Detecting Fake News

Abhay Chennagiri

***Abstract*—Convolution Neural Networks(CNN) have proven to be very successful in text classification tasks. Given the task of classifying the fake news dataset into 6 categories specifying varying levels of credibility is essentially a form of sentiment analysis. For this assignment, I have tested the dataset on two models namely CNN model and hybrid CNN model where the former uses only the statement while the latter takes into account the metadata information also for classifying.**

I. Introduction

The amount of fake content that is being generated in today's world is astounding. It is very appalling that fake impressionable content is being propagated using all forms of social media and influence people's opinion causing a lot of disharmony in the society. The US national elections is also allegedly a victim of this menace.

In this HW, we tackle the problem of fake news detection on a dataset containing political statements and other metadata.

II. Data

The dataset is obtained from PolitiFact.com, a a Pulitzer Prize-winning website. The LIAR dataset includes 12.8K human labeled short statements from Politifact.coms API's,and each statement is evaluated by a PolitiFact.Ccom editor for its truthfulness.

There are six fine-grained labels for the truthfulness ratings: pants-fire, false, barelytrue, half-true, mostly-true, and true. The distribution of labels in the LIAR dataset is relatively well-balanced: except for 1,050 pants-fire cases, the instances for all other labels range from 2,063 to 2,638.

III. Model

I have implemented two models both of which employ CNN. The first one uses only the statement label while the second one is a hybrid model which uses the metadata information. I have experimented using a subset of the meta-data information. These models have worked out to be the best as stated in [1].The only difference in my model w.r.t to the model implemented in [1] is that I have used the BiLSTM before using the ConvNet layer. Figure 1 describes the model.

IV. Results

I spent some time preprocessing the data and adjusted it according to the results on validation set.The size of my vocabulary was 16110. I implemented the models in Tensorflow.

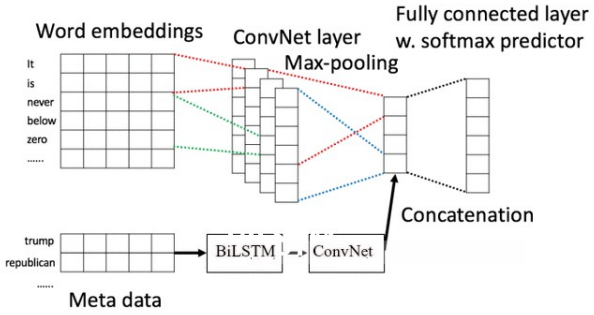

Fig. 1: Hybrid CNN Model

For both the models, I have used the 300 dimensional Glove word embeddings pre-trained on Wikipedia dataset containing 6B tokens. I tuned the hyper-parameters on the validation set. The best filter sizes for the CNN model was (3,4,5). In all cases, the number of filters was set to 128. The dropout keep probability was set to 0.5 for training and 0.8 for the validation set.The batch size for stochastic gradient descent optimization was set to 64. The number of epochs was set to 10. For the second model, we used filter sizes (3,4) for the meta-data and the number of filters was set to 20.

TABLE I: Simple CNN vs Hybrid CNN

Model	Valid Set	Test Set
CNN	0.247	0.258
Hybrid (all text)	0.257	0.279

V. Conclusion

It was really interesting tackling this problem. The result proves again that CNN's are the one of the best when it comes to text classification tasks. I would also like to mention about the running times of the model. The CNN models ran way faster compared to the neural network models that we had implemented as a part of last home work. I could do a lot of hyper parameter tuning thanks to it. If I had more time, I would try and implement an attention based model and compare the results.

References

[1] Wang, William Yang. "" Liar, Liar Pants on Fire": A New Benchmark Dataset for Fake News Detection." arXiv preprint arXiv:1705.00648 (2017).

[2] "Http://www.wildml.com/2015/12/implementing-a-cnn-for-text-classification-in-tensorflow/." Web log post. N.p., n.d. Web

Advisor: Dipl.–Ing. Firstname Lastname, Lehrstuhl für Nachrichtentechnik, TUM, WS 2050/2051.