DL tp n2

Mathieu Chalvidal

Janvier 2019

Abstract

In this assignment we will cover monolingual word and sentence embeddings, multilingual word embeddings and sentence classification with Bag-of-Vectors (BoV) and LSTMs. An ipython notebook for the full project is attached: nlp project.ipynb. It contains instructions on what you must code. Please refer to the "Send your answers" part for a description of what we expect from you in terms of deliverable.

1 Monolingual embeddings

See Jupyter Notebook

2 Multilingual word embeddings

Question Using the orthogonality and the properties of the trace, prove that, for X and Y two matrices:

$$\underset{W\in O_d(R)}{\operatorname{argmin}} \left\| WX - Y \right\|_F = UV^T, U\Sigma V^T = SVD(YX^T).$$

$$\begin{aligned}
\underset{W\in O_d(R)}{\operatorname{argmin}} \left\| WX - Y \right\|_F &= \underset{W\in O_d(R)}{\operatorname{argmin}} TR(X^TW^TWX) - X^TW^TY - Y^TW^TX - Y^TY) \\
&= \underset{W\in O_d(R)}{\operatorname{argmin}} TR(X^TX + Y^TY) - 2TR(Y^TWX) \\
&= \underset{W\in O_d(R)}{\operatorname{argmax}} \left\langle W \middle| X^TY \right\rangle \\
&= \underset{W\in O_d(R)}{\operatorname{argmax}} \left\langle W \middle| U\Sigma V^T \right\rangle \\
&= \underset{W\in O_d(R)}{\operatorname{argmax}} \left\langle U^TWV \middle| \Sigma \right\rangle
\end{aligned}$$

Given than U, W and V are orthogonal matrices, the maximum of this product is reached for $U^TWV = I_d$ Hence for $W = UV^T$

3 Sentence classification with BoV

Question : What is your training and dev errors using either the average of word vectors or the weighted-average?

BoV + Logistic regression model		
reference	Accuracy	f1-score
training set No$_Idf$	0.47	0.42
dev set No-Idf	0.43	0.37
training set Idf	0.47	0.43
dev set Idf	0.41	0.36

4 Deep Learning models for classification

Question : Which loss did you use? Write the mathematical expression of the loss you used for the 5-class classification.

The loss used for classification purpose is **categorical crossentropy**. It is a usefull loss function coming from information theory that can be interpreted as a mesure of divergence between two distributions. It is defined as follow for our 5-class classification problem.

$$H(\mathbf{y}_{true}, \mathbf{y}_{pred}) = \sum_{i=1}^{5} y^i_{true} log \frac{1}{y^i_{pred}}$$

with n being the number of training exemples, y_{true} the true distribution class of our data and y_{pred} the distribution predicted by our model.

Question : Plot the evolution of train/dev results w.r.t the number of epochs.

One of the major problems with recurent Neural net architectures is overfitting. In order to keep the model to overfit the data, we added Dropout(0.5) layers and performed early stopping in train (5 epochs only). As the curves show below, the loss function has stopped descending on the dev set and the accuracy is plateauing at 40%.

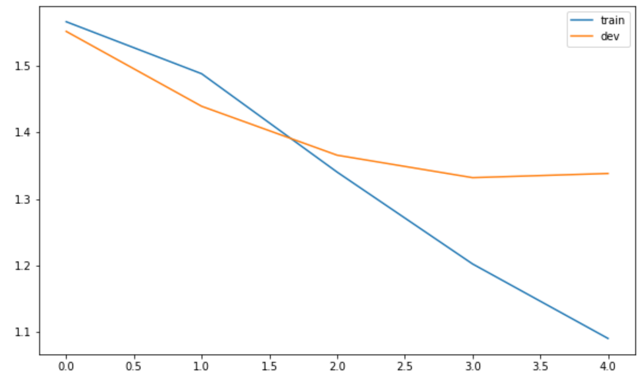


Figure 1: Fonction de coût en fonction du nombre d'epochs

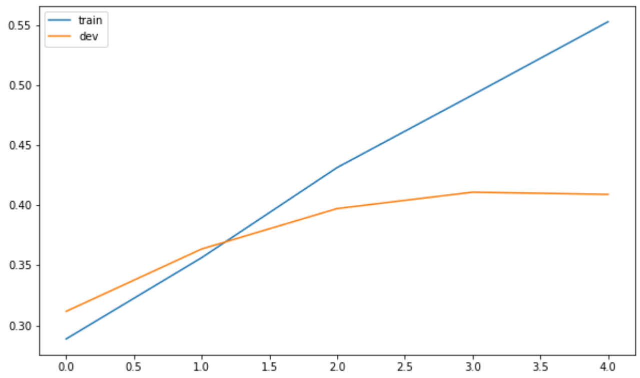


Figure 2: Accuracy en fonction du nombre d'epochs

Question : Be creative: use another encoder. Make it work! What are your motivations for using this other model?

In order to explore an other kind of Neural network architecture, I decided to implement a CNN architecture + word2vec embeddings as described in the paper **A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional** (Zhang et al, 2014) The architecture presented use the 300-dimensional word2vec embeddings as encodings for the sentences and perform different convolution with trained filters on 3 different region sizes: (2,3,4). Then a pooling layer is apllied and a classifier architecture tops the architecture.

CNN have proven to be efficient on language classification tasks given their power of representation when it comes to relational-structured data such as sentences. The architecture is depicted in Figure 3. In addition to the architecture, the use of word2vec embeddings might yeald better results than crafted embeddings trained within the precendent model framework. Indeed, the dev accuracy is plateauing at 40%.

However, confusion matrix of the 2 models tends to show that CNN is better at representing middle-class sentence. Thus validating our hypothesis that CNN have a good power of representation even when it comes to fine-grained classification.

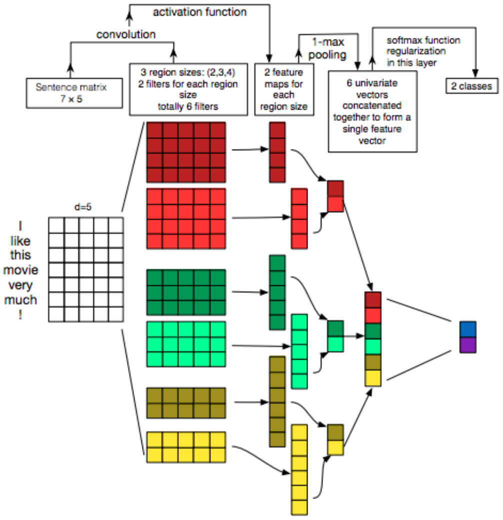
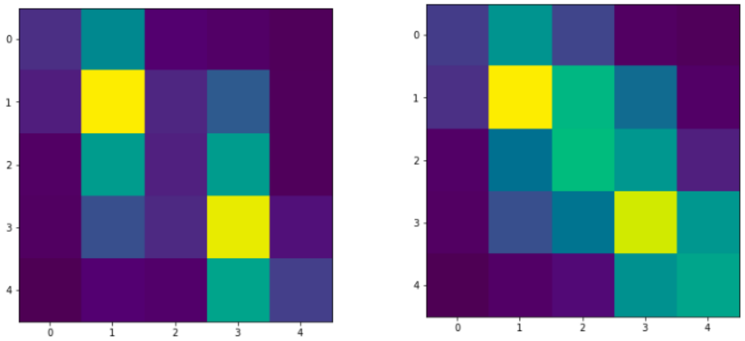

activation function
convolution
1-max pooling
softmax function regularization in this layer
Sentence matrix 7 x 5
3 region sizes: (2,3,4) 2 filters for each region size totally 6 filters
2 feature maps for each region size
6 univariate vectors concatenated together to form a single feature vector
2 classes
d=5
I
like
this
movie
very
much
!

Figure 3: architecture of implemented CNN



(a) LSTM confusion matrix

(b) CNN confusion matrix