Processamento de Linguagem Natural Utilizando o TFIDF e Bag of Words

Caio Eduardo Kikuti Machado - Mateus Felipe Larrosa Furlan

***Abstract*—Natural Language Processing (NLP) is an Artificial Intelligence field that aims to perform interactions between computers and human language, more specifically on how to make computers process and analyze a large amount of data in natural language. One of the areas of greatest focus within the PLN is analysis and classification of feelings based on texts written in natural language. For this, there is a need to perform several procedures in the database used, such as the use of algorithms for the extraction of characteristics from the text, in order to achieve better results from the use of some classifier. In this work, an algorithm will be implemented for the analysis and classification of feelings based on the critical basis of films present on the IMDB platform. For this, two different feature extractors will be used, Bag of Words (BoW) and Term Frequency - Inverse Document Frequency (TF-IDF). In addition, a comparative analysis will be carried out between different classifiers in order to define which is the most suitable for a database used in this work, among them are: Support Vector Machine (SVM), Decision Tree, k-Nearest Neighbors (kNN) and Classifier Naive Bayes.**

***Resumo*—O Processamento de Linguagem Natural (PLN) é um campo da Inteligência Artificial que busca realizar interações entre os computadores e a linguagem humana, mais especificamente em como fazer os computadores processar e analisar uma grande quantidade de dados em linguagem natural. Uma das áreas de maior enfoque dentro do PLN é a análise e classificação de sentimentos a partir de textos escritos em linguagem natural. Para isso, há a necessidade da realização de diversos procedimentos na base de dados utilizadas, como por exemplo a utilização de algoritmos para a extração de características do texto, a fim de alcançar melhores resultados a partir da utilização de algum classificador. Neste trabalho será implementado um algoritmo para análise e classificação de sentimentos tendo como base críticas de filmes presentes na plataforma IMDB. Para isso, serão utilizados dois diferentes extratores de características, *Bag of Words* (BoW) e *Term Frequency - Inverse Document Frequency* (TF-IDF). Além disso, será realizada uma análise comparativa entre diferentes classificadores a fim de definir qual o mais indicado para a base de dados utilizada neste trabalho, entre eles estão: *Support Vector Machine* (SVM), *Decision Tree*, *k-Nearest Neighbors* (kNN) e *Naive Bayes Classifier*.**

I. Introdução

O Processamento de linguagem natural pode ser entendido como a capacidade de um computador tratar diversos aspectos da linguagem humana, considerando por exemplo sons, palavras em sentenças e discursos em inúmeros contextos. Segundo [1] o PLN visa fazer o computador se comunicar em linguagem humana, nem sempre em todos os níveis de entendimento, sendo que esse níveis são divididos em: fonéticos e fonológico, morfológico, sintático semântico e pragmático.

Como uma das aplicações do processamento de linguagem natural, pode-se realizar o processamento de textos escritos com a finalidade da realização de uma análise de sentimento. A análise de sentimento é também chamada de mineração de opinião, sendo um dos ramos mais estudados da área de PLN. Segundo [2], a análise de sentimento tem como objetivo analisar a opinião, a avaliação a emoção e a atitude de um documento em relação a um evento, produto, serviço, organização ou outra entidade. Dentro da análise de sentimentos um campo amplamente explorado é a análise de polaridade, essa análise visa classificar textos em uma escala entre positivo e negativo.

O escopo deste trabalho encontra-se no campo do processamento de linguagem natural, mais especificamente abordando a análise de sentimento e explorando a polaridade de um conjunto de 50 mil textos. Os textos analisados são críticas cinematográficas feitas no conhecido portal de avaliações de filmes, IMDB. Portanto, pretende-se por meio do processamento de linguagem natural e pela análise de sentimento, classificar as críticas cinematográficas em positivas e negativas.

Os objetivos deste trabalho são o aprofundamento dos conhecimentos acerca do campo da inteligência artificial que envolve a análise de sentimento e processamento de linguagem natural, bem como o desenvolvimento prático de um algoritmo que realize a classificação de críticas cinematográficas por meio da utilização de algoritmos de extração de características como *Bag of Words* e *Term Frequency–Inverse Document Frequency* e de classificadores disponibilizados pela biblioteca *sklearn* da linguagem Python. Por fim, pretende-se analisar os resultados obtidos pelo algoritmo enfocando a acurácia da solução desenvolvida.

II. Problema

Conforme abordado na Seção 1, o problema proposto neste artigo é a classificação de 50 mil críticas cinematográficas disponibilizadas pelo site IMDB. As críticas estão contidas em um documento csv e por meio da aplicação do algoritmo de classificação de polaridade, devem ser classificadas em críticas de caráter positivo ou negativo.

III. Metodologia

Para a implementação deste algoritmo foi necessária a separação do trabalho em algumas partes: primeiramente realizou-se o processamento do texto, na segunda parte realizou-se a extração de características e por fim a classificação e a avaliação dos resultados por meio da métrica *F1-Score*.

O processamento do texto está relacionado com a limpeza de impurezas que podem estar contidas no texto. Primeiramente o arquivo csv é lido por meio das funções disponibilizadas pela biblioteca pandas, após a leitura mais uma

coluna é adicionadas ao csv, a coluna que identifica se a mensagem transmitida pela crítica é positiva ou negativa, essa coluna fora nomeada de labelNum e mapeia os valores 0 para criticas negativas e 1 para críticas positivas. Uma vez lido o arquivo, fora realizado os procedimentos de limpeza de impurezas no texto removendo *tags* html, símbolos e caracteres especiais. Posteiormente realizou-se a uniformização de todas as palavras do texto para letra minúscula, a separação das palavras da *string* em um vetor, por meio do uso da função *split* e a remoção de *stop words* (stop words ou palavras vazias, contemplam um conjunto de palavras que não tem grande valor para o processamento do texto entre elas estão algumas preposições e pronomes que não tem um real impacto na extração de características). Após a extração das palavras vazias é realizada a normalização dos textos. Para a normalização do texto deste problema, foi escolhida a técnica conhecida como *lemmatatization*. A lematização é um processo que determina o lema ou forma canônica da palavra baseado no significado daquela palavra. Diferentemente de outras técnicas de normalização, a lematização depende da correta identificação do que é pretendido na parte do texto, do significado da palavra bem como da maneira com a qual a palavra é utilizada no contexto da frase do parágrafo ou do texto. Após a normalização os textos são retornados novamente para o formato de *string*.

Na extração de características, primeiramente são definidos os conjuntos de treino e teste, para este experimento definiu-se que 80% do conjunto de texto seria utilizado para o conjunto de teste e os 20% restantes seria utilizado para a realização dos testes. Após essa definição são aplicados os algoritmos para extração de características, conforme citado na introdução neste trabalho forma utilizados dois métodos extração, TF-IDF e Bag of Words.

A abordagem Bag of Words realiza a análise de um documento e o divide o documento por palavras, sendo que essa palavras também são conhecidas como *tokens*, assim pode-se chamar todo esse processo de ”tokenização“. Sendo assim os *tokens* são coletados de todo o documento e são ordenados formando um vocabulário. Posteriormente um vetor de tamanho equivalente ao do vocabulário é criado contendo o valor de frequência que dada palavra do vocabulário aparece no texto. A imagem abaixo contempla corretamente o processo de obtenção do vocabulário.

Fig. 1. Processo Bag of Words

Para a utilização do algoritmo Bag of Words (BOW), forma definidos os seguintes parâmetros. O primeiro dos parâmetros é o modelo de n-gramas utilizado, para esse experimento utilizou-se o modelo um para dois, ou seja o vocabulário leva em consideração uma ou duas palavra por vez (cada posição do vetor é representado por uma palavra ou um conjunto de duas palavra). O segundo parâmetro utilizado é o *analyzer* no qual definiu-se que o vocabulário seria composto por palavras. Por último definiu-se o tamanho máximo definido para o vocabulário como sendo um total de 100 palavras.

Já o TF-IDF é uma estatística numérica que tem como objetivo definir o quanto uma palavra é importante para um documento que faz parte de uma coleção de documentos, ou *corpus*. Essa técnica é comumente utilizada para recuperação de informação e mineração de textos. A importância de uma palavra aumenta proporcionalmente com o número de vezes que ela aparece dentro de um documento, mas diminui pela frequência que a palavra aparece em outros documentos que fazem parte do corpus, o que auxilia a ajustar a métrica, uma vez que há palavras que aparecem mais frequentemente por padrão.

O TF-IDF é composto por dois termos: o primeiro calcula a frequência de um termo normalizado (TF), ou seja, o número de vezes que uma palavra aparece em um documento dividido pelo número total de palavras neste documento. O segundo termo é a frequência inversa do documento (IDF), o qual é dado pelo logaritmo do número de documentos no corpus dividido pelo número de documentos em que um termo específico aparece.

- **TF:** responsável por medir a frequência que um termo aparece em um documento. Levando em conta que cada documento tem um tamanho diferente, é possível que um termo apareça muito mais vezes em documentos longos do que em curtos. Portanto, a frequência do termo é dividida pelo tamanho do documento (número total de palavras que formam tal documento) como uma forma de normalização.

$$TF(t,d) = f_{t,d}/n \tag{1}$$

  Onde $f_{t,d}$ representa o número de vezes que um termo t apareceu em determinado documento d, e n o número total de termos no mesmo documento.
- **IDF:** responsável por medir o quão importante um termo é. Durante a computação do TF, todos os termos são considerados igualmente importantes. Contudo, é preciso levar em conta que certos termos tendem a aparecer muito mais que outros mas ter muito pouca importância. Logo, é preciso diminuir o peso desses termos menos importantes mas que aparecem muito, enquanto aumenta o peso das palavras que aparecem menos, mas que agregam para a análise.

$$IDF(t,D) = \log \frac{N}{n} \tag{2}$$

  Onde N representa o número total de documentos que fazem parte do corpus D e n o número de documentos em que determinado termo t aparece.

Com isso, temos que o cálculo do TF-IDF é dado por:

$$TFIDF(t,d,D) = TF(t,d) \cdot IDF(t,D) \tag{3}$$

Após a realização da extração de características do texto, fora realizada a classificação das críticas em positivas e negativas. Para que fosse realizada essa classificação forma utilizados alguns conhecidos algoritmos de classificação, sendo eles:

Support Vector Machine, Decision-tree, k-Nearest Neighbors e Naive Bayes Classifier.

O *Support Vector Machine* (SVM) foi elaborado com o estudo proposto por [3]. O SVM é um algoritmo de aprendizado supervisionado cujo objetivo é classificar determinado conjunto de pontos dados que são mapeados para um espaço de características multidimensional usando uma função kernel (abordagem utilizada para classificar problemas). Nela, o limite de decisão no espaço de entrada é representado por um hiperplano em dimensão superior no espaço [4].

Neste trabalho foi utilizado o SVM Linear, o qual realiza a separação de um conjunto de objetos com diferentes classes, ou seja, utiliza o conceito de plano de decisão. As classes presentes, no caso de estudo, foram *Positive* e *Negative*, referentes às críticas positivas e negativas, respectivamente, realizadas em filmes no portal IMDB. Tendo as classes definidas, a separação é realizada por meio de uma "linha" que define o limite de cada classe. Ao entrarem novos objetos na análise, estes serão classificados como *Positive* ou *Negative* dependendo de qual lado da linha eles estiverem.

A árvore de decisão é um método de aprendizado não parametrizado utilizado para classificação e regressão. O objetivo deste algoritmo é a criação de um modelo que prediz os valores de uma dada variável, por meio do simples aprendizado, utilizando regras de decisão inferidas pelas características dos dados analisados. O aprendizado de uma árvore de decisão pode ser descrito por uma curva que se assemelha a uma curva senoidal, formada devido as ao conjunto de regras formadas por construções *if-then-else*. Na prática, quanto mais profunda for a árvore de decisão, mais complexas são as regras de decisão e mais complexo é o modelo. O comportamento deste método pode ser vizualizado por meio da Figura 2 [1]

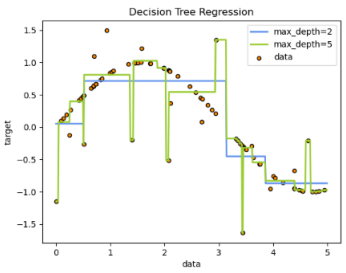

Fig. 2. Exemplo de comportamento de uma árvore de decisão

Para o classificador baseado em árvore de decisão, foram utilizados os parâmetros de profundidade máxima da árvore definida em 5 e de *random state*, que controla a aleatoriedade do estimador, definido em 0.

O classificador k-Nearest Neighbors oferece métodos de aprendizados baseados em vizinhos de maneira supervisionada e não supervisionada. O principio por trás deste método é o descobrimento de um número pré-definido de amostras de treinamento que sejam próximas ao novo valor que se visa encontrar, e realizar a predição deste valor com base nessa descoberta. O número de amostras para que se faça uma estimativa pode ser definida pelo usuário do método ou então ser estipulados com base na variação de densidade local das amostras analisadas. Para a utilização deste método, foi utilizado o parâmetro número de vizinhos definido com o valor 5, este parâmetro define o número de vizinhos a serem usados, por padrão, para consultas de novos vizinhos.

Os classificadores *Naive Bayes* são modelos probabilísticos de aprendizagem de máquina usados para a realização de classificações e análises de sentimentos em textos. Tais classificadores são baseados no teorema de Bayes. Neste trabalho foi utilizado o classificador *Multinomial Naive Bayes*, o qual é o mais indicado para a resolução do problema de classificação de documentos. As métricas utilizadas por este classificador são baseadas na frequência em que os termos aparecem nos documentos.

A métrica utilizada para avaliar os resultados gerados pelos classificadores foi a *F1-Score*. Esta métrica é calculada com base em duas métricas: precisão e *recall*. A primeira métrica é muito utilizada para determinar qual o custo dos falsos positivos, ou seja, dentre os casos analisados pelo modelo que foram definidos como positivos, quantos deles realmente são positivos. Já o *recall* é utilizado para calcular a proporção dos casos positivos que foram identificados corretamente, e é muito utilizado para identificar qual modelo é melhor com relação ao número de falsos negativos. O cálculo da precisão e do *recall* são dados pelas seguintes fórmulas:

$$Precisao = \frac{Positivo}{Positivo + Falso\ Positivo} \tag{4}$$

$$Recall = \frac{Positivo}{Positivo + Falso\ Negativo} \tag{5}$$

Com isso, o *F1-Score* é calculado pela média harmônica entre a precisão e o *recall*:

$$F1 = 2 \cdot \frac{Precisao \cdot Recall}{Precisao + Recall} \tag{6}$$

Essa métrica é utilizada quando se deseja encontrar um equilíbrio entre precisão e *recall*. O valor ótimo para o *F1-Score* é alcançado quando tanto a precisão quanto o *recall* são 100%. Além disso, caso um dois dois seja igual a 0, o *F1-Score* também será 0.

A. Ambiente de desenvolvimento

O ambiente no qual os testes foram executados contemplam a seguinte configuração de hardware:

- Processador: Ryzen 5 2600 Six-core
- Memória RAM: 8 GB
- Sistema operacional: Windows 10 PRO

IV. Análise e discussão

Conforme citado na seção anterior a métrica utilizada para a avaliação dos classificadores foi a F1-Score, que é calculada por uma função pronta disponibilizada pela biblioteca Sklearn. Considerando ambos métodos de extração de dados pode-se notar que o algoritmo SVM teve o maior destaque mantendo um *score* médio de 0.9, sendo que o algoritmo implementado

[1]Imagem e informações retiradas de https://scikit-learn.org/stable/modules/tree.html

utilizando o TF-IDF obteve uma precisão de 92% para a classificação de críticas negativas e 90% de precisão para críticas positivas. Já na implementação utilizando o algoritmo Bag of Words, a precisão para críticas positivas foi 89% e de críticas negativas foi de 90%.

No caso da árvore de decisão nota-se que entre a abordagem realizada pelo TF-IDF e pelo Bag of words obteve-se uma diferença de cerca de 5% em relação a métrica F1-Score. Além disso, ao realizar a análise do gráfico da Figura 4, nota-se que entre as duas abordagens houve uma diferença de cerca de 8% na métrica recall, sendo que essa métrica tem forte influência nos valores do F1-score, conforme explicado na seção anterior.

Assim como o a árvore de decisão, na classificação utilizando k-Nearest Neighbors percebe-se uma nítida diferença no uso do TF-IDF e no BOW. Para a métrica F1-Score a implementação utilizando TF-IDF é quase 50% mais precisa, sendo que os baixos valores para F1 na implementação utilizando a abordagem BOW, podem novamente ser explicados pelo baixo valor de recall encontrado. Outro ponto perceptível para esse classificador é a baixa precisão obtida na classificação das críticas, atingindo não mais que 55% de precisão quando implementado juntamente ao Bag of Words.

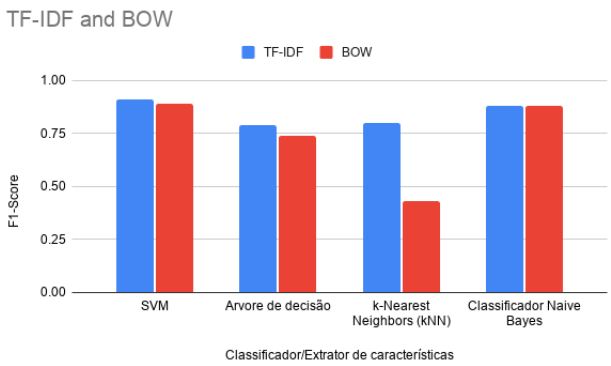

Fig. 3. Gráfico de desempenho dos classificadores, segundo a métrica F1-Score

O classificador que obteve o segundo melhor desempenho foi o Naive bayes Classifier, esse classificador atingiu um *score* de 0.89, sendo muito próximo a o SVM. Para este classificador, tanto os valores de precisão como os de recall foram altos em ambas implementações, sendo que na implementação com Bag of Words, atingiu um *recall* médio de 0.9 sendo apenas 1% menor do que a implementação com o TF-IDF. Em relação a precisão a diferença de 1% entre as implementações se manteve, tendo a maior precisão sendo atingida pelo TD-IDF cerca de 89%.

Como elucidado anteriormente, pode-se observar uma grande vantagem do classificador SVM comparado à maioria dos classificadores analisados, com exceção do Naive Bayes, o qual teve um resultado próximo ao SVM. Um dos fatores cruciais para o bom desempenho do SVM é a divisão clara entre as diferentes classes de dados presentes na base de estudo, como pode ser observado no gráfico 4, indicando um acerto de aproximadamente 90% na classificação dos casos de estudo. Outro fator que corrobora para o melhor desempenho do SVM

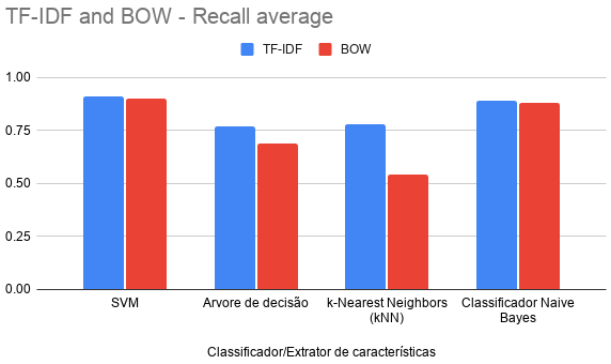

Fig. 4. Gráfico da média do recall

é a dependência mútua das características dos documentos presentes no corpus utilizado neste trabalho, uma vez que este classificador possui uma maior capacidade de análise entre essas dependências.

V. Conclusão

Neste trabalho realizou-se a implementação de um algoritmo para classificação automática de críticas cinematográficas, por meio da utilização de conceitos teórico-práticos de processamento de linguagem natural, mais especificamente abordando a análise de sentimento. Dessa maneira os experimentos foram realizados e os resultados foram compilados e expostos, por meio deste artigo.

Ao enfocar-se a Seção 4 é possível concluir que o algoritmo mais preciso para a classificação das críticas é o algoritmo que utiliza o classificador Support Vector Machine (SVM). Por meio da análise gráfica é possível notar que este classificador atingiu altos níveis em relação as métricas *recall* e precisão, as quais lhe conferiram um alto valor para a principal métrica analisada neste trabalho, o F1-Score. Outro classificador que apresentou um excelente resultado foi o Naive Bayes Classifier, esse classificador obteve resultados cerca de 2% menos precisos do que o SVM. É importante salientar que esperava-se que o classificador Naive Bayes se destaca-se, uma vez que ele é tido como um dos classificadores mais adequados para analise de sentimentos em processamento textuais.

Por fim pode-se concluir também que os objetivos elencados na Seção 1 foram atingidos. Por meio da implementação do algoritmo, foi possível a maior compreensão e aplicação dos conceitos e das técnicas de processamento de linguagem natural, bem como o desenvolvimento de um algoritmo que classifique de maneira satisfatória os textos contidos na base de dados disponibilizada. Para trabalhos futuros, uma opção é realizar mais experimentos utilizando algumas modificações nos parâmetros dos classificadores, afim de avaliar-se o impacto que estes podem gerar sobre as métricas analisadas neste artigo.

References

[1] M. Gonzalez and V. L. S. Lima, “Recuperação de informação e processamento da linguagem natural,” in *XXIII Congresso da Sociedade Brasileira de Computaçao*, vol. 3, 2003, pp. 347–395.

[2] B. V. Arosa, “Analise de sentimento em textos curtos baseada em processamento de linguagem natural e aprendizado de maquina,” Ph.D. dissertation, Universidade Federal do Rio de Janeiro, 2018.
[3] B. E. Boser, I. M. Guyon, and V. N. Vapnik, “A training algorithm for optimal margin classifiers,” in *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, ser. COLT ’92. New York, NY, USA: Association for Computing Machinery, 1992, p. 144–152. [Online]. Available: https://doi.org/10.1145/130385.130401
[4] V. V. Saradhi, H. Karnik, and P. Mitra, “A decomposition method for support vector clustering,” in *Proceedings of 2005 International Conference on Intelligent Sensing and Information Processing, 2005.*, 2005, pp. 268–271.