Homework 1

1 Monolingual embddings

2 Multilingual word embeddings

Question: Prove the following:

$$W^* = \arg\min_{W \in O_d(\mathbb{R})} ||WX - Y||_F = UV^T, \text{ where } U\Sigma V^T = SVD(YX^T).$$

Answer:

- First, note that $\arg\min_{W \in O_d(\mathbb{R})} ||WX - Y||_F = \arg\min_{W \in O_d(\mathbb{R})} ||WX - Y||_F^2$
- Consider $||WX - Y||_F^2$:

$$\begin{aligned} ||WX - Y||_F^2 &= ||WX||_F^2 + ||Y||_F^2 - 2\langle WX, Y\rangle_F \\ &= ||X||_F^2 + ||Y||_F^2 - 2\langle WX, Y\rangle_F, \end{aligned}$$

  since $||WX||_F^2 = Tr(WXX^TW^T) = Tr(XX^TW^TW) = Tr(XX^T) = ||X||_F^2$, using the cyclic property of trace, and the fact that W is orthogonal.

- Now we can reformulate our problem:

$$\arg\min_{W \in O_d(\mathbb{R})} ||WX - Y||_F = \arg\max_{W \in O_d(\mathbb{R})} \langle WX, Y\rangle_F$$

- Consider $\langle WX, Y\rangle_F$:

$$\begin{aligned} \langle WX, Y\rangle_F &= Tr(X^TW^TY) \\ &= Tr(YX^TW^T) && \text{cyclic property} \\ &= Tr(U\Sigma V^TW^T) && \text{SVD decomposition} \\ &= Tr(V^TW^TU\sqrt{\Sigma}^T\sqrt{\Sigma}) && \text{cyclic property + } \Sigma \text{ has non-negative values} \\ &= Tr(M\sqrt{\Sigma}^T\sqrt{\Sigma}) && M \text{ - orthogonal as a multiplication of orthogonal matrices} \\ &= \langle M\sqrt{\Sigma}, \sqrt{\Sigma}\rangle_F \\ &\leq ||M\sqrt{\Sigma}||_F||\sqrt{\Sigma}||_F && \text{Cauchy-Schwartz inequality} \\ &= ||\sqrt{\Sigma}||_F||\sqrt{\Sigma}||_F && \text{as M - orthogonal} \\ &= ||\sqrt{\Sigma}||_F^2 \\ &= Tr(\Sigma) \end{aligned}$$

- We see that $\langle WX, Y\rangle_F$ reaches its maximum when $V^TW^TU\Sigma = \Sigma$. So we derive $W^* = UV^T$.

3 Sentence classification with BoW

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

Answer: To improve the result presented in the Table 1, I have tried different values of C parameter of the logistic regression. For weighted-average vectors $C = 0.7$ shows the best result, for encoding by mean of word vectors, the default C=1 is optimal. We see that the option with weighted-average vectors works better, so as a final result I took the prediction on test set using idf-weighted mean of word vectors encoding by the classifier trained on both train and validation sets.

Mode	Train accuracy, %	Validation accuracy, %
Average	42.9	39.1
Weighted-average	46.9	42.8

Table 1: Logistic regression results for BoV.

As a model for the bonus question I took a neural network classifier by scikit learn (MLPClassifier). Obtained results are a bit better for average mode that these by logistic regression, but a bit worse for idf-weighted one: Table 2. The neural network classifier tends to overfit fast, so I used the early stopping option which interrupt training when validation loss stops improving.

Mode	Train accuracy, %	Validation accuracy, %
Average	46.3	41.1
Weighted-average	47.7	41.5

Table 2: MLPClassifier results for BoV.

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.

Answer: As we deal with a multiclassification task, I took categorical cross-entropy as the loss function:

$$loss = -\frac{1}{N}\sum_{i=0}^{N}\sum_{c=0}^{4}\mathcal{I}_{y_i \in C_c} log(p_{model}(y_i \in C_c)),$$

where N is number of observations, $\mathcal{I}_{y_i \in C_c}$ - indicator if ith observation belongs to cth class. As we use Softmax activation, $p_{model}(y_i \in C_c) = \frac{e_i^s}{\sum_{c=0}^{4} e_c^s}$, where s_i - the network score.

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

Answer: The results are shown on the figure 2. We see that the model learns only until the third epoch, after it overfits. Interesting, that this model classify validation set examples only for two classes which are the most popular in the training set. This may be solved by paying attention to class balancing, and I used it in the next question.

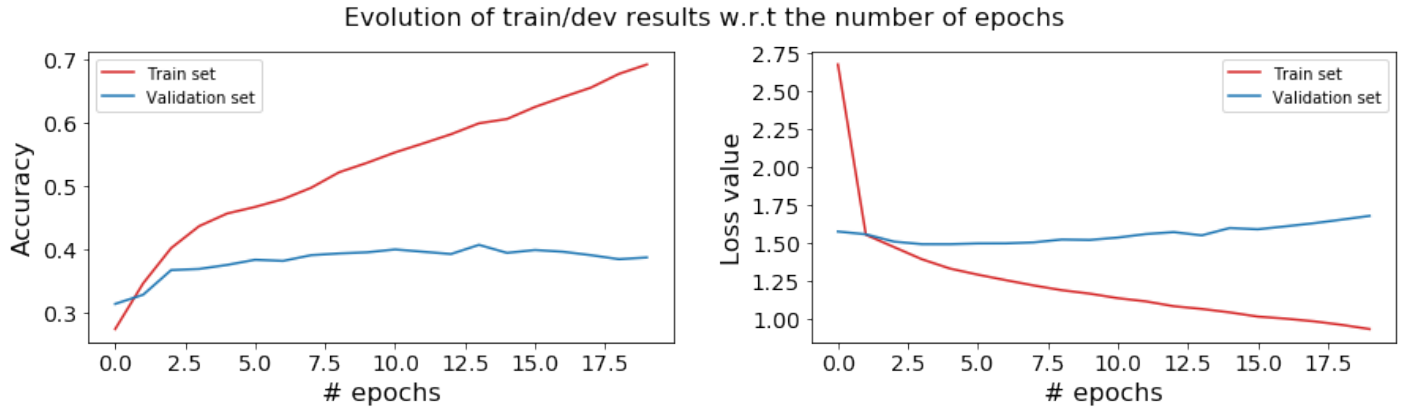

Figure 1: Evolution of train and dev results w.r.t the number of epochs.

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

Answer: What I have tried:

- Use pretrained by word2vec embeddings embedding layer.
- Balance classes defining classes weights.
- Make adaptive learning rate decreasing with number of epochs increasing.
- Use bidirectional LSTM.
- Use convolution layers followed by maxpooling.
- Use Tokenizer

These methods didn't give the significant improvement, but the combination of following methods showed the best result: pretrained embedded layer, classes weights, bidirectional LSTM.