Universidade Federal da Paraíba
Centro de informática

CLASSIFICAÇÃO DE DISCURSOS POLÍTICOS

Disciplina: Processamento de Linguagem Natural

Equipe: Adriel dos Santos Araújo Cabral, Deivison Rodrigues Jordão

Paraíba
2023

Sumário

1. Apresentação do problema

Dado o contexto em que vivemos atualmente, um aspecto muito importante para o dia-a-dia de uma parcela da população, principalmente os que trabalham com jornalismo, redes sociais e no meio jurídico por exemplo, é a política.

Assim, se faria útil a criação de um modelo de processamento de linguagem natural para classificação de posicionamento político em textos que pode ser aplicado em diversos contextos relevantes. Primeiramente, ele seria valioso em ambientes de análise de mídia e jornalismo, permitindo a categorização automatizada de discursos políticos em artigos, discursos e comentários online. Além disso, em plataformas de redes sociais, esse modelo poderia ajudar a identificar e monitorar tendências políticas, facilitando a compreensão das opiniões predominantes. Em cenários políticos mais amplos, o modelo poderia ser empregado para análise de pesquisas de opinião, oferecendo insights sobre a inclinação ideológica da população em relação a diferentes questões. Em resumo, a utilidade desse modelo se estende por diversos setores, contribuindo para uma compreensão mais eficaz e rápida do cenário político em constante evolução.

2. Objetivos

O principal objetivo no desenvolvimento deste trabalho foi estudar o desempenho dos modelos de inteligência artificial, em especial os voltados para processamento de linguagem natural, na tarefa de classificação de conceitos tão abstratos, como um posicionamento político, a partir de um discurso ou texto dado.

3.Dados utilizados

3.1 Coleta

Para que pudéssemos treinar um modelo de inteligência artificial voltada ao processamento de linguagem natural, com o objetivo de classificar o posicionamento ideológico a partir de um instância de texto, precisaríamos de uma quantidade considerável de texto rotulados, em uma quantidade que não seria possível criar de forma manual ou encontrar com facilidade.

Assim, devido a essas dificuldades, foi utilizado um código de web scraping(Código cedido pelo Isaque), o qual tem a capacidade de buscar nos sites de transparência do governo brasileiro os discursos feitos por nossos deputados e senadores na câmara, além de outras informações que seriam de muita relevância para nós, como por exemplo o partido o qual o político, do discurso em questão, está vinculado.

Dessa forma, conseguindo uma boa quantidade de dados, onde no final da coleta de dados, chegamos a ter por volta de 12 mil discursos e suas informações complementares. Dados esses que ainda não estavam prontos para serem utilizados para o treinamento da rede, uma vez que, ainda não havia um rótulo que representasse o posicionamento ideológico do discurso em si. Porém, com alguns pré-processamentos descritos na seção 4, reservada para pré-processamento dos dados, foi elaborada uma forma de rotulá-los.

3.2 Divisão

A divisão dos dados foi feita utilizando a função train_test_split do sklearn, **10%** para teste e **20%** para validação. A divisão foi feita de tal forma que a proporcionalidade das classes seja a mesma para as 3 sub amostragens. Primeiro foi feito o split para os dados de teste e treino-validação e por fim foi feita de fato a divisão entre treino e validação.

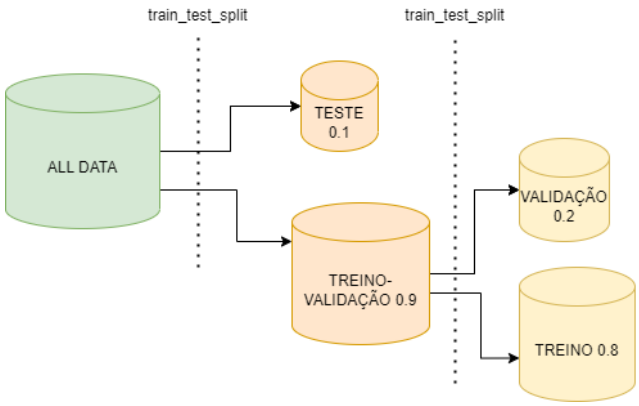

Figura 1: Processo de divisão dos dados.

A quantidade de dados e a proporcionalidade das classes ficou da seguinte forma:

Conjunto de Treino: Conservador (label 0): 3128 exemplos, Neutro (label 1): 1005 exemplos, Progressista (label 2): 4684 exemplos, 8817 no total.

Conjunto de Validação: Conservador (label 0): 782 exemplos, Neutro (label 1): 252 exemplos, Progressista (label 2): 1171 exemplos, 2205 no total.

Conjunto de Teste: Conservador (label 0): 434 exemplos, Neutro (label 1): 140 exemplos, Progressista (label 2): 651 exemplos, 1225 no total

4. Pré-processamento

Como discutido no ponto anterior "**Dados utilizados"**, em nosso dataset não havia nenhuma informação que de fato fosse útil para ser utilizada para rotular e representar de forma satisfatória os discursos para o objetivo estabelecido para esse modelo de linguagem, que seria de classificar o posicionamento ideológico de texto.

Dessa forma, foi feito um tratamento de dados, a fim de separar do texto do discurso, informações que não faziam parte do mesmo e o qual a presença poderia vir a atrapalhar ou enviesar o modelo, como por exemplo nome do orador, seu partido, data, entre outras. Essas informações não foram perdidas, apenas realocadas para outras colunas no dataset.

Assim, ao analisar as informações que se tinha à disposição, foi notado que tínhamos em nossa posse, a informação do partido político do orador vinculado ao discurso, o que nos dá margem para rotular o texto do discurso de acordo com o posicionamento político anunciado publicamente pelo partido.

Para isso, fizemos uma pesquisa sobre o posicionamento político de cada partido presente nos discursos, dessa forma decidimos rotular os mesmos dentro do domínio de três classes, sendo

elas: Conservador, Neutro e Progressista. Os oradores sem partido, como representava uma quantidade ínfima com relação ao total de amostras, foram apenas retirados do dataset.

Baseado nas pesquisas feitas, foi definido que seria usado as seguinte correlação entre os partidos e posicionamento ideológico:

Tabela da relação partido / posicionamento ideológico usado para rotular os discursos(apenas algumas amostras)

MDB	NEUTRO
NOVO	CONSERVADOR
CIDADANIA	PROGRESSISTA
PSD	NEUTRO
PT	PROGRESSISTA
UNIÃO	CONSERVADOR
REPUBLICANOS	CONSERVADOR
PSOL	PROGRESSISTA

Tabela 1:Exemplos de alguns partidos e seu posicionamento ideológico.

Por fim, após uma análise das informações disponíveis concluímos que as seguintes features eram irelevantes ou nocivas para a confiabilidade do modelo(como por exemplo o partido do orador, uma vez que o rótulo foi inferido dele). Assim, ficamos com as seguintes features para utilizarmos para o treinamento do modelo:

	discurso	label
	Sra. Presidente, Sras. e Srs. Deputados, gosta...	Progressista
	No correr da vida, isto que o poeta diz que, à...	Progressista
	Sra. Presidente, o PV quer chamar a atenção pa...	Progressista
	Sr. Presidente Arthur Lira, Sras. Deputadas e ...	Progressista
	Então, se se tem realmente a intenção de defen...	Progressista
	...	...

Figura 2: Exemplo com 5 linhas de como ficou o dataset para o treinamento.

5. Arquitetura da rede neural

Para a classificação dos discursos, foi utilizado o modelo DeBERTaV3 pré-treinado com uma parte do corpus em português do C4 database. Os detalhes sobre como foi feito o pré-treino estão disponíveis neste link do HuggingFace.

O modelo possui 12 camadas, cada uma com 12 attention heads. Para efetuar a classificação, foi adicionado outra camada denominada “head”. Também mudamos

“attention_probs_dropout_prob” e “hidden_dropout_prob” para 0, os quais são, respectivamente, dropout aplicado ao attention tensor e aplicado as camadas Linear (Feedforward).

A saída da layer-11 possui dimensão (batch size, context length, hidden size), então pegamos esse tensor e o transformamos para uma dimensão (batch size, hidden size) para que possa ser utilizada pela camada Linear (Feedforward) com uma entrada de 768 e saída 3, as quais seriam cada classe do nosso problema.

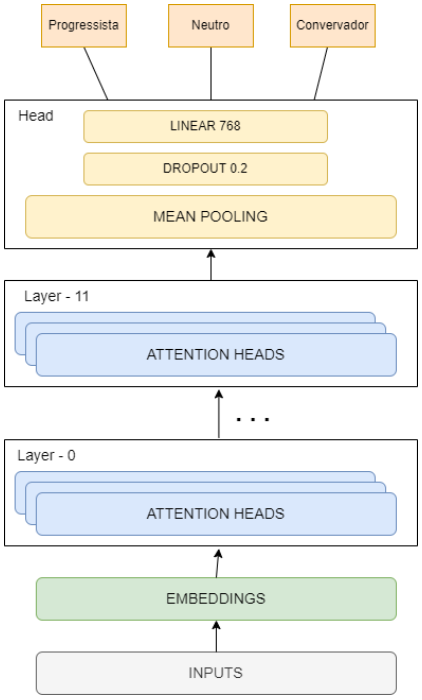

Figura 3: Explicação em alto nível de como funciona a arquitetura do modelo.

6. Treinamento (fine-tuning)

O fine-tuning foi feito utilizando: Adamw, CrossEntropyLoss, Linear schedule e float16 para acelerar o tempo de treinamento, diminuindo de 42 minutos para 18 minutos. Os ajustes de alguns hiperparâmetros foram feitos a fim de melhorar os resultados nos dados de validação e por fim rodamos o código nos dados de teste.

6.1 Hiperparâmetros

Os seguintes hiperparâmetros foram utilizados no fine-tuning: learning rate: **1*10-5**, minimum learning rate: **0**, weight decay: **0.01**, context length: **256** tokens, epochs: **5**, batch size: **20**, warmup steps: **265**.

Apenas os hiperparâmetros: batch size, learning rate, warmup steps foram ajustados.

6.2 Grouped parameters

Como explicado na seção de arquitetura, o modelo possui 12 camadas mais a camada de classificação. Dessa forma, utilizamos uma estratégia diferente para ajustar os parâmetros durante o treinamento.

Fizemos uma divisão do modelo em grupos, cada um associado a uma taxa de aprendizagem diferente. As camadas iniciais têm uma taxa ligeiramente menor para um ajuste mais delicado, enquanto as camadas mais distantes possuem uma taxa de aprendizagem maior. Decidimos usar essa técnica, pois não tínhamos muitos dados para o fine-tuning e queríamos preservar mais o que foi aprendido nas camadas iniciais.

A divisão foi feita da seguinte forma: Embeddings: lr * **0.89**, layer(**0-2**): lr * **0.91**, layer (**3-5**): lr * **0.93**, layer**(6-8)**: lr * **0.95**, layer**(9-11)**: lr * **0.97** e a camada de classificação com a taxa de aprendizagem padrão.

épocas	treino (loss)	treino (acurácia)	validação (loss)	validação (acurácia)
1	0.9367	0.5344	0.8976	0.6201
2	0.5986	0.7284	0.5797	0.7476
3	0.4086	0.8353	0.5202	0.7868
4	0.2760	0.8984	0.5352	0.7941
5	0.1913	0.9365	0.5620	0.7886

Tabela 2: Resultados da loss e acurácia para os dados de treino e validação durante o treinamento.

É perceptível que a partir da época 3 o modelo começa a melhorar bastante nos dados de treinamento, mas não tem uma melhora significativa nos dados de validação. Na época 4 essa diferença se torna ainda maior e já dá o indício de que teríamos um overfitting nas próximas épocas, o que ocorre de fato na 5.

7. Resultados

O modelo utilizado para chegar a esses resultados foi gerado na época 4, a qual foi a última em que houve uma melhora na acurácia nos dados de validação:

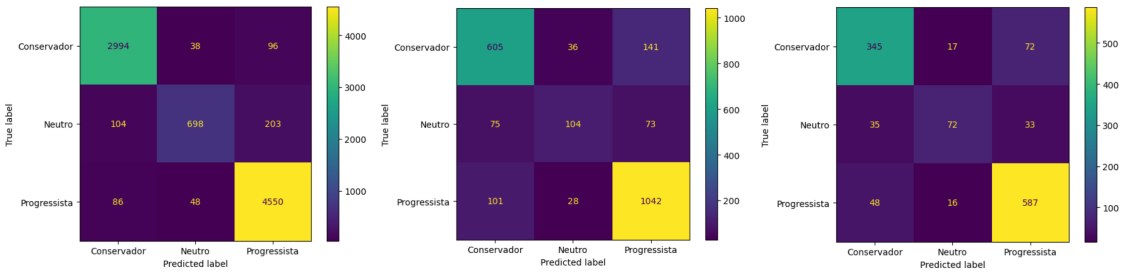

Figura 4: Matrizes de confusão geradas para os dados de, respectivamente, treino, validação e teste.

	TREINO	VALIDAÇÃO	TESTE
Loss	0.2003	0.5352	0.4991
Acurácia	0.9347	0.7941	0.8195

Tabela 3: Resultados obtidos utilizando a função que usa o modelo em modo de avaliação, o que explica o resultado diferente para o treino em relação à tabela anterior.