UNIVERSIDADE DE SÃO PAULO

INSTITUTO DE CIÊNCIAS MATEMÁTICAS E CIÊNCIAS COMPUTACIONAIS

SCC0633/SCC5908 - Processamento de Linguagem Natural, 2024

Tarefa 4 - implementação das propostas apresentadas

- **Nome dos integrantes do grupo: GPTrouxas**
  - André G De Mitri, NUSP: 11395579
  - Daniel Carvalho , NUSP: 10685702
  - Fernando, NUSP: 11795342
  - Lucas Henrique Sant'Anna, NUSP: 10748521
  - Magaly L Fujimoto, NUSP: 4890582

- **Tópico do trabalho:**
  - Análise de sentimentos de avaliações de filmes, como Positivo ou Negativo.

- **Lista de córpus escolhidos para o trabalho:**
  - Base de dados no Kaggle
  - Base de dados auxiliar para avaliação extra do BERT e Aplicação Web

Resumo com Visão Geral da Solução Simbólica

A arquitetura da solução simbólica envolve duas etapas principais: a primeira é usar uma ferramenta léxica chamada SentiWordNet e a segunda será uma abordagem simbólica, na qual serão utilizadas as frequências de palavras advindas de uma *WordCloud*.

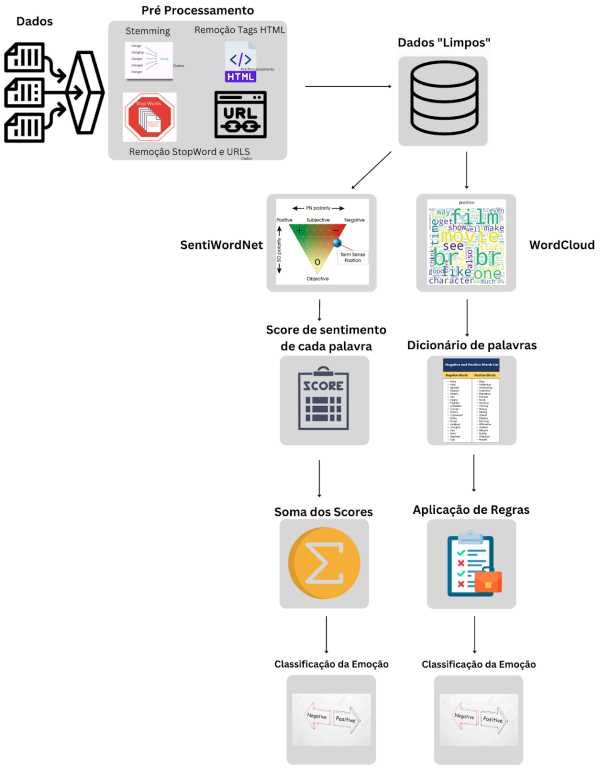
Para a abordagem utilizando o SentiWordNet realizamos um pré-processamento, com a tokenização, anotação e filtragem de *stopwords* para focar apenas nas palavras mais significativas para a tarefa. Aplicamos o SentiWordNet para cada palavra, obtendo os *scores* de sentimentos positivos e negativos. Com base no cálculo total dos *scores* é possível classificar os sentimentos utilizando a comparação dos *scores*. Se o *score* positivo total for maior que o *score* negativo, classificamos o texto como "positivo". Caso contrário, como "negativo".

A segunda abordagem para análise de sentimentos envolve o uso de *WordClouds* para identificar as palavras mais frequentes em textos classificados como positivos e negativos. Utilizamos as *reviews* classificadas como "positivas" e "negativas" para criar duas *WordClouds* separadas, uma para cada categoria. A partir das

WordClouds geradas, extraímos as palavras mais frequentes. Estas palavras formam os dicionários de palavras positivas e negativas, representando os termos que estão associados a sentimentos positivos e negativos, respectivamente. Para classificar novos textos, contamos a ocorrência das palavras nos dicionários de palavras positivas e negativas. Além disso, aplicamos regras adicionais para capturar nuances mais sutis na linguagem, como:

- "not" seguido de uma palavra positiva: Se a palavra "not" for seguida de uma palavra positiva, isso indica um contexto negativo.
- "too" seguido de uma palavra negativa: Se a palavra "too" for seguida de uma palavra negativa, isso reforça o sentimento negativo.
- "very" seguido de uma palavra positiva: Se a palavra "very" for seguida de uma palavra positiva, isso reforça o sentimento positivo.
- "but" entre palavras positivas e negativas: Se a palavra "but" aparecer entre uma palavra positiva e uma palavra negativa, a palavra após "but" geralmente tem mais peso no sentimento do texto.

Podemos apresentar o pipeline utilizado nessa etapa do trabalho da seguinte maneira:



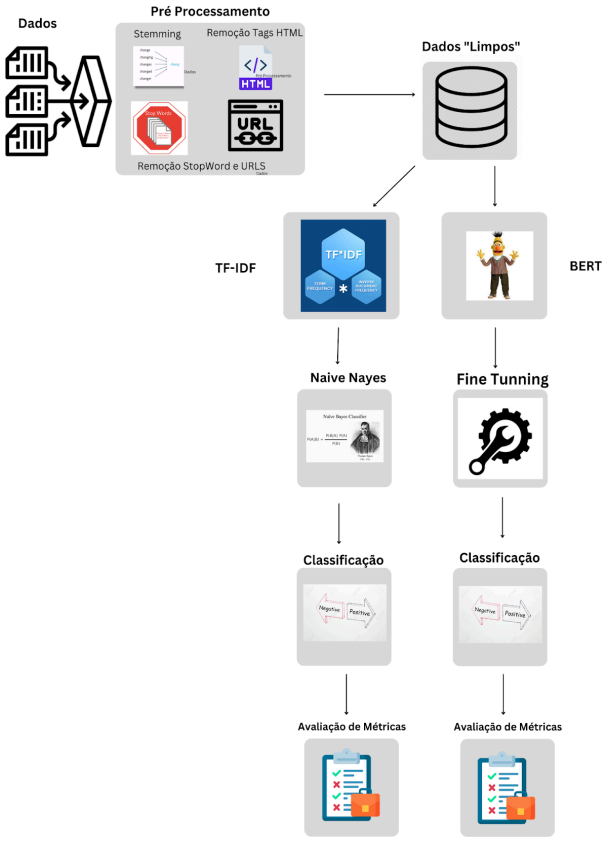
Resumo com Visão Geral da Solução Estatístico/Neural

A arquitetura da solução estatística/neural envolve duas abordagens.

Primeira abordagem: Utilizaremos o TF-IDF, que leva em consideração a frequência de ocorrência dos termos em um córpus e gera uma sequência de vetores que serão fornecidos ao Naive Bayes para classificação de cada *review* como positiva ou negativa.

Segunda abordagem: Será necessário preparar os dados para entrada para serem consumidos pelo BERT no treinamento. Após dividir em treino e teste, faremos o fine tuning de um modelo do tipo BERT para o nosso problema e dataset específico. Com o BERT adaptado, faremos a classificação de nossos textos, medindo o seu desempenho com F1 score e acurácia.

Podemos apresentar o pipeline utilizado nessa etapa do trabalho da seguinte maneira:



Etapas da solução estatística/neural

1. **Bibliotecas**: Importamos as bibliotecas necessárias, considerando pandas para manipulação de dados, *train_test_split* para dividir o conjunto de dados em conjuntos de treinamento e teste, *TfidfVectorizer* para vetorização de texto usando TF-IDF, *MultinomialNB* para implementar o classificador Naive Bayes Multinomial e algumas métricas de avaliação.
2. **Conjunto de dados**: Carregar o conjunto de dados e armazená-lo em um *DataFrame* usando pandas.

3. **Dividir o conjunto de dados**: Usamos *train_test_split* para dividir o *DataFrame* em conjuntos de treinamento e teste.
4. **TF-IDF**: Usamos *TfidfVectorizer* para converter as revisões de texto em vetores numéricos usando a técnica TF-IDF. Em seguida, ajustamos e transformamos tanto o conjunto de treinamento quanto o conjunto de teste.
5. **Naive Bayes**: Treinamos um classificador Naive Bayes Multinomial e usamos o modelo treinado para prever os sentimentos no conjunto de teste usando predict.
6. **Avaliação e Resultados**: Salvamos os resultados em um novo *DataFrame results_df* contendo as revisões do conjunto de teste, os sentimentos originais e os sentimentos previstos pelo modelo. Além disso, avaliamos o modelo verificando algumas métricas e a matriz de confusão

Etapas na classificação do BERT

1. **Pré-processamento do Texto:** Transformamos cada revisão de texto em uma sequência de tokens utilizando um tokenizer específico para o modelo BERT. O texto é tokenizado, e as sequências são preenchidas (padding) e truncadas para um comprimento máximo especificado. Experimentalmente, decidimos cortar o texto nos primeiros 200 tokens, pois se provou suficiente para nossa classificação de sentimentos.
2. **Criação de DataFrames Processados:** Criamos DataFrames contendo os tokens, máscaras de atenção e rótulos de sentimento para os conjuntos de treinamento e validação.
3. **Conversão para Dataset do Hugging Face:** Convertimos os DataFrames processados em `Datasets` do Hugging Face, que são otimizados para o treinamento de modelos transformadores.
4. **Carregamento do Modelo Pré-treinado:** Carregamos um modelo BERT pré-treinado específico para classificação de emoções. Esse modelo já foi treinado em um grande conjunto de dados para entender a linguagem e identificar emoções.
5. **Fine-tuning do Modelo:** Ajustamos o modelo BERT pré-treinado no nosso conjunto de dados de revisões de filmes (fine-tuning). Durante essa etapa, o modelo é treinado especificamente para a tarefa de classificação de sentimentos (positivo e negativo) usando nossos dados.
6. **Treinamento e Avaliação:** Treinamos o modelo no conjunto de dados de treinamento e avaliamos seu desempenho no conjunto de validação. Calculamos métricas de avaliação como precisão e F1-score para cada época de treinamento, e ajustamos o modelo com base nesses resultados.

Detalhes Técnicos das Implementações

Vamos dividir os detalhes técnicos das implementações abordando cada solução separadamente:

1. **Solução simbólica**

Para a solução simbólica, utilizamos algumas bibliotecas que foram importantes para a resolução do problema:

- NTLK (*Natural Language Toolkit*): Importante para diversas partes do processo, pois é dela que utilizamos o tokenizador, tags de POS (*part-of-speech*), *stopwords* e a *sentiwordnet*.
- Pandas: Utilizada para leitura e manipulações de *DataFrames*
- Matplotlib: Utilizada para visualização das imagens
- Wordcloud: Gera as *wordclouds* a serem visualizadas
- Sklearn: Utilizada para métricas de avaliação dos resultados

2. **Solução estatística/neural**

Da mesma forma, para a solução estatística neural, foram utilizadas algumas bibliotecas:

- NLTK (*Natural Language Toolkit*): Assim como na solução simbólica, utilizada para o tokenizador e, além disso, foi utilizada para lemetizar os tokens.
- Pandas: Utilizada para leitura e manipulações de DataFrames
- Re: Biblioteca de expressões regulares, utilizada no pré-processamento dos textos
- Sklearn: Utilizada para métricas de avaliação e para o uso de modelos de *Machine Learning*
- Matplotlib: Utilizada para visualização de imagens
- Transformers: Necessário para o Tokenizador do BERT e para usar a API da HuggingFace
- PyArrow: Usada para conversão de algumas tabelas
- Evaluate: Utilizada para check de métricas usando a API de treino

3. **HuggingFace**

Para a nossa aplicação final, utilizamos os serviços da HuggingFace, a fim de hospedar nossa aplicação de forma gratuita e não precisar que alguém rode os programas necessários de forma local. Os serviços utilizados foram os seguintes:

- Datasets: Utilizamos esse serviço para disponibilizar o dataset que é lido pela nossa aplicação, que fará o seu pré-processamento adequado e mostrará como a solução neural/estatística e a simbólica avaliaram o problema
- Models: Serviço utilizado para hospedar o modelo que é utilizado na aplicação e que foi treinado na nossa base de interesse, usando a GPU gratuita disponibilizada pelo Google Colab.
- Spaces: É o serviço usado para a aplicação em si, hospedado no link: https://huggingface.co/spaces/danielcd99/IMDB_Reviews

Avaliação dos Sistemas Produzidos

Para avaliação das abordagens estamos utilizando a matriz de confusão e o relatório de classificação que contém métricas importantes como precisão, recall, F1 e acurácia.

A matriz de confusão é uma tabela que é frequentemente usada para descrever o desempenho de um modelo de classificação em problemas de aprendizado supervisionado, como é o caso da análise de sentimentos. É possível identificar na matriz: os Verdadeiros Positivos (TP), que são os casos em que o modelo previu corretamente uma opinião como positiva; os Verdadeiros Negativos (TN), que são os casos em que o modelo previu corretamente uma opinião como negativa; os Falsos Positivos (FP), que são os casos em que o modelo previu incorretamente uma opinião como positiva, mas na verdade era negativa; e os Falsos Negativos (FN), que são os casos em que o modelo previu incorretamente uma opinião como negativa, mas na verdade era positiva.

As métricas consideradas para avaliação são: a precisão que é a proporção de verdadeiros positivos (TP) em relação a todos os exemplos classificados como positivos (incluindo os falsos positivos, FP); a revocação que é a proporção de verdadeiros positivos (TP) em relação a todos os exemplos que são realmente positivos (incluindo os falsos negativos, FN); o F1-score é a média harmônica entre a precisão e a revocação; e a acurácia que avalia o percentual de acertos, ou seja, é a razão entre a quantidade de acertos e o total de entradas.

Assim, para a abordagem simbólica SentiWordNet, temos que:

- A precisão para a classe positiva (0.60) é relativamente menor do que a precisão para a classe negativa (0.66), indicando que o modelo tende a classificar corretamente mais textos negativos do que positivos.

- O recall para a classe positiva (0.75) é maior do que o recall para a classe negativa (0.49), sugerindo que o modelo identifica com mais eficácia os textos positivos do que os negativos.
- O F1-score para ambas as classes é relativamente equilibrado, com uma pontuação mais alta para a classe positiva (0.66) em comparação com a classe negativa (0.57).
- A acurácia geral do modelo é de 0.62, o que significa que ele classifica corretamente cerca de 62% dos textos.

Para a abordagem simbólica *WordCloud*, temos que:

- A precisão para ambas as classes (positiva e negativa) é mais equilibrada, com valores de 0.58 e 0.59, respectivamente.
- O recall para a classe negativa (0.55) é maior do que o recall para a classe positiva (0.62), indicando que o modelo identifica com mais eficácia os textos negativos do que os positivos.
- O F1-score para ambas as classes é mais equilibrado em comparação com o modelo WordNet, com pontuações de 0.57 para a classe negativa e 0.60 para a classe positiva.
- A acurácia geral do modelo é de 0.58, o que significa que ele classifica corretamente cerca de 58% dos textos.

Considerando ambas as abordagens simbólicas têm suas vantagens e desvantagens. A abordagem usando SentiWordNet tende a ser mais precisa na classificação de textos positivos, enquanto a abordagem usando *WordCloud* tem uma precisão mais equilibrada para ambas as classes. Também é possível mencionar que existem palavras em comum em ambas as WordClouds, o que pode estar influenciando na baixa acurácia geral da abordagem.

Para a abordagem estatístico neural serão utilizadas as mesmas métricas. Para a abordagem usando TF-IDF e Naive Bayes, é possível verificar que a precisão e o recall estão variando entre 86 a 87%. A métrica F1-Score combina precisão e recall, possui valor de 86%, o que indica um bom equilíbrio entre precisão e recall. A Acurácia geral do modelo é de 86%, o que significa que ele classificou corretamente aproximadamente 86% de todos os exemplos no conjunto de testes.

O modelo Naive Bayes com vetorização TF-IDF conseguiu alcançar uma precisão, recall e F1-Score bastante equilibrados para ambas as classes, com uma acurácia geral de 86%. Podemos afirmar que o modelo é capaz de fazer previsões precisas em relação ao sentimento das revisões. Assim, podemos afirmar que o modelo estatístico possui um desempenho consideravelmente superior em relação à abordagem simbólica.

Para o modelo BERT, podemos concluir que ele demonstrou um desempenho promissor considerando as métricas avaliadas. Com uma avaliação de perda (eval_loss) de 0.283 indicando uma adaptação eficaz aos dados de validação. A alta acurácia de 88.3% (eval_accuracy) reflete a precisão geral das previsões do modelo, enquanto o F1-score de 88.3% demonstra um bom equilíbrio entre precisão e recall. Assim, o modelo alcançou o melhor desempenho dentre todos os modelos avaliados, superando inclusive a abordagem estatística.

Escolhemos o melhor modelo estatístico neural e o melhor modelo simbólico para testá-los também em uma base similar, a do RottenTomatoes, outro site de avaliação de filmes. Em nossa aplicação, então, mostramos a classificação que esses dois algoritmos realizam em cima dessa base.

Medimos, também, como o BERT (melhor estatístico/neural) e a Wordnet (melhor simbólico) desempenham na base do Rotten Tomatoes: No Wordnet, tivemos 54% de precisão, 55% de recall, 53% de f1 e 54% de acurácia. Já, no BERT, o modelo alcançou uma precisão de 85% e um valor de F1 de 86%.

Exemplo de Execução dos Sistemas

Para todas as abordagens, temos como entrada o arquivo csv contendo os *reviews* do IMDB.

Para esta explicação, estaremos considerando um exemplo de review positivo: “One of the other reviewers has mentioned that after watching just 1 Oz episode you'll be hooked. They are right, as this is exactly what happened with me.

The first thing that struck me about Oz was its brutality and unflinching scenes of violence, which set in right from the word GO. Trust me, this is not a show for the faint hearted or timid. This show pulls no punches with regards to drugs, sex or violence. Its is hardcore, in the classic use of the word.

It is called OZ as that is the nickname given to the Oswald Maximum Security State Penitentary. It focuses mainly on Emerald City, an experimental section of the prison where all the cells have glass fronts and face inwards, so privacy is not high on the agenda. Em City is home to many..Aryans, Muslims, gangstas, Latinos, Christians, Italians, Irish and more....so scuffles, death stares, dodgy dealings and shady agreements are never far away.

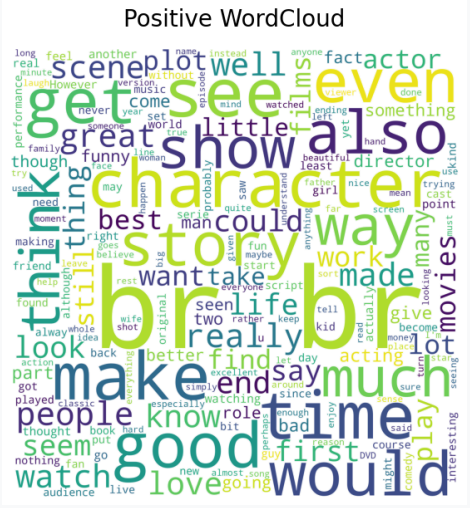
I would say the main appeal of the show is due to the fact that it goes where other shows wouldn't dare. Forget pretty pictures painted for mainstream audiences, forget charm, forget romance...OZ doesn't mess around. The first episode I ever saw struck me as so nasty it was surreal, I couldn't say I was ready for it, but as I watched more, I developed a

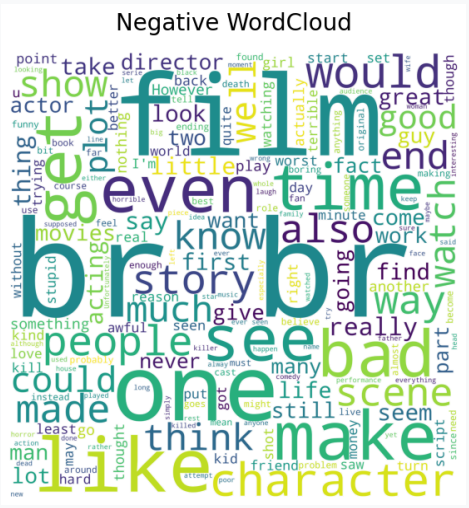
taste for Oz, and got accustomed to the high levels of graphic violence. Not just violence, but injustice (crooked guards who'll be sold out for a nickel, inmates who'll kill on order and get away with it, well mannered, middle class inmates being turned into prison bitches due to their lack of street skills or prison experience) Watching Oz, you may become comfortable with what is uncomfortable viewing....thats if you can get in touch with your darker side."

O código irá realizar a etapa de pré-processamento e em seguida apresentar a classificação considerando a SentiWordNet.

	review	sentiment	WordNet_classification
0	One of the other reviewers has mentioned that ...	positive	negative

Serão geradas as WordClouds, uma para o sentimento positivo e outra para o sentimento negativo.





E depois serão exibidas as classificações considerando a SentiWordNet e a WordCloud com as regras.

	review	sentiment	WordNet_classification	WordCloud_classification
0	One of the other reviewers has mentioned that ...	positive	negative	negative

Um ponto a se destacar são algumas regras implementadas na abordagem simbólica que visam melhorar a classificação, como a identificação da negação. Assim, se existirem *reviews* como “I don't hate that movie” ou “Not bad”, será possível identificar que se trata de um sentimento positivo. Já “I don’t like this movie” é identificado como sentimento negativo. Outra regra é a atenção com a palavra “but”, que indica um contraste de ideias. Nesse caso, consideramos um peso maior para o texto que está depois da palavra “but”.

Para a abordagem estatística considerando TD-IDF e Naive Bayes, apresentamos apenas uma amostragem dos dados, é realizado o treinamento e depois são apresentadas algumas linhas de resultado como na figura a seguir.

	review	original sentiment	predicted sentiment
34622	hard tell noonan marshall trying ape abbott co...	negative	negative
1163	well startbr br one reviewer said know youre r...	positive	positive
7637	wife kid opinion absolute abc classic havent s...	positive	positive
7045	surprise basic copycat comedy classic nutty pr...	positive	positive
43847	josef von sternberg directs magnificent silent...	positive	positive

Para a abordagem neural, treinamos o modelo e apresentamos alguns exemplos de predição utilizando o modelo treinado. Ao avaliar seu desempenho em outra base de dados auxiliar e muito similar, o RottenTomatos, foi obtido 86.08% de f1 score.

Ao final dos notebooks (Simbólico, Estatístico e Neural_Bert) são exibidas as avaliações, com a matriz de confusão e algumas métricas que já foram mencionadas no tópico anterior. No modelo neural é possível visualizar as métricas no *output* do trainer.train().

Instruções de como Compilar/Rodar os Códigos-Fonte

Os códigos fonte estão presentes em três notebooks explicativos no repositório do GitHub, que pode ser acessada através desse link, clonando **o repositório** e acessando a pasta *notebooks_explicativos,* basta executar os notebooks respectivos para cada abordagem. O aprendizado simbólico está no arquivo *Simbolico.ipynb,* a abordagem estatístico está no arquivo *Estatistico.ipynb* e o *fine tunning* do modelo bert está em *Neural_Bert.ipynb*.

Foi desenvolvida uma aplicação nesse mesmo repositório demonstrando o resultado final dos modelos na aba da aplicação, que pode ser encontrada neste link, ele utiliza o modelo Bert que treinamos para a modelagem neural e aplica nossas regras da wordnet, mostrando os dois resultados lado a lado.

Considerações Finais

É possível verificar que a abordagem simbólica requer um tempo de processamento menor, mas a sua implementação de modo a alcançar uma alta acurácia é muito desafiador dependendo do contexto em que está se trabalhando. No caso deste trabalho, em que trabalhamos textos livres escritos por pessoas para fazer reviews de filmes, que podem

possuir diferentes problemas, como erros de ortografia, vagueza, ambiguidade, entre outros, adiciona um nível de dificuldade considerável para ser possível definir um conjunto de regras que consiga abranger todos os possíveis problemas para obter uma alta acurácia.

Já a abordagem estatística possui um bom equilíbrio entre exigência de recursos computacionais e desempenho em relação a acurácia. Apesar de existir um treinamento, o modelo é relativamente rápido e alcança um bom resultado. Em contrapartida, a abordagem neural, apesar do ótimo desempenho, demanda muitos recursos computacionais. Considerando a abordagem inicial proposta, alteramos para não considerar o Word2vec e treinamos o BERT com apenas 1 época. A simplificação foi a solução encontrada para conseguir treinar o modelo apesar da falta de recursos computacionais necessários para realizar um treinamento adequado para obter uma acurácia maior. Apesar dessas limitações, o modelo neural obteve o melhor desempenho dentre todas as abordagens implementadas.