia	2990	748	934
Problema na face	2886	721	902
Problema de tensão arterial	2807	702	877
Problema na cabeça e pescoço	2738	685	856
Problema de saúde da mulher	2662	666	832
Problema de integridade cutânea	2624	656	820
Problemas inespecíficos	2451	613	766
Problema de Ansiedade	2264	566	707
Problema por ingestão de substâncias tóxicas	2050	513	641
Problemas nos membros inferiores – Tornozelo Pé	1910	478	597
Problemas nos membros inferiores - Anca	1634	409	511
Problema respiratório	1530	382	478
Problema de diabetes	1526	381	477
Problema de obstipação	1422	356	444
Problemas nos membros inferiores - Joelho	1385	346	433
Problemas geriátricos	1384	346	432
Problemas nos membros superiores – Ombro / Clavícula / Braço	1378	344	431
Problema na criança que chora (0-1 ano)	1234	308	386
Problema de alteração da cor das fezes	1198	299	374
Problema nos membros superiores – Punho / Mão	974	243	304
Problema de saúde do homem	925	231	289
Problema de alergias	870	218	272
Problemas nos dedos	845	211	264
Problema de desmaio ou lipotimia	604	151	189
Problema de depressão	561	140	175
Problemas de reação a vacinação	540	135	169
Emergência	465	116	145
Problema da mama	450	112	141
Problemas das queimaduras	450	112	140
Problema por corpo estranho (Inalação, Aspiração)	421	105	131
Problema na amamentação	235	59	73
Problema de asma ou pieira	213	53	67
Problemas no cotovelo	88	22	27
Problemas por sarampo	62	16	20
Problemas de adaptação em situação de crise	38	10	12
	172579	**43144**	**53931**
Total	**269 654**		

Tabela 4.1: Divisão estratificada das classes

4.5 Abordagem ao Problema

O problema de classificação alvo desta dissertação é um problema de classificação multi-classe pois existem 51 classes em vez de apenas duas e estamos perante um problema de aprendizagem supervisionada dado que os dados se encontram rotulados. A abordagem a este problema foi dividida em três fases:

Na primeira fase foram utilizados todos os algoritmos e técnicas descritas na próxima secção apenas com o atributo Motivo de Contacto, resultando num total de 40 experiências. Após a realização deste conjunto de experiências foram selecionadas as três abordagens que produziram melhores resultados, com o auxílio dos testes de McNemar, e foi feita uma otimização de parâmetros dos respetivos algoritmos fazendo uso da funcionalidade *GridSearch* da biblioteca scikit-learn.

A segunda fase corresponde à adição gradual de novos atributos à classificação das experiências com melhores resultados da primeira fase, após a otimização de parâmetros. Os atributos adicionados e o respetivo pré-processamento encontram-se na secção 4.3.

A terceira e última fase, corresponde às experiências finais realizadas com a melhor abordagem resultante das etapas anteriores, nomeadamente o *fine-tuning* do modelo utilizado e o cálculo de desempenho para as classes do *top* 3 e do *top* 5. A fase 3 é a única na qual foi utilizado o conjunto de teste para a classificação. Nas restantes fases a classificação é sempre referente ao conjunto de validação.

4.6 Conjunto de Experiências

O conjunto de experiências realizadas corresponde à junção entre os algoritmos e as técnicas abaixo detalhados.

4.6.1 Algoritmos selecionados

Os algoritmos selecionados para a realização do conjunto de experiências foram:

- SVM's

  As SVM's foram selecionadas por serem um dos algoritmos de ML mais utilizados e porque tal como foi anteriormente referido na secção 2.1.1 as SVM's são flexíveis graças à sua eficiência em espaços dimensionais elevados como é o caso do nosso conjunto de dados. Os núcleos selecionados para a realização das esperiências foram o rbf e o linear através das bibliotecas SVC e *LinearSVC* do scikit-learn.

- *Random Forest*

  O *Random Forest* para além de já ser um algoritmo eficiente a nível de custo computacional outra grande vantagem é o facto de permitir ajustar os parâmetros de modo a paralelizar os processos e fazer uso de vários processadores resultando assim num tempo de processamento ainda menor. Em Python o algoritmo é usado através do *RandomForestClassifier* disponibilizado pela biblioteca scikit-learn e a paralelização é efetuada ajustando o parâmetro "n_jobs" para um número fixo de processadores ou para -1 no caso de pretender usar todos os processadores disponíveis.

- *Multinomial Bayes*

  O modelo *Naive Bayes* trabalha com a probabilidade condicional que se origina da conhecida abordagem estatística "Teorema de Bayes", que se refere à "suposição" de que todos os atributos dos

exemplos são independentes um do outro dado o contexto da classe [MN01]. Por causa dessa suposição de independência, os parâmetros para cada atributo podem ser aprendidos separadamente e isso simplifica muito a aprendizagem, especialmente quando o número de atributos é grande [MN01].

O modelo multinomial faz uma suposição semelhante de Bayes: que a probabilidade de ocorrência de uma palavra num documento é independente do contexto da palavra e da posição no documento [MN01]. Este modelo foi selecionado pela sua alta aplicabilidade em tarefas de classificação de texto, e porque a suposição em que se baseia, mesmo não sendo verdadeira nas tarefas do mundo real, gera bons resultados em tarefas de classificação.

Como o *Multinomial Bayes* é um algoritmo que não lida com valores de entrada negativos todos os dados foram normalizados através do método *MinMaxScaler* da biblioteca scikit-learn.

4.6.2 Técnicas utilizadas

Word N-Grams

Foram realizadas experiências de *Word N-Grams* para N=1,2,3,4 e para todos os valores de N foram usados ambos o *CountVectorizer* e o *TF-IDFVectorizer* através das bibliotecas do scikit-learn *CountVectorizer* e *TfidfVectorizer*. Na tabela 4.2 é possível observar o número de N-Gramas do conjunto de treino para o atributo Motivo de Contacto usado em todas as experiências.

Número de N-Gramas	
N = 1	23 762
N = 2	160 434
N = 3	359 064
N = 4	478 626

Tabela 4.2: Número de N-Gramas do atributo Motivo de Contacto

BERT

O BERT (secção 2.2.2) foi selecionado graças à sua capacidade de identificar os contextos das palavras ao invés de criar o mesmo vetor para a mesma palavra independentemente das palavras que a rodeiam. Apesar de ser um modelo com mais elevado custo computacional este compensa graças aos seus modelos pré-treinados em quantidades de dados gigantes que têm vindo a obter bons resultados tanto em pequenos como grandes conjuntos de dados, como é o exemplo do SQuAD [DCLT18].

Foi utilizado um modelo BERT pré-treinado para a língua portuguesa que inclui dois modelos:

BERT_BASE (L=12, H=768, A=12, Número de Parâmetros=110M)

BERT_LARGE (L=24, H=1024, A=16, Número de Parâmetros=340M)

em que:

L: Número de camadas,

H: Tamanho oculto,

A: Número de cabeças de atenção[4]

Ambos os modelos se encontram disponíveis em: `https://github.com/neuralmind-ai/portuguese-bert?fbclid=IwAR29Yi4wA2MU5d-XiSzSFbp-hYa6zMypBfIAyo9gr5--AQ_P9qnbNuLAGdo`

Para a realização das experiências deste projeto foi utilizado apenas o modelo BERT_LARGE.

Flair

O Flair (secção 2.2.2) tal como o BERT foi selecionado devido à sua capacidade de obter a contextualidade não só das palavras mas também a nível de caracteres. É um modelo que se tem vindo a tornar estado da arte na área de Processamento de Língua Natural em várias tarefas e em várias línguas, entre estas o Inglês, Alemão e Português como é possível observar em [SCS+19] e [ABV18].

Para o Flair também foi utilizado um modelo pré-treinado para a língua portuguesa que inclui os modelos forward e backward.

Ambos os modelos se encontram disponíveis em: `https://github.com/jneto04/ner-pt`

4.7 Interpretação dos resultados

Para determinar que experiências devem ser aprofundadas após a execução do primeiro conjunto de experiências, são necessárias métricas para avaliar os resultados dos diferentes algoritmos. Porém, como a exatidão apenas fornece a quantidade de entradas classificadas corretamente relativamente ao número total de entradas do conjunto completo esta medida pode nem sempre ser a mais adequada para a tomada de decisões. Para neutralizar isso, a cobertura, que representa a relação entre as observações positivas classificadas corretamente numa classe e todas as observações nessa mesma classe, pode ser considerado em combinação com a precisão, que representa a proporção de observações positivas classificadas corretamente em relação ao total de observações positivas classificadas nessa classe. Para combinar os dois valores, é utilizado a Medida-F, que calcula a média harmónica entre os dois. A medida-F foi calculada para todas as classes e os resultados mais promissores da média balanceada em relação à medida-F e à exatidão foram então selecionados.

[4] *Attention heads*

5

Resultados

Neste capítulo são apresentados os resultados obtidos a partir das técnicas referidas no capítulo 4. Este capítulo inicia com a secção 5.1 com os resultados da primeira fase de experiências correspondentes à utilização do atributo Motivo de Contacto. Nesta secção são também apresentados os resultados da otimização de parâmetros para os melhores resultados. Na secção 5.2 são apresentados os resultados da segunda fase de experiências que corresponde à incrementação gradual de novos atributos às experiências que obtiveram melhores resultados na primeira fase. Para terminar as experiências principais, a secção 5.3 apresenta os resultados das experiências finais realizadas com a melhor abordagem encontrada. Na secção 5.4 serão apresentados os resultados de algumas experiências complementares que auxiliaram algumas decisões ao longo do percurso. Finalmente, o capítulo termina com a discussão dos resultados na secção 5.5.

Todos os resultados apresentados neste capítulo são referentes à média pesada de todas as classes[1]. Até à fase de otimização de parâmetros todos os algoritmos são usados com os seus parâmetros *default* e todos os resultados apresentados nas fases 1 e 2 são relativos ao conjunto de validação. O conjunto de teste apenas foi utilizado na fase 3, correspondente à fase final das experiências.

[1] weighted avg

5.1 Fase 1

Tal como mencionado anteriormente a primeira fase de experiências corresponde à junção de todos os algoritmos e técnicas detalhados na secção 4.6 apenas com a utilização do atributo de texto livre Motivo de Contacto. Como tal nesta secção serão apresentados os resultados de 40 experiências e esses resultados serão organizados em secções pelos diferentes algoritmos.

Para cada uma das tabelas apresentadas nesta secção foi efetuado o teste de McNemar com um nível de significância de 0,05, que permite fazer uma análise entre dois resultados distintos. Este teste foi feito entre o melhor e segundo melhor resultado observado em cada tabela, e foram assinalados os melhores resultados cuja diferença percentual para com os restantes se mostrou significativa a negrito. Os resultados destas análises podem ser observados no Apêndice A.

5.1.1 SVM's

Nas tabelas 5.1 e 5.2 encontram-se os resultados referentes a todas as experiências nas quais se utilizaram as SVM's com os núcleos rbf e linear respetivamente.

		SVC			
		Exatidão	**Precisão**	**Cobertura**	**F1**
Word N-grams (N = 1)	**Count**	76,15%	76,89%	76,15%	76,10%
	Tfidf	**76,97%**	**77,28%**	**76,97%**	**76,83%**
Word N-grams (N = 2)	**Count**	68,07%	68,74%	68,07%	67,40%
	Tfidf	70,27%	70,48%	70,27%	69,60%
Word N-grams (N = 3)	**Count**	51,63%	62,12%	51,63%	51,63%
	Tfidf	55,80%	63,72%	55,80%	55,15%
Word N-grams (N = 4)	**Count**	32,88%	62,59%	32,88%	32,57%
	Tfidf	34,75%	62,11%	34,75%	36,01%
BERT Large		75,83%	75,81%	75.84%	75.52%
Flair		76,58%	76,78%	76,59%	76,30%

Tabela 5.1: Resultados das diferentes técnicas com SVM's com núcleo rbf

		LinearSVC			
		Exatidão	**Precisão**	**Cobertura**	**F1**
Word N-grams (N = 1)	**Count**	76,28%	76,05%	76,28%	75,99%
	Tfidf	76,47%	76,14%	76,47%	76,10%
Word N-grams (N = 2)	**Count**	72,03%	71,74%	72,03%	71,72%
	Tfidf	73,34%	72,90%	73,34%	72,88%
Word N-grams (N = 3)	**Count**	61,23%	62,59%	61,23%	60,99%
	Tfidf	62,56%	63,46%	65,56%	62,07%
Word N-grams (N = 4)	**Count**	43,77%	54,17%	43,77%	44,64%
	Tfidf	44,75%	54,78%	44,75%	45,31%
BERT Large		76,39%	76,16%	76,39%	76,04%
Flair		**77,96%**	**77,49%**	**77,96%**	**77,51%**

Tabela 5.2: Resultados das diferentes técnicas com SVM's com núcleo linear

Como nas SVM's foram utilizados dois núcleos distintos foi efetuado o teste de McNemar, com nível de significância de 0,05, entre as melhores técnicas de ambas as tabelas. Os resultados deste teste (apêndice A) indicaram que em todos os casos existe diferença percentual significativa, sendo possível concluir que as SVM's lineares resultam num melhor desempenho ao nível de *word embeddings* e de N-Gramas, com exceção aos unigramas (N=1).

5.1.2 *Random Forest*

Os resultados referentes a todas as experiências realizadas com *Random Forest* podem ser observados na tabela 5.3. O melhor desempenho obtido com este algoritmo foi para os *Word N-Grams* (N=1) porém este resultado foi inferior aos anteriormente obtidos.

		Exatidão	Precisão	Cobertura	F1
Word N-grams (N = 1)	**Count**	73,84%	73,33%	73,84%	73,26%
	Tfidf	**74,93%**	**74,55%**	**74,93%**	**74,40%**
Word N-grams (N = 2)	**Count**	66,51%	66,14%	66,51%	65,54%
	Tfidf	68,10%	67,70%	68,11%	67,41%
Word N-grams (N = 3)	**Count**	55,83%	58,61%	55,83%	55,73%
	Tfidf	56,73%	58,98%	56,73%	56,39%
Word N-grams (N = 4)	**Count**	39,67%	58,15%	39,67%	44,67%
	Tfidf	39,82%	57,61%	39,82%	44,65%
BERT Large		69,39%	68,94%	69,39%	68,16%
Flair		68,36%	68,43%	68,37%	66,96%

Tabela 5.3: Resultados das diferentes técnicas com *Random Forest*

5.1.3 *Multinomial Bayes*

As experiências com *Multinomial Bayes* obtiveram resultados muito inferiores a qualquer uma das abordagens anteriores, e, como tal não foram utilizados nas próximas fases de desenvolvimento. Estes podem ser observados na tabela 5.4

		Exatidão	Precisão	Cobertura	F1
Word N-grams (N = 1)	**Count**	**66,26%**	**67,68%**	**66,26%**	**63,85%**
	Tfidf	57,83%	63,10%	57,83%	53,59%
Word N-grams (N = 2)	**Count**	60,09%	64,85%	60,09%	57,32%
	Tfidf	52,58%	65,96%	52,58%	49,35%
Word N-grams (N = 3)	**Count**	51,35%	62,37%	51,35%	49,35%
	Tfidf	42,70%	66,43%	42,70%	40,55%
Word N-grams (N = 4)	**Count**	36,18%	60,52%	36,18%	34,74%
	Tfidf	29,45%	62,11%	29,45%	26,20%
BERT Large		58,73%	60,16%	58,74%	57,73%
Flair		47,78%	55,54%	47,78%	45,77%

Tabela 5.4: Resultados das diferentes técnicas com *Multinomial Bayes*

5.1.4 Otimização de Parâmetros

A otimização dos parâmetros é um processo importante pois permite maximizar os resultados dentro dos algoritmos utilizados descobrindo os valores ótimos para cada parâmetro. Esta otimização foi efetuada para o grupo de experiências que obteve melhores resultados na primeira fase através da função GridSearchCV da biblioteca scikit-learn e cujo objetivo foi maximizar o resultado da métrica F1. Como tal foram realizadas otimizações nas três experiências seguintes:

- **Flair + SVM com núcleo linear**

  Valores testados: C = [0.1, 0.01, 0.1, 1, 10, 100, 1000]

  Parâmetros *Default*: C=1

  Parâmetros Ótimos: C=1

  No caso do Flair os parâmetros ótimos obtidos foram os parâmetros *default* anteriormente usados como podemos observar de seguida na tabela 5.5.

	Exatidão	**Precisão**	**Cobertura**	**F1**
Sem otimização	77,96%	77,49%	77,96%	77,51%
Com otimização	77,96%	77,49%	77,96%	77,51%

Tabela 5.5: Resultados da optimização de parâmetros para o Flair

- **BERT + SVM com núcleo linear**

  Valores testados: C = [0.1, 0.01, 0.1, 1, 10, 100, 1000]

  Parâmetros *Default*: C=1

  Parâmetros Ótimos: C=0.1

  Na tabela 5.6 podemos observar os resultados antes e após a otimização.

	Exatidão	**Precisão**	**Cobertura**	**F1**
Sem otimização	76,39%	76,16%	76,39%	76,04%
Com otimização	77,10%	76,54%	77,10%	76,61%

Tabela 5.6: Resultados da otimização de parâmetros para o BERT

  Estes resultados foram alvo de uma análise de significância (secção A.1.3) que mostrou que existe diferença significativa de desempenho entre ambos.

- ***Word N-Grams* (N=1) com Tfidf + SVM com núcleo rbf**

  Valores testados: C = [0.1, 0.01, 0.1, 1, 10, 100, 1000] e gamma = ['scale', 'auto', 1, 0.1, 0.01, 0.001]

  Parâmetros *Default*: C=1, Gamma = 'scale'

  Parâmetros Ótimos: C=1, Gamma = 'scale'

  Para os N-Gramas o melhor resultado foi obtido com os parâmetros *default* da SVM como podemos observar na tabela 5.7.

	Exatidão	Precisão	Cobertura	F1
Sem otimização	76,97%	77,28%	76,97%	76,83%
Com otimização	76,97%	77,28%	76,97%	76,83%

Tabela 5.7: Resultados da otimização de parâmetros para os N-Gramas

5.2 Fase 2

A Fase 2 corresponde à incrementação gradual de novos atributos às melhores experiências da primeira fase após a otimização de parâmetros dos seus algoritmos. A tabela 5.8 resume os melhores resultados obtidos na Fase 1[2], e que foram alvo de análises de significância entre si, cujos resultados podem ser observados na secção A.1.4. Estas foram então as três experiências utilizadas como referência para iniciar esta nova fase de experiências.

		Exatidão	F1
1	**Flair + SVM com núcleo linear**	**77,96%**	**77,51%**
2	Word N-Grams (N=1) com núcleo rbf	76,97%	76,83%
3	BERT + SVM com núcleo linear	77,10%	76,61%

Tabela 5.8: Resultados finais obtidos na Fase 1

5.2.1 Flair e BERT

Nas classificações efetuadas com o Flair e com o BERT com SVM Linear, foi incrementado em primeiro lugar o atributo Comentários por ser o atributo de texto livre restante de entre os atributos selecionados. Como podemos ver nas tabelas 5.9 e 5.10 nenhuma das abordagens obteve melhoria de desempenho, e, como tal, não foi efetuada nenhuma adição de novos atributos posteriormente a esta, devido ao elevado custo computacional que estes requerem na geração dos *embeddings*. Tendo em conta esta decisão de não proceder com a adição de mais atributos também se optou por não efetuar uma análise de significância entre os resultados obtidos pois estes não serão posteriormente utilizados em nenhuma das experiências seguintes.

	Flair			
	Exatidão	**Precisão**	**Cobertura**	**F1**
Motivo Contacto	77,96%	77,49%	77,96%	77,51%
Motivo Contacto Comentários	77,06%	76,49%	77,06%	76,50%

Tabela 5.9: Resultados obtidos na incrementação de atributos na classificação com Flair e SVM's com núcleo linear

[2]As matrizes de confusão podem ser consultadas em `https://drive.google.com/drive/folders/1ZnqJxQ7G-VVTOVzz0VmPQXq3OB_8OA8O?usp=sharing`

	BERT			
	Exatidão	**Precisão**	**Cobertura**	**F1**
Motivo Contacto	77,10%	76,54%	77,10%	76,61%
Motivo Contacto Comentários	75,36%	74,64%	75,36%	74,66%

Tabela 5.10: Resultados obtidos na incrementação de atributos na classificação com BERT e SVM's com núcleo linear

5.2.2 *Word N-Grams* (N=1) com Tfidf

Os resultados da incrementação de atributos na classificação com Unigramas com Tfidf e SVM's encontram-se na tabela 5.11. Estes foram divididos em diferentes fases, inicialmente incrementando os quatro atributos individualmente ao Motivo de Contacto. Após estas 4 experiências a que deu melhor resultado foi usada para as seguintes adições e assim sucessivamente.

	Exatidão	**Precisão**	**Cobertura**	**F1**
Motivo Contacto	76,97%	77,28%	76,97%	76,83%
Motivo Contacto Comentários	77,15%	77,18%	77,15%	76,94%
Motivo Contacto Idade	75,99%	76,51%	76,00%	75,88%
Motivo Contacto Género	76,42%	76,84%	76,42%	76,30%
Motivo Contacto Hora do Dia	76,33%	76,84%	76,33%	76,24%
Motivo Contacto Comentários Idade	75,94%	76,00%	75,94%	75,70%
Motivo Contacto Comentários Género	76,72%	76,70%	76,72%	76,48%
Motivo Contacto Comentários Hora do Dia	76,71%	76,80%	76,71%	76,48%
Motivo Contacto Comentários Género Idade	75,58%	75,69%	75,58%	75,34%
Motivo Contacto Comentários Género Hora do Dia	76,35%	76,46%	76,35%	76,11%
Motivo Contacto Comentários Género Idade Hora do Dia	75,13%	75,35%	75,13%	74,88%

Tabela 5.11: Resultados obtidos na adição gradual de atributos à classificação com *Word N-Grams* (N=1) e SVM's com núcleo rbf

Como é possível observar o único atributo que contribuiu para uma melhoria de resultados na classificação foi o campo dos Comentários sendo que quantos mais atributos iam sendo adicionados menor eficácia ia sendo obtida. Isto indica que nenhum dos restantes atributos contém capacidade discriminatória suficiente para ajudar na classificação deste conjunto de dados. Tendo em conta de que a única melhoria obtida foi mínima, aumentou o custo computacional, e, continuou a ser inferior ao resultado previamente obtido com Flair e SVM Linear (tabela 5.8), não foram efetuadas análises de significância entre modelos nesta fase.

5.3 Fase 3

A fase 3 representa a fase final de experiências para o melhor resultado obtido até ao momento. A abordagem selecionada para esta última fase foi com Flair e SVM Linear pois apesar do custo computacional ser ligeiramente mais elevado, o desempenho continua a ter uma diferença significativa quando comparado à alternativa de menor custo computacional, neste caso Word N-Grams (N=1) com Tfidf e SVM (Os resultados desta análise de significância podem ser observados na secção A.1.4).

Tal como foi referido anteriormente, todos os resultados apresentados na Fase 1 e 2 são referentes ao conjunto de validação, sendo que o conjunto de teste será aplicado apenas ao melhor resultado e apresentado nesta secção. De modo a maximizar o resultado anteriormente obtido foi realizada uma afinação - *fine-tuning* - ao modelo Flair pré-treinado para a língua portuguesa utilizado, dado ser o melhor resultado obtido nas fases anteriores, de modo a que fique mais adaptado ao domínio clínico do nosso conjunto de dados. Os resultados do *fine-tuning* serão apresentados na secção 5.3.1 e na secção 5.3.2 serão apresentados os resultados do desempenho das duas melhores abordagens obtidas anteriormente para a apresentação de 3 ou 5 classes no classificador automático.

5.3.1 *Fine-tuning*

Dado que o conjunto de dados utilizado possui cerca de 2,3 milhões de *tokens* no atributo Motivo de Contacto considerou-se relevante realizar o *fine-tuning* do modelo Flair para a língua portuguesa usado nas experiências anteriores de modo a adaptá-lo ao domínio do nosso conjunto de dados. Para esse efeito foi realizado um primeiro treino para 20 épocas, porém o valor da perplexidade[3] começou a repetir-se a partir das 18 épocas sendo que o treino final foi fixado nas 18 épocas e resultou nas seguintes métricas:

Forward TEST: valid loss 0.65 | valid ppl 1.92

Backward TEST: valid loss 0.66 | valid ppl 1.93

Este novo modelo foi utilizado no conjunto de validação e posteriormente no conjunto de teste e os resultados podem ser observados na tabela 5.12.[4]

[3] valid ppl

[4] As matrizes de confusão podem ser consultadas em `https://drive.google.com/drive/folders/1ZnqJxQ7G-VVTOVzz0VmPQXq3OB_8OA8O?usp=sharing`

		Exatidão	Precisão	Cobertura	F1
Validação	**Sem *fine-tuning***	77,96%	77,49%	77,96%	77,51%
Validação	**Com *fine-tuning***	78,59%	78,18%	78,59%	78,21%
Teste	**Com *fine-tuning***	78,80%	78,42%	78,80%	78,45%

Tabela 5.12: Resultados obtidos com o *fine-tuning* do modelo Flair

Para validar a melhoria de desempenho do modelo ao efetuar o *fine-tuning* do mesmo foi efetuada uma análise de significância entre ambas as classificações efetuadas com o conjunto de validação. Este resultado encontra-se na secção A.1.5 e indica que existe diferença percentual significativa entre ambos e que este *fine-tuning* permitiu de facto melhorar o desempenho do modelo apresentado.

5.3.2 *Top* 3 e *Top* 5

Para finalizar os resultados obtidos é possível observar nas tabelas 5.13 e 5.14 o desempenho da abordagem com unigramas (N = 1) com Tfidf e SVM com núcleo rbf e da abordagem com Flair e SVM Linear na apresentação das classes do *Top* 3 e do *Top* 5.

Word N-Grams (N = 1) + SVM		
Top 1	**Top 3**	**Top 5**
76,97%	94,08%	96,77%

Tabela 5.13: Valores da exatidão obtida de acordo com o número de classes consideradas para unigramas com Tfidf e SVM com núcleo rbf

Flair + SVM		
Top 1	**Top 3**	**Top 5**
78,80%	94,10%	96,82%

Tabela 5.14: Valores da exatidão obtida de acordo com o número de classes consideradas para Flair e SVM com núcleo linear

Apesar da diferença significativa entre ambas as abordagens na apresentação de apenas uma classe, essa diferença não se mantém na apresentação de 3 ou 5 classes. Por esse motivo, nesta tarefa de classificação a abordagem com N-gramas é a melhor opção na apresentação de mais do que uma classe na ferramenta, pois o seu custo computacional é inferior à abordagem com Flair.

5.4 Experiências complementares

Ao longo do desenvolvimento das experiências foi necessário realizar algumas experiências secundárias para auxiliar algumas decisões, é o caso do atributo Idade e do atributo Data Início. Estas experiências foram efetuadas com a abordagem de Word N-Grams (N=1) com Tfidf + SVM de núcleo rbf. Esta secção detalha os testes realizados e que decisões foram tomadas através dos resultados produzidos.

5.4.1 Idade

No caso do atributo Idade foi necessário escolher a distribuição dos grupos de idades/faixas etárias a utilizar nas experiências. Como tal foram realizados três testes adicionando cada uma das distribuições ao atributo Motivo de Contacto de modo a determinar qual seria mais discriminatória. Essas distribuições e resultados encontram-se na tabela 5.15.

	Exatidão	**F1**
<1; 1 a 5: 6 a 17; 18 a 40; 41 a 55; 56 a 70; 71 a 85; >85	75,99%	75,88%
<1; 1 a 5; e de 5 em 5 até aos 100; >100	75,88%	75,76%
<1; 1 a 10; e de 10 em 10 anos até aos 100; >100	75,90%	75,79%

Tabela 5.15: Resultados obtidos para os diferentes grupos de idade considerados

Como os resultados são todos relativamente próximos a distribuição selecionada foi:
<1; 1 a 5: 6 a 17; 18 a 40; 41 a 55; 56 a 70; 71 a 85; >85
por apresentar o melhor resultado e porque do ponto de vista clínico é a distribuição de faixas etárias mais lógica.

5.4.2 Data Início

Tendo em conta que os dados utilizados foram registados no período entre Janeiro e Março de 2018 não foi possível usar as estações do ano na classificação dos dados. Foram então usadas as horas do dia com o objetivo de determinar se existe alguma distribuição horária com efeito mais discriminatório que outra. Como tal foram testadas cinco distribuições horárias através da adição das mesmas ao atributo Motivo de Contacto na classificação. Os resultados e respetivas distribuições podem ser observados na tabela 5.16.

	Exatidão	**F1**
Diurno vs Noturno 8h-20h vs 20h-8h	76,33%	76,24%
Semana vs Fim-de-semana Sex 18h – Dom 8h vs Dom 8h – Sex 18h	76,36%	76,25%
Horas de mais afluência 23h-8h; 8h-18h; 18h-23h	76,24%	76,13%
Semana vs Fim-de-semana Prolongado Sex 18h – Seg 18h vs Seg 18h – Sex 18h	76,33%	76,22%
Hora de Pico 17h-23h vs 23h-17h	76,41%	76,28%

Tabela 5.16: Resultados obtidos para as diferentes distribuições horárias

É possível observar um resultado ligeiramente acima dos restantes quando comparando a hora de pico com o resto do dia sendo que essa foi a distribuição horária selecionada e usada em todas as experiências

relativas à hora do dia.

5.5 Discussão

Nesta secção são discutidos os resultados obtidos nas fases 1, 2 e 3 apresentados nas secções anteriores.

Os resultados obtidos na fase 1 são a base de todas as experiências realizadas nas fases seguintes. O melhor resultado obtido nesta fase foi da abordagem com *embeddings* de Flair e SVM linear com uma exatidão de 77,96% e um F1 de 77,51%, valores já considerados bastante bons considerando a complexidade e dimensionalidade dos dados em utilização. Como o objetivo desta fase é selecionar os três melhores resultados, foram selecionados os melhores resultados para cada uma das técnicas escolhidas, sendo que depois do Flair o melhor resultado obtido foi com unigramas (N=1) e SVM com núcleo rbf, com uma exatidão de 76,97% e um F1 de 76,83%, e, em terceiro lugar o melhor resultado foi da abordagem com BERT e SVM linear, com exatidão de 76,39% e F1 de 76,04%. Os resultados do *Random Forest* conseguiram aproximar-se dos anteriores para os unigramas com uma exatidão de 74,93% e F1 de 74,40% porém não foi suficiente para chegar ao desempenho das abordagens anteriores. O *Multinomial Bayes* teve resultados bastante abaixo do esperado sendo que nenhum dos seus resultados ultrapassou os 66% de exatidão. Para suportar as escolhas dos melhores resultados obtidos para cada técnica foram realizados diferentes testes de McNemar (secção 2.3.1) para comparar os resultados obtidos em cada um dos algoritmos, e, posteriormente em cada um dos núcleos de SVM utilizados. Estes testes (apêndice A) permitiram concluir que todos os resultados comparados tinham diferença percentual significativa de desempenho.

Para finalizar os resultados da fase 1 foi realizada a otimização de parâmetros às três melhores abordagens selecionadas anteriormente, mas a única melhoria encontrada comparativamente aos resultados anteriores foi com BERT e SVM linear, cuja exatidão aumentou de 76,39% para 77,10% e F1 de 76,04% para 76,61%, aumento este que, apesar de parecer ultrapassar os unigramas, uma análise de significância com teste de McNemar (secção A.1.3) indica que os modelos não apresentam diferença percentual significativa entre si, e, sendo que a abordagem com unigramas com Tfidf e SVM tem custo computacional inferior, a abordagem com BERT permanece em terceiro lugar na lista.

Passando para a segunda fase temos a incrementação gradual de novos atributos a cada uma das abordagens referidas anteriormente. Começando pelo BERT e Flair ao adicionar o atributo Comentários à classificação ambos tiveram uma perda de performance entre 0,7% e 1%. Por este motivo e pelo facto de ambos requererem um elevado custo computacional foi dada como terminada a fase de adição de atributos para ambas as abordagens. Na adição de atributos na classificação com unigramas começou-se por adicionar o atributo Comentários que permitiu aumentar o valor da exatidão de 76,97% para 77,15% e do F1 de 76,83% para 76,94%, este aumento foi menor do que o esperado e ao continuar com a adição dos restantes atributos selecionados o desempenho teve resultados muito semelhantes na maioria dos casos mas todos eles inferiores ao valor de referência anterior. Esta fase de experiências permite concluir que neste conjunto de dados nenhum dos restantes atributos contém informação suficientemente relevante para a classificação dos algoritmos clínicos. Um dos possíveis motivos para tal acontecer é a informação descrita nos motivos de contacto já ser discriminativa o suficiente para fornecer dados relativamente aos outros atributos, como por exemplo a idade do utente.

A terceira e última fase de experiências consiste no *fine-tuning* da abordagem com melhor resultado até ao momento, sendo esta do Flair com SVM linear, com uma exatidão de 77,96% e um F1 de 77,51%. Através deste *fine-tuning* o modelo Flair para a língua portuguesa utilizado passa a estar adaptado ao domínio clínico dos dados em utilização. Este *fine-tuning* permitiu maximizar o resultado anterior, a exatidão subiu para 78,80% e o F1 subiu para 78,45%, sendo este o resultado final obtido para a apresentação de uma única classe no decorrer de todas as experiências.

Complementarmente foi realizado na fase 3 o cálculo da exatidão obtida para o *Top* 3 e *Top* 5 nas classes das abordagens com *Word N-Grams* (N = 1) com Tfidf e SVM e Flair com SVM Linear, com o atributo Motivo de Contacto. Estas abordagens foram selecionadas de modo a comparar de entre as três melhores experiências, a experiência com menor custo computacional e a experiência que devolveu melhores resultados. Apesar da diferença significativa de quase 2% na apresentação de uma única classe é possível observar que apresentando mais do que uma classe os desempenhos são muito aproximados. Ambas ultrapassaram os 94% de exatidão na apresentação de três classes, sendo que para cinco classes o resultado chega perto dos 97% em ambos os casos. Não havendo diferença significativa de desempenho a abordagem com unigramas torna-se então a melhor experiência a utilizar ao apresentar várias classes na ferramenta desenvolvida.

Através das matrizes de confusão é possível distinguir algumas das interferências entre algoritmos clínicos que dificultam a classificação. Alguns exemplos são a classe 'Tosse' com a classe 'Síndrome Gripal', a classe 'Problemas de adaptação em situação de crise' com a classe 'Problemas de Ansiedade', e, a classe 'Problemas por Sarampo' com a classe 'Rash'. Estas matrizes permitem também perceber quais algoritmos têm em média melhor ou pior desempenho. Alguns dos algoritmos clínicos com performance acima de 80% de cobertura são 'Problema Ocular', 'Problema por Queimaduras' e 'Problema Urinário' e alguns dos algoritmos com pior desempenho, geralmente abaixo de 50% de cobertura são 'Problemas de adaptação em situação de crise', 'Emergência', 'Problemas por Sarampo' e 'Problemas Inespecíficos'. Este baixo desempenho nestas classes pode dever-se à menor quantidade de registos ou à interferência de classes referida acima e a interferência entre classes deve-se à quantidade de sintomas em comum entre elas.

6

Conclusão e Trabalho Futuro

Nesta dissertação foram aplicadas metodologias de aprendizagem automática e processamento de linguagem natural com o objetivo de classificar algoritmos clínicos nas chamadas recebidas pelo SNS24 e auxiliar a decisão do enfermeiro que atende a respetiva chamada. O conjunto de dados utilizado contém cerca de 270.000 registos que se dividem em 51 algoritmos clínicos para a classificação e que foram divididos em conjuntos de treino, validação e teste. Esta tarefa foi desenvolvida através de diversas experiências e abordagens distribuídas em várias fases e com utilização de diferentes atributos selecionados do conjunto de dados.

Os algoritmos selecionados para a realização dessas experiências foram SVM's, *Random Forest* e *Multinomial Bayes*, e as técnicas de processamento de linguagem natural utilizadas foram *word n-grams* e *word embeddings*. Dado que ambos o *Random Forest* e o *Multinomial Bayes* produziram resultados inferiores, as abordagens selecionadas para prosseguir com o conjunto de experiências foram todas elas com SVM's com núcleos rbf e linear. Estas abordagens com SVM's foram sujeitas a uma optimização de parâmetros com o intuito de maximizar os resultados sendo que o melhor resultado obtido nesta fase foi de 77,96% de exatidão e 76,83% de medida-F para a abordagem de Flair com SVM linear.

Posteriormente foram realizadas experiências para testar o desempenho do modelo com a adição de novos atributos na classificação, e, ao contrário do esperado, apenas o atributo de texto livre 'Comentários' deu

uma pequena melhoria na classificação com *Word N-grams* (N = 1) e todas as restantes experiências produziram resultados inferiores aos valores de referência anteriores.

Para finalizar o ciclo de experiências foi feito um *fine-tuning* ao modelo Flair utilizado devido a ser o melhor resultado obtido de entre todas as experiências realizadas anteriormente. Este *fine-tuning* devolveu o melhor resultado obtido com uma exatidão de 78,80% e medida-F de 78,45%. A adicionar a este resultado foi calculada a exatidão das abordagens com *Word N-Grams* e Flair para a apresentação das classes do *top* 3 e *top* 5 cujos resultados foram para ambos os casos acima de 94% e de 96% respetivamente.

Tendo em conta todo o trabalho desenvolvido e o objetivo mencionado no início desta dissertação, concluiu-se que os resultados obtidos têm potencial para auxiliar a redução da duração das chamadas, sendo que apresentando os três ou cinco algoritmos com maior probabilidade de classificação são atingidas medidas de performance acima de 90%. Esta redução de tempo na duração das chamadas será avaliada através de análises comparativas da duração média das chamadas e da percentagem de chamadas telefónicas em que há alteração do algoritmo inicialmente selecionado.

O modelo apresentado não representa o modelo final a utilizar pois ainda há algumas linhas de experiências distintas a seguir ambos no conjunto de dados aqui apresentado com registos correspondentes a três meses, datados entre Janeiro e Março de 2018, e no conjunto de dados que contém três anos e meio de registos, datados entre Janeiro de 2017 e Junho de 2020, que nos foi fornecido recentemente. O trabalho futuro divide-se então entre estes dois conjuntos de dados.

Para prosseguir com o conjunto de experiências apresentado nesta dissertação serão aplicadas novas abordagens de classificação com redes neuronais artificiais, nomeadamente redes neuronais profundas com mecanismos baseados em atenção, e classificação hierárquica de classes.

Para o conjunto de dados maior, com registos datados entre Janeiro de 2017 e Junho de 2020, equivalente a cerca de 4 milhões de chamadas, serão desenvolvidos e comparados diversos modelos, contendo diferentes porções dos dados, dado que estes já contêm os primeiros 4 meses em que foram implementados os novos algoritmos clínicos relativos à atual pandemia e que poderão, ou não, causar maiores interferências. Serão aplicadas as metodologias que apresentam melhores resultados no conjunto anterior, entre elas as abordagens com *word embeddings* e SVM's, e serão desenvolvidas novas experiências com adição da estação do ano na classificação ou outros atributos que sejam considerados relevantes.

A

Teste de McNemar

Para o conjunto de experiências efetuado no desenvolvimento desta dissertação foram efetuados alguns testes de McNemar (secção 2.3.1) para auxiliar a seleção dos melhores resultados.

Nas secções A.1.1, A.1.2 e A.1.3 encontram-se os resultados referentes às análises de significância realizadas na primeira fase de experiências (secção 5.1).

A secção A.1.1 representa as análises realizadas entre o primeiro e segundo melhores resultados obtidos em cada um dos algoritmos individualmente.

A secção A.1.2 representa a comparação feita entre os núcleos de SVM utilizados nas experiências, nomeadamente, linear e rbf. Estes testes comparam o desempenho entre Unigramas e Tfidf com SVC e LinearSVC, entre Flair com SVC e LinearSVC, e, entre BERT com SVC e LinearSVC.

Finalizando a primeira fase de experiências a secção A.1.3 apresenta o resultado da análise significativa entre o resultado de BERT com e sem otimização de parâmetros.

Antes de avançar para a segunda fase de experiências (secção 5.2) foi efetuada uma análise de significância entre os três melhores resultados obtidos na primeira fase. Essa análise de significância encontra-se na secção A.1.4.

Os últimos testes de McNemar realizados ao longo destas experiências serviram para comparar o desempenho entre a melhor abordagem obtida até ao momento, com e sem fine-tuning do modelo utilizado, neste caso, Flair com SVM Linear. Estes resultados podem ser observados na secção A.1.5.

A.1 Resultados

A.1.1 Comparação entre Algoritmos

As tabelas A.1, A.2 e A.3 apresentam os resultados obtidos para a comparação entre os dois melhores resultados de cada um dos algoritmos apresentados na secção 5.1.

		SVC			
		Exatidão	**F1**	**statistic**	**p-value**
Word N-Grams (N = 1)	**Tfidf**	**76,97%**	**76,83%**	6.017	0.014
Flair		76,58%	76,30%		

Tabela A.1: Resultados do teste de McNemar entre os melhores resultados da tabela 5.1

		LinearSVC			
		Exatidão	**F1**	**statistic**	**p-value**
Flair		**77,96%**	**77,51%**	86.25	0
Word N-Grams (N = 1)	**Tfidf**	76,47%	76,10%		

Tabela A.2: Resultados do teste de McNemar entre os melhores resultados da tabela 5.2

		Random Forest			
		Exatidão	**F1**	**statistic**	**p-value**
Word N-Grams (N = 1)	**Tfidf**	**74,93%**	**74,40%**	55.782	0
Word N-Grams (N = 1)	**Count**	73,84%	73,26%		

Tabela A.3: Resultados do teste de McNemar entre os melhores resultados da tabela 5.3

Como é possível observar, o valor p em todas as tabelas é inferior ao limite de significância de $\alpha = 0{,}05$ estabelecido, sendo que é rejeitada a hipótese e os modelos têm todos diferença significativa de desempenho entre si.

A.1.2 Comparação entre núcleos

Tendo em conta que foram efetuadas experiências com dois núcleos distintos, linear e rbf, foram efetuados testes de McNemar para analisar a significância entre os resultados de ambas as abordagens apresentadas nas tabelas 5.1 e 5.2.

			Exatidão	F1	statistic	p-value
SVC	**Word N-Grams (N = 1)**	**Tfidf**	**76,97%**	**76,83%**	13.181	0
LinearSVC	**Word N-Grams (N = 1)**	**Tfidf**	76,47%	76,10%		

Tabela A.4: Resultados do teste de McNemar entre *Word N-Grams* (N=1)

		Exatidão	F1	statistic	p-value
SVC	**BERT**	75,83%	75,52%	14.158	0
LinearSVC	**BERT**	**76,39%**	**76,04%**		

Tabela A.5: Resultados do teste de McNemar entre BERT

		Exatidão	F1	statistic	p-value
SVC	**Flair**	76,58%	76,30%	101.045	0
LinearSVC	**Flair**	**77,96%**	**77,51%**		

Tabela A.6: Resultados do teste de McNemar entre Flair

Tal como na secção anterior, todos os resultados obtidos obtiveram um valor p inferior ao limite de significância $\alpha = 0{,}05$, sendo que é novamente rejeitada a hipótese e todos os modelos têm diferença significativa de desempenho entre si.

A.1.3 Comparação da otimização de parâmetros para BERT

Ao realizar a otimização de parâmetros nas três melhores experiências resultantes da primeira fase de experiências, a experiência com BERT e SVM Linear foi a única que obteve melhoria de resultados. Como tal foi efetuada a análise de significância entre ambos os resultados da tabela 5.7 e essa análise pode ser observada na tabela abaixo.

	Parâmetro C	Exatidão	F1	statistic	P-value
BERT + SVM Linear	1	76,39%	76,04%	62,016	0
BERT + SVM Linear	**0.1**	**77,10%**	**76,61%**		

Tabela A.7: Resultado do teste de McNemar na otimização de parâmetros com BERT e SVM Linear

Como é possível observar o valor de p é inferior ao grau de significância estabelecido de 0,05 e assim sendo conclui-se que os modelos têm diferença significativa de desempenho.

A.1.4 Comparação entre os três melhores resultados obtidos

Nas tabelas seguintes é possível observar a análise de significância realizada entre os melhores resultados relativos às três melhores abordagens obtidas na primeira fase de experiências (secção 5.1), experiências estas que foram alvo de análises de significância apresentadas entre as secções A.1.1 e A.1.3 deste capítulo e selecionadas de acordo com as mesmas.

	Exatidão	F1	statistic	P-value
Flair + SVM Linear	**77,96%**	**77,51%**	33.774	0
BERT + SVM Linear	77,10%	76,61%		

	Exatidão	F1	statistic	P-value
Flair + SVM Linear	**77,96%**	**77,51%**	38.785	0
Word N-Grams (N=1) com Tfidf + SVM rbf	76,97%	76,83%		

	Exatidão	F1	statistic	P-value
BERT + SVM Linear	**77,10%**	**76,61%**	0.673	0.412
Word N-Grams (N=1) com Tfidf + SVM rbf	**76,97%**	**76,83%**		

Tabela A.10: Resultados dos testes de McNemar entre as três melhores abordagens obtidas

Através dos resultados obtidos é possível observar que o valor p nas comparações com Flair e SVM Linear é sempre inferior ao nível de significância de 0,05 estabelecido e, assim sendo, concluir que a abordagem com Flair e SVM Linear é a que tem melhor desempenho de entre as três.

Observando então o resultado da análise de significância entre as abordagens com BERT e SVM Linear e Unigramas com Tfidf e SVM de núcleo rbf, o valor p de 0.412 é superior ao nível de significância de 0,05, o que significa que os modelos não apresentam diferença significativa entre si a nível de desempenho. Assim sendo a abordagem com Unigramas com Tfidf e SVM de núcleo rbf é considerada nesta situação, melhor do que a alternativa, dado o seu custo computacional ser inferior. Através desta análise temos então a tabela A.11 que representa os melhores resultados obtidos até ao momento e pela sua devida ordem.

		Exatidão	F1
1	**Flair + SVM com núcleo linear**	**77,96%**	**77,51%**
2	**Word N-Grams (N=1) com núcleo rbf**	76,97%	76,83%
3	**BERT + SVM com núcleo linear**	77,10%	76,61%

Tabela A.11: Melhores resultados resultantes da primeira fase de experiências segundo os testes de McNemar

A.1.5 Comparação entre resultados com e sem fine-tuning

Na terceira e última fase de experiências foi realizado o fine-tuning do modelo Flair utilizado ao longo das experiências. Na próxima tabela é possível observar a comparação de desempenho entre o resultado de Flair e SVM Linear com fine-tuning e sem fine-tuning para o conjunto de validação apresentados na tabela 5.3.1.

		Exatidão	F1	statistic	P-value
Sem Fine-tuning	**Flair + SVM Linear**	77,96%	77,51%	23.214	0
Com Fine-tuning	**Flair + SVM Linear**	**78,59%**	**78,21%**		

Tabela A.12: Resultados do teste de McNemar entre abordagens de Flair com e sem fine-tuning

A tabela acima permite observar que ambos os modelos possuem diferença percentual significativa entre si, dado que o valor de p é inferior ao nível de significância de 0,05 estabelecido. Isto permite concluir que este fine-tuning permitiu melhorar o modelo e a respetiva classificação.

Bibliografia

[ABV18] Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics.

[AG97] Yali Amit and Donald Geman. Shape quantization and recognition with randomized trees. *Neural Comput.*, 9(7):1545–1588, October 1997.

[AJO+18] Oludare Isaac Abiodun, Aman Jantan, Abiodun Esther Omolara, Kemi Victoria Dada, Nachaat AbdElatif Mohamed, and Humaira Arshad. State-of-the-art in artificial neural network applications: A survey. *Heliyon*, 4(11):e00938, 2018.

[AX19] Felipe Almeida and Geraldo Xexéo. Word embeddings: A survey, 2019.

[Bre96] Leo Breiman. Bagging predictors. *Machine Learning 24*, 24:123–140, 8 1996.

[Bre01] Leo Breiman. Random forests. *Machine Learning 45*, 45:5–32, 10 2001.

[Bur19] A. Burkov. *The Hundred-Page Machine Learning Book*. Andriy Burkov, 2019.

[DCLT18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 10 2018.

[eJH18] D. Kirsh e J. Hurwitz. *Machine Learning for Dummies*. John Wiley & Sons, Inc, 2018.

[Eth19] Kawin Ethayarajh. How contextual are contextualized word representations? comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China, November 2019. Association for Computational Linguistics.

[Fab19] Samson Damilola Fabiyi. A review of unsupervised artificial neural networks with applications. *International Journal of Computer Applications*, 181:22–26, 02 2019.

[Fer06] Edberto Ferneda. Redes neurais e sua aplicação em sistemas de recuperação de informação. *Ciência da Informação*, 35, 08 2006.

[FGE14] Khaled Fawagreh, Mohamed Medhat Gaber, and Eyad Elyan. Random forests: from early developments to recent advancements. *Systems Science & Control Engineering*, 2(1):602–609, 2014.

[FR98] C. Fraley and A. E. Raftery. How many clusters? which clustering method? answers via model-based cluster analysis. *The Computer Journal*, 41(8):578–588, 1998.

[GRS98] S. Guha, R. Rastogi, and Kyuseok Shim. Cure: an efficient clustering algorithm for large databases. In *SIGMOD '98*, 1998.

[HA04] Victoria Hodge and Jim Austin. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22:85–126, 10 2004.

[Ham18] Ahmad Hammoudeh. A concise introduction to reinforcement learning, 02 2018.

[Hay09] Simon S. Haykin. *Neural networks and learning machines*. Pearson Education, Upper Saddle River, NJ, third edition, 2009.

[HKP06] Jiawei Han, M. Kamber, and J. Pei. Data mining: Concepts and techniques. pages 585–631, 01 2006.

[HM15] Mohammad Hossin and Sulaiman M.N. A review on evaluation metrics for data classification evaluations. *International Journal of Data Mining & Knowledge Management Process*, 5:01–11, 03 2015.

[HR18] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification, 2018.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[IPP16] Jasmine Irani, Nitin Pise, and Madhura Phatak. Clustering techniques and the similarity measures used in clustering: A survey. *International Journal of Computer Applications*, 134:9–14, 01 2016.

[JHS02] Li-Ping Jing, Hou-Kuan Huang, and Hong-Bo Shi. Improved feature selection approach tfidf in text mining. volume 2, pages 944 – 946 vol.2, 02 2002.

[Jin17] Kejin Jin. Language model: A survey of the state-of-the-art technology. 2017.

[Kal20] R. K. Kaliyar. A multi-layer bidirectional transformer encoder for pre-trained word embedding: A survey of bert. In *2020 10th International Conference on Cloud Computing, Data Science Engineering (Confluence)*, pages 336–340, 2020.

[KG19] Sang-Woon Kim and Joon-Min Gil. Research paper classification systems based on tf-idf and lda schemes. *Human-centric Computing and Information Sciences*, 9, 12 2019.

[LHH19] H. Lane, H. Hapke, and C. Howard. *Natural Language Processing in Action: Understanding, analyzing, and generating text with Python*. Manning Publications, 2019.

[LHP+15] Timothy Lillicrap, Jonathan Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *CoRR*, 09 2015.

[LSW14] Lijuan Liu, Bo Shen, and Xing Wang. Research on kernel function of support vector machine. *Lecture Notes in Electrical Engineering*, 260:827–834, 04 2014.

[LY17] Yang Li and Tao Yang. *Word Embedding for Understanding Natural Language: A Survey*, volume 26. 05 2017.

[Mac67] James B. MacQueen. Some methods for classification and analysis of multivariate observations. 1967.

[MBM+16] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning, 2016.

[MBPPM09] Popescu Marius, Valentina Balas, Liliana Perescu-Popescu, and Nikos Mastorakis. Multilayer perceptron and neural networks. *WSEAS Transactions on Circuits and Systems*, 8, 07 2009.

[McN47] Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, June 1947.

[MCPvD20] Timothee Mickus, Mathieu Constant, Denis Paperno, and Kees van Deemter. What do you mean, bert? assessing bert as a distributional semantics model. 01 2020.

[Meh20] Deepanshu Mehta. State-of-the-art reinforcement learning algorithms. *International Journal of Engineering and Technical Research*, V8, 01 2020.

[Mil17] Dubravko Miljković. Brief review of self-organizing maps. 05 2017.

[MKS+13] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. 12 2013.

[MN01] Andrew Mccallum and Kamal Nigam. A comparison of event models for naive bayes text classification. *Work Learn Text Categ*, 752, 05 2001.

[MSC+13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality, 2013.

[PNI+18] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations, 2018.

[Pra12] Ashis Pradhan. Support vector machine-a survey. *IJETAE*, 2, Agosto 2012.

[PSM14] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *In EMNLP*, 2014.

[PVG+11] F. Pedregosa, 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, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[Ras20] Sebastian Raschka. Model evaluation, model selection, and algorithm selection in machine learning, 2020.

[RKR20] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. A primer in bertology: What we know about how bert works, 02 2020.

[RM05] Lior Rokach and Oded Maimon. *Clustering Methods*, pages 321–352. 01 2005.

[RN09] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: a modern approach*. Pearson, 3 edition, 2009.

[RWC+19] A. Radford, Jeffrey Wu, R. Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[Sar15] Sarat Kumar Sarvepalli. Deep learning in neural networks: The science behind an artificial brain. Outubro 2015.

[Saz06] Murat Sazli. A brief review of feed-forward neural networks. *Communications, Faculty Of Science, University of Ankara*, 50:11–17, 01 2006.

[SCS+19] Joaquim Santos, Bernardo Consoli, Cicero Santos, Juliano Terra, Sandra Collonini, and Renata Vieira. Assessing the impact of contextual embeddings for portuguese named entity recognition. pages 437–442, 10 2019.

[SPG+17] Amit Saxena, Mukesh Prasad, Akshansh Gupta, Neha Bharill, op Patel, Aruna Tiwari, Meng Er, Weiping Ding, and Chin-Teng Lin. A review of clustering techniques and developments. *Neurocomputing*, 267, 07 2017.

[SQXH20] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. How to fine-tune bert for text classification?, 2020.

[SU12] Karanjit Singh and Shuchita Upadhyaya. Outlier detection: Applications and techniques. *International Journal of Computer Science Issues*, 9, 01 2012.

[SWB13] Cyrus Shaoul, Chris Westbury, and Harald Baayen. The subjective frequency of word n-grams. *Psihologija*, 46:497–537, 12 2013.

[Tin95] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd International Conference on Document Analysis and Recognition*, volume 1, pages 278–282, 1995.

[Tin98] Tin Kam Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.

[US19] Vidyadhar Upadhya and P. Sastry. An overview of restricted boltzmann machines. *Journal of the Indian Institute of Science*, 99, 02 2019.

[Vap95] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[WRR+18] Théophane Weber, Sébastien Racanière, David P. Reichert, Lars Buesing, Arthur Guez, Danilo Jimenez Rezende, Adria Puigdomènech Badia, Oriol Vinyals, Nicolas Heess, Yujia Li, Razvan Pascanu, Peter Battaglia, Demis Hassabis, David Silver, and Daan Wierstra. Imagination-augmented agents for deep reinforcement learning, 2018.

[ZJZ10] Yin Zhang, Rong Jin, and Zhi-Hua Zhou. Understanding bag-of-words model: A statistical framework. *International Journal of Machine Learning and Cybernetics*, 1:43–52, 12 2010.

[ZRL96] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: An efficient data clustering method for very large databases. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, SIGMOD '96, page 103–114, New York, NY, USA, 1996. Association for Computing Machinery.

[ZZY03] Shichao Zhang, Chengqi Zhang, and Qiang Yang. Data preparation for data mining. *Applied Artificial Intelligence*, 17:375–381, 05 2003.

UNIVERSIDADE DE ÉVORA
INSTITUTO DE INVESTIGAÇÃO
E FORMAÇÃO AVANÇADA

Contactos:
Universidade de Évora
Instituto de Investigação e Formação Avançada — IIFA
Palácio do Vimioso | Largo Marquês de Marialva, Apart. 94
7002 - 554 Évora | Portugal
Tel: (+351) 266 706 581
Fax: (+351) 266 744 677
email: iifa@uevora.pt