Ass 2 – Part 1

Dorin Keshales – 313298424

Model's Parameters

In both tasks:

- I used dropout, after the activation function, with probability of p=0.5.
- Optimizer: Adam
- Learning rate: 1e-3
- Size of the hidden layer: 150

POS:

- I used batches of size 32 for both training and dev sets.
- Number of epochs: 7

NER:

- I used batches of size 32 for the training set and batches of size 128 for the dev set.
- Number of epochs: 8

Answers to the considerations

- In order to deal with words that appear in the dev / test set and not in the training set, I added to the vocabulary (which is based on the words appear the training set) a token called '<unk>'. When we come across a word that does not appear in the vocabulary (for example a word that was on the dev set or test set), then i assign to it the word embedding vector of '<unk>'. In order to train this word embedding vector, I took from the training set all the rare words (which appear only once) and treated them as if they were unknown words, which means they got the embedding vector of the unknown token - '<unk>'.

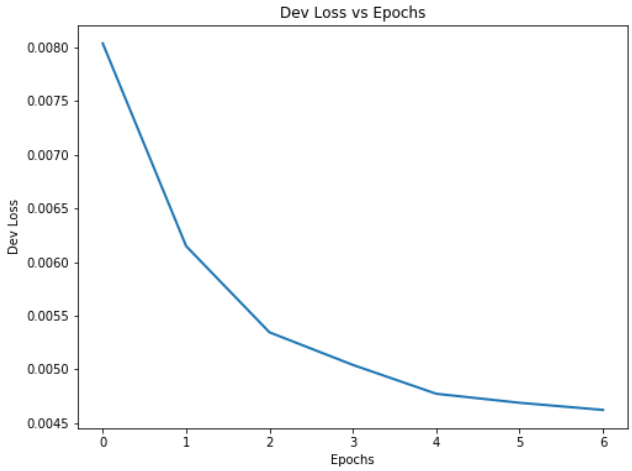
- In order to create a context window for the first word in a sequence, I added to the vocabulary a start token - <start> in which I used as the two words to the left side of the word being tagged (in this case the first word of a sequence). As a result, each start tokens in the context window will get

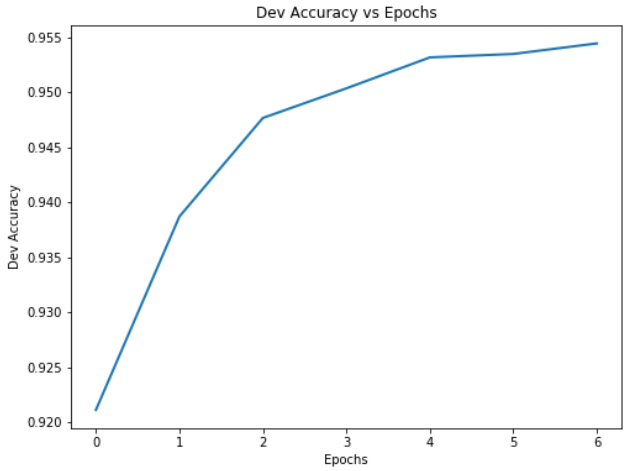
the embedding vector corresponding to it. This embedding vector will be learned during the training process.

- I used a similar idea in order to create a context window to the last word in a sequence. For this case I added to the vocabulary an end token - <end> in which I used as the two words to the right side of the word being tagged (in this case the last word of a sequence). As a result, each end token in the context window will get the embedding vector corresponding to it. This embedding vector as well will be learned during the training process.

My model's results

POS: Loss - 0.00462 Accuracy – 95.447%





NER: Loss - 0.00109 Accuracy – 80.251%