Neural Language Model - POS Tagging

Pedro V. Brum

[1]Universidade Federal de Minas Gerais (UFMG) – Belo Horizonte, MG – Brazil

pedrobrum@dcc.ufmg.br

1. Introdução

Nesse trabalho foi construído um modelo de *POS-Tagging* utilizando os dados disponibilizados, que consitem de sentenças em português. A tarefa de *POS-Tagging* consiste em categorizar cada token de uma sentença. Nesse trabalho, podemos considerar um token como uma palavra ou sinais de pontuação. Ou seja, dada uma sentença, consideramos tokens que não são palavras, como aspas e sinais de pontuação. Porém, antes de tudo podemos fazer um preprocessamento das sentenças, retirando as pontuações e caracteres especiais. Para implementar o trabalho foi utilizada a linguagem python e o pacote $Keras$.

2. Método

Para construiur um modelo de *POS-tagging* podemos utilizar uma rede LSTM (Long Short Memory Term). A rede recebe uma sentença (sequência de palavras) e atribui para cada token da sentença uma tag. A entrada pode ser processada de diferentes maneiras. Assim, para construir o modelo de *POS-tagging* a partir de uma LSTM podemos pensar em duas abordagens. A primeira abordagem é atribuir a cada palavra do vocabulário um índice, e a partir desses índices, formar a representação vetorial de uma sentença atribuindo a cada token o seu índice correspondente. A segunda abordagem é formar os *embeddings* das palavras que formam as sentenças. Dessa forma, uma sentença pode ser representada como uma lista de *embeddings*. A segunda abordagem é a que foi utilizada neste trabalho. Na figura 1 podemos observar um diagrama do modelo utilizado para a tarefa de *POS-Tagging*.

3. Análise experimental

Na figura 2 podemos observar a variação do valor de acurácia ao longo do treinamento do modelo para o conjunto de treino e para o conjunto de validação, quando consideramos a *tag* pontuação. Para o conjunto de treino, a acurácia aumenta até a última época. Dessa forma, é provável que a acurácia continuaria aumentando se o modelo fosse treinado por mais épocas, até atingir o valor 100%. Por outro lado, para o conjunto de validação, a acurácia aumenta até a época 3, aproximadamente.

Na figura 3 podemos observar a variação do valor de perda ao longo do treinamento do modelo para o conjunto de treino e para o conjunto de validação, quando consideramos a *tag* pontuação. Para o conjunto de treino, a perda diminui até a última época. Dessa forma, é provável que a perda continuaria diminuindo se o modelo fosse treinado por mais épocas, até atingir o valor 0. Por outro lado, para o conjunto de validação, a perda diminui até a época 3, aproximadamente.

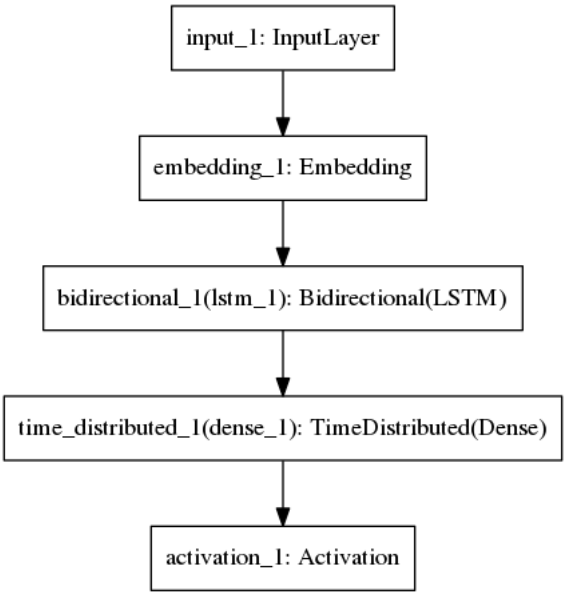

Figure 1. Diagrama do modelo utilizado para tarefa de *POS-Tagging*.

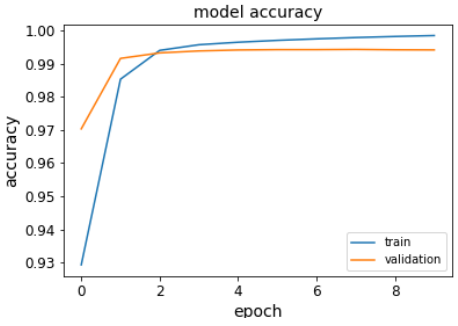

Figure 2. Valores da acurácia para o conjunto de treinamento e para o conjunto de validação a cada época, quando consideramos a *tag* pontuação.

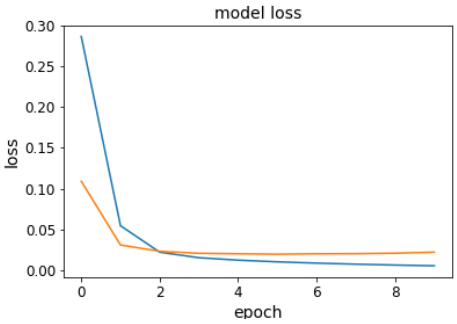

Figure 3. Valores de perda para o conjunto de treinamento e para o conjunto de validação a cada época, quando consideramos a *tag* pontuação.

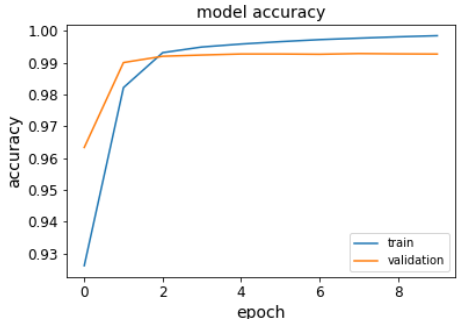

Figure 4. Valores da acurácia para o conjunto de treinamento e para o conjunto de validação a cada época, quando não consideramos a *tag* pontuação.

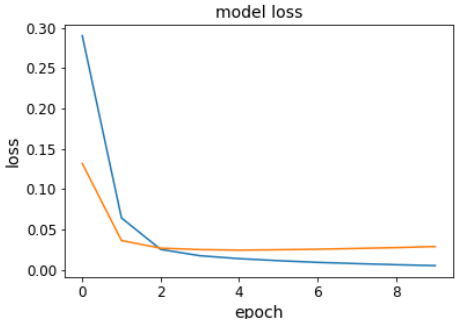

Figure 5. Valores de perda para o conjunto de treinamento e para o conjunto de validação a cada época, quando não consideramos a *tag* pontuação.

Portanto, quando consideramos a *tag* correspondente à pontuação, o modelo se comporta como esperado, oferecendo bons valores de acurácia e de perda ao fim do treinamento. Além dos ponto comentados, podemos ressaltar que o modelo atinge bons valores de acurácia e perda em poucas épocas. Dessa forma, poderíamos treinar o modelo por menos épocas se o tempo de execução fosse um critério muito importante.

Na tabela 1 podemos observar os valores de acurácia para cada uma das *tags* presentes na base de dados. A precisão do modelo é maior para as *tags* PREP+PROADJ, CUR e PU, e menor para as *tags* IN, PREP+PROSUB e PREP+ADV. Ou seja, o modelo erra mais vezes para as classes gramáticais relacianas as *tags* IN, PREP+PROSUB e PREP+ADV. Podemos notar que o modelo funciona bem para as *tags* PREP+PROKS, PREP+PROPESS, PREP+ART e PREP+PROADJ, mas não funciona bem para PREP+PROSUB e PREP+ADV. Dessa forma, é provável que o modelo erre mais vezes ao identificar preposição + pronome substantivo e preposição e advérbio. Também é possível notar que para as *tags* em que a precisão é menor existe poucas instâncias. De qualquer forma, o modelo funciona bem no geral, resultando em uma boa precisão para a maior parte das *tags*. No conjunto de de teste, o modelo resultou uma acurácia igual a 0.9245.

Quando não consideramos a *tag* pontuação os valores de acurácia e de perda se comportam de forma similar a quando consideramos. Nesse caso, no conjunto de teste, o modelo resultou uma acurácia igual a 0.9115.

4. Conclusão

Neste trabalho foi implemntado um modelo para a tarefa de *POS-Tagging*. O modelo constiste em um uma LSTM (Long Short Term Memory). O modelo funciona bem no geral, oferecendo uma alta precisão para a maior parte das classes gramáticais. Além disso, temos que os resultados quando consideramos a *tag* de pontuaçãp (PU) e quando não consideramos são bem próximos. Porém, quando não consideramos, os valores finais de precisão ficam um pouco melhores.

Table 1. Valores de acurácia para cada *tag*, quando consideramos a *tag* pontuação (PU).

tag	accuracy	# words
IN	0.4388	98
PREP+PROSUB	0.5705	156
PREP+ADV	0.6667	30
ADV-KS	0.7130	230
PROSUB	0.7738	1565
NPROP	0.7779	15936
KS	0.7879	2537
PDEN	0.8233	1092
PCP	0.8266	3640
ADJ	0.8488	8550
ADV	0.8649	5439
NUM	0.8917	2540
PRO-KS	0.9057	2194
PROADJ	0.9268	3417
PREP+PRO-KS	0.9310	58
N	0.9482	36529
V	0.9485	19709
PREP	0.9546	16773
PROPESS	0.9562	2875
PREP+PROPESS	0.9603	126
KC	0.9722	4530
ART	0.9851	12577
PREP+ART	0.9889	10214
PU	0.9918	26894
CUR	0.9966	296
PREP+PROADJ	0.9968	309

Table 2. Valores de acurácia para cada *tag*, quando não consideramos a *tag* pontuação (PU).

tag	accuracy	# words
IN	0.3163	98
ADV-KS	0.5957	230
PREP+ADV	0.6000	30
PREP+PROSUB	0.6859	156
PREP+PRO-KS	0.7069	58
PROSUB	0.7213	1561
NPROP	0.7759	15618
KS	0.7832	2537
PCP	0.8180	3615
PDEN	0.8402	1089
ADV	0.8418	5430
ADJ	0.8578	8457
NUM	0.8788	2516
PRO-KS	0.9038	2194
PREP+PROPESS	0.9286	126
PROADJ	0.9312	3415
V	0.9323	19670
N	0.9408	36121
PROPESS	0.9485	2874
PREP	0.9564	16773
KC	0.9713	4530
ART	0.9723	12576
PREP+ART	0.9883	10213
PREP+PROADJ	0.9935	309
CUR	0.9966	296