# NLP Assignment 1: POS Tagging
## NLP Course Project

**Alberto Luise, Angelo Quarta** and **Edoardo Fusa**
Master's Degree in Artificial Intelligence, University of Bologna
{ alberto.luise, angelo.quarta, edoardo.fusa }@studio.unibo.it

## Abstract

In studies of Machine Learning, especially of Natural Language Processing, the presented models are often backed up by huge datasets, sometimes spanning multiple languages. However, it's not often that the behaviour of a relatively small model is analyzed in a limited and controlled environment.

In this report we'll tackle the classification task of POS-tagging on a specific dataset: we'll try to train multiple models with a small depth on a fixed set of sentences related to the business world. We will see what are the main challenges of this tasks, what techniques could be employed in order to overcome them and what results can be obtained.

## 1 Introduction

**Part-Of-Sentence Tagging** is a word-to-word classification task in the wide field of Natural Language Processing, and as such lots of works have been published with very impressive results (Jacobsen et al., 2021), with multiple models achieving more than 96% of accuracy on a huge variety of classes. These studies often train their models on very big datasets, hence why they are able to achieve such results. In this work, however, we want to focus on the limitations of having only a handful of text at our disposal, to better understand how a model behaves during training and how a proper Training Set should be built.

We were given a dataset composed of 200 sentences, of which only 100 could be used for training, whereas in one of the more common sets for POS tagging a total of 69.000 sentences are present across multiple languages (Nivre et al., 2020). We built a pipeline that trained each model multiple times with different random seeds, each time resetting its weights to whatever they initially were, in order to average the results and remove results fluctuation.

We trained our models both with a single word at a time and with full sentences, using the Macro F1 Score as a metric for evaluation, to better emphasize the underlying issues in our task. Our results were satisfying anyway, since we managed to surpass 90% accuracy on two different models and get very close to an F1 Score of 0.8; much more than that would've been almost impossible, and we'll soon see why.

This shows that even in an unfavorable environment, Natural Language Models can prove to be very resilient and reliable.

## 2 System description

We have used a total of 4 different models:

- **Baselines**: Two simple Sequential architectures composed of a Bidirectional LSTM layer and a Dense layer, with the only difference being the type of input they accept; both of them process sequences, but one works word-by-word (iterating over the elements of their embedding vectors; 1 feature, length 100) while the other reads one sentence at a time, iterating over the words (100 features, variable length). In order to fit every sentence into a single Array, they were padded with empty vectors, to make them constant in size, and a Masking layer was added on top of the model to filter out all the non-word vectors fed to it.
- **Improvements**: After the baselines training, we tried to improve the best performing model (the sentence-to-sentence one) by adding to it an additional layer: we tried both an additional LSTM layer with double the units and an additional Dense layer.

  The main challenge for these models will be to generalize on all classes, even on those with very few training samples: we're dealing with a fixed, imbalanced dataset. The best performing model's architecture can be seen in Figure 1.

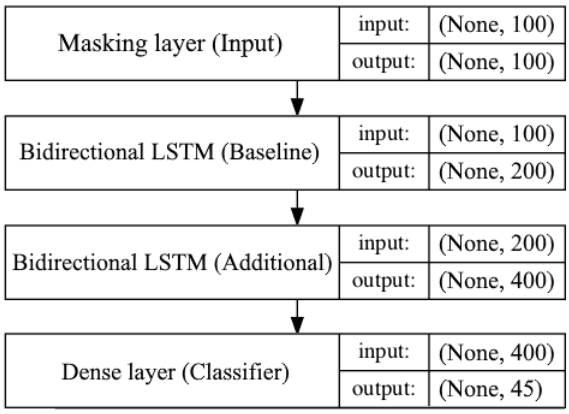


Figure 1: Best model's architecture

## 3 Experimental setup and results

For every model, the training pipeline consists in three random restarts with fixed seeds in order to average the results and eliminate variance. After training both baselines, the full-sentences approach proved to be superior, with a huge increase in accuracy and a small margin of F1 Score improvement. After that, we extended this baseline with an additional layer, choosing between an additional LSTM or an additional Dense layer.

- The double LSTM model got a 90.7% Validation Accuracy after 10 epochs and a Macro F1 Score of 0.765, with only 5 POS classes getting an individual score below 0.5;
- The double Dense model got a much higher 92.1% Validation Accuracy, but a slightly lower Macro F1 Score of 0.763, with 8 classes below 0.5. Since our reference metric is the F1 Score, the first model was chosen for further testing.

Lastly, we performed a bit of fine-tuning on the best-performing model, by boosting the worst classified POS tags in the Validation Set using **Class Weights**. We loaded the best checkpoint into the model's weights, performed fine-tuning for 20 epochs (aborted after 13 due to early stopping) and got interesting results: while there was no improvement in Validation Accuracy, the Macro F1 Score got boosted to 0.796, leaving only the classes that did not appear at all in the Training Set with an individual F1 Score of 0 (yes, there were actually two of those).

## 4 Discussion

A score of 0.796 would not usually be a very good result, but we're actually content with that due to the imbalanced nature of the dataset:

correctly classifying POS labels that appear less than 20 times in the Training Set is a task that maybe only a Large Language Model with ad-hoc prompting could do; therefore, since the F1 Score treats all classes equally, these extremely hard classes weight down the evaluation greatly. Considering that the first tested baseline got a Macro F1 Score of 0.690, this is actually a very good improvement, with **more than 0.1 total points** gained.

The Dense model, even if it was discarded, got a higher Validation Accuracy: this is probably because it focused on correctly identifying the majority classes, disregarding the smaller ones (as can be seen by the high number of totally unrecognized labels in the double digits).

The two classes that never appeared in the Training Set, -LRB- and -RRB-, are of course recognized by no model at all. Furthermore, the remaining low scores on some classes of the final model could also be due to Out-of-Vocabulary words: Last names, technical terms and so on are replaced with a static embedding, leaving no room to generalization. In the end, we can recognize two main problems with the task: mainly an imbalanced dataset, with a Training Set not representative of the Test Set and a huge imbalance in class magnitude, and secondarily pre-trained embeddings that, without proper fine-tuning, make certain classes very hard to distinguish.

(Note that the first problem is much more prominent than the second, since no magic embedding could teach a standard Sequential model to recognize a never-before-seen class.)

## 5 Conclusion

Overall, the takeaway message is probably this: Natural Language Processing is a very well-explored domain of Machine Learning, and even a small model can achieve good results on problems with a lot of background. However, Data preparation is key in this kind of tasks, and the unbalanced dataset is really what holds us back.

Since it's the fifth time I said it in this report, it's probably true that *"Data Scientists spend 80 percent of their time dealing with data preparation and the other 20 percent talking about it"*.

However, if limited datasets are unavoidable, the best approach would most likely be Transfer Learning, i.e. training a model on a much bigger Training Set and fine-tune it for the task at hand (Dione et al., 2023).

## 6 Links to external resources

The weights of the trained models can be found, alongside the code, in our GitHub repository.

## References


Cheikh M. Bamba Dione, David Adelani, Peter Nabende, Jesujoba Alabi, Thapelo Sindane, Happy Buzaaba, Shamsuddeen Hassan Muhammad, Chris Chinenye Emezue, Perez Ogayo, Anuoluwapo Aremu, Catherine Gitau, Derguene Mbaye, Jonathan Mukiibi, Blessing Sibanda, Bonaventure F. P. Dossou, Andiswa Bukula, Rooweither Mabuya, Allahsera Auguste Tapo, Edwin Munkoh-Buabeng, victoire Memdjokam Koagne, Fatoumata Ouoba Kabore, Amelia Taylor, Godson Kalipe, Tebogo Macucwa, Vukosi Marivate, Tajuddeen Gwadabe, Mboning Tchiaze Elvis, Ikechukwu Onyenwe, Gratien Atindogbe, Tolulope Adelani, Idris Akinade, Olanrewaju Samuel, Marien Nahimana, Théogène Musabeyezu, Emile Niyomutabazi, Ester Chimhenga, Kudzai Gotosa, Patrick Mizha, Apelete Agbolo, Seydou Traore, Chinedu Uchechukwu, Aliyu Yusuf, Muhammad Abdullahi, and Dietrich Klakow. 2023. Masakhapos: Part-of-speech tagging for typologically diverse african languages. *ACL Anthology*.

Magnus Jacobsen, Mikkel H. Sørensen, and Leon Derczynski. 2021. Optimal size-performance tradeoffs: Weighing pos tagger models. *ArXiv*.

Joakim Nivre, Marie Catherine de Marneffe, Filip Ginter, Jan Haji, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, and Francis Tyers Daniel Zeman. 2020. Universal dependencies v2: An evergrowing multilingual treebank collection. *Proceedings of the Twelfth Language Resources and Evaluation Conference*.