An easy approach to NER for event detection

Bruno Iannone `iannone.1883472@studenti.uniroma1.it`

Abstract

The given task has as its goal the words classification based on their position, so it makes sense to use an LSTM. The report has the will to show how far a bilateral LSTM can be pushed to achieve the best results by using only the given dataset. This means, for example, avoiding the use of architectures like GloVe. The best result obtained on the test set among all the experiments resulted in an F1 of 0.721 on the test set.

1 Introduction

Named entity recognition (NER) state-of-the-art models have some pretty high scores. Bi-LSTM models can achieve very high F1 as we read in "Named Entity Recognition with Bidirectional LSTM-CNNs" (Chiu and Nichols, 2015) that achieved 91.62 on CoNLL 2003 (English) dataset, they get some improvements by using "Conditional random fields" (CRFs) as we read in "Deep contextualized word representations" (Peters et al., 2018) with an F1 of 92.22 on CoNLL 2003 (English). Must be noted that these models aren't so much outperformed from trasformers as "FLERT: Document-Level Features for Named Entity Recognition" model got 94.09 on CoNLL 2003 (English) dataset (Schweter and Akbik, 2020). These models used sophisticated word embeddings like GloVe to achieve these results, but on a very limited dataset as the given one, was possible to retrieve a vocabulary of a little more than 30k words, and using a such large embedding on a such small vocabulary would lead to bad performances as the most part of it won't be fine tuned during training. So the goal was to squeeze the model as much as possible with what was learned during the lectures in order to obtain the best results.

2 Experiments

Due to the nature of the task, the use of a bilateral LSTM (Bi-LSTM) seemed the most appropriate: the usage of this type of model allow the information flow to go from the past to the future and from the future to the past, making the model more robust for our task which strongly relies on words position, so a many to many LSTM is used. After the LSTM a linear layer was connected to it to map predictions to labels space. For the training phase, early stop with pacience was used in a slightly modified version: if the new F1 calculated from the epoch validation phase is lower than the previous one, patience is reduced but, if a new maximum F1 is reached, than the model is "forgiven" and it regains all chances. At every epoch, if the F1 is a new maximum, the model is saved to ensure that, at the end of the training, the saved weights are the best possible. For the experiments Adam optimizer was chosen as it behaved better than SGD (an example of a train with SGD is visible in Figure 1), with a batch size of 4096. In the very first experiments the embedding dimension was equal to 32, the Bi-LSTM hidden dimension was equal to 64, the number of layers was 2 and the learning rate was equal to 0.001. As 100 epochs weren't enough (it needed more than 150), the learning rate was moved to 0.01 which sped up a lot the training phase needing only about 50 epochs (and giving better results) as Figure 2 shows. From the previous plot is clear that F1 reached a stable point, so dropout was introduced in the embedding layer, in the LSTM itself and in the linear layer. From tuning these parameters F1 moved from 0.680 to 0.704 by setting the three values to 0.5,0.8,0.2 (see Figure 3) resulting in an F1 on the test set equal to 0.717. As increasing the number of layers didn't achieve anything and resulted in a worse model that needed more than 130 epochs to train (Figure 4), it was time to try different dimensions of embedding and hidden layers: the higher results obtained were with 25 and 150. This resulted in a F1 score of 0.702 that scored 0.721 on the test set. Unfortunately, due to lack of computational power, was impossible to go further, so the very last thing tried was to retrieve the word embedding from this last result and use it in the model so that it could just focus on the task without having to learn the embedding too. This, sadly, didn't gave any relevant results in terms of scores, but was a notable speed up to the training, as the F1 started to raise from the very first epochs as is possible to see from Figure 5.

Conclusions

The results of the chosen model are represented in the confusion matrix in Figure 6. From this plot we can observe that the model is pretty confident on the event with the "B"s label and even more by detecting "O"s which is predictable as they are the most frequent type of tokens in the sentences. Unfortunately there are problems with "I"s tokens, as we can see in the dataset, the "I"s tags are very few: there are only 30 "I-POSSESSION" samples and only 24 "I-SENTIMENT". As a general rule we can say that almost always, "I-TYPE"s tags are preceded by the same "B-TYPE" token, while the viceversa isn't true and much more often "B-TYPE"s are followed by "O"s: for example "I-ACTION" tokens are 457 and 441 are preceded by "B-ACTION", while more than 20000 times "O"s are preceded by B-ACTION so, statistically, the model is more incline to predict an "O" after a "B-ACTION" rather than an "I-ACTION", indeed we can see that "I-ACTION" has a 0.40 % to be predicted as "0". One strange result is the in "I-SCENARIO" which follow the same rule above, but has a score of 0.96, an explanation about this could be find in the fact that "I-SCENARIO" tokens having this label are often the same. So apart from the model which is, in the end, very simple, the task is harder because of this unbalancing occuring in the train set that makes the results worse. Speaking about parameters, the model has, in the end, 1.509.612 trainable parameters

References

Jason P. C. Chiu and Eric Nichols. 2015. Named entity recognition with bidirectional lstm-cnns. *CoRR*, abs/1511.08308.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *CoRR*, abs/1802.05365.

Stefan Schweter and Alan Akbik. 2020. FLERT: document-level features for named entity recognition. *CoRR*, abs/2011.06993.

2.1 Tables and figures

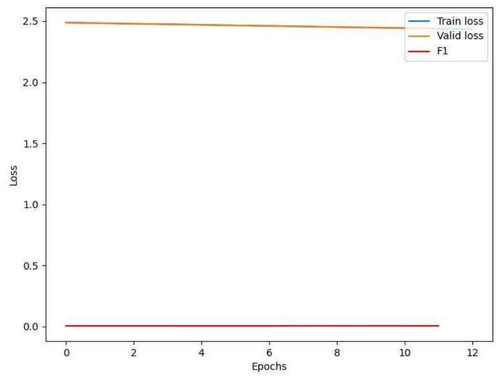


Figure 1: Train results using SGD optimizer

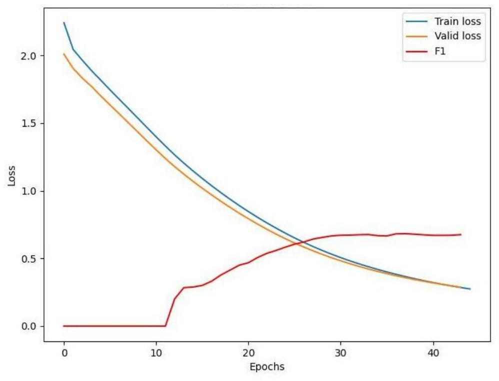


Figure 2: Train results in a F1 = 0.680 using Adam optimizer with lr = 0.01

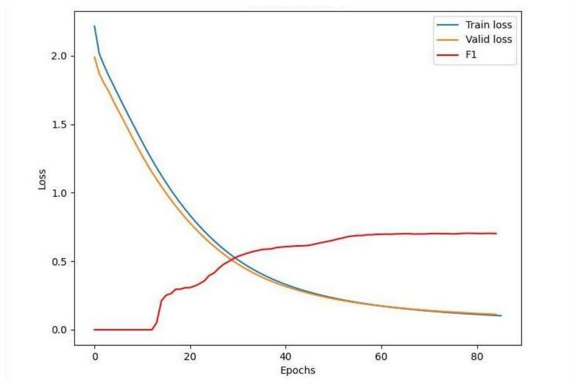


Figure 3: Train results in a F1 = 0.704 using Adam optimizer with lr = 0.01

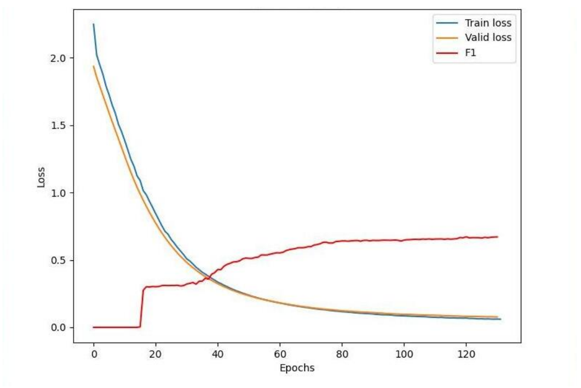


Figure 4: Train result with 4 layers

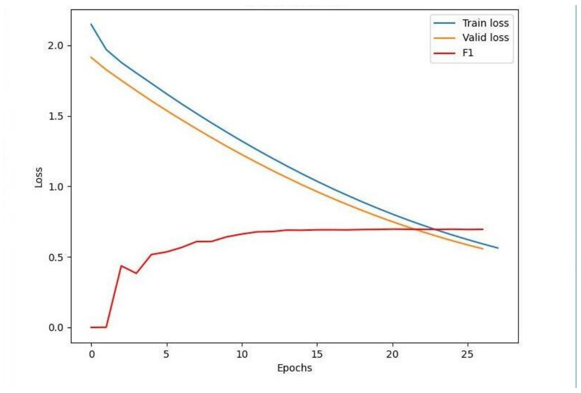
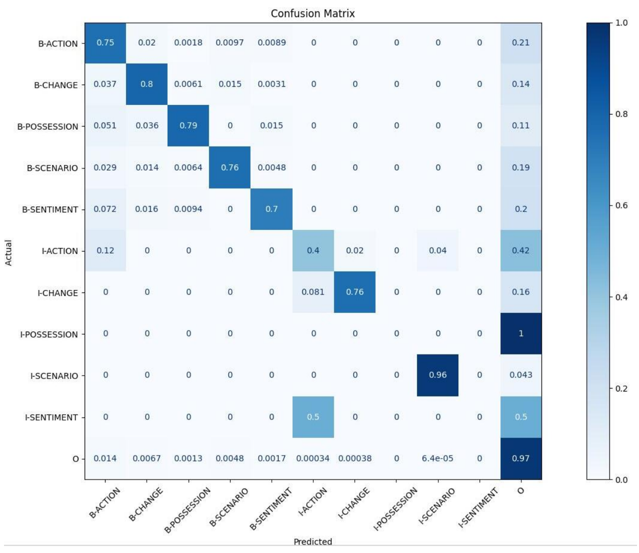


Figure 5: Train speed up using previous word embedding



Actual \ Predicted	B-ACTION	B-CHANGE	B-POSSESSION	B-SCENARIO	B-SENTIMENT	I-ACTION	I-CHANGE	I-POSSESSION	I-SCENARIO	I-SENTIMENT	O
B-ACTION	0.75	0.02	0.0018	0.0097	0.0089	0	0	0	0	0	0.21
B-CHANGE	0.037	0.8	0.0061	0.015	0.0031	0	0	0	0	0	0.14
B-POSSESSION	0.051	0.036	0.79	0	0.015	0	0	0	0	0	0.11
B-SCENARIO	0.029	0.014	0.0064	0.76	0.0048	0	0	0	0	0	0.19
B-SENTIMENT	0.072	0.016	0.0094	0	0.7	0	0	0	0	0	0.2
I-ACTION	0.12	0	0	0	0	0.4	0.02	0	0.04	0	0.42
I-CHANGE	0	0	0	0	0	0.081	0.76	0	0	0	0.16
I-POSSESSION	0	0	0	0	0	0	0	0	0	0	1
I-SCENARIO	0	0	0	0	0	0	0	0	0.96	0	0.043
I-SENTIMENT	0	0	0	0	0	0.5	0	0	0	0	0.5
O	0.014	0.0067	0.0013	0.0048	0.0017	0.00034	0.00038	0	6.4e-05	0	0.97

Figure 6: Confusion Matrix of the final model