Relatório - EP3

Piero Conti Kauffmann (8940810)

1 Modelos utilizados

1.1 Encoder-Decoder BiLSTM

Pelas especificações da tarefa, o primeiro modelo proposto para o problema de geração automática de títulos deve ser uma rede neural encoder-decoder com uma LSTM bidirecional ($\overrightarrow{g_e}$ e $\overleftarrow{g_e}$) como encoder acoplada a um mecanismo de atenção (Figura 1.1).

Na componente do decoder do modelo, escolhi incluir duas LSTMs unidirecionais em sequência. A primeira LSTM recebe o embedding do último token escrito no título, e é inicializada com os estados finais da rede bidirecional concatenados, portanto possuí o dobro de *hidden units* de $\overrightarrow{g_e}$ e $\overleftarrow{g_e}$. Além disso, é na primeira LSTM do decoder em que é feito o cálculo dos vetores de contexto por meio do mecanismo de atenção do modelo, descrito adiante. Esses vetores são concatenados aos *hidden states* da LSTM e são passados para a LSTM final como input, que finaliza a decodificação do próximo token da sequência.

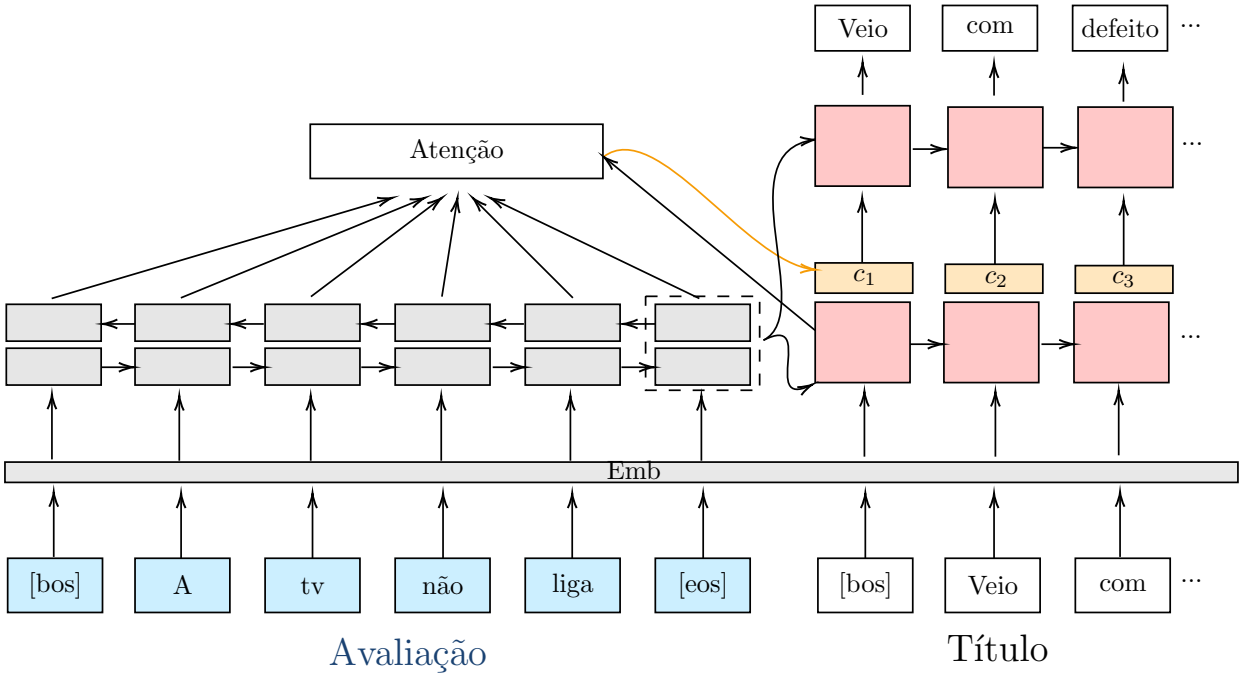

Figura 1: Diagrama do modelo encoder-decoder BiLSTM com camada de atenção escolhido. Os tokens especiais `[bos]` e `[eos]` delimitam respectivamente o início e fim das sequências de texto.

O mecanismos de atenção adotado para o modelo foi o mecanismo de atenção global proposto por Luong, Pham e Manning [1] com escores de atenção obtidos a partir do produto interno dos *hidden states*. Graças à segunda LSTM do decoder, o modelo também é capaz de utilizar a informação dos vetores de contexto dos tokens passados.

1.2 BERT-CLS e BERT-MASK

Neste trabalho iremos experimentar com duas variantes de soluções baseadas no BERT (representado na Figura 1.2) para geração de textos de maneira autoregressiva. A primeira variante consiste em utilizar o campo de classificação (token `[CLS]`) de um modelo BERT pré-treinado para a língua portuguesa (Souza, Nogueira e Lotufo [2]) para prever o próximo token de uma sequência.

A camada final associada ao token `[CLS]` do BERT é pré-treinada com a tarefa de *Next Sentence Prediction* (NSP), que consiste em tentar adivinhar se a segunda sentença fornecida (separada pelo token `[SEP]`) vem depois da primeira sentença em um texto corrido. Para o problema em questão, podemos descartar a camada densa de classificação binária usada para o objetivo de NSP e criar uma nova camada densa com a mesma dimensão final do vocabulário de saída, e utilizarmos essa nova componente para prever o próximo token do título de maneira autoregressiva. Chamaremos esse modelo de BERT-CLS.

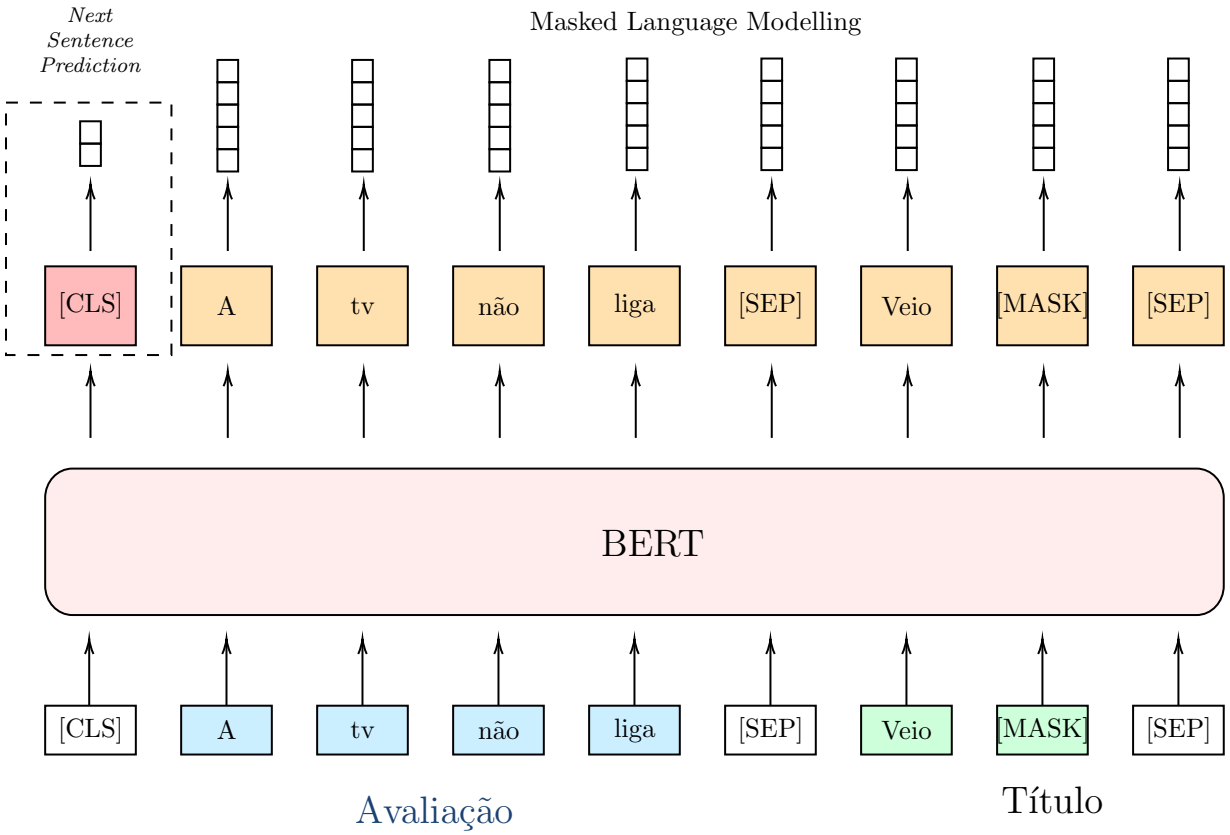

Figura 2: Diagrama ilustrativo do BERT para a task compartilhada de *Masked Language Modelling* e *Next Sentence Prediction*.

Alternativamente, podemos tentar aproveitar a semelhança do objetivo deste trabalho com a tarefa de *Masked Language Modelling* (MLM) do modelo pré-treinado, que tenta recuperar os tokens mascarados aleatóriamente nas sentenças. Apesar de parecer a melhor alternativa, existem algumas desvantagens aparentes: no objetivo de geração de texto deste trabalho, vamos sempre adicionar o token `[MASK]` no final da segunda sentença, que estará sempre seguido do token final `[SEP]`. Naturalmente, isto irá fazer o modelo original acreditar que o token mascarado é sempre um token próximo ao final de sentença (um caractere de ponto final, por exemplo) e irá prejudicar a performance do texto que iremos gerar de maneira causal autoregressiva. Ao fazermos o *fine tuning* segundo esta abordagem, estamos grosseiramente tentando converter um modelo treinado com a tarefa de *Masked Language Modelling* para um modelo causal de sumarização de textos.

É difícil de prever de antemão se o BERT-MASK produzirá resultados melhores ou piores que o BERT-

CLS (proposto originalmente pelo enunciado da tarefa), então, a título de curiosidade, decidi avaliar a performance destas duas abordagens diferentes.

2 Treinamento

2.1 Encoder-Decoder BiLSTM

O treinamento da encoder-decoder BiLSTM é feito utilizando a técnica *teacher forcing* (Williams e Zipser [3]), que passa a sequência correta inteira de tokens para o decoder que tenta prever o próximo token relativamente a cada item da sequência passada. Essa técnica na prática acelera a convergência de modelos sequenciais e usualmente também facilita a implementação destes modelos.

Treinamos o modelo com *early-stopping* em uma amostra de 72% do conjunto total de dados, validando o modelo no final de cada época em um conjunto de validação que corresponde a 8% dos dados. O treinamento do modelo é interrompido se o custo calculado no conjunto de validação não diminuir em 5 épocas consecutivas. Quando o treinamento é interrompido, apenas o modelo com menor custo no conjunto de validação é salvo.

2.2 BERT-CLS e BERT-MASK

Seguindo o procedimento de preparação dos dados descrito no enunciado do exercício, utilizamos a mesma amostra de 80% dos dados extraidos para o treinamento da LSTM. Ao dividir a geração de texto token a token, o processamento destes dados produz uma amostra com mais instâncias. Por limitações computacionais, utilizei apenas 2/3 destes dados durante apenas uma única época de treinamento. Por sua vez, como a função de custo foi calculada apenas em dados que não foram antes vistos pelo modelo, não foi necessário utilizar *early-stopping* para previnir *overfitting*.

O modelo BERT-CLS convergiu de maneira estável, sem precisar de um tamanho de *batch* muito elevado, finalizando após cerca de 20h de treinamento no *Google Colab*. O modelo BERT-MASK exigiu maior esforço de treinamento e teve convergência difícil, o que é justificável, visto que alteramos a tarefa base em que o modelo foi treinado originalmente. Para garantir estabilidade no treinamento do BERT-MASK foi necessário utilizar um tamanho de *batch* elevado com *gradient accumulation* para não exceder a capacidade da infraestrutura disponível. A configuração exata do treinamento segue reproduzida na tabela a seguir

3 Resultados experimentais

Neste trabalho, foram feitos 6 experimentos diferentes segundo as combinações dos hiperparâmetro da camada de *Dropout* (0%, 25% e 50%) entre as duas modalidades de LSTM (unidirecional e bidirecional). Os resultados registrados em treinamento nos conjuntos de validação e treino são apresentados nas Figuras 2 e 3.

Verificamos dos gráficos de treinamento que os modelos de LSTM unidirecionais apresentaram problemas graves de *underfitting*, que impediram o custo de diminuir significativamente em 100 época de treinamento. Isso é esperado, dado que foram feitas simplifcações no modelo, como utilizar uma dimensão baixa na camada de *embeddings* e congelar o treinamento dos mesmos. Uma alternativa para aumentar a complexidades destes modelos seria incluir mais uma camada recorrente unidirecional logo após a primeira.

Para as LSTMs bidirecionais, observamos o comportamento de overfitting descrito no enunciado. Após cerca de 25 épocas, o custo de validação começou a crescer para os três modelos. Verificamos também que a presença de *Dropout* reduziu a diferença entre o custo de treinamento e o custo de validação, conforme o esperado.

Referências

[1] Minh-Thang Luong, Hieu Pham e Christopher D Manning. "Effective approaches to atib tention-based neural machine translation". Em: *arXiv preprint arXiv:1508.04025* (2015).

[2] Fábio Souza, Rodrigo Nogueira e Roberto Lotufo. "BERTimbau: pretrained BERT models for Brazilian Portuguese". Em: *9th Brazilian Conference on Intelligent Systems, BRACIS, Rio Grande do Sul, Brazil, October 20-23 (to appear)*. 2020.

[3] Ronald J Williams e David Zipser. "A learning algorithm for continually running fully recurrent neural networks". Em: *Neural computation* 1.2 (1989), pp. 270–280.