# HW2

Gabriel ROMON

## 2 Multilingual word embeddings

Remember that the Frobenius norm on $\mathbb{R}^{d,m}$ is induced by the inner product $\langle A, B\rangle_F = \mathrm{tr}(A^T B)$. Consequently

$$\|WX - Y\|_F^2 = \|WX\|_F^2 + \|Y\|_F^2 - 2\langle WX, Y\rangle_F$$

Since $W$ is orthogonal, $\|WX\|_F^2 = \mathrm{tr}(X^T W^T W X) = \mathrm{tr}(X^T X) = \|X\|_F^2$ and by the cyclic property of trace $\langle WX, Y\rangle_F = \mathrm{tr}(X^T W^T Y) = \mathrm{tr}(W^T Y X^T) = \langle W, YX^T\rangle_F$, hence

$$\|WX - Y\|_F^2 = \|X\|_F^2 + \|Y\|_F^2 - 2\langle W, YX^T\rangle_F$$

We may therefore look for $\arg\max_{W\in O_d(\mathbb{R})}\langle W, YX^T\rangle_F$.
Let $U, V \in O_d(\mathbb{R})$ and $\Sigma \in \mathbb{R}^{d,d}$ a diagonal matrix with positive entries such that $YX^T = U\Sigma V^T$. From the cyclic property of trace,

$$\langle W, U\Sigma V^T\rangle_F = \langle U^T W V, \Sigma\rangle_F$$

Note that $Z := U^T W V$ is orthogonal as a product of orthogonal matrices and $\langle Z, \Sigma\rangle_F = \sum_{i=1}^d \Sigma_{ii} Z_{ii}$. Since $Z$ is orthogonal its entries are $\leq 1$ in absolute value and since the singular values $\Sigma_{ii}$ are $> 0$ we infer that $\langle Z, \Sigma\rangle_F \leq \sum_{i=1}^d \Sigma_{ii}$ with equality when $Z = I_d$, which yields the optimal $W$ as $UV^T$.

## 3 Sentence classification with BoV

We tune the value of the regularization parameter on the dev set, with and without IDF weighting. The accuracies of the corresponding best models are reported in Figure 1.

| | Average | IDF-weighted average |
|---|---|---|
| Train set | 48.68% | 48.85% |
| Dev set | 43.05% | 41.78% |

Table 1: Accuracy of the best model

## 4 Deep Learning models for classification

(a) The loss that yielded the best results is the categorical cross-entropy. For a single data point $x$ with true label $k$ it is equal to $-\log p$ where $p$ is the predicted probability that $x$ belongs to class $k$. It may also be written as

$$-\sum_{i=0}^{4} 1_{c(x)=k} \log(p_k)$$

where $(p_0, \ldots, p_4)$ is the vector of probabilities returned by the model for the instance $x$.

(b) I chose not to use IDF weighting. The accuracies on the train and dev set for each epoch are reported in Figure 1. Since there is not so much data very few epochs (around 4) are sufficient to reach convergence.

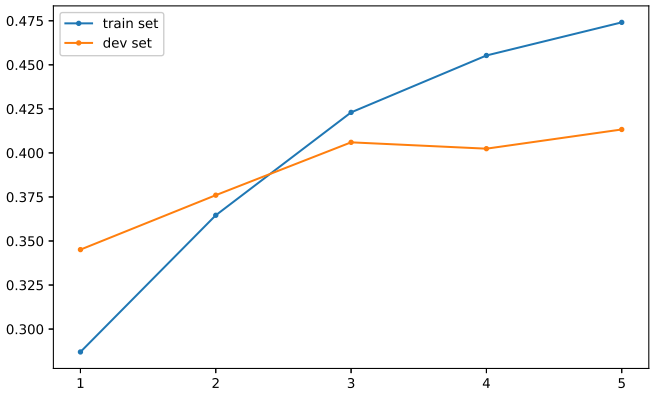


Figure 1: Accuracies on the train and dev set

(c) The architecture I experimented with is a 1D CNN followed by Maxpooling and a LSTM. Because of the Maxpooling it is much faster to train than the vanilla LSTM. Some dropout is added to prevent overfitting. For some runs it slightly outperformed the previous architecture.