Assignment 1

Due to: 20/12/2021 (dd/mm/yyyy)

Credits: Andrea Galassi, Federico Ruggeri, Paolo Torroni

Summary: Part-of Speech (POS) tagging as Sequence Labelling using Recurrent Neural Architectures

Intro

In this assignment we will ask you to perform POS tagging using neural architectures

You are asked to follow these steps:

- Download the corpora and split it in training and test sets, structuring a dataframe.
- Embed the words using GloVe embeddings
- Create a baseline model, using a simple neural architecture
- Experiment doing small modifications to the baseline model, choose hyperparameters using the validation set
- Evaluate your two best model
- Analyze the errors of your model

Task: given a corpus of documents, predict the POS tag for each word

Corpus: Ignore the numeric value in the third column, use only the words/symbols and its label. The corpus is available at: https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/packages/corpora/dependency_treebank.zip

Splits: documents 1-100 are the train set, 101-150 validation set, 151-199 test set.

Features: you MUST use GloVe embeddings as the only input features to the model.

Splitting: you can decide to split documents into sentences or not, the choice is yours.

I/O structure: The input data will have three dimensions: 1-documents/sentences, 2-token, 3-features; for the output there are 2 possibilities: if you use one-hot encoding it will be 1-documents/sentences, 2-token labels, 3-classes, if you use a single integer that indicates the number of the class it will be 1-documents/sentences, 2-token labels.

Baseline: two layers architecture: a Bidirectional LSTM layer and a Dense/Fully-Connected layer on top; the choice of hyper-parameters is yours.

Architectures: experiment using a GRU instead of the LSTM, adding an additional LSTM layer, and adding an additional dense layer; do not mix these variantions.

Training and Experiments: all the experiments must involve only the training and validation sets.

Evaluation: in the end, only the two best models of your choice (according to the validation set) must be evaluated on the test set. The main metric must be F1-Macro computed between the various part of speech. DO NOT CONSIDER THE PUNCTUATION CLASSES.

Metrics: the metric you must use to evaluate your final model is the F1-macro, WITHOUT considering punctuation/symbols classes; during the training process you can use accuracy because you can't use the F1 metric unless you use a single (gigantic) batch because there is no way to aggregate "partial" F1 scores computed on mini-batches.

Discussion and Error Analysis : verify and discuss if the results on the test sets are coherent with those on the validation set; analyze the errors done by your model, try to understand which may be the causes and think about how to improve it.

Report: you are asked to deliver the code of your experiments and a small pdf report of about 2 pages; the pdf must begin with the names of the people of your team and a small abstract (4-5 lines) that sums up your findings.

Out Of Vocabulary (OOV) terms

How to handle words that are not in GloVe vocabulary? You can handle them as you want (random embedding, placeholder, whatever!), but they must be STATIC embeddings (you cannot train them).

But there is a very important caveat! As usual, the element of the test set must not influence the elements of the other splits!

So, when you compute new embeddings for train+validation, you must forget about test documents. The motivation is to emulate a real-world scenario, where you select and train a model in the first stage, without knowing nothing about the testing environment.

For implementation convenience, you CAN use a single vocabulary file/matrix/whatever. The principle of the previous point is that the embeddings inside that file/matrix must be generated independently for train and test splits.

Basically in a real-world scenario, this is what would happen:

1. Starting vocabulary V1 (in this assignment, GloVe vocabulary)
2. Compute embeddings for terms out of vocabulary V1 (OOV1) of the training split
3. Add embeddings to the vocabulary, so to obtain vocabulary V2=V1+OOV1
4. Training of the model(s)
5. Compute embeddings for terms OOV2 of the validation split
6. Add embeddings to the vocabulary, so to obtain vocabulary V3=V1+OOV1+OOV2
7. Validation of the model(s)

8. Compute embeddings for terms OOV3 of the test split
9. Add embeddings to the vocabulary, so to obtain vocabulary V4=V1+OOV1+OOV2+OOV3
10. Testing of the final model

In this case, where we already have all the documents, we can simplify the process a bit, but the procedure must remain rigorous.

1. Starting vocabulary V1 (in this assignment, GloVe vocabulary)
2. Compute embeddings for terms out of vocabulary V1 (OOV1) of the training split
3. Add embeddings to the vocabulary, so to obtain vocabulary V2=V1+OOV1
4. Compute embeddings for terms OOV2 of the validation split
5. Add embeddings to the vocabulary, so to obtain vocabulary V3=V1+OOV1+OOV2
6. Compute embeddings for terms OOV3 of the test split
7. Add embeddings to the vocabulary, so to obtain vocabulary V4=V1+OOV1+OOV2
8. Training of the model(s)
9. Validation of the model(s)
10. Testing of the final model

Step 2 and step 6 must be completely independent of each other, for what concerns the method and the documents. But they can rely on the previous vocabulary (V1 for step 2 and V3 for step 6) THEREFORE if a word is present both in the training set and the test split and not in the starting vocabulary, its embedding is computed in step 2) and it is not considered OOV anymore in step 6).

Report

The report must not be just a copy and paste of graphs and tables!

The report must not be longer than 2 pages and must contain:

- The names of the member of your team
- A short abstract (4-5 lines) that sum ups everything
- A general description of the task you have addressed and how you have addressed it
- A short description of the models you have used
- Some tables that sum up your findings in validation and test and a discussion of those results
- The most relevant findings of your error analysis

Evaluation Criterion

The goal of this assignment is not to prove you can find best model ever, but to face a common task, structure it correctly, and follow a correct and rigorous experimental procedure. In other words, we don't care if you final models are awful as long as you have followed the correct procedure and wrote a decent report.

The score of the assignment will be computed roughly as follows

- 1 point for the general setting of the problem
- 1 point for the handling of OOV terms
- 1 point for the models
- 1 point for train-validation-test procedure
- 2 point for the discussion of the results, error analysis, and report

This distribution of scores is tentative and we may decide to alter it at any moment. We also reserve the right to assign a small bonus (0.5 points) to any assignment that is particularly worthy. Similarly, in case of grave errors, we may decide to assign an equivalent malus (-0.5 points).

Contacts

In case of any doubt, question, issue, or help we highly recommend you to check the course useful material for additional information, and to use the Virtuale forums to discuss with other students.

You can always contact us at the following email addresses. To increase the probability of a prompt response, we reccomend you to write to both the teaching assistants.

Teaching Assistants:

- Andrea Galassi -> a.galassi@unibo.it
- Federico Ruggeri -> federico.ruggeri6@unibo.it

Professor:

- Paolo Torroni -> p.torroni@unibo.it

FAQ

- You can use a non-trainable Embedding layer to load the glove embeddings
- You can use any library of your choice to implement the networks. Two options are tensorflow/keras or pythorch. Both these libraries have all the classes you need to implement these simple architectures and there are plenty of tutorials around, where you can learn how to use them.