# Assignment 2 – Window-based Tagging

Daniel Bazar 314708181

Peleg shefi 316523638

## Part 4 – Convolution-based sub-word units

### Architecture

* Both tasks implementing the same Network: MLP with one hidden layer and a tanh activation function. The MLP feeded from a **Convolution-based sub-word embedding**
* The Convolution-based sub-word units architecture is the following:  
   each word is represented as a max-pooled vector over the vectors resulting from a convolution over the word’s characters. The pre-trained word-vectors and the character-based CNN vectors are combined via concatenation
* The network trained with a cross-entropy loss
* We Experimented with several network configurations and chose the best configuration based on the DEV accuracy

### Best parameters

* NER:
  + Hidden layer size: 130
  + Dropout probability: 0.3
  + Batch size: 128
  + Optimizer: Adam (Learning rate: 1e-4)
  + Epochs: 6
  + CNN filters: 30
  + CNN embedding dimension: 30
  + CNN kernel size: 3
  + CNN padding size: 2
  + CNN dropout: 0.5
* POS:
  + Hidden layer size: 90
  + Dropout probability: 0.2
  + Batch size: 64
  + Optimizer: Adam (Learning rate: 5e-05)
  + Epochs: 8
  + CNN filters: 30
  + CNN embedding dimension: 30
  + CNN kernel size: 3
  + CNN padding size: 2
  + CNN dropout: 0.5

### Considerations

* The Convolution-based sub-word units method can be combined with the pretrained embedding so we had to consider all the things we take into account in the pretrained part:
  + We handle words that appear in the training file but not in the embedding file as we handle those words in part1 (words that don’t appear in dev file), we assign them the UNK token.
  + If we used the pre-trained embedding vectors, we transform the training and the dev data to lower case because the embedding vocabulary being lower-case.
  + Because the embedding vocabulary contains special words like “DGDGDGDG”, “DG.DG”, “+DG”, “NNNUMMM”, etc. we treated those words as digits patterns.
  + We padded the sentences with SOS (start of string) and EOS (end of string) at the beginning and end of the sentence.
* The Convolution-based sub-word units cannot combined with the sub-word units method so we prevents this from happening.
* Compared to what we implemented in Part 4 (sub-word units) this method ---------
* Trying fewer filters result with ------ and trying more filter results with ------
* Trying different window sizes results with ---------------------

### Results

* NER:
  + Loss validation: 0.101
  + Accuracy validation: 83.13%
* POS:
  + Loss validation: 0.135
  + Accuracy: 95.84%

### Brief analysis of the results

|  |  |  |
| --- | --- | --- |
| Tagger | Best DEV Accuracy | conclusion |
| standard | 82.95 | Baseline |
| Pre-trained & Sub-word | 79.54 | worser |
| Sub-word | 79.25 | No significant difference |
| Pre-trained | 77.51 | worst |

We can see that all methods are harming our standard tagger. Although, its seems like the sub-word units are more useful than the pre-trained embedding and there is no significant difference in combining them together

### Graphs