Neural NotWork Work Journal

Week of 2/9 Journal:

Meeting day: 2/9/2020

Time: 3:00 PM

Location: DSI 324

Members Attendance: All presented

We first discussed what the topic was and tried to load data in python. The dataset was quite large and we were not familiar with the format of the data. Thus, we spent an hour trying to understand how to load the data. Our final goal is to build a robust model to correctly perform optical character recognition on Bengali.

We proposed and agreed on the works below of this following week:

Data collection: finished, Hanxiao downloaded data from the site

EDA: In progress, Yi will do this in the following week.

Domain knowledge: we found and Weihao will read the following papers

1. https://ieeexplore.ieee.org/document/8281853
2. https://www.hindawi.com/journals/cin/2018/6747098/
3. https://arxiv.org/abs/1712.09872

Week of 2/16 Journal:

Meeting day: 2/16/2020

Location: Online discussion

Members Attendance: All presented

Pre-EDA discussion

Some of our members are travelling for Datathon this weekend and thus our discussion remained online. We mainly discussed the sample codes on Kaggle and will continue our discovery in the next few days. Since the dataset contains only images, EDA seems a tricky thing to perform and we have a thought to plot the distribution of each word and try to seperate images into RGB channels.

Week of 2/23 Journal:

Meeting day: 2/23/2020

Location: Sayes Hall

Members Attendance: All presented

Model selection discussion

We discussed model selection issues and CNN seemed to be the optimal solution to the problem for the following two reasons.

- The dataset contains only images and some of them rotate or shift.
- We need multiple outputs of the model. Since the convolutional layers of CNN are able to extract the major features of the images, a simple method to solve this problem is to construct three dense output layers based on the convolutional layers.

After completing Datathon this weekend, we had a meeting about our plan for next week and we planned to focus on finding the optimal hyperparameters of the model and training the model.

Week of 3/1 Journal:

Meeting day: 03/01/2020

Location: DSI

Members Attendance: All presented

Model selection discussion 2

This week, we have attempted several useful models including AlexNet, ResNet, and RNN. We first attempted to use some sample data for each model which could give us an insight into how each model performs. We saw that all the sample models had reasonable accuracies which gave us an insight that we should keep trying on these models. After that, we applied full data to each model and through a tremendous amount of time on tuning on parameters such as activation, padding, and a possible number of layers, the results were somewhat similar among the models but there were still improvements that could be augmented. In these few days, we will try to apply some other new methods to the textbooks and start working on our final blog post.

To be specific, the method we will try are CNN + GABOR + DROPOUT, All Convolutional Network, Network in Network (NiN), Densely Connected Network (DenseNet).

We also want to do some feature engineering and customization on loss function, if time permitted.

Week of 3/5 Final Journal:

Meeting day: 03/05/2020

Location: DSI

Members Attendance: All presented

This week we were finalizing our models' improvement and started editing our final blog post page. We all agreed upon the use of 30 epochs for all the models as accuracy scores would converge after this amount of epochs steps. We had issues with FractalNet and DenseNet. In the case of FractalNet, we first constructed a joint Layer class and successfully constructed the model with such a customized layer. However, it was another story when saving and loading our model on Kaggle. We spent hours solving the incompatibility of keras and tensorflow.keras and the issue remained after hours of research. Keras did not allow us to save and then load the model constructed with our customized module. Tensorlfow.keras, in contract, was theoretically allowed to load a customized model, yet this package did not have the Layer class as template that we could do inheritance. We also tried to train our FractalNet model directly on Kaggle, but unfortunately, that did not work also. In this case, we could not register our FractalNet model to Kaggle.

We were almost done with DenseNet as it should be built upon ResNet as they share similar architectures. We first implemented a ResNet and it worked perfectly as the we hit 95.58% on Kaggle. This gave us a good sense that DenseNet would be a better approach. However, due to the difficulty in FractalNet as mentioned above, we could have enough time to find another time for the whole group to meet and discuss the implementation of DenseNet.

The allConv model was one of the first models we tried after the baseline model, it has a relatively simple structure. The number of layers, including batch normalization and pooling, was just 20. Therefore, the training time is short. It achieved 92.2% accuracy on Kaggle. The Deep CNN was a reasonable try when we saw a decent performance of allConv model, and it did improve the accuracy over 95% benchmark.

We finally summarized our results into a blog and a video. They have been posted on Medium and YouTube (bit.ly/2TwZAmo and https://youtu.be/mxfHMGFszZI). We spent the entire afternoon on Wednesday assembling our thoughts and filmed the screencast.

In our future work, To further improve the accuracy, feature engineering is necessary. Hence, we will have a discussion with domain experts and study some basic linguistic knowledge by ourselves to enhance our way of data processing. Moreover, ensemble methods should be applied to improve the performance. The major issue of bagging is that it requires much more RAM than the limit set by Kaggle, and thus we will firstly optimize our way of memory utilization. We also want to try stacking models together if time permitted. Finally, there are lots of open source trained models available on Kaggle and GitHub. We will try to understand their thoughts and experiences and use transfer learning to further enhance their models.

Also, we will try to continuously improve our models on Kaggle to get the extra credits.