ATCS Project 1 Report. InferSent-based architectures

Arsen Sheverdin
University of Amsterdam
arsen.sheverdin@student.uva.nl

Table 1: Performance on transfer tasks (accuracy)

	Transfer Tasks (accuracy)							
Model	**MR**	**CR**	**SUBJ**	**MPQA**	**SST**	**TREC**	**MRPC**	**SICK-E**
AWE	62.31	81.33	99.6	87.59	79.01	75.83	72.01	58.6
LSTM	66.48	78.82	99.6	88.07	78.78	69.61	72.84	82.8
BiLSTM	67.72	79.45	99.6	89.1	78.9	70.1	72.89	85.4

Table 2: Performance on transfer tasks and NLI task (accuracy)

		NLI	Transfer Tasks	
model	**dev**	**test**	**micro**	**macro**
AWE	74.12	72.31	82.23	77.035
LSTM	81.41	79.96	83.06	79.63
BiLSTM	82.12	79.97	85.11	80.39

One can see clearly from table 2 that the LSTM model performs surprisingly better on transfer tasks, than on the NLI, on which it was trained which can be attributed to high performance on SUBJ and MPQA tasks. The classifier of the model was trained with dropout value of 0.1, leading to a better generalization, which could also be one of the reasons for such a high performance on the tasks outside of the NLI task. What was surprising also is that the values on SUBJ tasks outperformed the values the ones reported in the paper.

It is apparent that AWE under-performs for computationally demanding metrics such as SICK-E, due to probably lower complexity of the model.

One trend that is noticable is that AWE performs better on binary or multi-class classification tasks such as: MR,SST,MPQA,CR. It is particularly interesting that on CR and MPQA metric AWE outperformed more complex architectures. It could be that by under-performing on NLI task, it gained more generalizable embeddings, that are very efficient across those tasks. It also explains why it performed so poorly on SICK-E: SICK-E can be considered as an out-of-sample NLI task.

Regarding BiLSTM, main conclusion is that it did not have a strong impact on the performance in general, despite of considerably longer training time. One can only see strong 3% improvement for SICK-E task, while having marginal improvement on NLI task itself.