Homework 1: Named Entity Recognition

Mattia Pannone
pannone.1803328@studenti.uniroma1.it

Abstract

In the few last year the field of Artificial Intelligence has made great strides, especially the sub-field related to deep neural networks. This allow the improvements of many tasks such as the Natural Language Processing (NLP). NLP is an important task because there exists many different of spoken languages and a simple text (or spoken sentence) of each one can assume different meaning depending on the context. So if this feature, sometimes, can trick a human, for a computer is very hard deal with human languages, so were developed different techniques for Natural Language Processing and there are still many ongoing studies.

1 Introduction

This paper refers to a specific sub-task of the NLP: **Named Entity Recognition (NER)**. NER is a natural language processing technique that can automatically scan entire sentences and pull out some fundamental entities in a text and classify them into predefined categories. Entities may be for example person names, location names, company names, etc. So NER is a kind of information extraction from which (together other techniques) a computer can learn to define the context of a text and understand it.

In this paper this task is done following a specific schema of NER, that is che **BIO** schema, it stands for **Beginning**, **Inside** and **Outside**. Beginning means that the tag is at the beginning of a tag ('B-' prefix before the tag), Inside indicate that the tag is inside a chunk ('I-' prefix before the tag), Outside indicates that a token belongs to no entity/chunk ('O-' prefix before the tag). So, for this study we have a set of sentences and the goal is to predict the right BIO-tag for each word of every sentence. For example in the sentence *Robert was born in New York*, the model have to predict "Robert" as "I-PER", New as "B-LOC", "York" as "I-LOC" and all other as "O".

The following paragraphs will be structured like this: an overview on the format of data and they organization for the task, the related work that will explore the different neural network and NLP practices used, then the evaluations on the results obtained and finally the conclusions.

2 Data Organization

First of all it is important to organize data because from them depend a major part of learning of a neural network. In particular each sentence, or better each word, must be encoded in a way that can be used to train a neural network; there are different ways to encode each word of a vocabulary in a list of feature (tensors) that differentiate between them, ,so that can be represented in a feature space. One way is to train an own embedding environment, however I opted for a pre-trained word-embedding, which is more accurate and more fast and allow to concentrate on the NER task. In particular I used GloVe (Global Vector for word representation) which have different pre-trained dictionaries and I used one with 400K of words with a representation of the words with a 300-dimensional features each one. Hoping that all words of dataset were included in the vocabulary, I added for each missing word a "unk" token encoded with a fixed tensor of 300 of all "1".

Since this is a classification task, in this work we have 13 classes which correspond to NER tags: "PER" (person), "LOC" (location), "GRP" (group), "CORP" (corporation), "PROD" (product), "CW" (creative work), each one that can have a prefix of "B-" or "I-", and the final class "O". I encoded this classes in a one-hot encoding notation so we have for each word a tensor with dimension 13 and where an index represent a class (0-12) the relative value represent the probability that those word belong to that class.

Furthermore, since every sentence have a different length but a neural network model need to take

as input data with same shape, I added the padding so to have every sentence as long as the longest sentence in the dataset. I encoded the "pad" words with a fixed tensor of 300 of all "0" in data and I added "PAD" as a 14th class which will not be included in the training procedure.

3 Related Work

In this section I will show the different models with which I tried to reach high performance. Since there are many methods I focused in particular on 3 types of models: the fist use a **BiLSTM** layer plus a classifier, the second is the same model but adding a NLP method that is **POS (Part-Of-Speech) tagging** and the third add an **CFR (Condition Random Field)** layer. Each model trained with different hyperparameter tuning.

3.1 BiLSTM + Classifier

This simple model use a kind of Recurrent Neural Network, that is a type of artificial neural network commonly used in speech recognition and natural language processing, capable to recognize data's sequential characteristics. In particular the layer used is LSTM (Long Short Term Memory) which is explicitly designed to avoid the long-term dependency problem, remembering information for long periods of time, so this allows to model the behavioe of different NER tags in a sentence. Moreover it is used a particular implementation of the LSTM which is the Bidirectional LSTM, it works making any neural network to have the sequence information in both directions backwards (future to past) or forward(past to future) so that a sentence can be analyzed in both verses. The output of BiLSTM is passed to a linear classifier which can predict 13 probabilities for each word. The linear classifier is made by 4 linear layer and dropout layer.

3.2 Adding POS

POS tagging is the process to assign to each word a particular part of speech based on both its definition and its context, for example a word can be identified as nouns, verbs, adjectives, adverbs, etc. As in the case of word embedding there are pre-trained pos tagger, I used nltk library to assign a POS tag to each word. I created also a dictionary with all the POS tags used and I encoded each of them with random tensors (I chosen 100 feature for each tag). First trial was to train the previous network only with POS tags as data and corresponding NER tags as labels. Seen the poor performance, I tried to concatenate each POS tag to its corresponding words so to have more information in the feature space of the words, information related also to the part of speech of the words.

3.3 Adding CRF layer

A Conditional Random Field is a standard model for predicting the most likely sequence of labels that correspond to a sequence of inputs. In particular this layer takes as input the output of BiLSTM + classifier, so the likelihoods of each word of the sentence and the true labels associated and construct a transition matrix where are indicated the likelihoods of passing from a NER tag to another, this for all tags (in this case we have 14x14 matrix including the padding as class). For this reasons this layer should help the classifier and improve the performances of the model.

4 Training and Evaluation

Evaluation of different models are made with the F1 score that is the harmonic mean of precision and recall, this metric can be better with respect to accuracy because an high accuracy do not mean necessarily good performance in that, as shown in Figure 1 and Figure 2 there an imbalanced distribution of data with respect to class. In particular there are two kind of F1 used, one from skelearn library that include all 13 classes and the other from seqeval library that considers classes without prefixs "B-", "I-" and "O". Table 1 shows some results of different trials. The model that use BiLSTM plus a classifier reach good performance after 35 epochs and improve a bit but very slowing augmenting the number of epochs, figures (3, 4) refers to the model trained with 35 epochs, figures (5, 6) refers those trained with 100 epochs. An important and relevant trial is done with the same model where the LSTM is not bidirectional where both F1 metrics result lower, this show the importance of bidirectional LSTM. Instead training the model using also POS tags the performance are not good (Figure 7), probably need more work to improve to improve architecture and data organization for POS tagging. From the result with the model that use also the CRF layer it would seem that performance are not improved so more, however this model reach the result shown with less epochs so the total performance are improved (Figures 8, 9, 10).

All models are trained using a batch size of 256, different trials shown that changing batch size, performance does not change much. Models use the Cross Entropy Loss, except that with the CRF, where the minimization is done on the loss returned by CRF layer for training, while for the validation data I defined a loss function which compute the error rate (this matter doesn't require the gradient and the optimization step, this explain the graph of different trend of losses in Figure 11).

5 Conclusions

In this homework I tried to implement and train a model that was able to classify words with the respective named entities. The first important task to do this is organize dataset to collect features that are useful for the classifier model. It should be useful have a vocabulary with many other words to resolve the miss of words problem. Then there are many other possible approaches for example improve the POS tagging task to support NER, improve the use of CRF, mix the models together to take advantage from different features, also should be useful deal with data imbalance and with the padding used.

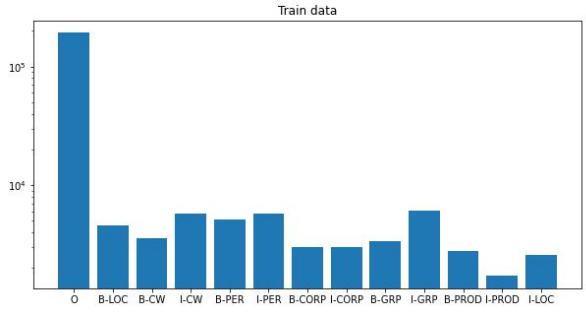

Figure 1

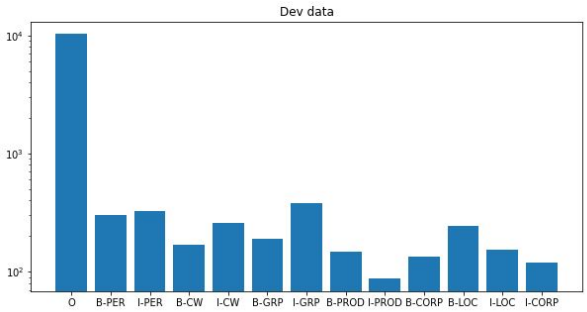

Figure 2

References

1.Understanding LSTM Networks: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

2.Complete Guide To Bidirectional LSTM (With Python Codes): https://analyticsindiamag.com/complete-guide-to-bidirectional-lstm-with-python-codes/

3.Exploring Conditional Random Fields for NLP Applications: https://hyperscience.com/tech-blog/exploring-crfs-for-nlp-applications/

4.Pytorch-crf: https://pytorch-crf.readthedocs.io/en/stable/torchcrf.CRF.forward

Model	F1 sklearn	F1 seqeval
BiLSTM + classifier , 35 epochs	0.71	0.58
BiLSTM + classifier , 100 epochs	0.73	0.61
LSTM (No Bi) + classifier, 35 epochs	0.43	0.46
BiLSTM + classifier with POS, 50 epochs	0.31	0.17
BiLSTM + classifier + CRF, 50 epochs	0.73	0.63

Table 1: F1 scores.

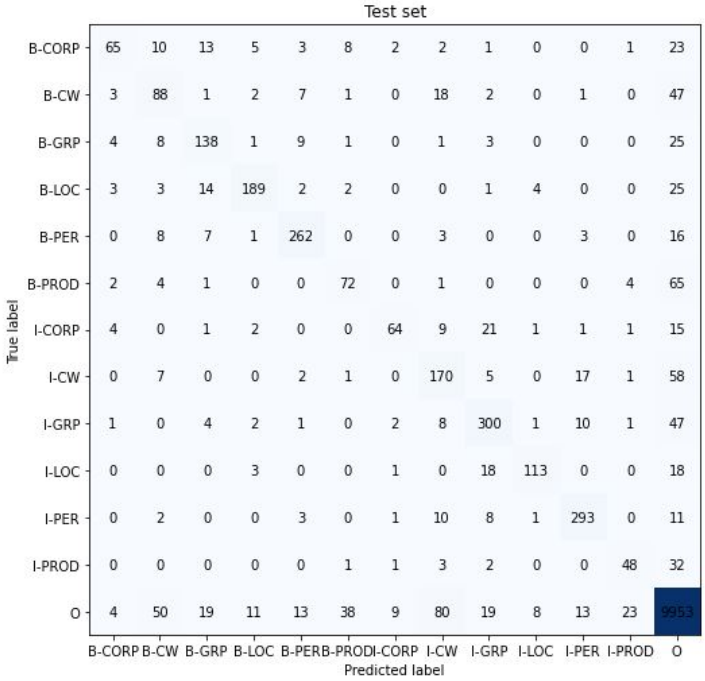

Figure 3

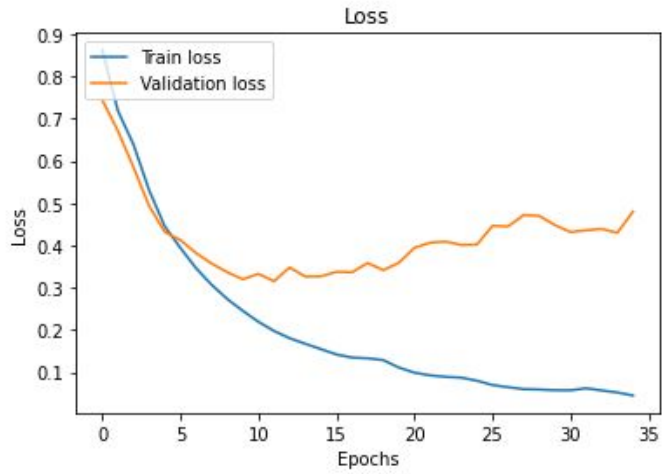

Figure 4

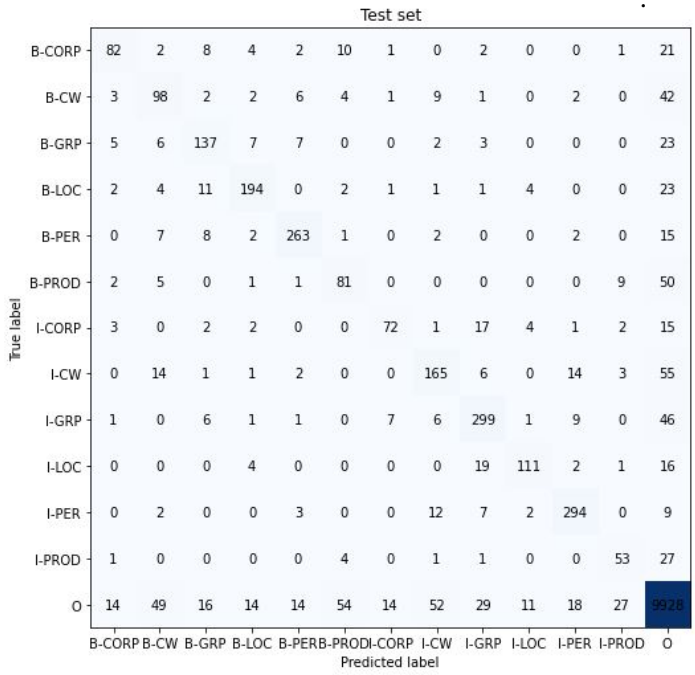

Figure 5

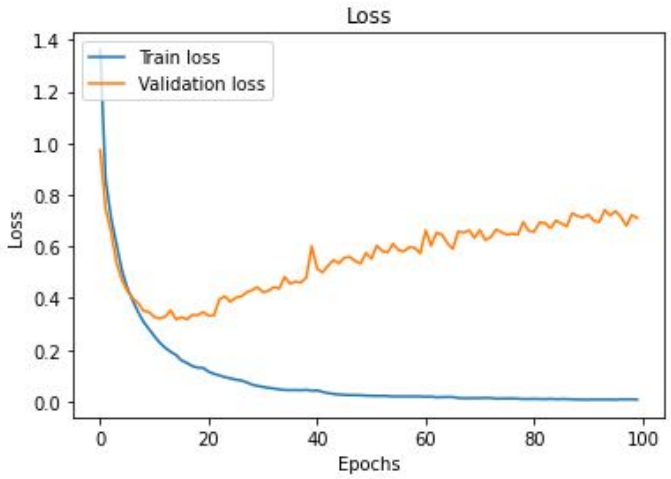

Figure 6

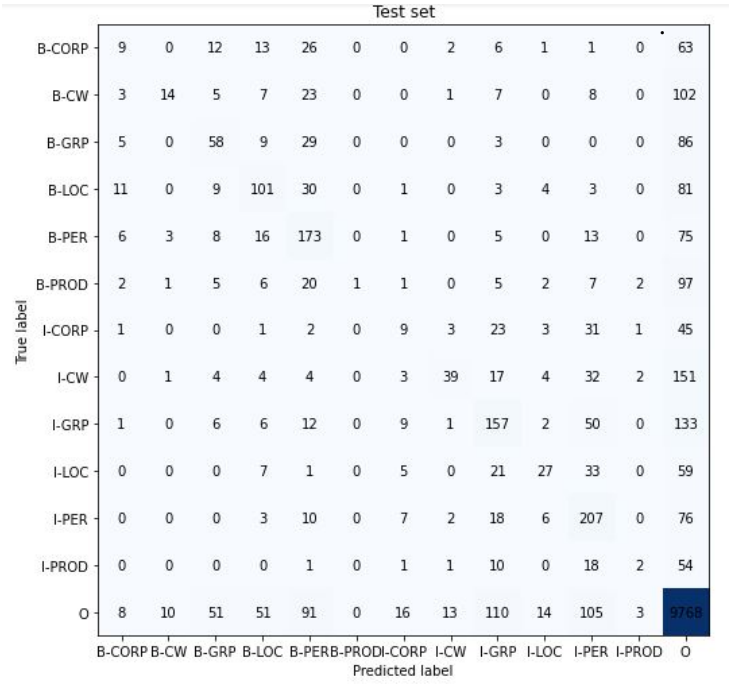

Figure 7

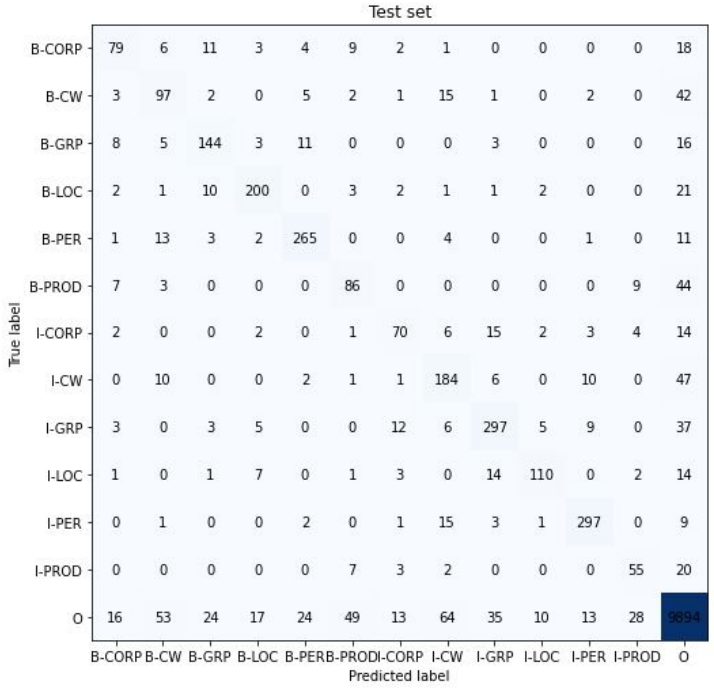

Figure 8

Classification Report

	precision	recall	f1-score	support
B-CORP	0.65	0.59	0.62	133
B-CW	0.51	0.57	0.54	170
B-GRP	0.73	0.76	0.74	190
B-LOC	0.84	0.82	0.83	243
B-PER	0.85	0.88	0.86	300
B-PROD	0.54	0.58	0.56	149
I-CORP	0.65	0.59	0.62	119
I-CW	0.62	0.70	0.66	261
I-GRP	0.79	0.79	0.79	377
I-LOC	0.85	0.72	0.78	153
I-PER	0.89	0.90	0.89	329
I-PROD	0.56	0.63	0.59	87
O	0.97	0.97	0.97	10240
accuracy			0.92	12751
macro avg	0.73	0.73	0.73	12751
weighted avg	0.93	0.92	0.92	12751

Figure 9

F1 score: 0.6268581169488764

	precision	recall	f1-score	support
CORP	0.56	0.53	0.54	133
CW	0.44	0.51	0.47	170
GRP	0.62	0.66	0.64	190
LOC	0.79	0.78	0.78	243
PER	0.81	0.86	0.83	300
PROD	0.47	0.51	0.49	149
micro avg	0.65	0.68	0.66	1185
macro avg	0.61	0.64	0.63	1185
weighted avg	0.65	0.68	0.66	1185

Figure 10

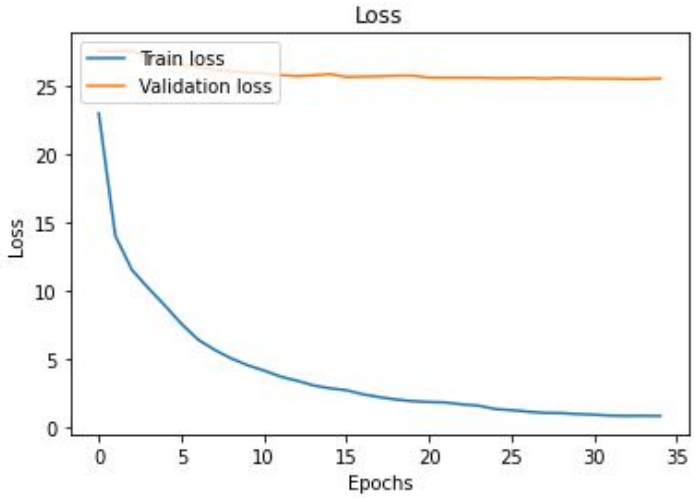

Figure 11