Deep Learning for Natural Language Processing

-

Project

Sarah Lina HAMMOUTENE

February 2018

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.

1 Multilingual word embeddings

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

$$W^* = \underset{W \in O_d(R)}{argmin} ||WX - Y||_F^2 = UV^T; with UV^T = SVD(YX^T)$$

Proof. To derived the method we first expand $\|WX - Y\|_F^2$

$$\begin{aligned}\|WX - Y\|_F^2 &= tr(WX - Y)(X^TW^T - Y^T) \\ &= tr(WX)(X^TW^T) - tr(WXY^T) - tr(YX^TW^T) + tr(YY^T) \\ &= \|WX\|_F^2 + \|Y\|_F^2 - 2tr(YX^TW^T)\end{aligned}$$

Thus, minimizing $||WX - Y||_F^2$ equals to maximizing $tr(YX^TW^T)$.
Let $U\Sigma V^T$ be the singular value decomposition of YX^T.

$$\begin{aligned}tr(YX^TW^T) &= tr(U\Sigma V^TW^T) \\ &= tr(\Sigma V^TW^TU)\end{aligned}$$

We define Z as $Z = V^TW^TU$, Z is orthogonal since it is the product of orthogonal matrices.
The objective is thus to maximize $tr(\Sigma Z)$ through the choice of W. Because Σ is diagonal we have $tr(\Sigma Z) = \sum_{i=1}^{n} \Sigma_{ii} Z_{ii}$. The Σ_{ii} are non-negative and Z is orthogonal for any choice of W.
The maximum is achieved by choosing W such that all of $Z_{ii} = 1$ which implies $Z = I$. So an optimal W is UV^T.

2 Sentence classification with BoV

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

1. Logistic Regression

After tuning the L2 regularization on the dev set, we obtain the following results:

```
BoV-mean (average of word vectors):
-----------------------------------
Accuracy of the training set: 0.49
Accuracy of the dev set: 0.44
Best C: 1


BoV-idf (weighted-average of word vectors):
-------------------------------------------
Accuracy of the training set: 0.49
Accuracy of the dev set: 0.42
Best C Idf:0.5
```

Figure 1: Results of the logistic regression using BoV-mean and BoV-Idf

We can see that the best validation accuracy is obtained with the average of word vectors.

2. Models Comparison

In this part we experiment different classifiers: Decision Tree, Random Forest, SVM, Gradient Boosting, Logistic Regression and MLP. We use those classifiers with both BoV mean and BoV idf. The results are the following:

```
################################################################################################
######                1. BoV-mean (average of word vectors):                                ######
################################################################################################


Testing DecisionTreeClassifier
====================================
Learing time 4.50616693497s
Predicting time 0.0692739486694s
Accuracy 0.28792007266121705


Testing RandomForestClassifier
====================================
Learing time 1.16601610184s
Predicting time 0.00454807281494s
Accuracy 0.31607629427792916


Testing SVC
====================================
Learing time 53.8812880516s
Predicting time 3.47730088234s
Accuracy 0.2706630336058129


Testing GradientBoostingClassifier
====================================
Learing time 166.208421946s
Predicting time 0.0393218994141s
Accuracy 0.4159854677565849


Testing LogisticRegression
====================================
Learing time 5.06242799759s
Predicting time 0.002137184143075s
Accuracy 0.44323342415985467


Testing MLPClassifier
====================================
Learing time 28.6762418747s
Predicting time 0.0101850032806s
Accuracy 0.3787465940054496
----------------------------------------
```

Figure 2: Results of the classifiers on the dev set using BoV-mean

```
################################################################################################
######              2. BoV-idf (weighted-average of word vectors):                          ######
################################################################################################


Testing DecisionTreeClassifier
====================================
Learing time 4.40980601311s
Predicting time 0.0193340778351s
Accuracy 0.2706630336058129


Testing RandomForestClassifier
====================================
Learing time 1.72169494629s
Predicting time 0.0112888813019s
Accuracy 0.3151680290644868


Testing SVC
====================================
Learing time 53.9700307846s
Predicting time 4.28054785728s
Accuracy 0.3923705722070845


Testing GradientBoostingClassifier
====================================
Learing time 229.294970036s
Predicting time 0.0158298015594s
Accuracy 0.3923705722070845


Testing LogisticRegression
====================================
Learing time 4.60899710655s
Predicting time 0.00233197212219s
Accuracy 0.4178019981834696


Testing MLPClassifier
====================================
Learing time 34.0905849934s
Predicting time 0.00441098213196s
Accuracy 0.37693006357856496
----------------------------------------
```

Figure 3: Results of the classifiers on the dev set using BoV-idf

3. Parameters tuning

To improve the accuracy of the models, we have decided to tune the parameters. The chosen model is Random forest using both BoV mean and BoV idf.
After running the optimization, we obtain the following dev accuracy: As we

```
Parameters tuning for random forest:
----------------------------------------
('n_estimators : ', 500)
('max_feature : ', 'auto')
('min_samples_leaf : ', 3)
('min_samples_split : ', 5)
('max_depth : ', 50)
0.41900822007260796
```

Figure 4: Parameters tuning for random forest

```
Parameters tuning for random forest:
----------------------------------------
('n_estimators : ', 500)
('max_feature : ', 'auto')
('min_samples_leaf : ', 5)
('min_samples_split : ', 15)
('max_depth : ', 50)
0.4178395175141354
```

Figure 5: Parameters tuning for random forest

can see, we obtain the same score as with the logistic regression

3 Deep Learning models for classification

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

We build a network uses that the efficient Adam gradient descent optimization algorithm with **"categorical cross entropy"** as loss function. We choose this particular loss function since we have a multi-class classification problem where since each of our sentences can only belong to one category.

The categorical cross entropy is based on the fact that any directly decodable coding scheme for coding a message to identify one value x_i out of a set of possibilities X can be seen as representing an implicit probability distribution $q(x_i) = 2^{-l_i}$ over X, where l_i is the length of the code for x_i in bits. Therefore, cross entropy can be interpreted as the expected message-length per datum when a wrong distribution Q is assumed while the data actually follows a distribution P. That is why the expectation is taken over the probability distribution P and not Q.

$$\begin{aligned} H_{(p,q)} = E_p[l_i] &= E_p\left[\log \frac{1}{q(x_i)}\right] \\ &= \sum_{x_i} p(x_i) \log \frac{1}{q(x_i)} \\ &= -\sum_{x} p(x) \log q(x) \end{aligned}$$

2. Plot the evolution of train/dev results w.r.t the number of epochs. As we can see, the train accuracy increases with the increase of the number

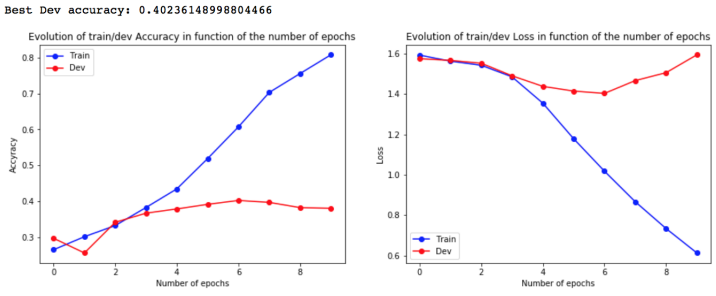

Figure 6: Evolution of train/dev results w.r.t the number of epochs

of epochs while the loss is decreasing. For the dev test, we can see that the accuracy start increasing and then remains constant starting from number of epoch = 3 same while the loss decreases and then increases from epoch = 5. The best accuracy given this model is **40%**.

3. Be creative: use another encoder. What are your motivations for using this other model?

Convolutional neural networks excel at learning the spatial structure in input data, and since the SST data have a spatial structure in the sequence of words in the sentences, the CNN may be able to pick out invariant features for the classification. This learned spatial features may then be learned as sequences by an LSTM layer. We can also add a one-dimensional CNN and max pooling layers after the Embedding layer which feed the consolidated features to the LSTM.

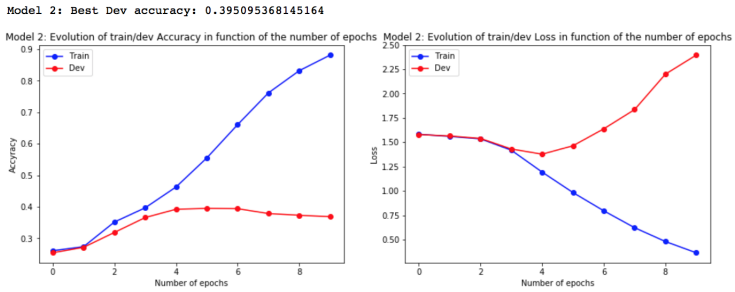

Figure 7: Results of the classifiers using BoV-idf

We can see that the score did not improve by using this model. One wants to try another approach by using a bidirectional LSTM that will run the inputs in two ways, one from past to future and one from future to past and what differs this approach from unidirectional is that in the LSTM that runs backwards preserve information from the future and using the two hidden states combined we are able in any point in time to preserve information from both past and future. The results given by this third model are the following:

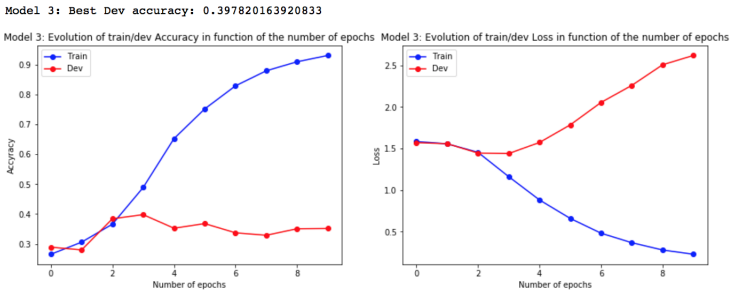

Figure 8: Results of the classifiers using BoV-idf