Lab 5: CNNs for Sentence Classification

In this assignment, you will extend the ConvNet you have as an example in the *jupyter notebook*. In particular, you will change the activation function, implement a new regularizer and use deeper models.

In the skeleton code (see below for clarifications) we provide an implementation of a model similar to Kim Yoon's Convolutional Neural Networks for Sentence Classification [1]. The model presented in the paper achieves good classification performance across a range of text classification tasks (like Sentiment Analysis) and has since become a standard baseline for new text classification architectures.

The dataset we use in this assignment is the Movie Review data from Rotten Tomatoes, one of the datasets also used in the original paper. The dataset contains 10,662 example review sentences, half positive and half negative. The dataset has a vocabulary of size around 20k. Note that since this dataset is pretty small we are likely to overfit with a powerful model. Also, the dataset does not come with an official train/test split, so we simply use 10% of the data as a dev set. The original paper reported results for 10-fold cross-validation on the data.

In the following sections we describe what you have to do for completing the assignment.

1 Activation

Replace the *tanh* non-linearity by a *rectified linear* (RELU) unit [2], where RELU(x) = max(0,x). When you train these models, it might help to change the initialization [3]. **Make sure that you choose the learning rate carefully**. If it is too small, training will be slow whereas if it is too big, you will not learn anything useful.

2 Regularization

Add Dropout [5, 7], a regularizer that has been applied with great success on multiple tasks, including object recognition [6] and language modeling [8]. During training, drop each hidden layer unit with a probability of 1/2. In general, the units you remove will be different for every training sample, although you may also use a single mask for every mini-batch [4]. When testing your model, you should use all hidden units, but multiply the outgoing weights by 1/2 to account for this change.

3 Depth

Finally, try to adjust the number of convnet layers[1]. Depth allows you to learn multiple levels of representation. Play with the architecture of your network (depth, width, filter size, activation, regularization) to see what works best.

1 Try at least increasing depth in one layer.

4 Write-up

You should write a short report (maximum 2 pages!) describing your experiments and results. Also try to analyze the behavior of your systems. Which changes were the most effective? How would you further improve your models?

5 Code and data files for the assignments

For completing the assignment we provide the skeleton code that consist in several files. You just need to change "*text_cnn.py*" python script, which is the one that describes the CNN model used in the assignment. The rest of the python script are used to invoke the model for training ("*train.py*") or testing ("*eval.py*"). Please, refer to *README.md* to know about the usages of the training and testing scripts.

Data is stored in "*data/rt-polaritydata*" folder, and contains negative and positive examples in separated files: "rt-polarity.neg" and "rt-polarity,pos".

The code serializes log data under "runs" folder. You can launch tensorboard to inspect log data:

```
$ tensorboard --logdir=path/to/log-directory
```

Finally, we add jupyter notebook file ("*CNN_for_Images_example.ipynb*") that might be helpful to understand how the CNNs work.

6 References

[1] Kim, Yoon. "Convolutional neural networks for sentence classification." arXiv preprint arXiv:1408.5882 (2014).

[2] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In International Conference on Artificial Intelligence and Statistics, pages 315323, 2011.

[3] Ben Graham, Jeremy Reizenstein, and Leigh Robinson. Efficient batchwise dropout training using submatrices. CoRR, abs/1502.02478, 2015.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. arXiv preprint arXiv:1502.01852, 2015.

[5] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhut-dinov. Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580, 2012.

[6] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pages 10971105, 2012.

[7] Nitish Srivastava,Geoffrey Hinton,Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 15(1):19291958, 2014.

[8] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. Recurrent neural network regularization. ArXiv preprint arXiv:1409.2329, 2014.