MVA 2020 - Deep Learning

NLP project

Ariane ALIX

March 1st, 2020

---

2 Multilingual word embeddings

2.1 Question

Let X and Y be two matrices of dimension $d \times n$.

We consider the minimization problem:

$$\arg \min_{W \in O_d(\mathbb{R})} \|WX - Y\|_F$$

Which is equivalent to:

$$\arg \min_{W \in O_d(\mathbb{R})} \|WX - Y\|_F^2$$

Since W is orthogonal, we have $\|WX\|_F = \|X\|_F$.

Therefore the problem can be simplified to:

$$\begin{aligned}
& \arg \max_{W \in O_d(\mathbb{R})} \langle W, YX^T \rangle_F \\
= \quad & \arg \max_{W \in O_d(\mathbb{R})} \langle W, U\Sigma V^T \rangle_F \\
= \quad & \arg \max_{W \in O_d(\mathbb{R})} \langle U^T WV, \Sigma \rangle_F \\
\leq \quad & \arg \max_{W \in O_d(\mathbb{R})} \|U^T WV\|_F \|\Sigma\|_F \quad \text{using Cauchy-Schwarz} \\
= \quad & \|\Sigma\|_F \quad \text{since } U^T WV \text{ is orthogonal}
\end{aligned}$$

Therefore, W maximizes the previous equation when we have an equality at the next-to-last line, which happens if and only if $U^T WV = I_d$.

The solution is therefore $W* = UV^T$.

3 Sentence classification with BoW

3.1 Question

Accuracies obtained using the average of word vectors:

- Train set: 46.898%
- Dev set: 41.871%

Accuracies obtained using the IDF weighted average:

- Train set: 47.647%
- Dev set: 41.508%

4 Deep Learning models for classification

4.1 Question

We used the loss of Tensorflow called *sparse_categorical_crossentropy*, which is used for multiple categorical classes (here the order of the numbers 0 to 5 of the classes does not mean anything). The *sparse* option allows us to use the class directly without one-hot encoding.

$$L(y, \hat{y}) = -\sum_{i=1}^{n} y_i \log(\hat{y}_i) \tag{1}$$

4.2 Question

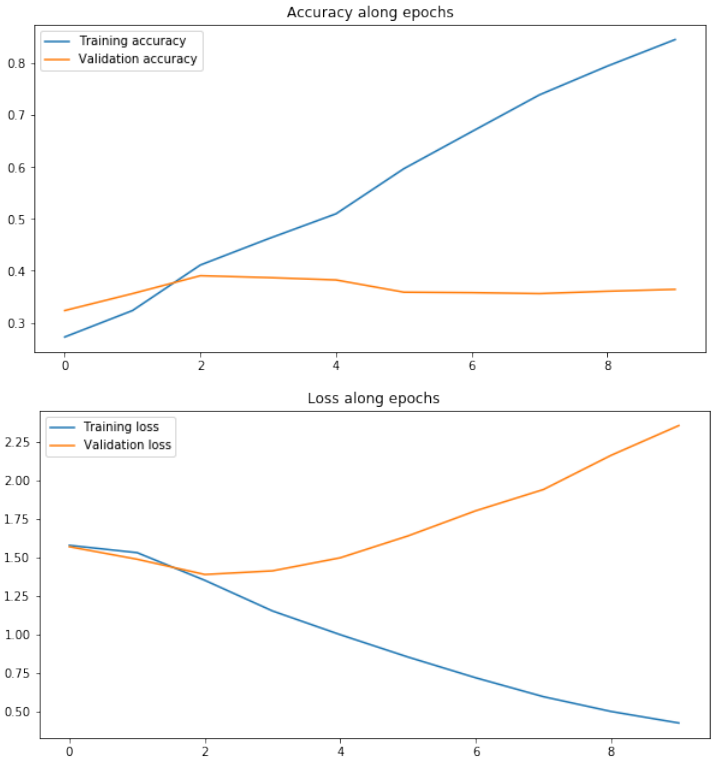

Figure 1: Accuracy and loss of the given LSTM

4.3 Question

We then tried with a different approach: a bidirectionnal LSTM with a Convolutional Neural Network. It is interesting in the case of NLP since CNN look at patterns of words, hence 'understanding' syntax, and bidirectionnal LSTM can also take into account the context of following words.

We only put 1 convolutional and 1 linear layer and a few number of parameters to avoid overfitting since the dataset is quite small. For that same reason, we added a 'dropout' and reduced the size of the embedding of words.