Explanation of solution

Georgios Tzannetos

March 17, 2018

1 Introduction

This problem belongs to the case of text classification. Two ideas were evaluated. The first one, was to use an LSTM Recurrent neural network, which have been shown to perform well in sequences of data, like a text. The second method was to use a Convolutional neural network, which also has been proven to perform well for text classification. In Yoon Kim's paper "Convolutional Neural Network for Sentence Classification", it is showed that a simple CNN can perform really well on various benchmarks, using also pretrained word vectors. Although, in our case the fact that our data are obfuscated, makes the above method not viable. However, the patterns between the characters are preserved. For that reason we can use a CNN for classification of the sentences on the character level, as shown on the paper "Character-level Convolutional Networks for Text Classification" authored by Xiang Zhang, Junbo Zhao, Yann LeCun.

1.1 Explanation

To solve the given problem, I was based on the Character Level CNN text classification and the code was written on the PyTorch framework. The code folder consists of four files:

- `data_loader.py`, where the dataset class is created, and moreover one-hot encoding of the lines.
- `model.py`. Here the CNN model is created. Different architectures were experimented. I started with the initial architecture introduced in the paper, but then tried simpler architectures. Shortly, I have finally used Conv→ReLU→MaxPool, 3 times followed by a FC → ReLU→FC→Softmax output layer.
- `train.py`. This is the file responsible for the training. At this point I have to mention that for the characters the alphabet is needed. I used the default one as in the paper, including letters plus special characters and newline, in total 70, although in our dataset only 26 letters and newline appear. For the loss function the negative log likelihood was used.

- `test.py`. Runs model and predicts for each line of the test data the respective novel.

Furthermore, for the training the Adam optimizer was used, with learning rate 0.0007, and for batch size 64 was used. Also a splitting of the training dataset to 80 % training set and and to 20 % validation set. The accuracy and loss of the validation set was tested after 50 batches as the training was proceeding. Also the use dropout is suggested in the paper. However, since I used a simpler architecture, I have omitted the dropout, without any negative effects on the generalization. An interesting observation was that the distribution of the training data was not close to equal, for example class 0 appears only 543 times, whereas class 7 appears 5097 times. That is not ideal and should be treated although is this case was ignored.

The training was done until the model overfits, as can be seen in Figures 1 and 2.

We can clearly see at Figure 1 that after the 80th evaluation the model's loss function starts to increase, indicating that it overfits. So the model for the prediction was chosen before that happens(early stopping).

2 Results

I include some metrics at Table 1, computed while training, regarding the *validation set*, which led to the choice of the best model. The same values for the training set at that point are shown at Table 2. Moreover metrics, such as precision and recall, have been computed for that case and can be seen in Table 3.

With all that in mind I expect the accuracy on the test set to be in the range from $\approx$ 65-70 %

Comment: By looking at the results (training accuracy can improve a lot) it is probable that a more complicated, deeper model could have led to better results, but then dropout and other regularization techniques should have been introduced to avoid overfitting.

References

[1] Xiang Zhang, Junbo Zhao and Yann LeCun *Character-level Convolutional Networks for Text Classification*, in CoRR,2015

[2] Yoon Kim *Convolutional Neural Networks for Sentence Classification*, Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) ,pages 1746-1751, October 2014, Doha, Qatar.

Figures and Tables

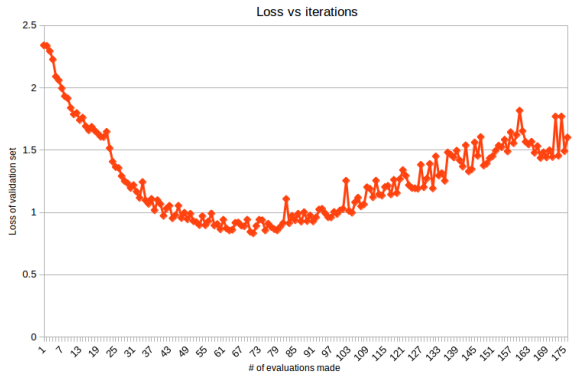

Figure 1: Loss vs evaluations for validation set

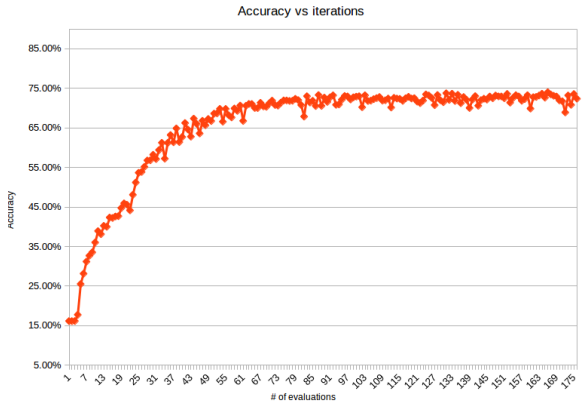

Figure 2: Accuracy vs evaluations for validation set

Table 1: Validation set loss and accuracy

Evaluation	Loss	Accuracy	Correctly Classified
	0.855490	72.347 %	4704/6502

Table 2: Training set loss and accuracy

Evaluation	Loss	Accuracy
	0.475587	83.38 %

Table 3: Detailed metrics for validation set

Label	Precision	Recall	F-Score
0	65.3 %(49/75)	51.0 %(49/96)	57.3%
1	72.1 %(483/670)	68.7 %(483/703)	70.4%
2	80.7 %(117/145)	43.7 %(117/268)	56.7%
3	71.9 %(561/780)	71.0 %(561/790)	71.5%
4	81.1 %(374/461)	79.6 %(374/470)	80.3%
5	88.2 %(402/456)	85.9 %(402/468)	87.0%
6	80.1 %(595/743)	68.8 %(595/865)	74.0%
7	70.3 %(933/1327)	88.9 %(933/1049)	78.5%
8	85.1 %(606/712)	83.4 %(606/727)	84.2%
9	57.1 %(68/119)	34.5 %(68/197)	43.0%
10	50.9 %(356/700)	59.7 %(356/596)	54.9%
11	51.0 %(160/314)	58.6 %(160/273)	54.5%