Assignment 1: recurrent neural models for sequence labeling

Piero Castriota, Usha Padma Rachakonda

July 6, 2022

Abstract

This project aims to train four different models able to perform a Sequence Labelling of Part of Speech tagging on the dependency treebank dataset. Only the two models best performing on the validation set, considering their accuracy, will be evaluated on the test set. The evaluation on the test set will be carried out considering f1-macro instead. The following models represent the best combination of hyperparameters tried during the training.

1 Data Preparation

The dataset has been downloaded and divided in the three sets: train, validation and test.

For each set, a dataframe has been created, containing a row for each sentence and two columns: one containing a list of all the words in the sentence and the other containing a list for all the labels associated with each word.

2 Embedding

In order to make the dataset available for a network to train on, Glove word embedding was performed, an unsupervised learning algorithm for obtaining vector representations for words.
The chosen value for the dimension of the embedding is 100. Many attempts were made and 100 represent the lowest value that doesn't compromise the performance of the models. Higher values (300) were also chosen at the beginning, but the results weren't good enough to accept the slowness of the training.

3 Vocabulary and Embedding Matrix

A set containing the words in the training set was created and it was compared to the vocabulary of the GloVe embedding model to find the out of vocabulary (OOV) terms.
The embedding matrix was initialized stacking up all the embedding vectors associated to each word: for the non-OOV words the associated embedding vector was added while for the OOV words a new vector was randomly initialized and stacked up to the matrix. Also, the vocabulary was enlarged with the OOV terms.
The validation and test sets were used to update the vocabulary and the embedding matrix in the same way, adding at each step the corresponding embedding vectors to the embedding matrix and adding the new OOV terms to the vocabulary.
In this way, a word considered OOV when analysing the training set is not considered OOV anymore if found in the validation or test set.

4 Models

The first model instantiated, the baseline model, is composed in total of four layers: input, embedding, Bidirectional LSTM and TimeDistributed Dense layer. The training involves only the last two layers. To avoid overfitting, after some attempts, the following hyperparameters have been chosen for the LSTM: Dropout rate: 0.2, Recurrent Droput: 0.2, Nodes: 16.
Also, to maximize the performance of the training, an exponential decay schedule for the learning rate

and an early stopping with patience = 3 were defined.
Of all the tried optimizers, Adam was the one giving the best results in the shortest time.
Other three models were analysed, they were all created just by making some modifications to the baseline model, more specifically:
Second model: Bidirectional GRU layer instead of Bidirectional LSTM
Third model: Double Bidirectional LSTM layer
Fourth model: Double TimeDistributed Dense layer
The learning rate is for all the models 0.01 except for the fourth model which is 0.1, since it was very necessary to increase it.
By default, the number of epochs set is 30.

5 Scores

Summary of evaluation on val data		
	val_loss	val_accuracy
model1	0.027203	0.915274
model2	0.029269	0.907738
model3	0.025233	0.922971
model4	0.086384	0.749896

According to these results, the models that performs best on the validation set are the first and the third (the baseline model and the Double Bidirectional LSTM variation).
The first model didn't encounter an early stopping while the third got it at exactly the 30th epoch, this means that we could spend even more time tuning some parameters and training the models in order to achieve better results, although it might not be that helpful since the exponential decay of the LR.

6 Evaluation

The two best models were evaluated on the test set considering the metric f1-macro.11 In order to make the evaluation significant the labels corresponding to punctuation and symbols were removed.
The f1-macro uses a weighted average, considering each class' support, useful to balance the significance for underrepresented and over represented terms.

	f1-macro
model1	0.820352
model3	0.911771

7 Error analysis

The confusion matrices' show a pretty satisfying result for both the models.
The main sources of error are the misclassification of "nn" to "nnp" and vice versa, but since those labels also represent the highest number of elements, the proportion of wrong predictions with respect to true predictions is more than acceptable for both of them (less than 5% of error considering only these two).
The "jj" label also presents an important number of error samples, but also in this case it gets significally misclassified only with the "nn" and the "nnp" labels.
To improve the performances it's possible, as already said, to spend more time in the training process for sure, and also having a more balanced dataset could help since there is a big disproportion in this one between the different classes.