Text Mining – Análise de Sentimentos de *Tweets*

Gonçalo Alface, 69209
Mestrado em Engenharia Informática
ISCTE-IUL
gpaea@iscte-iul.pt

Bruno Coitos, 64647
Mestrado em Engenharia Informática
ISCTE-IUL
bmfcc@iscte-iul.pt

***Abstract*— Neste artigo abordamos o tema de Análise de Sentimentos presente em *tweets*. A nossa abordagem passa por fazer um tratamento dos *tweets* de uma forma especifica, passando pelo tratamento de hashtags, URL's, negação e palavras alongadas. Após este tratamento, aplicamos um léxico de sentimentos a cada *tweet*, testando a contribuição de *Stemming* e *Lemmatization* na classificação da polaridade de um *tweet*. Por último, demonstramos duas formas de abordar o tratamento da negação.**

Keywords—text mining, sentiment analysis, tweets

I. INTRODUCÃO

O tema de Análise de Sentimentos tem vindo cada vez mais a ser um ponto importante para a maioria das organizações, pois permite saber a opinião do público em relação a determinado assunto, podendo este ser relacionado com os seus produtos, serviços, ou até mesmo a organização em si. Uma das principais fontes de dados para este tipo de análise tem sido o *Twitter*. Atualmente a maioria das pessoas utiliza o *Twitter* para exprimir a sua opinião e os seus sentimentos através de *tweets*, fazendo com que o interesse no tema de Análise de Sentimentos em *tweets* seja cada vez maior.

A análise em *tweets* traz vários desafios visto que não há um cuidado em relação à gramática/escrita correta das frases. Para além disto os *tweets* têm características próprias tais como mencionar outros utilizadores, hashtags, acrónimos e alongar palavras, e por isso vamos abordar estes casos mais à frente no artigo.

II. ANÁLISE DA LITERATURA

O trabalho anteriormente realizado nesta área, indica-nos maneiras de lidar com a negação contida nas frases, tal como a utilidade de utilização de palavras adjacentes, como unigramas, bigramas, trigramas e tetragramas, em conjunto com o algoritmo *Naive Bayes* [1]. Na área de análise de sentimentos de *tweets*, podemos verificar a importância do tratamento do texto dos *tweets*, visto que estes, por vezes, não se apresentam da melhor forma para serem analisados, desde a incoerências gramaticais/sintáticas, a espaços em branco, caracteres especiais, *stop-words*, *emoticons*, *hashtags*, abreviações, entre outros [2]. No tratamento dos *tweets* conseguimos verificar a importância de tratar a negação no texto, causando grandes melhorias dos resultados [3].

Conseguimos concluir também, que os algoritmos mais utilizados para abordar este tipo de problemas de classificação são, o algoritmo de SVM (*Support Vector Machine*) e o *Naive Bayes*, e apesar do algoritmo de SVM demorar um pouco de mais tempo a processar, geralmente apresenta melhores resultados do que o *Naives Bayes* [7].

III. PREPARAÇÃO DOS DADOS E *BASELINE*

O conjunto de dados que foi escolhido para abordar este tema foi *"Tweets_EN_sentiment.json*". Este conjunto de dados era caracterizado por cada elemento do *JSON* ser um tweet, composto por vários elementos, onde nos concentramos, somente em dois, no texto do tweet e na sua classificação, ou seja, se era um tweet de classificação positiva ou de classificação negativa.

Este conjunto de dados foi dividido em duas partes, a primeira, cerca de 80% dos dados, será usada para o treino dos modelos, enquanto que 20% dos dados, cerca de 10.000 tweets, vão ser usados para testar os modelos.

A. Baseline

Para a criação de uma *Baseline* usámos a biblioteca *TextBlob* para verificar a polaridade de cada tweet. Esta biblioteca foi aplicada sem qualquer tipo de tratamento dos dados, para se perceber melhor o impacto do tratamento.

Nestas condições obtivemos os seguintes resultados:

- *True positives*: 4599
- *True negatives*: 592
- *False positives*: 533
- *False negatives*: 4261

Com esta informação percebe-se que houve uma correta classificação de 5191 *tweets* e 4794 tweets com uma classificação incorreta. Estas classificações foram obtidas da seguinte forma: se a palavra for positiva, adicionamos o seu número de ocorrências a uma variável positiva, caso contrário, adicionamos a uma variável negativa. Após a classificação de cada *tweet*, se o número de palavras positivas for superior ao número de palavras negativas, classificamos o *tweet* como positivo, caso contrário terá uma classificação negativa. No final, se o *tweet* tiver uma classificação de base positiva e o resultado obtido for positivo, temos então um resultado *true positive* (TP), já se o *tweet* tiver uma classificação de base negativa e o resultado obtido for negativo, então temos um resultado *true negative* (TN). Caso a classificação do *tweet* for positivo, mas, no entanto, o resultado é negativo, deparamo-nos com um *false positive* (FP), e por último, se o *tweet* for negativo, e o resultado obtido for positivo, temos um *false negative* (FN) [2]. Estes resultados são importantes para se conseguir calcular as métricas de performance, nomeadamente, *Accuracy, Recall, Precision* e *F1-measure.*

Para o cálculo de cada uma destas métricas foram usadas as seguintes formulas:

a) *Accuracy:* (TP+TN)/(TP+TN+FP+FN), a

accuracy permite demonstrar o rácio de previsões corretas pelo total de *tweets* testados;

b) *Precision:* TP/(TP+FP), representando o rácio de *tweets* positivos, corretamente previstos, para o total positivos previstos;

c) *Recall:* TP/(TP+FN), que revela o rácio de *tweets* positivos corretamente previstos, pelo número total número de positivos verdadeiros mais o número de negativos falsos;

d) *F1:* (2**Precision***Recall*)/(*Precision*+*Recall*), demonstra o peso da *Precision* e do *Recall.*

Assim obtivemos as seguintes métricas na avaliação da *Baseline*:

- *Accuracy*: 52.0%
- *Precision*: 89.6%
- *Recall*: 51.9%
- *F1*: 65.7%

Apesar de não serem resultados muito elevados, se tomarmos em consideração que não houve qualquer tipo de tratamento dos dados, estes resultados são positivos, demonstrando uma *accuracy* acima dos 50%.

B. Preparação dos dados

Nesta fase realizámos o tratamento dos dados, onde nos focamos essencialmente em casos específicos do nosso conjunto de dados, isto é, tendo em conta que o nosso conjunto de dados é proveniente de *tweets*, optámos por fazer um tratamento à medida dos nossos dados.

Este tratamento teve em conta as seguintes fases:

- Hashtags
- URL's
- Palavras alongadas
- Maiúsculas para minúsculas
- Abreviações
- Pontuação

1) Hashtags

Para o tratamento de hashtags optámos por retirar apenas o símbolo de hashtag (*#*), considerando o hashtag uma palavra "normal", podendo assim verificar-se no léxico o sentimento associado.

2) URL's

Nos tweets é comum encontrar *links* para websites e assim sendo, decidimos tratar destes casos. Para isso optámos por remover estes URL's pois considerámos que não teriam impacto na classificação de sentimentos de um tweet. Para ajudar nesta remoção utilizamos uma expressão *Regex* para identificar URL's [4], e assim removê-lo.

3) Palavras alongadas

Para além dos casos já referidos, é normal encontrar também em tweets palavras alongadas, por exemplo, "*happyyyy*". Na nossa perspetiva estas palavras transmitem um intensificador do sentimento expresso pela palavra, sendo que, no exemplo mencionado, se exprime um sentimento positivo mais intenso.

Para estes casos a nossa abordagem foi retirar os caracteres duplicados e duplicar a palavra, provocando assim um "peso" maior da palavra no sentimento expresso no tweet. Usando o exemplo já mencionado, caso seja verificado num tweet a existência da palavra "*happyyyy*", esta será substituída por "*happy happy*", fazendo com que esta palavra conte duas vezes como uma palavra positiva para a classificação do sentimento do tweet.

4) Maiúsculas para minúsculas

Uma abordagem muito comum em problemas de *Text Mining* é a passagem de todos os caracteres para minúsculas sendo mais fácil a comparação em léxicos e manter a coerência de todas as frases. Posto isto, nesta fase decidimos passar todas as palavras do conjunto de dados para minúsculas.

5) Abreviações

Um caso, facilmente observável em tweets, é o uso de abreviações, tais como "lol" ou "btw". Tendo em conta este caso optámos por criar uma tabela de abreviações que consideramos mais comuns. Esta tabela contém 158 abreviações e o seu significado, por exemplo, "lol" aparece na tabela como "laughing out loud" e "btw" como "by the way".

Assim, com esta tabela decidimos substituir as abreviações pelo seu significado, pois considerámos que poderiam ter impacto na classificação do sentimento do tweet [5].

6) Pontuação

Tal como a passagem de todos os caracteres para minúsculas, o tratamento da pontuação é muito comum na área de *Text Mining*. Neste caso optámos por remover toda a pontuação presente nos tweets.

IV. APLICAÇÃO DE LÉXICO DE SENTIMENTOS

Após o tratamento dos dados, passaremos ao desenvolvimento de um classificador de sentimentos, a partir da utilização de um léxico, nomeadamente *NCR-lexicon*. Este ficheiro CSV contém várias colunas, das quais apenas as três primeiras vão ser usadas, nomeadamente "*English*", "*Positive*" e "*Negative*". Estas colunas permitem classificar cada palavra (contidas na coluna "*English*") quanto à sua polaridade, se é positiva ou negativa, ou seja, se tem valor na coluna "*Positive*" ou na coluna "*Negative*". Posto isto, decidimos criar um dicionário, em que como *key* teríamos a palavra do léxico, e como *value* o valor 1, se esta palavra fosse positiva, ou o valor -1, caso fosse negativa. Se as palavras tivessem tanto valor positivo como negativo, ou não tivessem valor positivo e negativo, estas não seriam tomadas em consideração, considerando-as neutras. O léxico contém um número total de 14183 palavras, das quais 8708 considerámos neutras e 5474 positivas ou negativas. No processo de classificação de cada tweet verificamos todas as palavras contidas no mesmo e verificamos qual é a sua classificação de acordo com o léxico utilizado. Para isso, recorremos à função *FreqDist()* importada da biblioteca *NLTK*, que transforma a mensagem do tweet num dicionário, permitindo-nos saber quais as palavras que estão nesse tweet e o número de ocorrências de cada uma delas, ou seja, a palavra será a *key* do dicionário e o *value* será o número de ocorrências da mesma.

Para aplicação do léxico de sentimentos foram efetuadas

as seguintes abordagens:

1. Aplicação direta do Léxico; nesta abordagem, cada palavra contida no dicionário passará pelo léxico para validar a polaridade da mesma, se esta for positiva, soma-se o seu número de ocorrências a uma variável de contador de palavras positivas. Caso contrário, soma-se o número de ocorrências a uma variável responsável por contar as palavras negativas.

2. Aplicação do Léxico com *Lemmatization* ou com *Stemming*; nestas abordagens, o processo inicial é igual ao anterior, no entanto, nesta fase cada palavra do *tweet* passa pelo léxico para se verificar a validade da mesma, se esta não existir no dicionário, aplicamos a *Lemmatization* à palavra ou então *Stemming* à palavra, correndo posteriormente a palavra pelo léxico para encontrar o seu valor. Depois disto, fazemos a mesma validação de resultados como na abordagem anterior.

Os resultados obtidos nestas abordagens foram os seguintes:

	Accuracy	*Precision*	*Recall*	*F1*
Léxico	79.0%	91.6%	83.4%	87.3%
Léxico com *Lemmatization*	78.5%	91.2%	83.0%	86.9%
Léxico com *Stemming*	78.6%	91.2%	82.6%	86.9%

Podemos verificar que, em comparação com os valores de *Baseline* houve uma melhoria substancial nos resultados obtidos, sendo que a *accuracy* quase que chega aos 80% nos três casos. Podemos também verificar que a aplicação do *Lemmatization* e do *Stemming* não apresentaram melhores resultados, muito pelo contrário, houve um pequeno decréscimo dos valores.

De maneira a melhorar os resultados, foram desenvolvidas duas opções para o tratamento da negação. Nos dois casos, para o tratamento da negação foram consideradas as seguintes palavras "*no", "not"* e qualquer palavra com a seguinte terminação "*n't*" [1]. Qualquer palavra após uma das palavras anteriormente identificadas, é adicionado um prefixo de "*not_"*. Por exemplo, na frase "*The ball is not red enough*", o resultado seria, "*The ball is not not_red not_enough*". As palavras com o prefixo "*not_*" serão consideradas como negativas se a sua polaridade for positiva no léxico, e positivas se o oposto se verificar, ou seja, causa uma inversão da polaridade da palavra original. No entanto, na primeira opção de negação adicionamos o prefixo "*not_*" até encontrarmos um terminador de frase, ou seja, qualquer um dos símbolos de pontuação seguintes, ".", ",", "!", "?", ";", ":", ou qualquer uma das seguintes palavras: "*but*", "*however*", "*although*", "*nevertheless*", "*still*", "*yet*" e "*though*". A segunda opção, é igual à primeira opção, adicionando o prefixo "*not_"* até encontrar um dos terminadores de frase, anteriormente apresentados, ou então até encontrar uma palavra com polaridade positiva [6]. Depois de desenvolvidas, as opções anteriores foram implementadas nas abordagens anteriormente aplicadas, portanto, temos os seguintes casos:

1. Aplicação direta do Léxico com a primeira opção de negação;

2. Aplicação direta do Léxico com a segunda opção de negação;

3. Aplicação do Léxico com *Lemmatization* ou *Stemming* e a primeira opção de negação;

4. Aplicação do Léxico com *Lemmatization* ou *Stemming* e a segunda opção de negação.

Os resultados obtidos nestas abordagens foram os seguintes:

	Accuracy	*Precision*	*Recall*	*F1*
Léxico com 1ª Opção de negação	79.6%	92.3%	83.3%	87.6%
Léxico com 2ª Opção de negação	79.6%	92.3%	83.4%	87.6%
Léxico com *Lemmatization* com 1ª Opção de negação	79.2%	92%	83.0%	87.3%
Léxico com *Lemmatization* com 2ª Opção de negação	79.3%	92.0%	83.1%	87.3%
Léxico com *Stemming* com 1ª Opção de negação	79.0%	91.9%	82.7%	87.1%
Léxico com *Stemming* com 2ª Opção de negação	79.1%	91.9%	82.8%	87.1%

Podemos verificar que, houve uma melhoria dos resultados após o tratamento da negação, tanto com a primeira opção de tratamento, como com a segunda. Continuamos a verificar que a aplicação direta do Léxico continua a ser a melhor opção, visto que os resultados apresentam uma accuracy de 79.6%, um crescimento de 0.6% em relação aos resultados anteriores, ou seja, sem o tratamento de negação. Os resultados da aplicação de Lemmatization e Stemming também apresentaram melhorias até 0.8% e 0.5%, respetivamente.

V. CONCLUSÃO

Como foi abordado anteriormente, a análise de sentimentos é cada vez mais um processo importante, tanto para a análise das opiniões dos utilizadores quanto aos produtos/serviços de uma empresa, como também para o controlo de cyberbullying ou violência nas redes sociais. Neste relatório concluímos que um bom tratamento de dados pode melhorar bastante os resultados da classificação dos sentimentos presentes num tweet. Como podemos verificar, a Baseline produzida a partir da ferramenta de TextBlob, demonstra valores de 52.0% de accucacy na classificação de sentimentos nos tweets, e após terem sido aplicados métodos de tratamento do conjunto de dados e, também, a aplicação de um léxico de sentimentos, os resultados obtidos aumentaram substancialmente, passando para 79.0% de accuracy. Foi também aplicado métodos de Lemmatization e Stemming, mas estes não demonstraram melhoria nos valores, em comparação com os valores obtidos pela aplicação direta do Léxico de sentimentos. Por fim, foram efetuadas medidas de tratamento de negação nos tweets,

nomeadamente duas abordagens. Estas demonstraram ser benéficas nos resultados obtidos, obtendo um valor máximo de 79.6% de accuracy no caso da aplicação direta do Léxico, com qualquer uma das abordagens. A segunda opção de tratamento de negação revelou-se mais eficaz, tanto com Lemmatization como Stemming, mas por apenas 0.1% em ambos os casos. A aplicação de Lemmatization e Stemming demonstrou valores de accuracy de 79.3% e 79.1%, respetivamente.

CONTRIBUIÇÃO PARA O TRABALHO:

- Gonçalo Alface, 69209: 50%
- Bruno Coitos, 64647: 50%

VI. REFERÊNCIAS

[1] Garg, S. K., & Meher, R. K. (2015). Naive Bayes model with improved negation handling and n-gram method for Sentiment classification. *Department of Computer Science and Engineering National Institute of Technology Rourkela*.

[2] Bhagyashri Wagh, Prof. J. V. Shinde, P. P. A. K. (2017). A Twitter Sentiment Analysis Using NLTK and Machine Learning Techniques, *9359*(12), 37–44.

[3] Patodkar, V. N., & I.R, S. (2016). Twitter as a Corpus for Sentiment Analysis and Opinion Mining. Ijarcce, 5(12), 320–322. https://doi.org/10.17148/IJARCCE.2016.51274

[4] Regextester.com. (2018). Detect URL - Regex Tester/Debugger. [online] Available at: https://www.regextester.com/96504 [Accessed 1 Nov. 2018].

[5] Kouloumpis, E., Wilson, T., & Moore, J. (2011). Twitter Sentiment Analysis : The Good the Bad and the OMG !, 538–541.

[6] Reitan, J., Bungum, L., Faret, J., & Gambäck, B. (2015). Negation Scope Detection for Twitter Sentiment Analysis. *Proceedings Ofthe 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, (Wassa), 99–108.

[7] Agardwal, A., Xie, B., Vovsha, I., Rambow, O., & Passonneu, R. (2017). Semantic Sentiment Analysis of Twitter Data, (June), 30–38. https://doi.org/10.1098/rsif.2011.0356