FCTUC FACULDADE DE CIÊNCIAS
E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Projeto - Meta 2

Processamento de Linguagem Natural

Diana Isabel de Sá Janeiro e Martins
2023218119
João Frederico Correia da Costa Marques Lopes
2022231718

1. Introdução

Processamento de linguagem natural integra uma área bastante relevante da Inteligência Artificial contemporânea, permitindo a análise, compreensão e geração de linguagem por meio de máquinas. A crescente disponibilidade de dados textuais, aliada ao avanço de técnicas de aprendizagem computacional, tem impulsionado o desenvolvimento de soluções capazes de realizar tarefas complexas, como classificação de textos.

As experiências apresentadas em baixo, fruto da continuidade deste estudo, visam aplicar um vetorizador a um modelo de aprendizagem computacional de modo a conseguir uma classificação de texto fidedigna para o dataset escolhido.

2. Materiais

Para este estudo, foi utilizado o dataset *'FactNews'*, um dataset proposto por Francielle Vargas composto por 6,191 linhas. O mesmo é constituído pelos seguintes atributos: *file* (categórico); *id_sente* (numérico); *id_article* (numérico); *domain* (categórico); *year* (numérico); *sentences* (categórico); *classe* (numérico).Tem frases extraídas de notícias em português do Brasil retiradas de três fontes diferentes de notícias fidedignas como São Paulo, O Globo, e Estadão, e as suas anotações foram disponibilizadas junto dos dados que permitem classificar as frases em 'citação', 'facto' e 'viés'.

De entre as bibliotecas usadas foram aplicadas para predição os modelos da biblioteca scikit-learn, que é bastante completa e crucial em tarefas de classificação. Serve também para construção e projeção de uma matriz de confusão que permite analisar métricas de precisão e 'matplotlib' foi utilizado para construção de gráficos.

Finalmente, foi também recorrido à biblioteca 'optuna', para otimização de parâmetros.

3. Metodologia

O procedimento aplicado para conseguir este estudo e tirar conclusões valiosas passou por experimentar combinações de vetorizadores, mas também isoladamente, sendo estes aplicados também a um conjunto de modelos de classificação multiclasse. Deste modo, cria-se um universo de testes significativo para comparar compatibilidades entre os diferentes arranjos de algoritmos para servir uma classificação eficiente.

Tanto os parâmetros dos modelos de representação quanto os dos modelos de classificação foram otimizados com Optuna e utilizámos o F1-score como métrica de avaliação, uma vez que o dataset original é desbalanceado e o F1-score penaliza de forma mais severa erros nas classes minoritárias do que, por exemplo, a accuracy.

a. Pré processamento

Para o pré-processamento neste estudo, foi aplicada uma complementação ao do anterior, nomeadamente com as técnicas de matização e oversampling.

É feita, primeiramente, a divisão do dataset em treino e teste nas proporções de 70%/30%, respetivamente; de seguida é dividido o treino em treino e validação nas proporções de 80%/20%, por esta ordem. Deste modo, obtemos uma tripartição do conjunto de dados que irá servir posteriormente a modelagem.

De forma a evitar viés na classificação, fruto dos modelos terem um treino desequilibrado, foi aplicado oversampling aos conjuntos de treino, teste e validação, replicando exemplos das classes menores. Deste modo, todas as classes, -1, 0 e 1 têm cada uma, para o conjunto de treino: 2375; para o conjunto de teste: 1273; para o conjunto de validação: 594 elementos. Ficam, assim, distribuídas de modo semelhante e evitam que os modelos se tornem tendenciosos.

Finalmente, é realizada tokenização de cada frase em palavras, são removidas as stopwords do corpus português do NLTK é feita uma matização das palavras, com recurso ao *RSLP Stemmer* do NLTK.

b. Vetorização

De entre os vetorizadores selecionados apresentam-se: Count Vectorizer, TF-IDF(Term Frequency - Inverse Document Frequency), Word2Vec e Glove.

Enquanto o Count Vectorizer e TF-IDF se baseiam exclusivamente na frequência das palavras no corpus, o Word2Vec e Glove, conseguem captar contexto, local e global, e representar a semântica das palavras. Além da utilização de modelos pré-treinados, no caso do Word2Vec foi também treinada uma versão utilizando os nossos dados.

i. Count Vectorizer

Os parâmetros *max_features* e *ngram_range* foram selecionados para otimização, pois permitem controlar a dimensão do vocabulário e capturar contexto local no texto, enquanto o *token_pattern* garante que palavras e sinais de pontuação relevantes sejam considerados.

ii. TF-IDF

Para este vetorizador, o parâmetro *max_features* foi otimizado de forma a selecionar a melhor dimensionalidade e evitar overfitting, sendo o mesmo objetivo de otimizar os parâmetros *ngram_min* e *ngram_max*, permitindo também controlar o tamanho do vocabulário.

iii. Word2Vec

No Word2Vec treinado com os nossos dados, otimizámos *vector_size*, *window*, *min_count*, *sg* e *epochs*, ajustando a dimensão dos vetores, o contexto considerado, a frequência mínima das palavras, o tipo de modelo e o número de épocas, a fim de capturar o contexto local e global das palavras.

iv. GloVe

O "*glove_s100.tx*t", usado neste estudo, é um vetorizador pré-treinado que representa palavras por vetores de 100 dimensões. Foi escolhido pois, as nuances semânticas das palavras são melhor capturadas que por vetores de 50 dimensões, e evitam o overfitting, caso fossem representadas por vetores de 200 ou 300 dimensões.

c. Modelagem

Foram selecionados vários modelos de machine learning supervisionados para ter uma base sólida de comparação: Naive Bayes, SVM, KNN, Decision Tree, Random Forest, Neural Network e Logistic Regression. O Naive Bayes foi utilizado em duas variantes: o Multinomial para lidar com matrizes esparsas (Count Vectorizer e TF-IDF) e o Gaussiano para matrizes com valores contínuos (Glove e Word2Vec), garantindo que cada tipo de representação textual fosse tratado de forma adequada.

4. Experiências

Todos os modelos de machine learning foram inicialmente avaliados individualmente com cada modelo de representação. Após identificar os melhores classificadores, estes foram aplicados aos dados pré-processados segundo a Meta 1, com o objetivo de comparar os dois métodos de pré-processamento. Partindo da análise dos gráficos abaixo é possível comparar o desempenho dos vários modelos segundo duas métricas escolhidas: Accuracy, sendo simples e fácil de interpretar, e F1-score por ser mais rígida, já que leva em conta o equilíbrio entre falsos positivos e falsos negativos.

Sendo que os modelos de representação baseados em frequência obtiveram melhores resultados do que aqueles baseados em semântica, surgiu o interesse em combinar os dois vetorizadores de matrizes esparsas Count Vectorizer + TF-IDF de modo a observar se surgiriam melhores resultados.

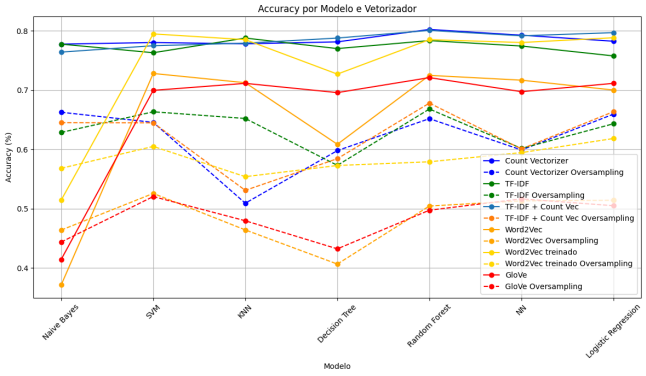

Fig.1 Comparação de Accuracy

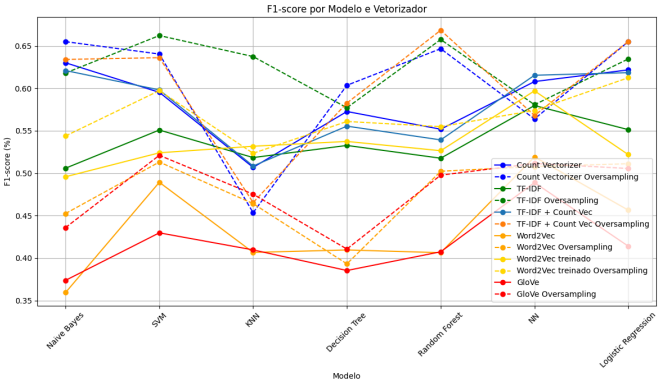

Fig.2 Comparação de F1-score

5. Resultados

Optando escolher distinguir o melhor método segundo a métrica mais rígida (F1-score), foi possível observar que a melhor classificação foi feita partindo da combinação do Count Vectorizer com o modelo de Regressão Logística, o que demonstra que são uma combinação fidedigna para captar contexto e obter boa interpretação para o fim de classificação de texto.

Desse modo, foram comparados os classification reports para os três tipos de conjuntos de dados: conjunto de dados da Meta 1; conjunto de dados do presente estudo; e conjunto de dados do presente estudo com oversampling. É possível observá-los em baixo:

	precision	recall	f1-score	support
-1	0.81	0.66	0.73	417
0	0.81	0.92	0.86	1273
1	0.28	0.13	0.18	168
accuracy			0.79	1858
macro avg	0.64	0.57	0.59	1858
weighted avg	0.77	0.79	0.77	1858

Fig. 3 Classification report, meta 1

	precision	recall	f1-score	support
-1	0.81	0.69	0.74	417
0	0.82	0.91	0.87	1273
1	0.30	0.17	0.22	168
accuracy			0.79	1858
macro avg	0.64	0.59	0.61	1858
weighted avg	0.77	0.79	0.78	1858

Fig.4 Classification report

	precision	recall	f1-score	support
-1	0.83	0.72	0.77	1273
0	0.55	0.77	0.64	1273
1	0.65	0.46	0.54	1273
accuracy			0.65	3819
macro avg	0.67	0.65	0.65	3819
weighted avg	0.67	0.65	0.65	3819

Fig.5 Classification report, oversampling

Assim os respectivos resultados são:

- **F1-Score:** 60.76%
- **F1-Score c/oversampling:** 65.02%
- **F1-Score c/meta1:** 59.05%

Analisando os F1-scores em cada classe, é possível perceber que a técnica de oversampling resolveu uma disparidade entre o f1-score da classe 0 e da classe 1, pois tornam-se valores mais aproximados e o modelo, desta forma, tornou-se menos tendencioso.

6. Conclusão

Este estudo demonstrou que surgem vantagens claras da combinação entre vetorizadores e modelos de machine learning para classificação de texto. A otimização dos parâmetros permitiu melhorar o desempenho, com destaque para a combinação do Count Vectorizer e regressão logística, que obteve os melhores resultados em F1-score. A aplicação de oversampling e matização contribuiu para mitigar o desbalanceamento e melhorar a qualidade dos dados.

No geral, foi validado que, no contexto do nosso dataset, os modelos de frequência apresentaram consistentemente melhor desempenho do que os modelos baseados em semântica, incluindo os pré-treinados. Isto evidencia que vetorizadores de frequência, treinados diretamente sobre os nossos dados e devidamente otimizados, podem oferecer classificações mais robustas e eficazes do que representações semânticas pré-treinadas.