Assignment 1

Antonio Lopez, Edoardo Merli, Matteo Vannucchi and **Alessandra Blasioli**
Master's Degree in Artificial Intelligence, University of Bologna
{ antonio.lopez2, edoardo.merli, matteo.vannucchi, alessandra.blasioli }@studio.unibo.it

Abstract

This study conducts a comparative analysis of some RNN architectures in the context of Part-Of-Speech (POS) tagging, to investigate their performance. Specifically, we focused on architectures composed of Bidirectional LSTMs and dense layer(s) on top. We observed the model composed of a 2-layer Bidirectional LSTM and a dense layer to perform best among those in the analysis, reaching a macro F1 score of 0.8267 in the validation set and of 0.83524 in the test set.

1 Introduction

The objective of this assignment was to conduct Part-of-Speech (POS) tagging on the Penn Tree-Bank Corpus dataset (Marcus et al., 1993) using deep learning models.

To tackle POS tagging, "deep learning methods have shown much better tagging performance than the machine learning-oriented methods in terms of learning features by themselves. But these methods are more complex and need high computing resources." (Chiche and Yitagesu, 2022).

Our analysis considers three different combinations of Bi-LSTM (Huang et al., 2015) and dense layers for the architectures. The metric used for model evaluation is the macro F1 score, excluding punctuation.

2 System description

We decided to work directly with embeddings instead of defining an Embedding layer. The dense embeddings we used are GloVe (6B tokens) embeddings (Pennington et al., 2014). Initially, every token in the corpus dataset is converted to lowercase, since the GloVe vocabulary used is uncased. The mapping from word tokens to embeddings is not done upfront of training, but computed on each batch before feeding the embeddings to the models in an on-demand manner. We handled OOV tokens and the '[UNK]' token by sampling, for each one of the dimensions of the GloVe embeddings, from a different normal distribution centered around the respective mean and standard deviation. The sequences are padded before being fed to the models.

We defined three different model architectures:

- Bidirectional LSTM + dense layer (baseline)
- 2 layer Bidirectional LSTM + dense layer (model 1)
- Bidirectional LSTM + dense layer (model 2)

Figure 1 shows diagrams of the network architectures. We implemented these models in PyTorch, using a framework called PyTorch Lightning (Falcon and The PyTorch Lightning team, 2019).

3 Experimental setup and results

The dataset is split into training (documents 1-100), validation (documents 101-150), and test set (documents 151-199). Different values for the hidden layer dimension in the (first) dense layer were tested, before settling for 256, while the embedding dimension was set to 300.

In Model 2 we used again 256 as the second dense layer's dimension and a ReLU activation function between the two dense layers. This choice was motivated by the fact that, since a linear layer implements a matrix multiplication, two consecutive linear layers without any activation function in between do not increase the expressiveness of the model.

We trained using Adam optimizer with an initial learning rate of 10^{-3} and a step decay schedule. Early stopping was employed, with a maximum of 100 epochs (all models reached convergence around the 30th epoch) and a batch size of 64.

The metric used to evaluate our model is the macro F1 score, excluding punctuation. Figure 2 shows plots of the learning curves during training.

Model	Validation F1	Test F1
Baseline	0.7967 ± 0.0208	
Model 1	**0.8267 ± 0.0058**	0.83524
Model 2	0.8133 ± 0.0153	

Table 1: F1 scores on validation (with standard deviation computed over three seeds) and test set (only for the best performing model)

We tested with three different seeds (6, 90, 157) for robust estimation. The best performing model turned out to be Model 1, as we can see from Table 1. The test set performance is computed on the best model instance across the three trained for Model 1, i.e. seed = 60.

4 Discussion

The results highlight Model 1 as the best-performing model, with a mean macro F1 score of 0.8267 across the three seeds on the validation set. This is a significant increase with respect to the baseline performance, being more than a standard deviation above it.

The F1 score computed on the test set confirms the quality of the validation results and the robustness of model 1, yielding an even higher metric value.

The error analysis highlights several error scenarios. First, the most confused POS tags are respectively:

- NNP, confused with NN with 122 errors
- NN, confused with JJ with 96 errors
- JJ, confused with NN with 78 errors

where: NN = "noun, singular or mass", NNP = "proper noun, singular" and JJ = "adjective".

From this, it's clear that the model struggles most with singular nouns and their distinction with proper nouns and adjectives.

The proper noun confusion, in particular, can be justified by the fact that we convert all the tokens in lowercase as a preprocessing step. This makes it harder to distinguish between proper nouns and common nouns. This capitalization problem influences also the distinction between nouns and adjectives, as we can see from an example sentence presented in Table 2 below:

The model is fed the sentence "old spaghetti warehouse ...", which justifies its errors in interpreting these tokens as adjectives and common nouns due to the missing capital letters.

token	ground_truth	prediction
Old	**NNP**	**JJ**
Spaghetti	**NNP**	**JJ**
Warehouse	**NNP**	**NN**
rose	VBD	VBD
1	CD	CD
to	TO	TO
16	CD	CD
1\/8	CD	CD
.	.	.

Table 2: example sentence 1

It's also worth mentioning that sometimes the ground truth labels are misleading/wrong, as in the example sentence below:

token	ground_truth	prediction
GRAINS	**NNPS**	**NNS**
AND	**NNP**	**CC**
SOYBEANS	**NNPS**	**NNS**
:	:	:

Table 3: example sentence 2, CC stands for coordinating conjunction

This is the title of an article, but the words "GRAINS" and "SOYBEANS" should be labeled as proper nouns, and, even worse, the conjunction "AND" should be a proper noun.

We quantified these capitalization errors to be 310 out of 1212 total errors, the 25.58% of errors.

5 Conclusion

In this report, we tackled the problem of POS tagging using RNN architectures and GloVe word embeddings. Model 1 exhibited superior performance on both validation and test sets compared to the other models. However, the main challenges we detected were related to the presence of capitalized tokens in the dataset and misleading ground truth labels, which caused ambiguities leading to errors. The lack of capitalization handling, in particular, is one of the main limitations of our model, with 25.58% of errors being in capitalized words. Using cased embeddings (e.g. GloVe cased 840B tokens) seems like a natural and promising direction to address this weakness.

6 Links to external resources

- GitHub repository: Repository
- Models weights: Drive

References

Alebachew Chiche and Betselot Yitagesu. 2022. Part of speech tagging: a systematic review of deep learning and machine learning approaches. *Journal of Big Data*, 9(1).

William Falcon and The PyTorch Lightning team. 2019. PyTorch Lightning.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Comput. Linguist.*, 19(2):313–330.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

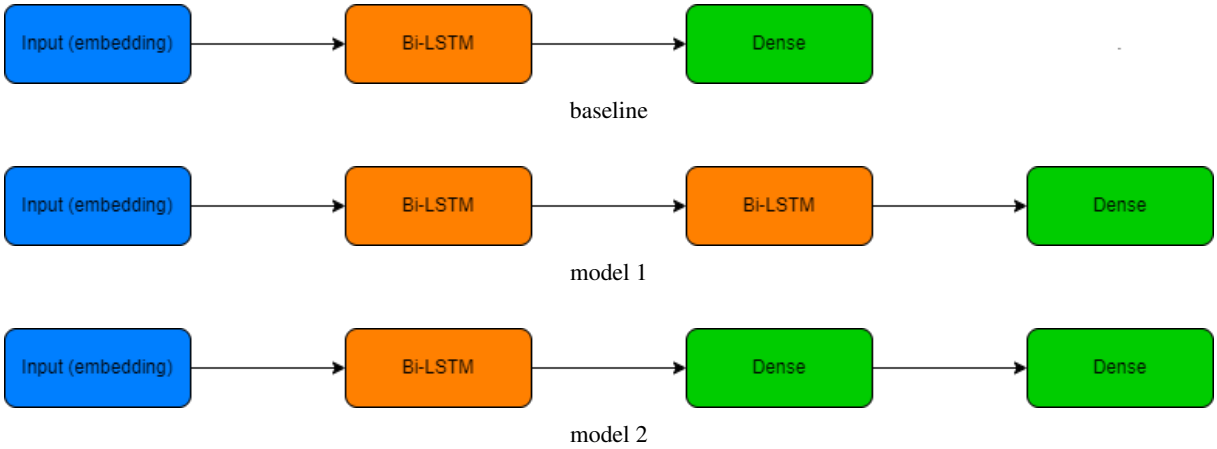

Figure 1: Diagrams of the three networks architectures implemented

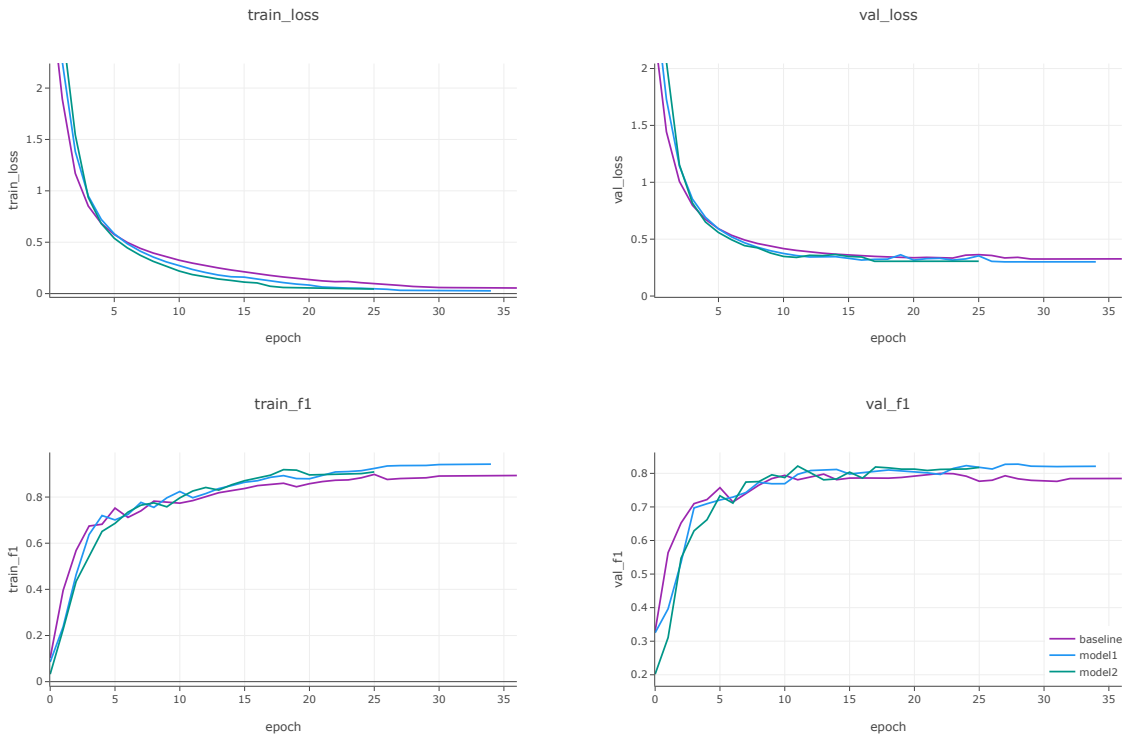

Figure 2: Learning curves for the three models. Loss and F1 score are plotted against epoch for training and validation set.