

AnTeDe Lab 8

1 Notebook B

The goal of a dependency parser is to find out the relations between different words and what type of relationships they have to each other. This is done by analysing the grammatical structure of the sentence.

To build the dependency between the words in the sentence an algorithm named Transition-based dependency parsing is used. It uses a Stack, Buffer and A (a set of arcs). The Stack contains which words the algorithm is currently looking at. The Buffer contains all other words from the sentence and A contains the arcs that the algorithm already classified. An Arc describes how words relate to each other starting from the root word.

1.1 Is `n_classes` a modifiable hyper parameter?

The neural network in the dependency parser has the job of deciding which of the three possible actions should be taken at each step of the algorithm.

1. Shift: Moves words sequentially into the stack
2. Left Arc: The word to the left of the Head (right most word in the stack) is a dependent of the Head.
3. Right Arc: The Head is a dependent of the word to its left (just the different direction as the Left Arc)

The `n_classes` parameter in the Neural Network corresponds to the three actions that can be taken at each step in the algorithm. Thus this parameter can not be changed since there are only three possible actions to be taken. From this follows that `n_classes` is not an hyper parameter that can/ should be tuned.

1.2 Hyper parameter tuning

In this section the effect of hyper parameter tuning on the model performance is analysed. The effect on the model performance of two hyper parameters will be evaluated. To start with the baseline performance will be evaluated.

1.2.1 Evaluation

All models will be compared by using the UAS (Unlabeled Attachment Score)

ref	hidden_size	dropout_prob	UAS
baseline	200	0.5	89.27
1	300	0.5	89.50
2	100	0.5	87.68
3	200	0.8	86.65
4	200	0.2	89.42

All models were trained for 10 Epochs and the final test UAS is noted in table. Looking at the above results it seems that a hidden_size between 200 and 300 is a very reasonable size. This makes sense since the model needs some capacity to accurately predict the next action. The best dropout_prob seems to be somewhere between 0.2 and 0.5. This is could be due to the fact that the model converges faster with a lower dropout_prob.