Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Recurrent Neural Networks

Pietro Epis, Michele Milesi, Anna Valanzano
Master's Degree in Artificial Intelligence, University of Bologna
{ pietro.epis, michele.milesi, anna.valanzano}@studio.unibo.it

Abstract

POS tagging automatically assigns POS categories to words according to their contextual information in a sentence. We solved the POS tagging task with different architectures based on Bidirectional Recurrent Neural Networks. In particular, we started from a baseline architecture composed of a Bidirectional LSTM and a Dense Fully connected layer and then we experimented with some modifications:(1) substituting the Bidirectional LSTM layer with Bidirectional GRU layer (2) adding an LSTM layer (3) adding a dense layer. After hyperparameters tuning, we selected the most promising architectures.

1 Introduction

1.1 Techniques

Neural networks learn a relationship between words and POS tags during their training and use this relationship to predict POS tags of words. While traditional neural networks consider all words in a sentence to be independent of each other, Recurrent Neural Networks (RNN) keep a hidden state that represents all the previous words in the sentence. Bidirectional Recurrent Neural Networks (BRNN) allow the networks to have both backward and forward information about the sequence at every time step. RNN suffers from the vanishing gradient problem, for which the hidden state of a word is influenced more by words near it than words far away. In other words, simple RNN does not have a "long-term memory". As the long-term dependencies between words are fundamental in POS tagging, we considered two variations of simple RNN (LSTM and GRU), that alleviates the vanishing gradient problem by having a forget gate layer to decide which words to "remember" and which words to "forget" (Xue and Zhang, 2021).

1.2 Dataset and Experiments

The dataset is composed of a set of documents which in turn are divided into several sentences. We structured the original dataset (available at this link) as a DataFrame. In our DataFrame for each word we store the corresponding POS tagging, the split (training, validation, or test), and the membership document.

2 System description

We started from a **Baseline Model**, a two layers architecture composed of a Bidirectional LSTM with 128 units and a Dense Fully connected layer with 45 outputs (that is the number of POS tagging classes). We defined three modified version of these architectures:

- the **GRU Model**, in which the Bidirectional LSTM layer is substituted with a Bidirectional GRU layer;
- the **Double LSTM Model**, which has an additional Bidirectional LSTM layer;
- the **Dense Model**, which has an additional dense layer.

We accurately show the two architectures with best performances in Figure 1 and in Figure 2.

3 Experimental setup and results

We performed hyper-parameter tuning for all four architectures, varying the number of epochs, the number of units of the network, the size of mini-batches and applying the dropout technique to regularize training. Moreover, we tried two different optimizers, `RMSProp` and `Adam` and we observe that the second one leads to better performance. After testing the four architectures, we selected only two architectures looking at the accuracy on the validation set. We report the performances on the validation set for the most promising combinations

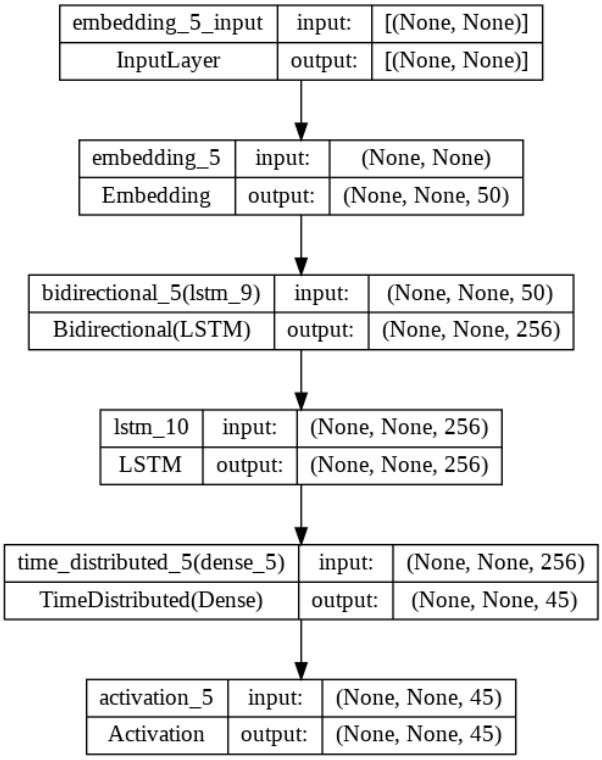

Figure 1: Dense Model architecture

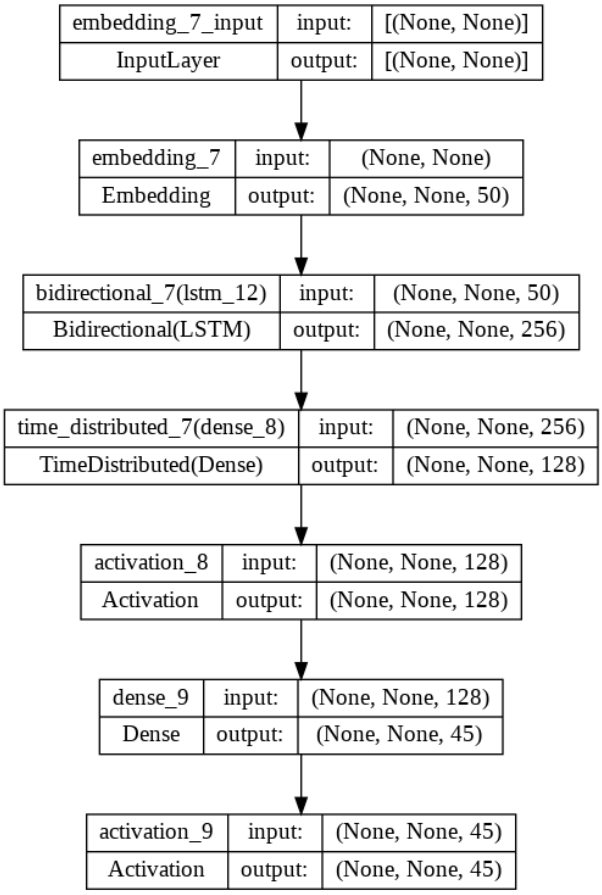

Figure 2: Double LSTM Model architecture

of hyper-parameters in Table 1, Table 2, Table 3, Table 4.

units	mini-batch	dropout	accuracy
100	32	0.2	0.8639
128	16	-	0.8805

Table 1: Validation performance of the Baseline Model

units	mini-batch	dropout	accuracy
100	64	-	0.8735
128	16	-	0.8757

Table 2: Validation performance of GRU Model

units	mini-batch	dropout	accuracy
100, 64	32	-	0.8685
128, 128	32	0.2	0.8787

Table 3: Validation performance of Dense Model

4 Discussion

4.1 Quantitive Results

Relying on the metrics and the experiments reported in Section 3 we can conclude that:

- the Baseline Model reaches about 88% of accuracy;
- the GRU Model reaches about 87% of accuracy;
- the Dense Model reaches about 88% of accuracy;
- the Double LSTM Model reaches about 88% of accuracy.

As the performances of the four models are very similar, we decided to take as the best models the two that seem to generalize better on the validation data (looking at the difference between validation and training accuracy). It turns out that our best models are the Double LSTM Model and the Dense Model. In Table 5, we report their performances on the test set (obtained with the best combination of hyper-parameters). The performances on the test set decrease with respect to those on the validation set, but the results are coherent: the Double LSTM Model achieves better results (even if only slightly).

units	mini-batch	dropout	accuracy
50, 100	32	-	0.8714
128, 256	32	0.2	0.8800

Table 4: Validation performance of Double LSTM Model

Model	f1-score
Double LSTM	0.7177
Dense	0.6937

Table 5: Performance of the two best models on test set

4.2 Error Analysis

We performed an error analysis looking at which are the most confusing classes and which classes are the most confusing to each other. We report some of these results in Table 6. First of all, we can observe that some classes have an f1-score equal to 0 (RP, LS, RBR, PDT, SYM) as they are very less frequent in the dataset. Indeed, the class SYM has only 1 instance, PDT 8 instances, LS 13, RBR 42, RP 60.

The class NNPS (proper noun plural) is very confused (with 0.22 f1-score) with NNP (proper noun) and NNS (noun plural). It does make sense because proper and nonproper nouns have the same function in a sentence, but only a slightly different meaning. They syntactically differ only in the initial capital letters, which are lost due to the preprocessing.

The class WDT (wh-deter) is highly confused with NN (sing noun) and VBP (verb non-3sg present). The classes WDT and VBP are very different from a syntactic and meaning point of view, but this error can be warranted by the fact that the class WDT is not so frequent in the dataset. Instead, the confusion between WDT and NN makes more sense, because these classes are more similar (for instance, they both can be used as subject in a sentence).

The tags VBN, VBP, VBG (representing verbs with different tenses) are confused by each other: this is understandable as these words differ only by the ending of the word, but have the same meaning and function.

5 Conclusion

As explained in Section 1, LSTM and GRU architectures similarly learn the relationships among the words and the corresponding POS tagging. However, we expected the LSTM architecture to be

Tag	Meaning	f1	confused with
RP	particle	0	RB, IN
LS	list item	0	NNP, CD
RBR	comparative adverb	0	JJ, JJR
PDT	predeterminer	0	JJ DT
SYM	symbol	0	CC
NNPS	proper noun plural	0.22	NNP,NNS
WDT	wh-deter	0.64	NN VBP
VBN	verb past ten	0.68	JJ, VBD
VBP	verb non-3sg present	0.72	NN, VB
VBG	verb gerund	0.76	VBN NN

Table 6: Mostly misclassified tags

more accurate than the GRU one, which uses fewer training parameters. Contrary to what we expected, on this dataset, the models reached similar performances. In addition, we expected to improve the Baseline Model's performance by adding a Dense layer or an LSTM layer. However, these modifications did not give significant enhancements. Regardless of the architecture, we observed that most of the errors occur for similar classes and this makes even more difficult the attempt of improving the model. However, as the model misclassifies lots of tags that are not so frequent in the dataset, the use of a bigger dataset, or of a dataset containing more instances of the misclassified classes, may highly increase the performance of the model.

References

Xiaorui Xue and Jiansong Zhang. 2021. Part-of-speech tagging of building codes empowered by deep learning and transformational rules. *Advanced Engineering Informatics*, 47:101235.