Escola de Tecnologias e Arquitetura

Departamento de Ciências e Tecnologias da Informação

Análise de Sentimento em Microblogues com base em Cascatas de Classificacão

Fernando Manuel Dias Rebelo

Dissertação submetida como requisito parcial para obtenção do grau de Mestre em **Engenharia Informática**

Orientador:

Doutor Fernando Manuel Marques Batista, Professor Auxiliar

ISCTE-IUL – Instituto Universitário de Lisboa

Coorientador:

Doutor Ricardo Daniel Santos Faro Marques Ribeiro, Professor Auxiliar

ISCTE-IUL – Instituto Universitário de Lisboa

Setembro, 2016

Resumo

Esta dissertação descreve uma plataforma de classificação que permite utilizar e aplicar classificadores binários em cascata. A plataforma utiliza algoritmos de aprendizagem automática existentes no software WEKA que disponibiliza uma API para esse efeito, tendo sido testada na classificação de sentimento de *tweets* e blogues. Esta plataforma permitiu também analisar e comparar diferentes cascatas de classificação com os classificadores Naïve Bayes, Regressão Logística e Support Vector Machines que implementa o algoritmo Sequential Minimal Optimization para otimização da fase de treino. Neste caso de estudo foram exploradas várias arquiteturas de classificação com um máximo de três níveis, combinando diversos classificadores binários, para classificação em quatro e seis classes. Como entrada para os classificadores, foram extraídas características de cada um dos documentos e utilizados léxicos de polaridade associados às palavras. Em geral, as arquiteturas que utilizam Support Vector Machines obtêm os melhores resultados. Os diferentes classificadores obtêm os seus melhores resultados com diferentes arquiteturas.

Abstract

This thesis describes a classification platform that enables to use and apply binary classifiers in cascade. The platform uses existing machine learning algorithms from WEKA software that provides an API for this purpose, having been tested in sentiment classification of tweets and blogs. This platform allowed also to analyze and compare different classification cascades with the classifiers Naïve Bayes, Logistic Regression and Support Vector Machines which implements the Sequential Minimal Optimization algorithm to optimize the training phase. In this case study were explored various classification architectures with a maximum of three levels, combining different binary classifiers for classification in four and six classes. As input for the classifiers, were extracted characteristics of each of the documents and utilized polarity lexicons associated to the words. In general, architectures based on Sequential Minimal Optimization get the best results. The different classifiers get their best results with different architectures.

Palavras Chave
Keywords

Palavras chave

Cascatas de Classificação

Aprendizagem Automática

Análise de Sentimento

WEKA

Keywords

Classification Cascades

Machine Learning

Sentiment Analysis

WEKA

Agradecimentos
Acknowledgements

Quero agradecer ao meu orientador e co-orientador, Fernando Batista e Ricardo Ribeiro, por todo o apoio e orientação, esta dissertação não teria sido possível sem eles. Queria também agradecer ao Afonso e à empresa Ongagement, pelo fornecimento dos dados de marketing. Por fim, mas não menos importantes, ficam os agradecimentos à minha família, à Sofia e aos meus amigos, em particular ao Vasco, camarada de longas horas de trabalho.

Lisboa, Setembro de 2016

Fernando Rebelo

Conteúdo

Lista de Figuras

Lista de Tabelas

1 Introdução

1.1 Motivação e objetivos

A proliferação das redes sociais permite que estas sejam utilizadas como uma fonte de informação, nas quais os seus utilizadores exprimem sentimentos. Dada a enorme quantidade de dados, surge o natural interesse na investigação de técnicas que permitam uma classificação automática destes conteúdos, que além de suprimir ou aliviar o esforço humano envolvido, permite extrair conhecimento importante desses dados, podendo ser uma mais-valia para inúmeras aplicações. O Twitter é uma rede social que surgiu em 2006 com a particularidade de permitir a produção de conteúdos por qualquer utilizador, conteúdos esses que ficam acessíveis publicamente. As mensagens de texto, apelidadas de *tweet*s, não podem exceder os 140 caracteres, têm os denominados *hashtags* (#) que permitem identificar tópicos de uma forma fácil, *users* (@) que permitem identificar pessoas e os *re-tweets* (RT) que não são mais do que reenviar uma mensagem de outro utilizador. Os sentimentos exprimidos nas redes sociais e em particular no Twitter têm a particularidade de surgir praticamente em tempo real, provenientes de pessoas que realmente querem exprimir a sua opinião, ao contrário dos tradicionais questionários, em que existe uma obrigação ou tarefa associada. Com frequência contêm linguagem informal, por vezes, com erros ortográficos e com a utilização de abreviaturas e acrónimos. Assim, os tweets apresentam um conjunto de fenómenos e particularidades que obrigam a uma necessária adaptação das tarefas de processamento da informação bem como dos métodos de classificação de texto. Por sua vez, os textos de blogues, têm em comum com os *tweets* a presença de endereços web e têm como grande diferença, um limite muito superior de caracteres permitido por texto, cerca de sete vezes mais.

Esta dissertação descreve uma plataforma que permite a classificação hierárquica de sentimento em microblogues, tendo em conta múltiplas classes. Esta plataforma utiliza os classificadores existentes na coleção de software para aprendizagem automática WEKA através da sua API. No âmbito deste trabalho exploram-se diferentes cascatas de classificação e algoritmos de aprendizagem automática sobre dois conjuntos de dados, nomeadamente: tweets espanhóis e críticas nacionais extraídas de blogues sobre produtos. A utilização de léxicos de polaridade na

língua portuguesa e espanhola permitiu ainda melhorar o desempenho dos vários classificadores.

1.2 Enquadramento

Nesta secção é realizado o enquadramento da ferramenta WEKA utilizada e dos classificadores escolhidos, com base na literatura.

1.2.1 WEKA

Para realizar experiências com diferentes algoritmos de aprendizagem automática, foi utilizada a ferramenta WEKA (Waikato Environment for Knowledge Analysis), que permite realizar análise computacional de dados com ferramentas para pré-processamento, classificação, regressão, *clustering* e regras de associação e de visualização. Esta ferramenta disponibiliza uma API (Application Programming Interface) para a linguagem JAVA (Hall et al., 2009), a qual foi utilizada para realização do trabalho descrito neste documento.

Dos múltiplos formatos de entrada de dados suportados pelo WEKA, um dos mais populares é o ARFF (Attribute-Relation File Format). Consiste num ficheiro ACSII (American Standard Code for Information Interchange) que resumidamente contém uma série de instâncias que partilham o mesmo conjunto de *features*. Este ficheiro é composto por duas zonas, a zona de cabeçalho e a zona de dados. A Figura 1.1 mostra um exemplo do conteúdo de um ficheiro ARFF. Na zona de cabeçalho encontra-se definido o nome da relação (linha 1) e o conjunto de *features* que são considerados pelo modelo e seu respetivo tipo (linha 3 a 12), com a exceção que na última linha das instâncias são declaradas as classes possíveis em classificação (linha 12). Na zona de dados (a partir da linha 14), existe uma instância para cada registo existente no conjunto de entrada, com os diversos pares atributo/peso que o cabeçalho define, com a exceção do último par que define a atual polaridade da instância, que terá de ser uma das definidas no último atributo do cabeçalho. Quando se tem grandes conjuntos de dados, surge a probabilidade dos dados poderem ser muito esparsos podendo ser usada este tipo de representação.

1.2.2 Algoritmos de aprendizagem automática

A aprendizagem supervisionada pode geralmente ser feita através de classificação ou regressão. Classificação é considerado o processo em que quando se tem determinados dados de entrada e se pretende mapeá-los à sua respetiva classe, com a característica de que as classes

```
1  @relation analise_de_sentimento
2
3  @attribute baratos numeric
4  @attribute trabas numeric
5  @attribute verte numeric
6  @attribute Gandia numeric
7  @attribute robó numeric
8  @attribute *TARGET numeric
9  @attribute argumento numeric
10 @attribute *URL numeric
11 ...
12 @attribute Sent6C {P,O}
13
14 @data
15 {1 1.0,3 2.0,4 1.0,5 1.0,12 O}
16 ...
```

Figura 1.1: Excerto do conteúdo de um ficheiro ARFF.

classificadas são discretas. Regressão é mais uma tarefa de valor contínuo, no qual em vários problemas se tenta prever um valor contínuo, por exemplo no intervalo [0,1] ou um número real. Foram explorados no WEKA os algoritmos de classificação RL (Regressão Logística), SVM (Support Vector Machines), o qual usa uma implementação baseada no algoritmo de otimização SMO (Sequential Minimal Optimization) e o *NB* (Naïve Bayes).

O *SMO* é uma maneira de resolver o treino de um *SVM*, que se serve de uma heurística para particionar o problema do treino em problemas mais pequenos, ou seja, como outros algoritmos de treino do *SVM*. Particiona problemas quadráticos grandes em pequenos problemas, com a particularidade de que o *SMO* em relação aos outros utilizar os que são realmente mais pequenos, que lhe garante rápidas execuções, que são resolvidas analiticamente. Estas heurísticas estão relacionadas com os dados e o nível de complexidade é passível de ser alterado de maneira a otimizar o modelo (Platt, 1999).

O *R*L é uma implementação do classificador de Cessie and van Houwelingen (1992), com uma ligeira adaptação para lidar com os diferentes pesos das instâncias, prevê acertar não na classe nominal mas no seu valor numérico. Pode sofrer de *overfitting* caso a dimensão dos dados seja muito grande ou os dados de treino serem escassos.

Em relação ao NB (John and Langley, 1995), é um classificador probabilístico, que se baseia na aplicação do teorema de Bayes com a assunção de "naïve" que diz que todas as *features* são independentes, o que torna o modelo menos complexo, passível de aceitar assim grandes conjuntos de dados de entrada.

positivo	negativo	← classificado como
tp	fn	positivo
fp	tn	negativo

Tabela 1.1: Representação de uma Matriz de Confusão para um problema binário.

$$P(c|T) = \frac{P(c) \cdot P(T|c)}{P(T)} \tag{1.1}$$

A Fórmula 1.1 apresenta o Teorema de Bayes, onde *c* representa a classe e *T* o *tweet*, *P(c|T)* representa a classe dado o *tweet*, *P(c)* representa a polaridade inicial do *tweet*, *P(T|c)* representa a probabilidade do *tweet* dada a classe e *P(T)* representa a probabilidade inicial do *tweet*.

1.3 Análise de sentimento

A análise de sentimento é uma sub-categoria de classificação de textos, sendo que outras tarefas de classificação de textos são por exemplo a de identificação de tópico ou do autor com recurso da utilização de técnicas de PNL (Processamento Natural da Língua). Pode ter vários níveis de complexidade, mais simples, com positivo e negativo, mais complexo, com a atribuição de valores numa escala, e num nível mais avançado detetar a entidade a que se refere o sentimento. A análise de sentimento pode ter diversas aplicações, como por exemplo no cinema, através da análise de sentimento do que foi dito sobre o filme, ou por exemplo sobre o sentimento sobre determinado produto, se é bom ou mau.

As principais medidas de análise utilizadas frequentemente em tarefas de classificação binária são: taxa de acerto (*accuracy*), precisão (*precision*), cobertura (*recall*), medida F (*F-measure*) e muitas vezes é também relevante uma análise mais cuidada à matriz de confusão. A matriz de confusão, cuja representação é apresentada na Tabela 1.1, permite fazer uma análise mais detalhada dos resultados, utilizando para isso contagens do número de elementos identificados como positivos que são realmente positivos (tp), elementos negativos que foram identificados como positivos (fp), elementos positivos que foram falsamente identificados como negativos (fn) e elementos negativos que foram corretamente identificados como negativos. A *accuracy* pode ser obtida usando a Equação 1.2, indicando a percentagem de elementos corretamente classificados em relação ao total dos elementos existentes. A *precisão*, expressa pela Equação 1.3, permite perceber a percentagem de classificações previstas como positivas que realmente são positivas. O *recall*, expresso pela Equação 1.4, permite perceber que percentagem de positivos que foram classificados como positivos. A *F-measure* é uma medida que combina

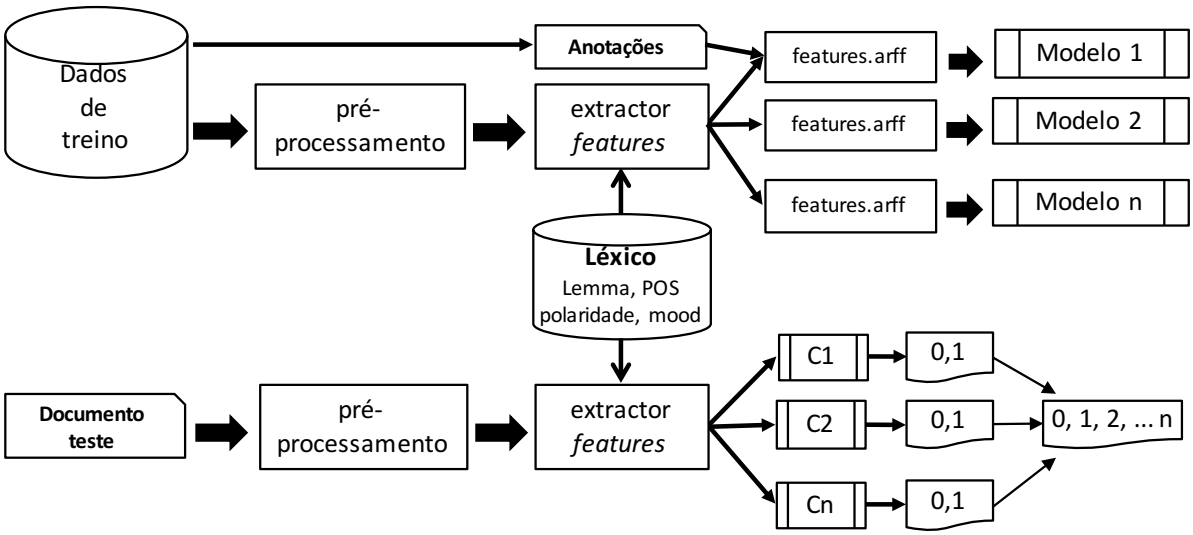

Figura 1.2: Fase de treino e de teste do classificador binário.

a *precisão* e o *recall*, como mostra a Equação 1.5.

$$\text{accuracy} = \frac{tp+tn}{tp+tn+fp+fn} \tag{1.2}$$

$$\text{precision} = \frac{tp}{tp+fp} \tag{1.3}$$

$$\text{recall} = \frac{tp}{tp+fn} \tag{1.4}$$

$$\text{f-measure} = 2 * \frac{precision * recall}{precision + recall} \tag{1.5}$$

1.4 Abordagem

Num problema de classificação com múltiplas classes é comum assumir que as classes são independentes. No entanto, as diversas classes podem ter diferentes níveis de afinidade entre si e uma abordagem que tire partido desse fator pode obter um melhor desempenho. Por exemplo, num cenário em que estão definidas as classes positivo, muito positivo, negativo e muito negativo poderá ser mais adequado realizar a tarefa de classificação em duas fases. Numa primeira fase, distinguir entre positivo e negativo e numa segunda fase calcular a intensidade associada. As experiências aqui reportadas descrevem estratégias de classificação para problemas multi-classe em que as classes não são completamente independentes que fazem uso de uma cascata

de classificadores binários para tratar a classificação por grupos de classes. Os problemas reportados neste artigo usam uma abordagem supervisionada que, em geral, pressupõe uma fase de treino na qual é criado um modelo com base em dados de treino e respetivas anotações e uma fase de teste na qual se usa o modelo anteriormente criado para produzir uma classificação automática. A Figura 1.2 ilustra este processo, no qual é também utilizado um léxico que, sendo um recurso externo, permite produzir *features* adicionais para os dados. Na fase de treino são criados vários modelos binários, que na fase de teste podem ser combinados de diferentes formas, com o fim de produzir uma classificação com várias classes.

1.5 Estrutura deste documento

Este tese tem a seguinte estrutura. O capítulo 2 realiza uma introdução ao domínio da análise de sentimento e uma revisão da literatura de classificação hierárquica e de outros modelos e termina com a apresentação da abordagem. No capítulo 3, é realizada análise de sentimento a *tweets* espanhóis, onde inicialmente são apresentados os conjuntos de dados utilizados, a abordagem ao problema de classificação com o modelo supervisionado e seus componentes, os classificadores binários e as arquiteturas que resultam da combinação dos diversos classificadores binários. São apresentadas as experiências e resultados, iniciando com a fase de treino dos classificadores binários, a avaliação dos resultados em cascata e do efeito das diferentes *features*, os resultados com o conjunto de dados de teste, uma análise de erros e algumas conclusões. No capítulo 4 são feitas experiências de classificação com os modelos existentes a outros conjuntos de dados, de língua portuguesa e são apresentados e discutidos os resultados. Finalmente no capítulo 5, descreve-se a análise efetuada, abordam-se os novos desafios tratados durante este trabalho e são discutidas conclusões.

Trabalho Relacionado

2

Este capítulo analisa os vários trabalhos ligados à análise de sentimento, com o intuito de desvendar algumas áreas de investigação a uma das redes sociais que recentemente tem demonstrado elevada importância, o Twitter. Foca-se também noutras explorações feitas à informação linguística para melhorar a classificação de sentimento e termina descrevendo as diferentes abordagens com cascatas de classificação, reportadas na literatura.

2.1 Análise de sentimento

A análise de sentimento tem sido investigada no contexto de diversas áreas e tópicos, tais como análise de notícias (Bautin et al., 2008), análise e marketing de produtos (Dave et al., 2003), críticas de filmes (Turney, 2002) e análise de sentimento da população durante desastres naturais e crises (Nagy and Stamberger, 2012). As técnicas de classificação da análise de sentimento podem ser divididas em duas, uma abordagem supervisionada, como no trabalho de Boiy and Moens (2009), ou distante como no trabalho de Pang et al. (2002), ou uma combinação das duas como no trabalho de Melville et al. (2009). Uma abordagem diz-se supervisionada quando existem dados anotados para treinar os modelos. Uma abordagem diz-se distante, quando não existem dados anotados e é necessário construir uma classificação automática através de diversos métodos (Pak and Paroubek, 2010; Read, 2005; Somasundaran et al., 2009), como por exemplo a procura de padrões.

Um dos primeiros trabalhos que aborda a análise de sentimento de conteúdos digitais, numa abordagem distante, surge por Pang et al. (2002). É efetuada a classificação automática de críticas de filmes, como sendo positivas ou negativas. Os dados para aprendizagem e avaliação foram obtidos de bibliotecas online, que contém indicadores do nível de sentimento desses dados, baseado na escala de classificação dada pelos autores das críticas. Os autores exploram os classificadores de *NB*, Maximum Entropy e *SVM*. Existe tratamento dos dados antes da avaliação dos modelos, é tido em conta o efeito provocado pela negação no contexto. Como *features* foram utilizadas diversas combinações: unigramas, bigramas, unigramas com bigra-

mas, bigramas, unigramas com anotações POS (parts-of-speech), adjetivos, os unigramas mais utilizados e unigramas com a respetiva posição. Para os unigramas foi testada a sua presença e a frequência, para os restantes a presença. Os unigramas, apresentam os melhores resultados quando é calculada apenas a presença da *feature*, com 82,90% de *accuracy* para o *SVM*. Na experimentação das restantes *features* a *accuracy* não é melhorada.

No trabalho de Kim and Hovy (2004), a abordagem passa por explorar a análise de sentimento após identificar o tópico e o autor do sentimento em determinado documento, com a classificação de três possíveis classes: positiva, negativa ou neutra. Esta classificação é feita em dois passos distintos, primeiro ao nível da palavra e depois ao nível da frase. Para o nível da palavra, foram propostos dois modelos. O primeiro assenta na criação de duas listas de palavras e sua respetiva polaridade, anotadas à mão, que servirão de semente para a procura de sinónimos. Para as ambiguidades (sinónimos podem não verificar o mesmo sentimento) verificadas no primeiro modelo, é proposto outro que visa medir a força do sentimento da palavra, baseia-se em encontrar que classe tem mais sinónimos de determinada palavra. Na análise de sentimento ao nível da frase, é considerado o pressuposto que a expressão de sentimento se encontra muitas vezes na região do tópico e titular da frase. Inicialmente é feita uma análise do tópico e do titular (pessoa/organização). São definidas várias regiões onde as palavras são tidas em conta. Para classificar as frases, foram definidos três modelos. No primeiro modelo, verifica-se uma anulação de sentimentos negativos que levam a que frases como "the California Supreme Court agreed that the state's new term-limit law was constitutional" e "the California Supreme Court disagreed that the state's new term-limit law was unconstitutional" tenham a mesma polaridade. No segundo modelo, a classificação é feita com base na média da força do sentimento das palavras em determinada região. No terceiro modelo, faz a classificação através da contagem do número de palavras com sentimento mais forte em determinada região. Para avaliação dos resultados, de uma lista de adjetivos e verbos, uma percentagem deles foram anotados por humanos com uma das três classes. Estas listas foram usadas como semente para classificação das restantes. Na classificação do sentimento da frase, na experimentação dos três modelos, testando os vários modelos de classificação de palavra, em conjugação com as diferentes regiões, os melhores resultados são obtidos pelo primeiro modelo, utilizando a classificação das palavras o modelo dois, com uma *accuracy* de 81% quando é conhecido o titular do sentimento e de 67% quando é desconhecido. Estes modelos apresentam algumas limitações, na classificação da palavra, existe a ambiguidade de uma palavra poder representar um sentimento negativo e positivo ao mesmo tempo, na classificação da frase, a abordagem focada na entidade a que se refere o sentimento, apenas foca o sentimento exprimido mais próximo da entidade.

Diferentes técnicas são utilizadas para os casos em que existe escassez de dados anotados. No trabalho explorado por Read (2005), é desenvolvido um classificador que tem a capacidade de ser treinado independentemente do tópico, domínio e tempo, com base em dados anotados

automaticamente de acordo com as emoções presentes. Numa primeira fase os autores exploram estas dependências, comprovando-as com experimentação, implementando uma versão própria de *NB* e a implementação de *SVM* de Joachims (1999) com o nome SVM *light*. Em relação à dependência do tópico, é demonstrando que as técnicas de classificação automática diminuem a sua *accuracy* quando o tópico dos dados de treino é diferente do tópico dos dados de teste. Quanto à dependência de domínio, também é verificado uma diminuição significativa da *accuracy*. Também se verifica uma dependência temporal quando os dados de treino são temporalmente diferentes dos dados de teste. Para treinar o modelo, foi obtida uma amostra relativamente grande de artigos contendo emoções, e testadas vários configurações. Foram obtidos resultados médios de 61,5% para o classificador *NB* e 70,1% para o classificador *SVM*. Para a baixa *accuracy* são apontados dois possíveis fatores, a baixa cobertura ao nível do número de *features* por parte dos classificadores e o possível ruído presente nos artigos. Noutro trabalho de Yang et al. (2007), são utilizadas as mensagens de blogues com emoções, que são utilizadas como corpus, para testar implementações nos classificadores *SVM* e CRF (Condition Random Field). A arquitetura do modelo, passa primeiro pela construção de um léxico com emoções, que serão as principais *features* que permitirão fazer a classificação ao nível da frase. Para treinar o classificador é adotado uma versão do *SVM* de LIBSVM e uma implementação de *CRF* de MALLET, que além das *features* tem em conta o sentimento das frases na sua vizinhança. Na fase de experimentação, além dos classificadores *SVM* e *CRF* é também testada uma implementação de Bayes. Para classificação ao nível do documento foram tidos em conta três critérios, atribuir a emoção presente em mais frases com classificação, na série mais longa com a mesma emoção e a atribuir a emoção verificada na última frase. Como resultados ao nível da frase, num primeiro teste com duas classes de emoções, o melhor classificador foi o *CRF* com 82,20% de *accuracy*, com 50 *features* e *SVM* com melhores resultados do que *NB*. Qualquer das implementações demonstra uma redução da *accuracy* aquando do aumento do número de *features*. Já no teste com quatro classes de emoções possíveis, o aumento do número de *features* melhora a *accuracy* do classificador *SVM* e de Bayes, com a melhor *accuracy* por parte do *CRF* com 50 *features* de 56%. Em relação à *accuracy* ao nível do documento, os melhores resultados são obtidos pelo terceiro critério, quando se tem conta o sentimento da última frase. Vosoughi et al. (2015) afirmam que perante a crescente quantidade dados existente para o Twitter, os classificadores que utilizam supervisão distante tendem a superar os classificadores que utilizam dados anotados à mão.

Os trabalhos que efetuam análise de sentimento às mensagens da rede social Twitter são mais recentes, após o seu grande crescimento de popularidade, um dos primeiros trabalhos é de Go et al. (2009). Foram utilizados os classificadores *NB*, *ME*, *SVM* e um outro baseado em palavras-chave, que foram treinados usando a abordagem utilizado por Read (2005), através de *tweets* contendo emoções. As *features* utilizadas para testar os classificadores foram uni-

gramas, bigramas e anotações POS. Os resultados obtidos foram muito semelhantes aos obtidos por Pang et al. (2002) com os mesmos classificadores, usando como *features* unigramas e bigramas. A adição de anotações POS em conjunto com os unigramas decresceu a *accuracy* dos três classificadores. Os autores apontam algumas ideias a ser exploradas para melhorar a *accuracy* dos classificadores, ao nível da semântica, que nome está mas associado com determinado verbo e classificá-lo de acordo. Ao nível do domínio, quando não se entra no contexto de vários. No trabalho de Barbosa and Feng (2010), também são classificados *tweets*. Os autores implementam um método de classificação em dois passos, com dados obtidos de três fontes que efetuam análise de sentimento sobre *tweets*, fornecendo dados anotados. No primeiro passo é analisada a subjetividade do *tweet*, no segundo passo é classificado o *tweet* subjetivo como positivo ou negativo. É explorado a limitação de os *tweets* serem pequenas mensagens e são implementadas duas novas *features*, meta-informação das palavra contidas nos *tweets* e a sintaxe do *tweets*. Foram testados diferentes algoritmos de aprendizagem automática e apresentados resultados para o *SVM*. Na classificação da subjetividade foi atingida uma taxa de erro de 18,10%, com prévio tratamento dos dados. Na classificação da polaridade, o melhor resultado apresenta uma taxa de erro de 18,70%. Esta abordagem mostra que estas *features* mostraram ser resistentes ao tamanho dos dados de treino e consistentes quando os dados das fontes poderão ser algo tendenciosos. A combinação de diferentes fontes de dados mostra ser eficiente. Trabalho futuro apontado pelos autores passa por identificar onde está o focus do *tweet*. Outro trabalho sobre o Twitter, é de Agarwal et al. (2011) Apresentam três modelos para a classificação de sentimento. Um baseado em unigramas, que servirá de base para comparar com um modelo baseado em *features*, algumas anteriormente usadas e propostas novas, e outro modelo baseado em "tree kernel", que permite a representação dos *tweets* na forma de árvore, permitindo combinar várias categorias de *features* numa representação. É introduzido um pré-processamento dos dados através do auxílio de um dicionário de emoções e de acrónimos e calculada a polaridade de cada palavra, parte das *features* utilizadas tem este peso em conta. Os melhores resultados obtidos na classificação para a classificação de duas classes, positiva e negativa, foram obtidos com o modelo de unigramas juntamente com o modelo de *features*, com uma *accuracy* de 75,40%. As *features* com melhores resultados são as que combinam uma polaridade prévia das palavras com anotações POS. A utilização das características da sintaxe dos *tweets* melhora os resultados ligeiramente. Numa das combinações entre os diferentes modelos a utilização unigramas em conjunto com a utilização de *features* de anotações POS com polaridade prévia obtiveram resultados de 75,10%, muito perto dos resultados obtidos utilizando todas as *features*.

2.1.1 Abordagens baseadas em informação linguística

Numa análise linguística mais profunda, existem trabalhos que exploram a existência de sarcasmo, que no caso de frases curtas como os *tweets*, podem alterar a polaridade de uma frase que possa parecer positiva como negativa e vice-versa (Davidov et al., 2010; González-Ibáñez et al., 2011). No trabalho de González-Ibáñez et al. (2011) , com um conjunto de dados de *tweets* anotados pelos próprios, são utilizadas *features* linguísticas para análise de sarcasmo. O modelo construído foi testado e comparado com a classificação humana, e os autores destacam que ambos têm um mau desempenho. Com esta conclusão afirmam que a anotação de dados de *tweets* com sarcasmo por parte de humanos pode não ser confiável, a não ser que as anotações sejam feitas pelo próprios que escreveram frases sarcásticas.

Várias análises linguísticas são encontradas na literatura, outra está relacionada com as técnicas para lidar com os erros ortográficos, no trabalho de Mullen and Collier (2004), introduz no pré-processamento dos dados um corretor ortográfico sobre palavras consideradas mal escritas (não encontradas no dicionário), sendo substituídas pela palavra corretamente escrita mais próxima. Os autores de Saif et al. (2012) apresentam uma abordagem na qual exploram a semântica presente no texto. Para cada entidade presente no *tweet*, é adicionado um conceito semântico como *feature* adicional e é comparada a *accuracy* em relação à utilização de outras *features*. A *feature* de semântica supera o modelo base com unigramas e anotações POS, observando que esta abordagem é apropriada quando o conjunto de dados existente é grande e abrange uma larga quantidade de tópicos. No trabalho de Aue and Gamon (2005), é abordada a dificuldade dos classificadores terem uma boa performance em dados de teste de domínio diferente que os dados de treino. Com dados de quatro domínios diferentes, propõem quatro abordagens nas quais exploram a pouca utilização dos dados anotados do domínio no qual se está a fazer a classificação. Os resultados entre as diferentes abordagens de treino em três domínios e teste com o domínio que sobra chega a ser de cerca de 10%. Numa diferente utilização de n-gramas, os autores de Vicente and Saralegi (2014), obtiveram uma melhor performance em relação ao seu modelo anterior, destacando como fatores mais importantes n-gramas baseados na sintaxe e com a combinação de vários léxicos de polaridade, melhoramento que só refletiu na fase de testes, destacam os autores.

A exploração sintática da frase em processamento natural da língua tem o nome de anotações POS, que consiste em identificar a que categoria morfossintática as palavras pertencem, nome, verbo, adjetivo, etc., e com isso construir padrões que ajudem na identificação da polaridade. A utilização de anotações POS pode permitir de alguma maneira a desambiguação do significado das palavras, textos subjetivos geralmente estão escritos na primeira pessoa ou dirigidas a alguém, na segunda pessoa, enquanto que nos textos objetivos na terceira pessoa (Pak and Paroubek, 2010). Os autores de Agarwal et al. (2011) utilizam uma combinação

de anotações POS com a polaridade das palavras, para construir diversas *features.* No trabalho de Pla and Hurtado (2014), a utilização de anotações POS serve para apenas incluir no modelo palavras que pertençam a determinadas categorias, nomes, verbos, adjetivos e advérbios.

O tratamento de negações na fase de extração de *features* desempenha um papel importante na análise de sentimento (Jia et al., 2009; Zhu et al., 2014). São procuradas palavras específicas da língua que neguem o contexto, e tradicionalmente o âmbito da negação é tudo até ser encontrado um caracter de pontuação ou o fim da frase (Batista and Ribeiro, 2013), onde todas as palavras são negadas através de um prefixo (ex: "não gosto" , "NOT_gosto) (Pang et al., 2002). Esta abordagem é de facto uma das mais referidas na literatura. No trabalho de Zhu et al. (2014), é explorado que diferentes palavras de negação podem ter efeitos diferentes no sentimento, e descriminam a palavra negada, tratando as diversas negações de maneira diferente (ex: "não gosto" fica "gosto_Nao" e "nunca gosto" fica "gosto_Nunca").

Modelos que usam fontes de dados que forneçam uma determinada pontuação ou polaridade, podem influenciar a *accuracy* dos dados. Modelos que usem mensagens com emoções para treinar modelos podem sofrer de problemas como a ambiguidade, quando se emite um sentimento positivo com uma emoção negativa presente, por outro lado, podem fornecer independência do tópico, domínio e tempo, ideias para uma rede social. Modelos que utilizam dados anotados à mão podem não providenciar dados suficientes para criar um modelo que seja resistente ao domínio, quando existe uma discrepância grande entre os dados de treino do modelo e de teste. Vários classificadores com diferentes *features* já mostram uma *accuracy* elevada quando é calculada a polaridade de duas classes (positiva e negativa) para *tweets* subjetivos, e três classes (positiva, negativa e neutra) quando não é observada a objetividade ou subjetividade. Melhorar a *accuracy* dos classificadores poderá passar por uma melhor qualidade e quantidade. Barbosa and Feng (2010) apresentam um modelo resistente ao tamanho dos dados de treino..

2.2 Classificação hierárquica e outros modelos

Outros trabalhos abordam a análise de sentimento através de modelos de classificação hierárquica, com vários níveis de decisão. Batista and Ribeiro (2013) implementam um modelo com uma dupla função, classificar os *tweets* quanto ao sentimento e quanto ao tópico. É utilizado um modelo de máxima entropia para uma classificação de seis possíveis classes: muito positivo, positivo, muito negativo, negativo, neutro e sem sentimento. De suporte ao modelo são utilizados unigramas, unigramas em conjunto com bigramas e um léxico de sentimento. Tanto os bigramas como o léxico de sentimento demonstraram não melhorar a *accuracy* dos resultados de treino e de teste. A informação acerca do autor e a utilização de marcas de pontuação mostraram ser *features* efetivas. É apontado como trabalho futuro, analisar o tipo de polari-

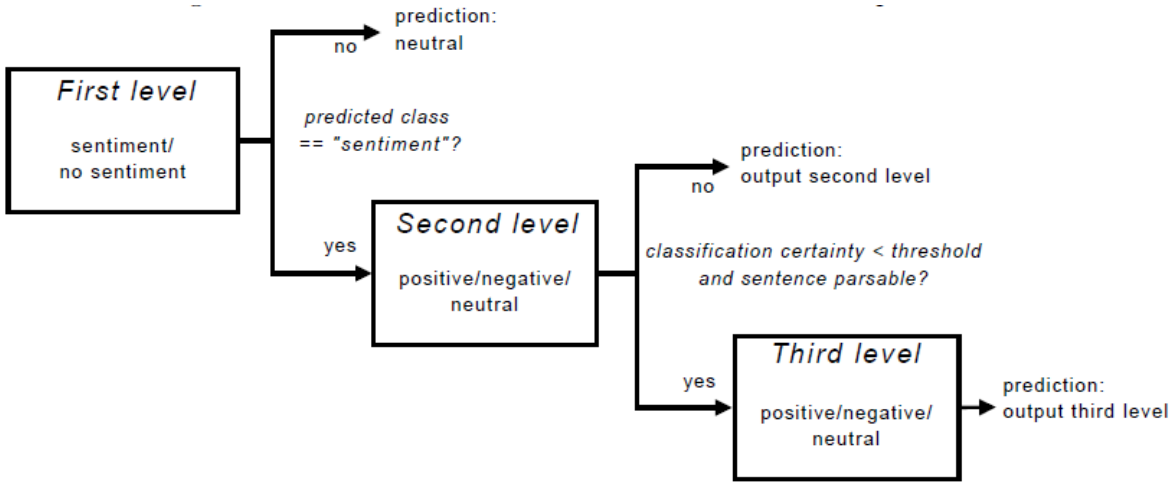

Figura 2.1: Cascata de três níveis usada por Boiy and Moens (2009).

dade do *tweet*, ou seja, a concordância ou discordância, meta-informação do autor do *tweet* e a utilização de um léxico com aprendizagem automática. No trabalho de Sánchez-Mirabal et al. (2014) é proposto um modelo com classificação em dois níveis, no primeiro nível o classificador K-NN (k-Nearest Neighbors) utilizando a métrica léxica de Levenshtein, que perante determinado limiar, procede a uma classificação automática, se chegar ao segundo nível é aplicado um classificador de regressão. O trabalho de Boiy and Moens (2009) analisa o sentimento sobre textos de fóruns, críticas e blogues em três línguas diferentes, de tópicos selecionados, com dados anotados manualmente em três categorias, positivo, negativo e neutro. Fizeram experiências de classificação em cascata com o classificador Multinomial Naïve Bayes, ME e SVM, até três níveis, com possível classificação final em cada nível. No primeiro nível é identificado se o texto tem a presença de sentimento, sendo catalogado logo como neutro. Ao chegar ao segundo nível é verificada se classe prevista contém sentimento, positiva ou negativa, se for avança para o terceiro nível onde é verificado se o valor da classe prevista ultrapassa determinado limite definido para saber se o texto é analisável. A Figura (2.1) mostra a cascata de três níveis usada pelos autores.

3 Análise de Sentimento de Tweets

Este capítulo descreve as experiências realizadas e apresenta os resultados obtidos com os algoritmos de aprendizagem automática na classificação de *tweets* espanhóis. É apresentado o corpus utilizado, a fase de pré-processamento dos dados, o processo de extração de *features*, a abordagem ao processamento de tweets com as arquiteturas em experimentação, os diversos classificadores binários e as suas combinações em cascata. Na parte final são apresentados resultados dos classificadores binários individualmente, das cascatas, os resultados com o subconjunto de teste e uma análise de erros das experiências realizadas.

3.1 Corpus TASS

As experiências aqui reportadas usam um conjunto de 68 000 *tweets* espanhóis, obtidos no contexto do workshop TASS (*Taller de Análisis de Sentimientos*) Villena-Román et al. (2015), evento satélite da conferência SEPLN (Sociedad Española para el Procesamiento del Lenguaje Natural). O corpus encontra-se dividido em dois subconjuntos, sendo 7 200 *tweets* utilizados para treino e cerca de 60 800 *tweets* usado para avaliação. Cada *tweet* contém a seguinte informação:

- Identificador único;
- Nome do utilizador;
- Mensagem;
- Data e hora;
- Língua;
- Polaridade que se divide em seis possíveis, positiva (P), muito positiva (P+), negativa (N), muito negativa (N+), neutra (NEU) e sem sentimento (NONE). A Tabela 3.1 apresenta a distribuição das várias classes de sentimento nos conjuntos de treino e teste;

Polaridade	# Treino	# Teste	% Treino	% Teste
Positivos (P)	1 019	1 488	14,12	2,45
Muito Positivos (P+)	1 764	20 745	22,44	34,12
Negativos (N)	1 221	11 287	16,91	18,56
Muito Negativos (N+)	903	4 557	12,51	7,50
Neutros (NEU)	610	1 305	8,45	2,15
Sem Sentimento (NONE)	1 702	21 416	23,58	35,22
Total	7 219	60 798	100,00	100,00

Tabela 3.1: Polaridade *tweets* do subconjunto de dados de treino e de teste.

- Tipo de polaridade, com dois valores possíveis, "AGREEMENT" e "DISAGREEMENT", que expressa o nível de acordo ou desacordo do sentimento dentro do conteúdo;
- Tópicos;
- Entidade.

As áreas cobertas pelos *tweets* abrangem dez tópicos, tais como, política, entretenimento, economia, comunicação, cultura, música, cinema, desporto, literatura e um último denominado outros. Existe também um ficheiro XML com informação dos autores que publicaram pelo menos um *tweet* no conjunto de dados e que inclui entre outras informações, o tipo de utilizador, com três tipos diferentes, jornalista, personalidade famosa ou político. O conjunto de dados abrange cerca de duzentos autores provenientes de vários países que adotam a língua espanhola como língua oficial (Villena-Román et al., 2015).

3.2 Pré-processamento dos dados

Depois da extração dos dados do XML, estes sofrem um processo de normalização com os seguintes passos:

- Tokenização: é o processo de segmentar um conjunto de dados, num conjunto de palavras, números ou símbolos, que se designam em PNL por *tokens*.
- Remoção de *stop words*: existem palavras que a sua presença pode não acrescentar valor na deteção de polaridade, pois podem aparecer em qualquer contexto de um conjunto de dados com objetividade ou subjetividade no seu conteúdo. Foi utilizado um ficheiro fornecido pelo TASS contendo 324 palavras espanholas (exemplos: tu, ya, uno).

Número de *tokens*	116 730
Número de *stop words*	45 687
Número de palavras capitalizadas	7 413
Número de *long words*	1 890
Número de endereços web	2 548
Número de *hashtags (#)*	1 671
Número de *targets (@)*	3 670
Número de negações	1 189
Número de emoções	213

Tabela 3.2: Estatísticas sobre os dados utilizados nas experiências.

Para o subconjunto de dados de treino, na Tabela 3.2 verificam-se que 39,13% dos *tokens* são *stop words,* 6,35% são palavras capitalizadas e 1,61% são *long words.* Em relação aos fenómenos particulares dos microblogues, verificam-se que 2,18% dos *tokens* são endereços web, 1,43% são *hashtags* e 3,14% são *targets*, todos juntos perfazem um total de 6,75% do total dos *tokens.* Verificam-se também que 1,02% de *tokens* vão provocar negação do contexto e que 0,20% são emoções.

3.3 Extrator de *features*

Após o pré-processamento dos dados, estes estão prontos para lhes serem extraídas características importantes para a deteção de sentimento. Existem *features* específicas ao Twitter (*hashtags* e *targets*) e de microblogues em geral como os endereços web. O tratamento de negações em específico, é dependente da língua, no sentido em que são determinadas palavras que indicam a negação. Foram exploradas as seguintes características:

Frequência (unigramas): representa a frequência com que o unigrama ocorre no conjunto de dados para ser considerado *feature*, permitindo limitar o número de *features.*

Frequência (bigramas): representa a frequência com que o bigrama ocorre no conjunto de dados para ser considerado *feature*, permite limitar o número de *features* e distinguir do valor da frequência dos unigramas para outras experiências.

Negações: as negações em frases curtas como os *tweets* podem ter um peso significante. Na análise de expressões como "eu gosto" e "eu não gosto", apesar de ambas conterem a palavra "gosto" que pode indicar um sentimento positivo, a presença de uma palavra a negar o contexto, inverte a polaridade. Para verificar a existência de uma negação, especificamente com palavras da língua espanhola que representam negação, "no" ou

"nunca", todas as palavras que lhe seguirem até ser encontrado um caracter de pontuação ou o fim da frase, serão negadas através da concatenação do prefixo "NO_". (exemplo: a frase "Eu não gosto de futebol." fica "Eu não NO_gosto NO_de NO_futebol"). Foi também testado o tratamento da negação com base no trabalho de Zhu et al. (2014), em que é a palavra que provoca a negação do contexto, é concatenada com as seguintes palavras negadas (ex: a frase "Eu não gosto de futebol" depois de negada fica "Eu não gosto_Nao futebol_Nao").

HTTP ou WWW: Verificação se o *token* começa pelas inicias "http" ou "www". O endereço web é substituído por "*HTTP" com o seu peso reduzido (Barbosa and Feng, 2010).

***Hashtags*:** Fenómeno particular do Twitter, que permite identificar tópicos e assuntos mais falados no momento, é identificado quando o primeiro caracter do *token* começa por "#". É utilizado como *feature* a substituição dos *hashtags* encontrados por "*HASHTAG" com um peso reduzido.

***Targets*:** Fenómeno particular do Twitter, que permite expressar o destinatário do *tweet* com a junção do nome de utilizador ao prefixo "@". É utilizado como *feature* a substituição dos *targets* encontrados por "*TARGET" com um peso reduzido.

Dicionário de emoções: Processamento de *emoticons* com base num dicionário que quando ocorrem são substituídos por _SMILE_ com peso aumentado. O dicionário utilizado encontra-se em [1];

***Long Words*:** Identificação de *long words* quando num *token* em que todos os caracteres são alfabéticos, existe a ocorrência do mesmo caracter pelo menos três vezes. É criada uma nova *feature* com a *long word* normalizada mantendo o ênfase (exemplo: a palavra "coi-soooooo" depois de normalizada passa a "coisoo").

Maiúsculas: Quando são identificados *tokens* em que todos os caracteres são alfabéticos e maiúsculos, tem um peso superior e é criada uma nova *feat*ure dessa palavra em minúsculas com menor peso.

***Long word* e maiúscula:** Identificação de *long word* e ser maiúscula em simultâneo, é criada uma nova *feature* com a palavra normalizada e outra com a palavra em minúsculas com um peso inferior.

Primeiro caracter capitalizado: Quando o *token* tem o seu primeiro caracter capitalizado, é criada uma nova *feature* dessa palavra em minúsculas com um peso menor.

[1] https://www.csh.rit.edu/∼kenny/misc/smiley.html

Bigramas: Os bigramas pretendem retirar mais contexto e ajudar a na desambiguação. A construção dos bigramas é feita sobre o vetor de *features* normalizado. Os bigramas são obtidos percorrendo o vetor, através da concatenação do *token* na posição n com o *token* na posição $n + 1$, até se atingir a última posição.

Informação do autor: Utilização de informação sobre o autor do conteúdo, com a utilização como *features* do seu nome e tipo de utilizador.

Presença do caracter de exclamação: Verificação da presença do caracter de interrogação em qualquer dos *tokens*.

Número de caracteres de exclamação: Contagem do número de caracteres de exclamação de todos os *tokens*.

Presença do caracter de interrogação: Verificação da presença do caracter de interrogação em qualquer dos *tokens*.

Número de caracteres de interrogação: Contagem do número de caracteres de interrogação de todos os *tokens*.

Tokens com menos de dois caracteres: Todos os *tokens* com menos de dois caracteres são removidos do vetor de *features*.

Remoção de caracteres de pontuação: São removidos alguns caracteres de pontuação e símbolos do *token*.

Remover caracteres não alfanuméricos: Remoção de todos os caracteres não alfabéticos do *token*.

Tamanho do *tweet*: Contagem do número de caracteres no *tweet*.

Caracteres minúsculos: Contagem do número de caracteres alfabéticos minúsculos.

Caracteres maiúsculos: Contagem do número de caracteres alfabéticos maiúsculos.

Caracteres alfanuméricos: Contagem do número de caracteres alfanuméricos.

Caracteres alfabéticos: Contagem do número de caracteres alfabéticos (Barbosa and Feng, 2010).

Tópicos: Os vários tópicos do *tweet* são considerados *features*, inspirado no trabalho futuro proposto por Vosoughi et al. (2015).

Léxicos de polaridade: Foram utilizados dois léxicos de polaridade de Perez-Rosas et al. (2012), com dois dicionários. Um com as palavras consideradas mais importantes, com 1347 palavras, que neste trabalho tem o nome de *LexicoFS* e outro de média força com 2496 palavras, que tem o nome de *LexicoMS*. Ambos contêm a informação da polaridade da palavra e um outro léxico com as línguas espanhola e portuguesa (Mohammad and Turney, 2010, 2013), que além da informação se a palavra é positiva e negativa, têm 10 estados de espírito possíveis, raiva, antecipação, desgosto, medo, alegria, tristeza, surpresa e confiança. Este léxico ao longo do trabalho tem o nome de *LexicoNRC*.

Número palavras positivas: Com base nos léxicos, são contabilizadas o número de palavras positivas presentes.

Número palavras negativas: Com base nos léxicos, são contabilizadas o número de palavras negativas presentes.

Lemmatização Transformação do *token* no seu lema, quando presente no dicionário de Reese et al. (2010).

Anotações POS Para as palavras que estão presentes no dicionário de Reese et al. (2010) que tem um total de 266 179 palavras (originalmente em inglês traduzido automaticamente para o espanhol), é acrescentado como *feature* a categoria morfossintática da palavra (nomes, verbos, adjetivos, advérbios, pronomes, determinantes, preposições, conjunções, interjeições e abreviaturas).

3.4 Processamento de *tweets*

Considere-se o cenário em que se pretende fazer classificação de sentimento em *tweets* utilizando seis classes possíveis: positivo (P), negativo (N), neutro (NEU), muito positivo (P+), muito negativo (N+) e sem sentimento (NONE). A classe NEU corresponde à existência de sentimentos positivos e negativos no mesmo documento, enquanto que a classe NONE corresponde a não existir qualquer tipo de sentimento. Este problema pode ser resolvido com base em quatro ou mais classificadores binários, tal como se pode observar na Figura 3.1. A arquitetura CS1 (Classificação Sentimento 1) é a mais simples e consiste em apenas dois níveis de classificação. Num primeiro nível, combinam-se os resultados de dois classificadores para distinguir entre quatro grupos de classes: NONE, P/P+, N/N+ e NEU. Um segundo nível é aplicado apenas aos grupos que contêm mais do que uma classe e corresponde a identificar a intensidade do sentimento. Uma vantagem adicional deste tipo de abordagem é a possibilidade de utilizar diferentes *features* em cada um dos classificadores. Por exemplo, no caso da língua portuguesa, adjetivos como “muito” ou “extremamente” podem constituir pistas fortes para ambos

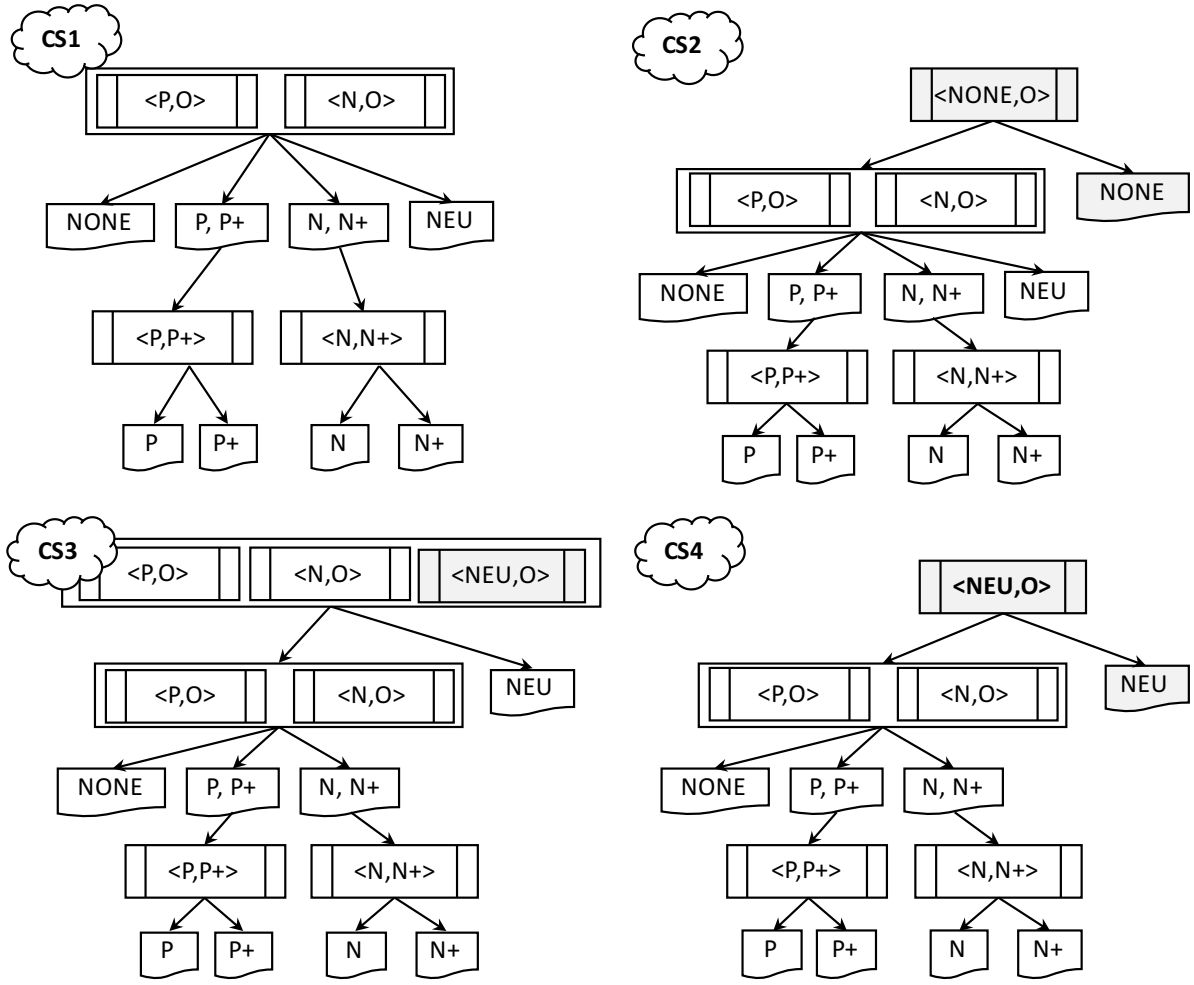

Figura 3.1: Arquiteturas de classificação em cascata para processamento de *tweets* (O - representa o conjunto das classes restantes).

os classificadores do segundo nível, apesar de poderem ser pouco úteis nos classificadores do primeiro nível. A arquitetura CS2 identifica logo no primeiro nível quais os *tweets* que têm ou não um sentimento associado, com a vantagem de filtrar uma porção de dados logo no primeiro nível. Os níveis seguintes correspondem à arquitetura CS1. A arquitetura CS3 surge devido à dificuldade acrescida de classificar *tweets* neutros, por terem tanto sentimentos positivos como negativos. A arquitetura CS4 surge também com o objetivo de tratar de forma particular o caso dos sentimentos neutros.

3.5 Classificadores binários

Abaixo encontra-se a todos os classificadores binários utilizados nas cascatas e a sua função em detalhe:

- <P,O> (Positivo - Outro): Os *tweets* na fase de tratamento com polaridade P ou P+ são anotados como P, todas as restantes polaridades serão anotadas como O. Este classificador tem como objetivo identificar se o *tweet* contém polaridade positiva ou outra.

- <N,O> (Negativo - Outro): Os *tweets* na fase de tratamento com polaridade N ou N+ são anotados como N, todas as restantes polaridades serão anotados como O. Este classificador tem como objetivo identificar se o *tweet* contém polaridade negativa ou outra.

- <P,P+> (Positivo - Muito Positivo): Os *tweets* na fase de tratamento com polaridade P, N, N+, NONE e NEU são anotados como P, os restantes mantém a polaridade P+. Este classificador tem como objetivo diferenciar a intensidade de sentimento nos casos em que o *tweet* é considerado positivo pelo classificador <P,O>.

- <N,N+> (Negativo - Muito Negativo): Os *tweets* na fase de tratamento com polaridade P, N, N+, NONE e NEU são anotados como N, os restantes mantém a polaridade N+. Este classificador tem como objetivo diferenciar a intensidade de sentimento nos casos em que o *tweet* é considerado negativo pelo classificador <N,O>.

- <NONE,O> (Sem Sentimento - Outro): Os *tweets* na fase de tratamento com polaridade P, P+, N, N+ e NEU são anotados como O, os restantes mantém a polaridade NONE. Este classificador tem como objetivo perceber se existe ou não subjetividade no *tweet*, quando classifica como O e que não existe sentimento quando classifica como NONE.

- <NEU,O> (Neutro - Outro): Os *tweets* na fase de tratamento com polaridade P, P+, N, N+ e NONE são anotados como O, os restantes mantém a polaridade NEU. Este classificador tem como objetivo identificar se o *tweet* contém polaridade neutra ou outra.

3.6 Resultados com classificadores binários

Com o subconjunto de dados de treino, foi avaliado numa primeira fase *o* desempenho dos classificadores binários individualmente. Foram utilizadas como *features* unigramas e bigramas com variação da frequência (igual para ambas). O principal objetivo nesta fase é tentar perceber nos classificadores em experimentação, os melhores resultados em combinação com a utilização do menor número de *features*, dado que um maior número de *features* alarga o tamanho do modelo e este aumento de complexidade pode não ser linear. Além disso, um número muito grande de *features* pode provocar o problema de *overfitting* ao confundir o classificador com demasiadas *features*. Nas tabelas abaixo encontram-se os melhores resultados para cada um dos classificadores binários, com a melhor configuração da frequência para *feature* e da combinação da utilização de unigramas e de unigramas em conjunto com bigramas, que servirá como base

classificador	*features*	n2f	# *features*	*recall*	*f-measure*	*accuracy* (%)
SMO	unigramas	7	1367	0,72	0,72	72,92
RL	unigramas	15	527	0,71	0,71	71,26
NB	unigramas + bigramas	2	8 077	0,67	0,67	67,45

Tabela 3.3: Melhores resultados de cada classificador para o <P,O>.

P	O	classificado como
306	228	P
176	734	O

Tabela 3.4: Matriz de confusão com a execução do *SMO*.

para exploração dos resultados das arquiteturas de combinação dos classificadores binários. O valor do parâmetro de complexidade nesta fase não foi alterado (valor por defeito é 1).

Classificador <P,O> (Positivo - Outro)

Teve o seu melhor resultado com o classificador *SMO*, com um resultado muito próximo do *RL*, com menos de metade do número de *features*, ambos utilizando apenas unigramas. O *NB* teve o melhor resultado combinando unigramas e bigramas com uma frequência baixa, com mais de 8 mil *features*, mas muito abaixo dos anteriores. A Tabela 3.3 mostra os resultados da melhor execução de cada classificador e a Tabela 3.4 mostra a respetiva matriz de confusão da melhor execução.

Classificador <N,O> (Negativo - Outro)

Novamente, o classificador *SMO* obteve o melhor resultado utilizando unigramas. Também com unigramas e com cerca de 30% do número de *features* o classificador *RL* ficou a 0,49% do melhor resultado. O *NB* com unigramas e bigramas ficou a cerca de 3% dos melhores resultados, utilizando um grande número de *features* comparativamente. A Tabela 3.5 mostra os resultados de cada classificador e a Tabela 3.6 a respetiva matriz de confusão da melhor execução.

classificador	*features*	n2f	# *features*	*recall*	*f-measure*	*accuracy* (%)
SMO	unigramas	6	1669	0,74	0,75	74,52
RL	unigramas	13	616	0,74	0,74	74,03
NB	unigramas + bigramas	2	8077	0,70	0,71	70,98

Tabela 3.5: Melhores resultados de cada classificador para o <N,O>.

N	O	classificado como
219	233	N
149	843	O

Tabela 3.6: Matriz de confusão da execução do SMO.

classificador	*features*	n2f	# *features*	*recall*	*f-measure*	*accuracy* (%)
SMO	unigramas	15	527	0,86	0,86	86,36
RL	unigramas	20	329	0,85	0,85	85,18
NB	unigramas	20	329	0,83	0,83	83,80

Tabela 3.7: Melhores resultados de cada classificador para o <N,N+>.

Classificador <N,N+> (Negativo - Muito Negativo)

Os melhores resultados de cada classificador foram todos obtidos com menos de 600 *features*. O *SMO* obteve o melhor resultado, seguido do classificador *RL* a cerca de 1% e o *NB* um pouco mais longe, a cerca de 3%. Todos também têm em comum a utilização de unigramas. A Tabela 3.7 mostra os resultados e a Tabela 3.8 a matriz de confusão da melhor execução.

Classificador <P,P+> (Positivo - Muito Positivo)

Desta vez com a utilização de unigramas e bigramas o *SMO* obteve novamente o melhor resultado com menos de 1000 *features*. O classificador *RL* fica a cerca de 1%, com menos de metade do número de *features*, utilizando apenas unigramas. O *NB*, mais uma vez em último, a cerca de 3% do melhor resultado. A Tabela 3.9 mostra estes resultados e a Tabela 3.10 a matriz de confusão da melhor execução.

Classificador <NONE,O> (Sem sentimento - Outro)

Novamente, com unigramas e bigramas, *SMO* tem o melhor resultado, mas desta vez com execuções próximas por parte dos classificadores *NB* e *RL* que ficaram ambos a cerca de 1%. Ao contrário do verificado até aqui, o *NB* não fica com o pior resultado. A Tabela 3.11 mostra

N	N+	classificado como
1 192	51	N
166	35	N+

Tabela 3.8: Matriz de confusão da execução do *SMO*.

classificador	*features*	n2f	# *features*	*recall*	*f-measure*	*accuracy* (%)
SMO	unigramas + bigramas	10	922	0,82	0,82	82,20
RL	unigramas	17	437	0,81	0,81	81,23
NB	unigramas	18	405	0,79	0,79	79,22

Tabela 3.9: Melhores resultados de cada classificador para o <P,P+>.

P	P+	classificado como
1 006	140	P
160	138	P+

Tabela 3.10: Matriz de confusão da execução do SMO.

os resultados de cada classificador e a Tabela 3.12 mostra a matriz de confusão da melhor execução.

Classificador <NEU,O> (Neutro - Outro)

Por último, mais uma vez o *SMO* com o melhor resultado, e por mais uma vez também com a utilização de unigramas e bigramas. O classificador *RL* com apenas unigramas e cerca de metade do número de *features* fica a menos de 0,50% do *SMO*. O *NB* com perto de 4 mil *features*, tem o pior resultado, como mostra a Tabela 3.13. A Tabela 3.14 mostra a matriz de confusão para a melhor execução.

O classificador *SMO* obteve os melhores resultados em todos os classificadores binários. O *NB* só por uma vez não obteve o pior resultado do três, que foi no classificador <NONE,O>. Em relação à complexidade dos modelos, o classificador *RL* apresenta resultados próximos, utilizando consideravelmente menos *features*, através de números de frequência altos. Este facto reduz bastante o tempo de treino dos modelos.

classificador	*features*	n2f	# *features*	*recall*	*f-measure*	*accuracy* (%)
SMO	unigramas + bigramas	13	636	0,81	0,81	81,44
NB	unigramas	8	1 180	0,80	0,80	80,47
RL	unigramas	18	405	0,80	0,80	80,12

Tabela 3.11: Melhores resultados de cada classificador para o <NONE,O>.

NONE	O	classificado como
148	994	NONE
165	137	O

Tabela 3.12: Matriz de confusão da execução com o NB.

classificador	*features*	n2f	# *features*	*recall*	*f-measure*	*accuracy* (%)
SMO	unigramas + bigramas	13	636	0,89	0,89	89,20
RL	unigramas	20	329	0,89	0,88	88,78
NB	unigramas	3	3 770	0,87	0,88	87,60

Tabela 3.13: Melhores resultados de cada classificador para o <NEU,O>.

3.7 Resultados com classificadores em cascata

Depois de avaliado o desempenho dos classificadores binários, os classificadores *SMO* e *RL* apresentam os melhores resultados para serem testados nas arquiteturas apresentadas. Nesta fase já não vai ser avaliado o desempenho do *NB* em cascata, pelo seus maus resultados nos diversos classificadores binários. Para facilitar a consulta das Tabelas 3.17 e 3.15, algumas *features* foram agrupadas em conjuntos de *features*. As *HTW features* são a utilização em conjunto das *features* extraídas para *hashtags*, *targets*, endereços web e números. As IP (Intensidade Palavra) são consideradas as *features* que capturam intensidade nas palavras (*long words* + *long word* e todos os caracteres maiúsculos + todos os caracteres maiúsculos + primeiro caracter maiúsculo). A *IA* representa a informação do autor que contém as *features* nome do utilizador e tipo de utilizador. As *IC* (Informação Caracter) são a utilização em conjunto da contagem do número de símbolos, de dígitos, de caracteres alfabéticos, de caracteres maiúsculos e de caracteres minúsculos. A pontuação contém a presença e frequência dos caracteres de interrogação e exclamação. O *RT<2* representa a *feature* que remove todos os *tokens* com com dois ou menos caracteres. A *feature TT* (Tamanho *Tweet*) representa o tamanho do *tweet*. O *LexicoNRC* e *numPosAndNegNRC* são *features* construídas da utilização do léxico de Mohammad and Turney (2010, 2013). Os léxicos *LexicoFS, LexicoMS* e *numPosAndNegFS* são *features* construídas com base no léxico de Perez-Rosas et al. (2012). A *feature negImp* representa o tratamento da negação com contexto. É considerada que a *feature* traz valor ao sistema quando aumenta a *accuracy* em pelo menos umas das classificações de quatro ou seis classes. A arquitetura CS1

NEU	O	classificado como
13	143	NEU
53	1 235	O

Tabela 3.14: Matriz de confusão da execução com o *SMO*.

(Arquitetura CS1) será utilizada como ponto de partida para a avaliação das *features* mais importantes para cada um dos classificadores. Na experimentação de *features*, é dada prioridade à classificação em quatro classes desde que não tenha excessivo efeito negativo *n* classificação em seis classes.

3.7.1 Sequential Minimal Optimization

Na Tabela 3.15 encontram-se as diversas execuções realizadas até encontrar a melhor configuração de *features* para o modelo. No primeiro passo foi encontrando o melhor valor para a frequência dos unigramas para obter a melhor *accuracy*. Com o valor três obteve-se 48,13% e 35,39% para quatro e seis classes respetivamente (execução 1). Depois foram adicionadas as *features smileDic* que provocaram um aumento de classificação em ambas as classificações (execução 2). Depois foram testadas as *features* de pontuação na execução 3, que apesar de um pequeno aumento na classificação em quatro classes, provocaram um aumento considerável em seis classes, com mais de 1% de aumento. A informação do autor, provocou um aumento na classificação em ambas as classificações (execução 4). As *features IP* resultaram num aumento de classificação em quatro classes e num pequeno decréscimo em seis classes (execução 5). Todas as execuções anteriores foram realizadas com o parâmetro de complexidade no seu valor por defeito (c=1). A otimização deste parâmetro, com um varrimento entre 0,40 e 2,00, foi encontrado o valor ótimo em 0,66 (execução 6). As *features HTW* e a feature *TT*, não se revelaram importantes, com descidas na classificação em ambas as classes (execução 7 e 8). A utilização de *RT<2*, reduziu ao total cerca de 100 *features*, mas com uma descida da classificação em quatro e seis classes (execução 9). A utilização do *LexicoFS* não representou ganhou em nenhuma das classificações (execução 10). As primeira *features* lexicais a ter impacto positivo são a utilização das *numPosAndNegNRC*, com aumentos em ambas as classificações (execução 11). A introdução das *features numPosAndNegFS* trouxe um aumento do resultado em quatro classes (execução 12). A introdução de negações piora os resultados em ambas as classificações (execução 13). As *features IC* principalmente na classificação em seis classes, são importantes (execução 14). O processo de lematização, provoca um aumento em quatro classes e um decréscimo no resultado em seis classes (execução 15). A introdução de tópicos trouxe ganhos em ambas as classificações, mais considerável em seis classes (execução 16). As *features* do LexicoNRC provocam um pequeno aumento em ambas as classificações, com o melhor resultado para seis classes com um resultado de 41,07% de *accuracy* (execução 17). A alteração do valor de frequência para 13 em simultâneo com a utilização da *feature* negImp provoca o melhor resultado na classificação em quatro classes com 56,93% de *accuracy* (execução 18). Quer a utilização de bigramas, quer a utilização de anotações POS, não melhoram os resultados (execuções 19 e 20).

execução	*features*	# *features*	4 cl. (%)	6 cl. (%)
1	unigramas	1407	48,13	35,39
2	(1) + smileDic	1409	49,31	36,43
3	(2) + pontuação	1413	49,86	37,60
4	(3) + IA	1494	50,55	38,30
5	(4) + IP	1558	51,04	38,23
6	(5) + c=0.66	1558	52,15	38,99
7	(6) + HTW	1430	51,04	38,50
8	(6) + TT	1559	51,66	38,64
9	(6) + RT<2	1441	50,07	37,26
10	(6) + LexicoFS	1613	52,01	38,85
11	(6) + numPosAndNeg NRC	1560	52,91	39,89
12	(11) + numPosAndNeg FS	1562	53,12	39,47
13	(12) + negações	1525	52,77	38,43
14	(12) + IC	1567	54,22	40,44
15	(14) + lemas	1589	54,99	39,68
16	(15) + tópicos	1599	55,61	40,72
17	(16) + LexicoNRC	2399	55,96	**41,07**
18	**(17) + negImp + f=13**	**347**	**56,93**	41,00
19	(18) + bigramas (f=13)	364	56,65	40,37
20	(18) + anotações POS	423	55,40	38,99

Tabela 3.15: Melhoramento das *features* do classificador *SMO*.

arquitetura	4 cl. (%)	6 cl. (%)
CS1	**56,93**	**41,00**
CS2	56,44	40,79
CS3	**56,93**	**41,00**
CS4	56,86	40,93

Tabela 3.16: Resultados das arquiteturas com o *SMO*.

As *features* mais importantes no desempenho do classificador *SMO* são o dicionário de emoticons, a informação dos caracteres e a informação do autor.

3.7.1.1 Avaliação das diferentes arquiteturas

A Tabela 3.16 mostra os resultados com as várias arquiteturas com o conjunto de *features* da execução 20 da Tabela 3.15. As arquiteturas CS1 e CS3 obtiveram o melhor resultado em quatro classes com 56,93%, com resultados muito próximos das arquiteturas CS2 e CS4. Em relação à classificação em seis classes, igualmente os melhores resultados são com as arquiteturas CS1 e CS3 com uma *accuracy* de 41,00%. As arquiteturas CS2 e CS4 ficam novamente com resultados muito próximos.

Para estas execuções, os classificadores binários obtiveram a *accuracy* de 76,39% para o <P,O>, de 78,81% para o <N,O>, de 64,89% para o <P,P+>, de 79,57% para o <N,N+>, de 83,52% para o <NONE,O> e de 89,20% para o <NEU,O>.

3.7.2 Regressão Logística

Na Tabela 3.17 encontram-se as diversas execuções realizadas até encontrar a melhor configuração de *features* para o modelo. A melhor frequência para os unigramas serem considerados *features* foi com o valor 13, com resultados de 47,16% e 34,90% em quatro e seis classes respetivamente (1). As *features smileDic* mostraram melhorias em ambas classificações (execução 2). As *features HTW* apesar de provocarem uma ligeira descida na classificação de quatro classes, melhora em cerca de 0,50% a classificação em seis classes, sendo que para já não é mantida, dada a prioridade à classificação de quatro classes. A utilização de pontuação trouxe um ganho considerável em ambas as classificações (execução 4). As *features* de *IP* e de *IC* não trouxeram aumentos em nenhuma das classificações (execuções 5 e 6). A informação do autor trouxe ganhos em ambas as classificações, com um aumento maior em quatro classes (execução 7). A remoção de caracteres menores que dois, não trouxe ganhos (execução 8). A *feature TT* provoca uma descida em quatro classes e seis classes (execução 9). A utilização de tópicos trouxe um aumento de quase 1% na classificação de quatro classes (execução 10). A

execução	*feature*	# *features*	4 cl. (%)	6 cl. (%)
1	unigramas	616	47,16	34,90
2	(2) + smileDic	621	47,78	35,25
3	(2) + HTW	564	47,71	35,87
4	(2) + pontuação	625	49,79	36,36
5	(4) + IP	669	49,58	36,29
6	(4) + IC	630	49,38	35,87
7	(4) + IA	707	50,48	36,91
8	(7) + RT<2	651	49,45	36,91
9	(7) + TT	708	49,72	36,50
10	(7) + tópicos	717	51,32	37,53
11	(10) + lematização	795	52,70	37,67
12	(11) + numPosAndNegNRC	797	55,40	39,34
13	(12) + numPosAndNegFS	799	55,96	39,47
14	(13) + bigramas (f=13)	822	54,85	38,50
15	**(13) + negações**	**786**	**57,27**	**40,79**
16	(15) + negImp	786	57,13	40,72
17	(15) + anotações POS	1 148	53,32	37,40

Tabela 3.17: Desempenho das *features* com a arquitetura CS1 utilizando Regressão Logística.

lematização, com mais cerca de 80 *features*, provocou mais de 1% de aumento em quatro classes, e uma ligeira subida em seis classes (execução 11), principalmente devido a cerca de 5% mais de *tweets* negativos corretamente classificados. A *feature numPosAndNegNRC* revelou-se importante, trouxe quase 3% de aumento à classificação de quatro classes e cerca de 1,50% à classificação em seis classes (execução 12). A *feature numPosAndNegFS,* trouxe ganhos ligeiros, ainda assim, quase 0,50% em quatro classes (execução 13). A utilização de bigramas e variação da sua frequência não trouxe ganhos na classificação, o melhor resultado obtido foi com o valor de frequência igual a 13 (execução 14). O tratamento de negações, em ambas as classificações, com mais de 1% de aumento, também se revelou importante (execução 15). A noção do contexto no tratamento das negações provocou um ligeiro decréscimo na classificação em ambas as classes (execução 16). A utilização de anotações POS, provoca uma grande descida em ambas as classificações (execução 17).

A melhor execução (15) tem uma *accuracy* de 57,27% para quatro classes e de 40,79% para seis classes, em que se destacaram como *features* mais importantes, a pontuação, a lematização, o número de palavras positivas e negativas do *LexicoNRC* e o tratamento de negações sem contexto.

arquitetura	4 cl. (%)	6 cl. (%)
CS1	**57,27**	**40,79**
CS2	55,82	40,44
CS3	55,75	40,10
CS4	55,75	40,10

Tabela 3.18: Desempenho das diferentes arquiteturas com o melhor conjunto de *features*.

3.7.2.1 Avaliação das diferentes arquiteturas

Na avaliação das diferentes arquiteturas para o classificador *RL*, a Tabela 3.18 mostra os resultados das execuções com a configuração de *features* da execução 15 da Tabela 3.17. Aqui também a arquitetura CS1 obteve o melhor resultado em ambas as classificações. Em quatro classes a diferença do melhor resultado para as restantes três arquiteturas é significativa, de mais de 1% para todas. Em seis classes, a diferença da melhor arquitetura para as restantes é menor, com resultados entre os 40% e os 41% de *accuracy* para todas.

Para estas execuções, as *accuracies* são de 76,18% para o classificador binário <P,O>, de 78,60% para o <N,O>, de 80,33% para o <P,P+>, de 84,00% para o <N,N+>, de 2% para o <NONE,O> e de 85,60% para o <NEU,O>.

3.7.3 Naïve Bayes

Na avaliação do desempenho das *features* do classificador *NB*, a Tabela 3.19 mostra os resultados. O ponto de partida foi a utilização de unigramas com um valor de frequência de três (execução 1). A utilização do *smileDic*, trouxe um ganho em ambas as classificações, mais considerável em quatro classes (execução 2). As *features* de pontuação, provocam o maior aumento já visto nas experiências, com quase 7% no resultado em quatro classes e mais de 3% em seis classes (execução 3). A informação do autor, provoca um resultado oposto nas classificações, com um aumento de mais de 3% em quatro classes e uma descida de cerca de 0,50% em seis classes. Os tópicos, apesar de ligeiro, provocam um aumento do resultado em ambas as classificações (execução 5). A *feature TT*, algo surpreendentemente, provoca cerca de 2% de aumento em quatro classes e também mais de 1% em seis classes (execução 6). A utilização de *RT<2*, além de provocar uma descida de cerca de 200 *features*, traz um pequeno aumento somente na classificação de quatro classes (execução 7). O tratamento de negações trouxe uma descida em ambas as classificações (execução 8). As *features HTW* não alteram o resultado em quatro classes mas provocam quase 1% de aumento em seis classes (execução 9). Igualmente, as *features IP* também não alteram a classificação em quatro classes e trazem um ligeiro aumento em seis classes (execução 10). As *features IC* não se mostram úteis, com

execução	*feature*	# *features*	4 cl. (%)	6 cl. (%)
1	unigramas	3 770	37,60	29,09
2	(1) + smileDic	3 778	38,16	29,29
3	(2) + pontuação	3 782	45,15	32,55
4	(3) + IA	3 909	48,68	31,99
5	(4) + tópicos	3 919	48,89	32,55
6	(5) + TT	3 920	51,04	34,00
7	(6) + RT<2	3 731	51,11	34,00
8	(7) + negações	3 671	50,90	33,93
9	(7) + HTW	3 415	51,11	34,83
10	(9) + IP	3 491	51,11	35,04
11	(10) + IC	3 496	50,90	34,21
12	(10) + lemas	3 081	51,87	34,90
13	(12) + lexicoNRC	4 657	52,56	35,60
14	(13) + lexicoFS	4 547	52,42	35,46
15	(13) + numPosAndNegNRC	4 659	53,81	37,12
16	(15) + numPosAndNegFS	4 661	54,36	38,09
17	(16) + anotações POS	6 790	54,92	38,02
18	(17) + negImp	6 790	55,40	**38,50**
19	(18) + bigramas (f=3)	9 354	**55,61**	38,30

Tabela 3.19: Desempenho das *features* com a arquitetura CS1.

a diminuição da classificação em ambas as classes (execução 11). A lematização provoca um redução de cerca de 400 *features* com um aumento do resultado na classificação em quatro classes e uma ligeira descida do resultado em seis classes (execução 12). No desempenho das *features* lexicais, a primeira a ser avaliada foi o LexicoNRC, com um aumento na classificação em ambas as classes (execução 13). Contrariamente, o LexicoFS não trouxe ganhos em nenhuma das classificações (execução 14). O número de palavras positivas e negativas dos dois léxicos provoca aumentos nos resultados (execuções 15 e 16). Em particular as *features numPosAndNegNRC* com um aumento de mais de 1% em ambas as classificações. A utilização de anotações POS, provoca algum aumento de complexidade, com mais de 2 000 *features,* mas com ganhos em ambas as classificações, em particular em quatro classes (execução 17). O tratamento das negações com contexto neste ponto, traz ganhos em ambas as classificações (execução 18). Por fim, a utilização de bigramas com o valor de frequência igual a três, provoca novo aumento considerável de complexidade devido ao aumento de 3 000 *features* mas perfaz o melhor resultado na classificação de quatro classes (execução 19).

arquitetura	4 cl. (%)	6 cl. (%)
CS1	**55,61**	38,30
CS2	55,40	**38,64**
CS3	48,27	34,21
CS4	48,20	34,14

Tabela 3.20: Desempenho das diferentes arquiteturas com o melhor conjunto de *features*.

3.7.3.1 Avaliação das diferentes arquiteturas

Para a melhor execução da Tabela 3.19, são mostrados os resultados obtidos com as diferentes arquiteturas na Tabela 3.20. Na classificação de quatro classes, o melhor resultado é obtido pela arquitetura CS1, com um resultado muito próximo pela arquitetura CS2. As arquiteturas CS3 e CS4 ficam a mais de 7% do melhor resultado. Em relação à classificação em seis classes, o melhor resultado é obtido pela arquitetura CS2, também com um resultado muito próximo pela arquitetura CS1. Novamente as arquiteturas CS3 e CS4 com os piores resultados.

Para estas execuções, os classificadores binários têm uma *accuracy* de 71,40% para o <P,O>, de 75,55% para o <N,O>, de 77,22% para o <P,P+>, de 77,70% para o <N,N+>, de 78,81% para o <NONE,O> e de 77,84% para o <NEU,O>.

3.8 Resultados com o subconjunto de teste

Na Tabela 3.21, encontram-se os resultados com o conjunto de dados de teste, com cerca de 60800 *tweets*, das 10 melhores execuções. O classificador *SMO* obteve os dez melhores resultados quer na classificação de quatro classes quer de seis classes. É importante destacar que o melhor resultado em ambas as classificações foi obtido com a mesma arquitetura, a CS2, e que, os três melhores resultados também são obtidos com ela. Os resultados restantes são obtidos com a arquitetura CS2 e um único com a arquitetura CS1. O melhor resultado obtido pelo classificador *RL* em quatro classes foi de 62,10% com a arquitetura CS2 e em seis classes foi de 52,95% também com a arquitetura CS2. Em relação ao classificador *NB*, não obteve nenhum resultado acima dos 60%, sendo que o melhor na classificação de quatro classes foi de 58,83% e de seis classes de 50,42% ambos com a arquitetura CS2.

Os resultados da *accuracy* dos classificadores binários da melhor execução com 63,26% em quatro classes é de 78,42% para o <P,O>, de 80,97% para o <N,O>, de 72,21% para o <P,P+>, de 88,39% para o <N,N+>, de 71,97% para o <NONE,O> e de 97,85% para o <NEU,O>. Para seis classes, com um valor de frequência igual a 20 e um resultado de 54,42% é de 78,51% para o <P,O>, de 80,92% para o <N,O>, de 70,97% para o <P,P+>, de 88,70%

arquitetura	alg.	n2f	# *features*	4 cl. (%)	6 cl. (%)
SC2	SMO	16	358	**63,26**	54,30
SC2	SMO	11	581	63,24	53,58
SC2	SMO	17	317	63,18	**54,40**
SC4	SMO	16	358	63,15	54,05
SC2	SMO	15	375	63,10	53,90
SC2	SMO	18	293	63,06	54,29
SC4	SMO	17	317	63,06	54,17
SC4	SMO	11	581	62,99	53,14
SC1	SMO	16	358	62,98	53,88
SC3	SMO	16	358	62,98	53,88

Tabela 3.21: Os 10 melhores resultados com o subconjunto de teste.

para o <N,N+>, de 71,22% para o <NONE,O> e de 97,85% para o <NEU,O>.

3.9 Comparação

Para os conjuntos de *features* utilizado pela melhor execução do classificador *RL* e do *SMO* da Tabela 3.21, a sua execução noutros classificadores, verificaram-se os resultados da Tabela 3.22, onde são comparadas todas as arquiteturas. O classificador *SMO* tem os melhores resultados com o modelo menos complexo, onde curiosamente todas as arquiteturas obtêm resultado próximos quer na fase de treino quer na fase de teste. Com a utilização de mais de 9 000 features, o *NB* tem o melhor resultado em seis classes enquanto que em quatro classes o classificador *SMO* tem novamente o melhor resultado, ambos com a arquitetura CS2. Os tempos de treino dos modelos com este número de features para o classificador *RL* são elevados pelo que existem resultados com o subconjunto de teste.

Um dos fatores que tornam esta tarefa de classificação de *tweets* mais complexa, são a identificação de *tweets* com sentimento neutro. Outro dos fatores que pode ter influência, se os tweets com menos caracteres são mais difíceis de classificar corretamente. Em mensagens tão curtas como os *tweets*, classificar corretamente esta classe é complexo, é necessário que os classificadores binários <P,O> e <N,O> identifiquem sentimento positivo e negativo respetivamente. Outra das possíveis explicações para a fraca *accuracy* na classificação de tweets neutros pode estar no pequeno conjunto de *tweets* neutro para treino do modelo (8,45%). As arquiteturas CS3 e CS4 exploravam uma melhoria da *accuracy* através da melhoria na classificação dos neutros. Foi analisado o comprimento médio dos *tweets* corretamente e incorretamente classificados, para perceber se existe alguma relação direta entre o tamanho do *tweet* e a complexidade da sua classificação. Foi verificado, na execução sobre o subconjunto de testes, uma melhor

				treino		teste	
classificador	# features	arq.	alg.	4 cl. (%)	6cl. (%)	4 cl. (%)	6cl. (%)
SMO	347	CS1	SMO	56,93	41,00	62,50	53,02
			RL	56,65	39,82	60,71	50,32
			NB	55,40	39,06	55,33	45,30
		CS2	SMO	56,44	40,79	62,83	53,57
			RL	56,09	39,96	61,29	51,29
			NB	54,16	40,17	55,00	45,61
		CS3	SMO	56,93	41,00	62,50	53,02
			RL	55,68	39,54	57,92	48,12
			NB	44,32	33,66	42,14	36,76
		CS4	SMO	56,86	40,93	62,71	53,23
			RL	55,68	39,54	58,27	48,47
			NB	44,18	33,52	42,25	36,86
RL	786	CS1	SMO	55,82	40,44	62,17	52,64
			RL	**57,27**	40,79	60,74	50,42
			NB	53,74	37,67	56,21	47,75
		CS2	SMO	55,47	40,51	62,39	53,09
			RL	55,82	40,44	61,29	51,39
			NB	53,60	39,06	57,68	49,61
		CS3	SMO	55,82	40,44	62,16	52,63
			RL	55,75	40,10	57,95	48,22
			NB	46,33	32,96	49,23	42,80
		CS4	SMO	55,68	40,30	62,34	52,81
			RL	55,75	40,10	58,26	48,54
			NB	46,33	32,96	49,25	42,82
NB	9 354	CS1	SMO	52,15	37,67	59,14	47,76
			RL	45,91	30,12	NA	NA
			NB	55,61	38,30	57,74	48,66
		CS2	SMO	50,69	37,26	59,80	49,00
			RL	37,12	27,70	NA	NA
			NB	55,40	38,64	58,80	50,35
		CS3	SMO	50,76	37,60	55,71	45,19
			RL	38,57	27,56	NA	NA
			NB	48,27	34,21	47,70	41,51
		CS4	SMO	50,69	37,53	56,49	45,97
			RL	39,13	28,12	NA	NA
			NB	48,20	34,14	47,77	41,59

Tabela 3.22: Comparação do desempenho das *features* com os subconjuntos de treino e teste.

	Corretos	Incorretos	Total
Média de caracteres	102	111	105

Tabela 3.23: Média de caracteres dos *tweets* corretos e incorretos.

média na *accuracy* de *tweets* mais curtos, cerca de 9 caracteres, em relação ao *tweets* que foram incorretamente classificados, como se vê na Tabela (3.23). Demonstra de alguma maneira que apesar da maior complexidade na tarefa de classificação em textos cada vez mais curto, existe um certo limiar em que isso não se reflete.

3.10 Conclusões

Relativamente aos resultados reportados a *accuracy* mais elevada é obtida pelo classificador *SMO* com 53,57% com a arquitetura CS2. Na avaliação do desempenho das *features* do classificador *NB*, as que se revelaram mais importantes foram as *features* de pontuação, o tamanho do *tweet*, a informação do autor e o número de palavras positivas e negativas do Léxico NRC. No classificador *RL*, a *feature* mais importante foi a utilização de lemas, mas também a pontuação e o número de palavras positivas e negativas. Por último, o classificador *SMO*, destacam-se a utilização do dicionário de emoções e das *features* construídas com base na intensidade das palavras e, à semelhança do verificado com o classificador *RL*, também a utilização de lemas foi importante.

Numa análise às diferentes arquiteturas na fase de treino, a arquitetura CS1 considerada a mais natural, obteve o melhor resultado em igualdade com a arquitetura CS3. Evidencia-se que todas as arquiteturas obtiveram taxas de acerto acima dos 40%. Já na fase de treino, o melhor resultado é obtido pela arquitetura CS2 com 53,57% e à semelhança do verificado na fase de treino, as restantes arquiteturas obtiveram resultados próximos, todos acima dos 53%.

Na classificação com o subconjunto de testes, os 10 melhores resultados são obtidos pelo classificador *SMO*. O melhor resultado obtido pelo classificador *SMO* é de 63,26% enquanto que o classificador *NB* não tem nenhum resultado acima dos 60%. A arquitetura CS2 destaca-se com os três melhores resultados na classificação de quatro e seis classes.

Comparando este trabalho com outros, das últimas edições do TASS, a melhor execução das experiências com o subconjunto de testes para a classificação com quatro e seis classes, comparativamente a 2013, perante 17 submissões, entre os 16,70% e 65,30%, ficaria em nono para seis classes com 54,42%, e entre os 35,10% e 71,10% ficaria em quinto para quatro classes com 63,01%. Em relação à edição de 2014, na classificação de quatro classes ficaria com 63,26% em 26º de um total de 50 submissões, com resultados entre os 33,03% e 70,96%.

Análise de Sentimento de Dados de Marketing

4

Este capítulo descreve as experiências realizadas com dados na língua portuguesa. É inicialmente apresentado o corpus utilizado e realizadas algumas estatísticas sobre ele. É depois apresentada a abordagem ao processamento de dados de marketing com as arquiteturas em experimentação, os resultados da classificação em cascata e por fim são discutidas as principais conclusões.

4.1 Corpus Marketing

O conjunto de dados de Marketing consiste num total de 1 893 textos, da página de uma marca na rede social Facebook em que o conteúdo dos textos está relacionado com a interação entre um representante da marca e os visitantes da página. Os textos foram anotados por humanos, classificando com três valores possíveis, sentimento positivo (P), sentimento negativo (N) e sem sentimento (NONE). Mais de metade dos textos têm sentimento positivo e os restantes negativos e sem sentimento estão em número semelhante, como mostra a Tabela 4.1. No conjunto de textos verificaram-se a presença de muitos *emoticons,* como ":)", ";) ou "(y)", sendo que no total foram encontrados 376. Algumas mensagens só continham mesmo o *emoticon* "(y)", que significa o polegar levantado indicando algo de positivo, que foram naturalmente classificadas como tendo sentimento positivo. Alguns textos, como o seguinte:

> *"Olá Bom dia :) Fiz há mais de uma semana atrás o registo para a vossa oferta da... . Desde dessa altura não recebi mais nenhum email..."*

apesar de conter o *emoticon* ":)" indicando sentimento positivo nos cumprimentos iniciais feitos pelo autor no texto, demonstra descontentamento no restante conteúdo. Com classificação automática por *emoticons* positivos ou negativos este era um texto que seria mal classificado.Este conjunto de dados foi subdividido em dois, num subconjunto com cerca de 80% dos textos para treino e dos restantes 20% para teste. Dos 80% de textos para treino, foi ainda criado outro subconjunto com cerca de 20% dos textos para desenvolvimento, como mostra a Tabela 4.1.

	#	# P	% P	# N	% N	# NONE	% NONE
# treino	1 200	703	58,58	258	21,50	239	19,92
# desenvolvimento	315	176	55,87	59	18,73	80	25,40
# teste	378	236	62,43	82	21,69	60	15,87
# total	1 893	1 115	58,90	399	21,08	379	20,02

Tabela 4.1: Classes dos textos do subconjunto de dados de treino, desenvolvimento e de teste.

Número de *tokens*	19 548
Número de *stop words*	684
Número de palavras capitalizadas	1 924
Número de *long words*	79
Número de endereços web	58
Número de negações	208
Número de *emoticons*	249

Tabela 4.2: Estatísticas sobre os dados utilizados nas experiências.

Do cerca de 80% dos textos do subconjunto de treino, a Tabela 4.2 mostra algumas estatísticas sobre os dados. De um total de 1 893 mensagens, existem 19 548 *tokens*, o que perfaz uma média de cerca de 10 *tokens* por mensagem. Do total, cerca de 10% dos *tokens* são palavras capitalizadas e cerca de 3% são *stop words*. O número de negações e de *emoticons* é um pouco superior a 1%. As negações são contabilizadas com a ocorrência de *tokens* iguais a *não*, *nunca* e *nao*. Ao dicionário de *emoticons* existente foram adicionados os *emoticons (y)* e *(n)*. Em relação à ocorrência quer de *long words* quer de endereços web, o seu peso é ainda menos significativo.

No pré-processamento dos dados, a remoção de *stop words*, é feita com base num universo de 220 palavras [1]. Em relação à extração de *features*, só serão utilizadas *features* do único léxico na língua portuguesa. Quer as anotações POS, quer o processo de lematização também não vai ser possível explorar.

4.2 Extrator de *Features*

Aqui o processo de extração de experimentação de features é semelhante ao descrito na Secção 3.3com as seguintes exceções:

- Não existe combinação de léxicos, só existe um léxico de polaridade para a língua portuguesa;

[1] http://snowball.tartarus.org/algorithms/portuguese/stop.txt

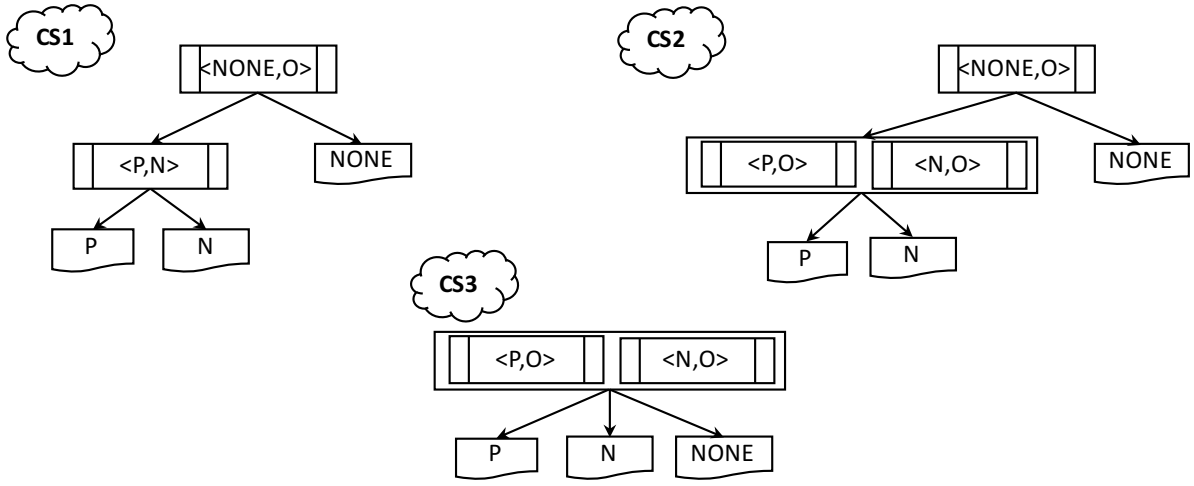

Figura 4.1: Arquiteturas de classificação em cascata para processamento de blogues (O - representa o conjunto das classes restantes).

- Não existe informação relativa ao autor;
- Não existe a noção de tópico;
- Não existe lematização.

4.3 Processamento de dados de marketing

Neste cenário pretende-se classificar o sentimento associado às interações entre um representante de uma dada marca e os visitantes da página do Facebook dessa marca, considerando três possíveis classes: positivo (P), negativo (N) e sem sentimento (NONE). Este problema é claramente uma simplificação do problema anterior, uma vez que deixa de existir a classe NEU, bem como as classes N+ e P+. Este problema pode ser resolvido usando dois ou mais classificadores binários, tal como se pode observar na Figura 4.1. A arquitetura CS1, composta por dois classificadores binários, surge de forma bastante natural no sentido em que o primeiro nível filtra os documentos sem sentimento, passando para o segundo nível apenas os documentos com sentimento associado. Além desta, foram exploradas as duas arquiteturas adicionais CS2 e CS3, que usam três classificadores distintos.

4.4 Resultados com classificadores em cascata

Para estas experiências não são apresentados os resultados obtidos com os classificadores binários individualmente. O desempenho da experimentação de features na *accuracy* é avaliado

execução	*features*	# *features*	CS1 (%)
1	unigramas	304	79,05
2	(1) + smileDic	307	80,32
3	(2) + negações	302	78,41
4	(2) + RT<2	259	80,00
5	(2) + pontuação	311	**80,63**
6	(5) + TT	312	80,32
7	(5) + IP	321	80,00
8	(5) + HTW	305	79,05
9	(5) + IC	316	79,68
10	(5) + numPosAndNegNRC	313	80,63
11	(5) + LexicoNRC	355	79,37
12	(5) + negImp	306	79,68
13	(5) + bigramas	346	80,63

Tabela 4.3: Desempenho das *features* com a arquitetura CS1 utilizando o *SMO*.

com a arquitetura CS1. As arquiteturas CS2 e CS3 só são exploradas com os classificadores Naïve Bayes e Regressão Logística com a utilização do valor da probabilidade para classificar textos em que em simultâneo o classificador binário <P,O> tem o valor P e o classificador binário <N,O> tem o valor N.

4.4.1 Sequential Minimal Optimization

A Tabela 4.3 mostra as várias execuções até ser encontrado a melhor configuração de *features*. Apenas o dicionário de *emoticons* e a utilização de pontuação contribuíram para o melhor resultado com uma *accuracy* de 80,63%.

Os resultados dos classificadores binários com a melhor configuração de *features* é de 92,38% para o <NONE,O>, de 83,81% para o <P,O> e de 84,44% para o <N,O> e de 79,37% para o <P,N>.

4.4.2 Regressão Logística

A Tabela 4.4 mostra para o classificador Regressão Logística, as várias execuções até ser encontrada a melhor configuração de *features*. O ponto de partida é a utilização de unigramas com um valor de frequência igual a 10, com uma *accuracy* de 72,38%. Para o melhor resultado, com uma *accuracy* de 76,19% contribuíram a utilização de RT<2, as *features* de informação dos caracteres e o número de palavras positivas e negativas do Léxico NRC.

execução	*features*	# *features*	CS1 (%)
1	unigramas	210	72,38
2	(1) + smileDic	213	71,43
3	(1) + negações	205	72,06
4	(1) + RT<2	170	72,70
5	(4) + pontuação	174	72,06
6	(4) + TT	171	71,11
7	(4) + IP	175	72,06
8	(4) + HTW	171	71,75
9	(4) + IC	175	74,29
10	(9) + numPosAndNegNRC	177	75,24
11	(10) + LexicoNRC	208	**76,19**
12	(11) + negImp	198	75,87
13	(11) + bigramas (f=10)	225	73,65

Tabela 4.4: Desempenho das *features* com a arquitetura CS1 utilizando Regressão Logística.

Para este classificador os resultados dos classificadores binários com a melhor configuração de *features* é de 87,94% para o <NONE,O>, de 78,41% para o <P,O> e de 83,17% para o <N,O> e de 65,40% para o <P,N>.

4.4.3 Naïve Bayes

Na experimentação de *features* com o classificador *NB* numa primeira execução com unigramas e com um valor de frequência de seis a *accuracy* é de 69,52%. O melhor resultado foi conseguido utilizando apenas o dicionário de *emoticons* e o tamanho do texto com uma *accuracy* de 73,33%. A Tabela 4.5 mostra os resultados com as diferentes *features*.

Os resultados dos classificadores binários com a melhor configuração de *features* é de 92,38% para o <NONE,O>, de 76,19% para o <P,O> e de 70,48% para o <N,O> e de 58,10% para o <P,N>.

4.5 Comparação

Na comparação dos diferentes classificadores, a Tabela 4.6 mostra a comparação dos resultados obtidos com o classificador *SMO*, *RL* e *NB*. Existe uma concordância entre os resultados com o subconjunto de treino relativamente ao subconjunto de teste. O classificador *SMO* com a arquitetura CS1 tem a melhor *accuracy* com 83,60%. Os resultados do classificador *NB* são semelhantes nas três arquiteturas, particularmente na fase de treino. A arquitetura CS2 apresenta

execução	*features*	# *features*	CS1 (%)
1	unigramas	351	69,52
2	(1) + smileDic	354	73,02
3	(2) + negações	348	72,70
4	(2) + RT<2	305	73,02
5	(2) + pontuação	358	72,06
6	(2) + TT	355	**73,33**
7	(6) + IP	373	73,02
8	(6) + HTW	349	73,02
9	(6) + IC	360	73,33
10	(6) + numPosAndNegNRC	357	72,70
11	(6) + LexicoNRC	404	72,38
12	(6) + negImp	349	73,02
13	(6) + bigramas	456	72,06

Tabela 4.5: Desempenho das *features* com a arquitetura CS1 utilizando o *NB*.

um resultado abaixo dos 60%, que pode ser explicado pelo baixo resultado dos classificadores binários <P,O> e <N,O> com uma *accuracy* de 76,19% e 68,57% respetivamente. Comparando os três classificadores, o *RL* tem os piores resultados, enquanto que o *NB* consegue resultados acima dos 80%, relativamente perto dos resultados obtidos pelo *SMO* com número semelhante de *features*. Curiosamente, os resultados do classificador *NB* nas arquiteturas CS2 e CS3 são iguais, não por uma simples coincidência na *accuracy* mas porque todos as mensagens foram classificadas com a mesma classe.

Pela observação da Figura 4.2, é possível verificar que na fase de treino as taxas de acerto são semelhantes para o classificador *NB* com as três arquiteturas em todas as configurações de *features*, tendo obtido o melhor resultado com as arquiteturas CS2 e CS3. Para o classificador *SMO*, as arquiteturas CS2 e CS3 ficam a cerca de 2% e 3% respetivamente do melhor resultado, que é superior a 80% obtido com a arquitetura CS1. Quanto ao *RL,* destacam-se com os piores resultados as execuções com a configuração de *features* do NB, abaixo dos 70%.

Na fase de teste, a Figura 4.3 permite ver relativamente ao *NB* a tendência verificada na fase de treino, com resultados próximos com as configurações de *features* do *SMO* e *NB*. No *RL,* é possível verificar que a arquitetura CS2 tem os piores resultados com as configurações de *features* dos três classificadores e resultados próximos nas arquiteturas CS1 e CS3. Na análise do *SMO,* os resultados obtidos com as configurações de *features* do *RL* e *NB* na fase de treino, são encurtadas na fase de teste para menos de 1% em ambas as configurações relativamente ao melhor resultado de 83,60%, obtido com a configuração de *features* do *SMO*.

conj. *features*	alg.	treino			teste		
		CS1 (%)	CS2 (%)	CS3 (%)	CS1 (%)	CS2 (%)	CS3 (%)
SMO	SMO	80,63			83,60		
	RL	71,43	64,44	67,94	70,11	65,08	71,69
	NB	72,06	75,56	75,56	81,48	80,95	80,95
RL	SMO	77,46			83,33		
	RL	76,19	73,33	75,87	73,28	72,49	76,98
	NB	70,16	71,75	71,75	76,72	76,46	76,46
NB	SMO	78,41			82,80		
	RL	68,89	66,35	69,52	70,37	67,72	71,69
	NB	73,33	74,29	74,29	80,95	80,95	80,95

Tabela 4.6: Comparação do desempenho das *features* com os subconjuntos de treino e teste.

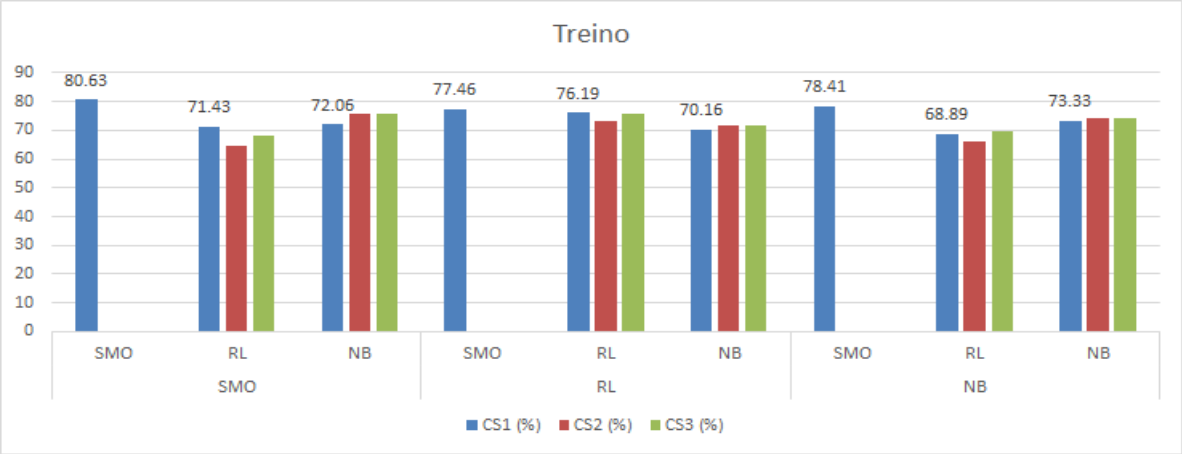

Figura 4.2: Comparação do desempenho das *features* com o subconjunto de treino.

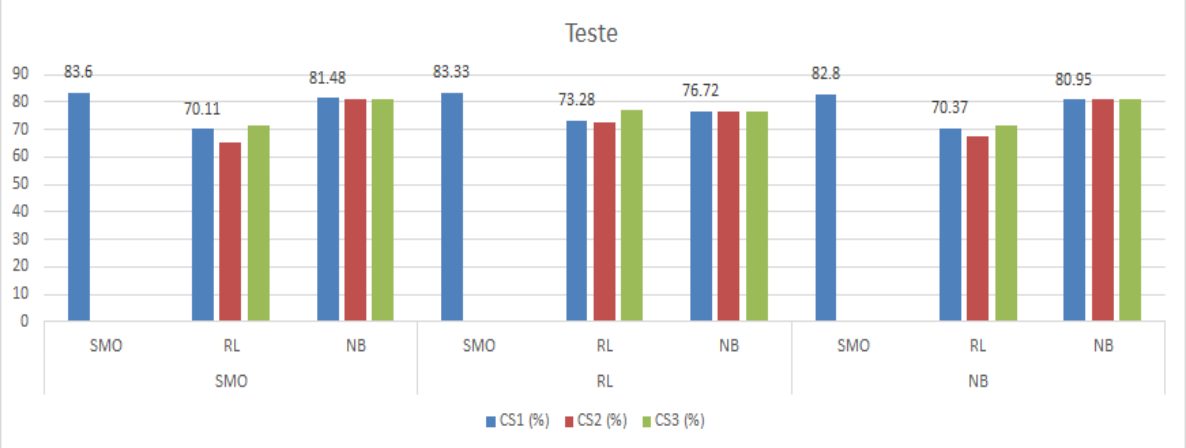

Figura 4.3: Comparação do desempenho das *features* com o subconjunto de teste.

4.6 Conclusões

Das experiências de classificação realizadas com os três classificadores, o *SMO* e *NB* têm os melhores resultados, com taxas de acerto acima dos 80% com o subconjunto de teste. O classificador *SMO* demonstra ser o mais resistente na variação do número de *features* e consegue uma *accuracy* superior a 83% com diferentes conjuntos de features.

Na comparação das arquiteturas propostas, na fase de treino a arquitetura CS1 têm um resultado distante das restantes duas arquiteturas. Na fase de teste, é mantida a força da arquitetura CS1, mas com resultados ligeiramente mais próximos das restantes arquiteturas, que obtêm taxas de acerto acima dos 80%. Na experimentação do desempenho das *features* no melhoramento da *accuracy*, a única *feature* comum que trouxe ganhos a mais do que um classificador foi o dicionário de *emoticons.* As *features* lexicais, ao contrário do verificado na classificação de *tweets*, apenas o Léxico NRC no classificador *RL* provocaram um aumento na *accuracy*, podendo ser explicado pela qualidade do léxico ou pela dimensão do conjunto de dados. Numa análise individual, no classificador *SMO*, também a pontuação foi utilizada como *feature*. No classificador *RL*, foram importantes as features que exploram a forma da palavra. Por fim, o classificador *NB*, teve também a utilização da feature com o tamanho do texto.

5 Conclusões

A tarefa de análise de sentimento é complexa, nem sempre os humanos estão em concordância relativamente a um sentimento, quer seja por fatores culturais, diferentes contextos ou simplesmente opiniões diferentes. Esta tarefa, com textos retirados das redes sociais torna-se ainda mais complexa, devido aos fenómenos particulares como a ocorrência de erros ortográficos, a existência de muitos endereços web, e no caso particular do Twitter, os *hashtags*, *targets* e *re-tweets* e a limitação de 140 caracteres.

Foram comparadas diferentes abordagens de classificação de sentimento em *tweets*, com a exploração de diversos classificadores de aprendizagem automática. Na fase de treino dos classificadores, o *RL* e o *SMO* tiveram resultados próximos na classificação de seis classes, com melhor resultado do *RL* e com o *SMO* a cerca cinco décimas. Já na classificação em quatro classes, o *SMO* tem o melhor resultado, com o *RL* igualmente a cerca de cinco décimas do melhor resultado. O classificador *NB* obteve os piores resultados, mais distantes na classificação de seis classes. Comparando a complexidade do treino dos modelos, o *RL* permite bons resultados com um menor número de *features*. Na fase de teste, os melhores resultados em ambas as classificações foram dominados pelo classificador *SMO*.

Na experimentação das diferentes cascatas propostas, a principal conclusão, é que as taxas de acerto obtidas, quer na fase de treino, quer na fase de teste são muito próximas. As arquiteturas CS1 e CS3 têm os melhores resultados na fase de treino mas na fase de testes a arquitetura CS2 obtém o melhor resultado.

Na avaliação das *features* que provocaram um aumento dos resultados de classificação, as que tiveram um ganho considerável quer no classificador *RL*, quer no *SMO*, foram o dicionário de emoções, a pontuação, a informação do autor, a utilização de tópicos, a lematização, o número de palavras positivas e negativas dos dois léxicos e o tratamento de negações. Individualmente, para o classificador *RL* a *feature* mais importante foi a utilização do número de palavras positivas e negativas do Léxico de NRC. Já para o classificador *SMO*, a mais importante foi o dicionário de emoções. Ao contrário do esperado, a utilização de *features* que exploram os fenómenos dos microblogues, não trouxe ganhos em nenhum dos classificadores. Em relação ao classificador *NB*, as *features* mais relevantes foram a utilização de pontuação, o tamanho dos

tweets e o número de palavras positivas e negativas do léxico NRC.

São também realizadas experiências noutras aplicações, na classificação de textos de língua portuguesa em três classes, positivo, negativo ou sem sentimento. Na fase de teste, os classificadores *SMO* e *NB* têm taxas de acerto acima dos 80%, com destaque para o melhor resultado pela arquitetura CS1. As restantes arquiteturas ficam algo distantes, principalmente na fase de treino. Na experimentação de *features*, em cada classificador foram poucas as que se revelaram importantes, com destaque para o dicionário de *emoticons* com impacto positivo quer no classificador *SMO* quer no classificador *NB*.

Os resultados apresentados nesta tese, para a classificação de *tweets* a quatro e seis classes, com o classificador *SMO* encontram-se na média dos resultados divulgados das últimas edições do TASS (Villena-Román et al., 2013; Rosenthal et al., 2014), pelo que se considera ser um sistema de classificação útil.

Esta tese tem alguns contributos, um deles consiste nas análises realizadas com os diferentes classificadores de aprendizagem automática e outro é a configuração de *features* utilizadas. Na literatura não existe nenhum modelo perfeito de classificação, há autores com melhores resultados apenas com unigramas, outros com unigramas e bigramas, e por exemplo nem sempre a utilização de anotações POS traz melhorias (Pang et al., 2002), nem de léxicos de polaridade (Batista and Ribeiro, 2013). Outro dos contributos é a classificação de dados anotados na língua portuguesa.

O trabalho futuro envolverá uma análise linguística mais cuidada, o enriquecimento dos léxicos de polaridade (português e espanhol) e a sua combinação, parece ser outro caminho com margem para muita evolução, conforme demonstra o trabalho de Zhu et al. (2014). Será também interessante explorar nas arquiteturas propostas, a combinação dos diferentes classificadores nos classificadores binários com melhores resultados.

Bibliografia

Agarwal, A., Xie, B., Vovsha, I., Rambow, O., and Passonneau, R. (2011). Sentiment analysis of twitter data. In *Proceedings of the Workshop on Languages in Social Media*, LSM '11, pages 30–38, Stroudsburg, PA, USA. Association for Computational Linguistics.

Aue, A. and Gamon, M. (2005). Customizing sentiment classifiers to new domains: a case study. In *Submitted to RANLP-05, the International Conference on Recent Advances in Natural Language Processing*, Borovets, BG.

Barbosa, L. and Feng, J. (2010). Robust sentiment detection on twitter from biased and noisy data. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, pages 36–44, Stroudsburg, PA, USA. Association for Computational Linguistics.

Batista, F. and Ribeiro, R. (2013). Sentiment analysis and topic classification based on binary maximum entropy classifiers. *Procesamiento del Lenguaje Natural*, 50:77–84.

Bautin, M., Vijayarenu, L., and Skiena, S. (2008). International sentiment analysis for news and blogs. In *Proceedings of ICWSM*.

Boiy, E. and Moens, M.-F. (2009). A machine learning approach to sentiment analysis in multilingual web texts. *Inf. Retr.*, 12(5):526–558.

Cessie, L. S. and van Houwelingen, J. C. (1992). Ridge Estimators in Logistic Regression. *Applied Statistics*, 41(1):191–201.

Dave, K., Lawrence, S., and Pennock, D. M. (2003). Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In *WWW*, pages 519–528.

Davidov, D., Tsur, O., and Rappoport, A. (2010). Semi-supervised recognition of sarcastic sentences in twitter and amazon. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, CoNLL '10, pages 107–116, Stroudsburg, PA, USA. Association for Computational Linguistics.

Go, A., Bhayani, R., and Huang, L. (2009). Twitter sentiment classification using distant supervision. *Processing*, pages 1–6.

González-Ibáñez, R., Muresan, S., and Wacholder, N. (2011). Identifying sarcasm in twitter: A closer look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, HLT '11, pages 581–586, Stroudsburg, PA, USA. Association for Computational Linguistics.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18.

Jia, L., Yu, C., and Meng, W. (2009). The effect of negation on sentiment analysis and retrieval effectiveness. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, CIKM '09, pages 1827–1830, New York, NY, USA. ACM.

Joachims, T. (1999). Making large-scale support vector machine learning practical. In Schölkopf, B., Burges, C. J. C., and Smola, A. J., editors, *Advances in Kernel Methods*, chapter Making Large-scale Support Vector Machine Learning Practical, pages 169–184. MIT Press, Cambridge, MA, USA.

John, G. H. and Langley, P. (1995). Estimating continuous distributions in bayesian classifiers. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, UAI'95, pages 338–345, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Kim, S.-M. and Hovy, E. (2004). Determining the sentiment of opinions. In *Proceedings of the 20th International Conference on Computational Linguistics*, COLING '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Melville, P., Gryc, W., and Lawrence, R. D. (2009). Sentiment analysis of blogs by combining lexical knowledge with text classification. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 1275–1284, New York, NY, USA. ACM.

Mohammad, S. M. and Turney, P. D. (2010). Emotions evoked by common words and phrases: Using mechanical turk to create an emotion lexicon. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, CAAGET '10, pages 26–34, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mohammad, S. M. and Turney, P. D. (2013). Crowdsourcing a word-emotion association lexicon. 29(3):436–465.

Mullen, T. and Collier, N. (2004). Sentiment analysis using support vector machines with diverse information sources. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 412–418. Poster paper.

Nagy, A. and Stamberger, J. (2012). Crowd sentiment detection during disasters and crises. In Rothkrantz, L., Ristvej, J., and Franco, Z., editors, *Proceedings of the 9th International ISCRAM Conference*.

Pak, A. and Paroubek, P. (2010). Twitter as a corpus for sentiment analysis and opinion mining. In Chair), N. C. C., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Rosner, M., and Tapias, D., editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

Pang, B., Lee, L., and Vaithyanathan, S. (2002). Thumbs up?: Sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*, EMNLP '02, pages 79–86, Stroudsburg, PA, USA. Association for Computational Linguistics.

Perez-Rosas, V., Banea, C., and Mihalcea, R. (2012). Learning sentiment lexicons in spanish. In Chair), N. C. C., Choukri, K., Declerck, T., Doğan, M. U., Maegaard, B., Mariani, J., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

Pla, F. and Hurtado, L.-F. (2014). *Natural Language Processing and Information Systems: 19th International Conference on Applications of Natural Language to Information Systems, NLDB 2014, Montpellier, France, June 18-20, 2014. Proceedings*, chapter Sentiment Analysis in Twitter for Spanish, pages 208–213. Springer International Publishing, Cham.

Platt, J. C. (1999). Fast training of support vector machines using sequential minimal optimization. In Schölkopf, B., Burges, C. J. C., and Smola, A. J., editors, *Advances in Kernel Methods*, chapter Fast Training of Support Vector Machines Using Sequential Minimal Optimization, pages 185–208. MIT Press, Cambridge, MA, USA.

Read, J. (2005). Using emoticons to reduce dependency in machine learning techniques for sentiment classification. In *Proceedings of the ACL Student Research Workshop*, ACLstudent '05, pages 43–48, Stroudsburg, PA, USA. Association for Computational Linguistics.

Reese, S., Boleda, G., Cuadros, M., Padró, L., and Rigau, G. (2010). Wikicorpus: A word-sense disambiguated multilingual wikipedia corpus. In Chair), N. C. C., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Rosner, M., and Tapias, D., editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA). POS TAGGER (ES,EN).

Rosenthal, S., Ritter, A., Nakov, P., and Stoyanov, V. (2014). Semeval-2014 task 9: Sentiment analysis in twitter. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 73–80, Dublin, Ireland. Association for Computational Linguistics and Dublin City University.

Saif, H., He, Y., and Alani, H. (2012). Semantic sentiment analysis of twitter. In *Proceedings of the 11th International Conference on The Semantic Web - Volume Part I*, ISWC'12, pages 508–524, Berlin, Heidelberg. Springer-Verlag.

Sánchez-Mirabal, P. A., Ruano Torres, Y., Hernández Alvarado, S., Gutiérrez, Y., Montoyo, A., and Muñoz, R. (2014). Umcc_dlsi: Sentiment analysis in twitter using polirity lexicons and tweet similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 727–731, Dublin, Ireland. Association for Computational Linguistics and Dublin City University.

Somasundaran, S., Namata, G., Wiebe, J., and Getoor, L. (2009). Supervised and unsupervised methods in employing discourse relations for improving opinion polarity classification. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 170–179, Stroudsburg, PA, USA. Association for Computational Linguistics.

Turney, P. D. (2002). Thumbs up or thumbs down?: Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 417–424, Stroudsburg, PA, USA. Association for Computational Linguistics.

Vicente, I. S. and Saralegi, X. (2014). Looking for features for supervised tweet polarity classification. In *TASS 2014*, Girona.

Villena-Román, J., García-Morera, J., Cumbreras, M. Á. G., Martínez-Cámara, E., Martín-Valdivia, M. T., and López, L. A. U., editors (2015). *Proceedings of TASS 2015: Workshop on Sentiment Analysis at SEPLN co-located with 31st SEPLN Conference (SEPLN 2015), Alicante, Spain, September 15, 2015*, volume 1397 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Villena-Román, J., Lana-Serrano, S., Martínez-Cámara, E., and Cristóbal, J. C. G. (2013). TASS - workshop on sentiment analysis at SEPLN. *Procesamiento del Lenguaje Natural*, 50:37–44.

Vosoughi, S., Zhou, H., and roy, d. (2015). Enhanced twitter sentiment classification using contextual information. In *Proceedings of the 6th Workshop on Computational Approaches*

to Subjectivity, Sentiment and Social Media Analysis, pages 16–24, Lisboa, Portugal. Association for Computational Linguistics.

Yang, C., Lin, K. H.-Y., and Chen, H.-H. (2007). Emotion classification using web blog corpora. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence*, WI '07, pages 275–278, Washington, DC, USA. IEEE Computer Society.

Zhu, X., Kiritchenko, S., and Mohammad, S. (2014). Nrc-canada-2014: Recent improvements in the sentiment analysis of tweets. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 443–447, Dublin, Ireland. Association for Computational Linguistics and Dublin City University.