Mini-project on NLP - REPORT

By : Thibault Lahire (thibault.lahire@student.isae-supaero.fr)

Attached programs

This mini-project on NLP was done for the MVA course "Deep Learning" under the supervision of Vincent Lepetit. There are 6 files for this mini-project. This pdf, the notebook, and the 4 .txt documents which are the predictions on the test set. Namely, we have : `logreg_bow_y_test_sst.txt`, `randforest_bow_y_test_sst.txt`, `logreg_lstm_y_test_sst.txt`, and `pretrainembedding_lstm_y_test_sst.txt`

1 Monolingual embeddings

We summarize the results obtained in the notebook for the test of the class Word2Vec :

Word 1	**Word 2**	**Cosine similarity**
cat	tree	0.264
cat	dog	0.708
cat	pet	0.675
Paris	France	0.689
Paris	Germany	0.405
Paris	baguette	0.294
Paris	donut	-0.0066

Moreover, the 5 most similar words of "*cat*", "*dog*", "*dogs*", "*Paris*" and "*Germany*" are:

Most similar	**2nd most similar**	**3rd most similar**	**4th most similar**	**5th most similar**
cat	cats	kitty	kitten	feline
dog	dogs	puppy	pup	canine
dogs	dog	cats	puppies	Dogs
Paris	France	Parisian	Marseille	Brussels
Germany	Austria	Europe	Berlin	Hamburg

In the second sub-part, we test the class BagOfWords we have implemented by finding the most similar sentences to a given sentence with "average of word embeddings" and "idf weighted average of word embeddings". For the first, the given sentence was "*1 smiling african american boy*" and for the second "*2 female babies eating cheaps*".

Rank	**Average of word embeddings**	**idf weighted average of word embeddings**
Most similar	1 smiling african american boy .	2 female babies eating chips .
2nd most similar	2 woman dancing while pointing .	2 kids holding hands and smiling .
3rd most similar	5 women and 1 man are smiling for the camera .	3 men , 2 shoveling dirt and the other laying down cement .
4th most similar	a small boy following 4 geese .	3 little kids are playing football .
5th most similar	2 female babies eating chips .	two men and 2 women running .

2 Multilingual word embeddings

First, we recall some useful definitions and properties. The Frobenius scalar product of two matrices A and B is : $\langle\ A,\ B\ \rangle_F = Tr(A^\top B)$, with Tr the trace. The Frobenius norm of matrix C is therefore : $\|C\|_F = \sqrt{Tr(C^\top C)}$. Moreover, by definition, $W \in \mathcal{O}_d(\mathbb{R}) \iff W^\top W = I_d$

For X and Y two matrices of compatible form with W such that $WX - Y$ has a sense, we are looking for :

$$W^* = \arg\min_{W \in \mathcal{O}_d(\mathbb{R})} \|WX - Y\|_F = \arg\min_{W \in \mathcal{O}_d(\mathbb{R})} \|WX - Y\|_F^2$$

$$\begin{aligned}
\|WX - Y\|_F^2 &= Tr\left[(WX - Y)^\top (WX - Y)\right] \\
&= Tr\left[(X^\top W^\top - Y^\top)(WX - Y)\right] \\
&= Tr\left[X^\top W^\top WX - X^\top W^\top Y - Y^\top WX + Y^\top Y\right] \\
&= Tr\left[X^\top X\right] - 2Tr\left[Y^\top WX\right] + Tr\left[Y^\top Y\right] \\
&= \|X\|_F + \|Y\|_F - 2\ \langle\ WX,\ Y\ \rangle_F
\end{aligned}$$

where we used the fact that $Tr(A) = Tr(A^\top)$. Moreover, since $Tr(AB) = Tr(BA)$, we have :

$$\langle\ WX,\ Y\ \rangle_F = Tr\left[Y^\top WX\right] = Tr\left[XY^\top W\right] = Tr\left[(YX^\top)^\top W\right] = \langle\ W,\ YX^\top\ \rangle_F$$

Therefore, our objective turns out to :

$$W^* = \arg\min_{W \in \mathcal{O}_d(\mathbb{R})} \|WX - Y\|_F^2 = \arg\max_{W \in \mathcal{O}_d(\mathbb{R})} \langle\ W,\ YX^\top\ \rangle_F$$

We now introduce three matrices U, Σ, and V such that the SVD-decomposition of $YX^\top$ can be written : $YX^\top = U\Sigma V^\top$. Hence,

$$\langle\ W,\ YX^\top\ \rangle_F = \langle\ W,\ U\Sigma V^\top\ \rangle_F = Tr\left[(U\Sigma V^\top)^\top W\right] = Tr\left[V\Sigma^\top U^\top W\right] = Tr\left[\Sigma^\top U^\top WV\right]$$

We introduce $A = U^\top WV$. At this step, we recall a property of the SVD decomposition : the matrix Σ is a matrix where all the coefficients are equal to 0 except for diagonal coefficients. We have:

$$Tr\left[\Sigma^\top A\right] = \sum_i (\Sigma^\top A)_{i,i} = \sum_i \sum_k (\Sigma^\top)_{i,k} A_{k,i} = \sum_i \sum_k \Sigma_{k,i} A_{k,i} = \sum_i \Sigma_{i,i} A_{i,i}$$

A is such that $A^\top A = I$, so, $\forall i,\ \sum_j A_{i,j}^2 = 1$. It implies that $A_{i,i} \leq 1$, and there is equality if and only if $A = I$. In other words,

$$Tr\left[\Sigma^\top A\right] \leq \sum_i \Sigma_{i,i}\ \text{, with equality iff } A = I$$

Hence, $Tr\left[\Sigma^\top A\right]$ is maximized for $A = I$, i.e. $U^\top WV = I$ i.e. $W = UV^\top$. This concludes the proof.

Here is what we obtained :

French word	English word 1	English word 2	English word 3
chat	cat	kitten	kitty
chien	dog	cat	pet
voiture	car	vehicule	automobile
zut	oops	Ah	ah

3 Sentence classification with BoW

As it can be seen on the attached jupyter, the outcome for logistic regression with a tuning of the L2-penalization is :

Method	Train score	Dev score
average	0.467	0.415
weighted-average	0.464	0.423

As required, I have tested several penalization and selected the one which gave the best scores :

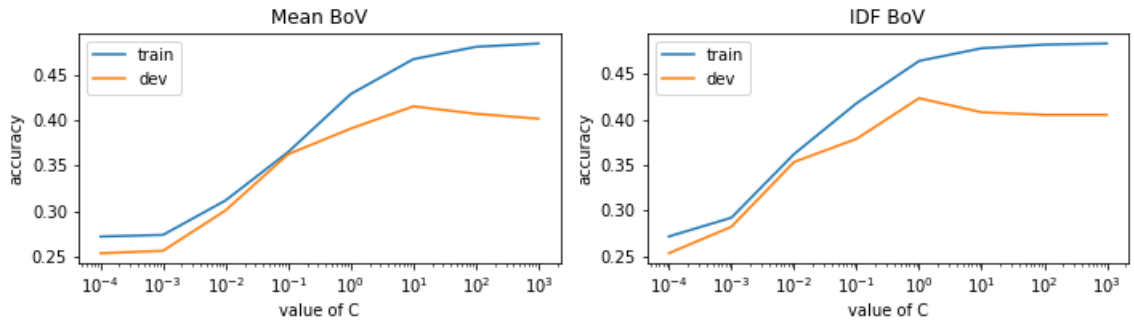

Figure 1: Best value of C for the logistic regression

As suggested in the provided notebook, I have also tested another classifier : RandomForestClassifier. Even with hyperparameters tuning, I did not manage to do better than logistic regression. The submission for this part is given in the file `rand_forest_bov_y_test_sst.txt`

4 Deep Learning models for classification

The loss we used is the categorical cross-entropy, in the case of a 5-class classification. We note n the number of samples, and $t_{i,j}$ the ground-truth score of the i-th sample for class j (which means $t_{i,j} = 1$ if sample i belongs to class j, 0 otherwise). The goal of the deep network is to approximate as best as possible the ground-truth scores $t_{i,j}$ for all samples $i \in [\![1; n]\!]$ for all classes $j \in [\![0; 4]\!]$. We note $s_{i,j}$ the prediction of the network. Then the categorical cross-entropy loss is :

$$L = -\frac{1}{n} \sum_{i=1}^{n} \sum_{j=0}^{4} t_{i,j} \log(s_{i,j})$$

With the proposed network, we obtain an evolution of the train/dev results w.r.t the number of epochs shown in Fig. [2] and [3].

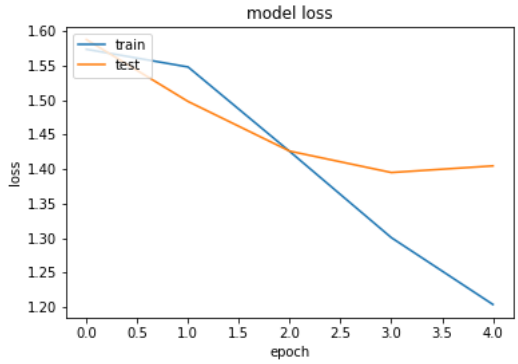

Figure 2: Loss of the proposed model over epochs

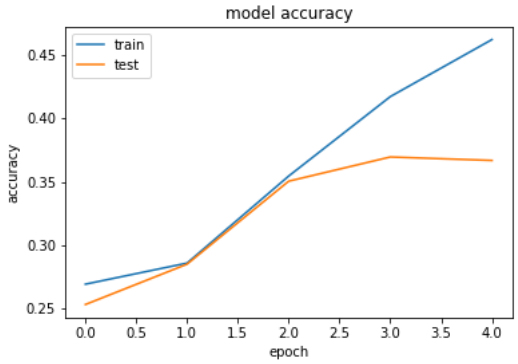

Figure 3: Accuracy of the proposed model over epochs

As we see, 5 epochs are enough to achieve the best performance of this model, which is below logistic regression. Indeed, we obtain an accuracy on the dev set about 0.38, whereas logistic regression provided an accuracy of 0.415 on the same set.

For the very last question, I propose the following innovations. Now, the network does not learn the embeddings, but use pretrained ones. More precisely, the network receives the pretrained embeddings used in the previous exercises. These embeddings are high quality since they were trained using more data. Furtheremore, I replaced the LSTM by a Bidirectionnal LSTM. Indeed, LSTM layer allows to capture the information in one direction, but going the other way round brings also information about the phrase. To prevent overfitting, I used dropout with rate 30%. I have tested several optimizers but I did not notice any big difference for "good" optimizers such as RMSprop, Adam, or ADAGRAD, so I decided so keep RMSprop.

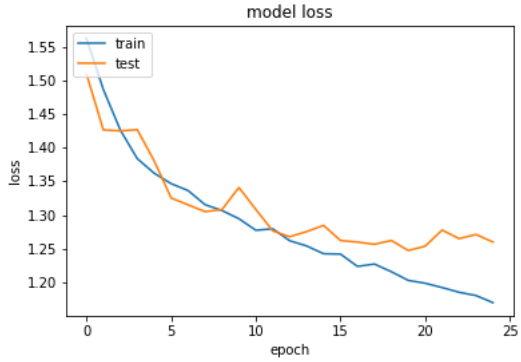

Figure 4: Loss of the new model over epochs

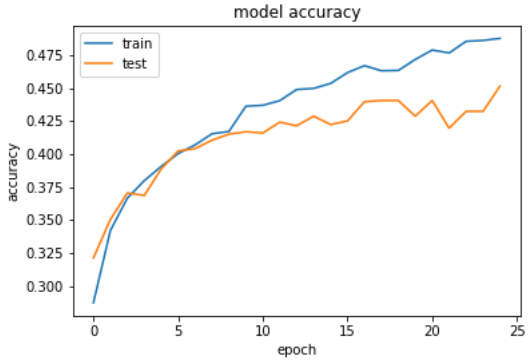

Figure 5: Accuracy of the new model over epochs

In the notebook, I run this network for a quite high number of epochs, i.e. 25. This is justified by the fact that the accuracy on the dev set is stagnating after 20 epochs, but

the loss on the dev set and the loss on the training set are still decreasing. Of course, the accuracy on the training set improves over epoch, but I don't think we are dealing with overfitting, since the dev loss is decreasing. So I ran the network over a large number of epochs, hoping that this will give better results on the test set. The submission for this part is given in the file `pretrainedbidirectional_lstm_y_test_sst.txt`

With this new network, I obtained an accuracy of 0.50 on the train set and an accuracy of 0.44 on the dev set. To conclude this report, an idea to improve even more the performance would be to add an attention mechanism, a widely-used technique in NLP.