BIRLA INSTITUE OF TECHNOLOGY AND SCIENCE - PILANI

NEURAL NETWORKS AND FUZZY LOGIC

COURSE PROJECT

MEME SENTIMENT ANALYSIS

Authors:
Aditya Ahuja
Rishabh Bajpai
Shubhad Mathur

November 25, 2019

MEME SENTIMENT ANALYSIS

Project Report

Aditya Ahuja*, Rishabh Bajpai*, Shubhad Mathur*

Birla Institute of Technology and Science - Pilani, Goa

Abstract. Meme Classification is a challenging and intriguing task. This report outlines our approach towards building a model for the problem. We have tried self-trained as well as fine-tuned pre-trained models in both Pytorch and Tensorflow.

1 Pre-Processing

The data provided was highly noisy. Pre-processing steps involved cleaning the text present in the csv file as well as recovering some image entries. This section includes the various ways the input was transformed for training our DL model.

1.1 Homogenizing the data

On a careful analysis of the .csv files, we noticed many of the entries were mixed. We identified these rows by counting the number of commas in the column 3 entries, and split them up manually to avoid issues. Apart from this, many entries had their labels shifted column-wise. This was noticed when plotting the class histograms, which had extra classes.

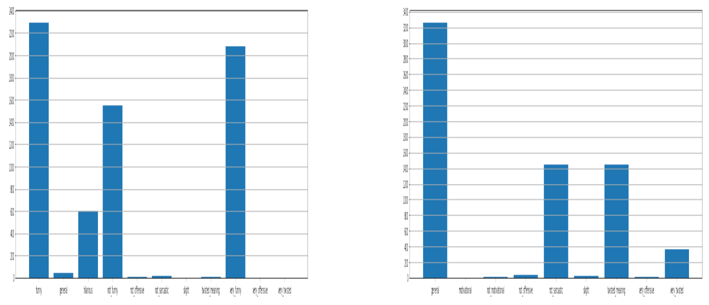

Fig. 1. Label histogram for tasks was found to have extra entries

Other changes included Meme image file extension Renaming and Re-mapping with their corresponding entries in the CSV file. It was noticed that the 3rd column was richer in information hence, only that was chosen as the text input for training purposes. The labels were converted into sparse categorical encoding. The text was cleaned using the following methods:

1. Removing special symbols. Many special symbols that are not a part of the written English language would add unnecessary noise to the input.
2. Removing website information. Website domain names were often scattered between the input, these included .com, .net, and other extensions.

1.2 Dealing with class Imbalance

Each task class had a good amount of imbalance. Training directly could cause over-fitting on a certain class, which would possibly lead to a bad F1 score.

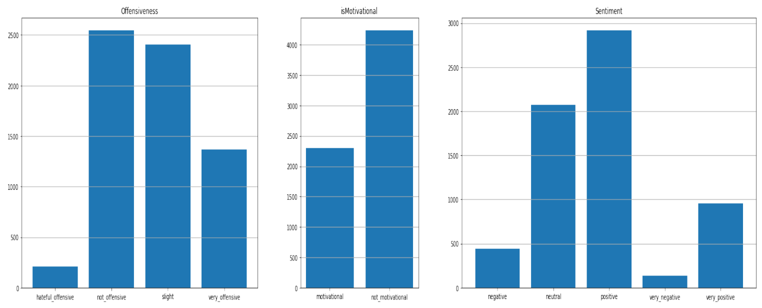

Fig. 2. Label histograms for each task

Baseline accuracies were calculated by assuming majority class prediction in each task, to help evaluate the model's score properly. These were as follows.

. `Hilariousness`	: `0.351050150021431`
. `Intentions`	: `0.501643091870267`
. `Offensiveness`	: `0.387912558936991`
. `Motivation`	: `0.647235319331333`
. `Sentiment`	: `0.447206743820545`

We looked into a few techniques from the library *imblearn* such as *SMOTE* and *ADASYN*, following advice from [1]. We used these 3 techniques throughout the model selection process. While in some cases they did provide a better f1 score, in our best case there wasn't a noticeable increase. One such example can be seen from the figure. We believe that input noise is at the heart of this issue.

Figure 3 shows that all techniques give very similar results on training a simple LSTM model. Further test were conducted with BERT and GRU models, all giving similar results. So, Oversampling was not aggressively pursued further.

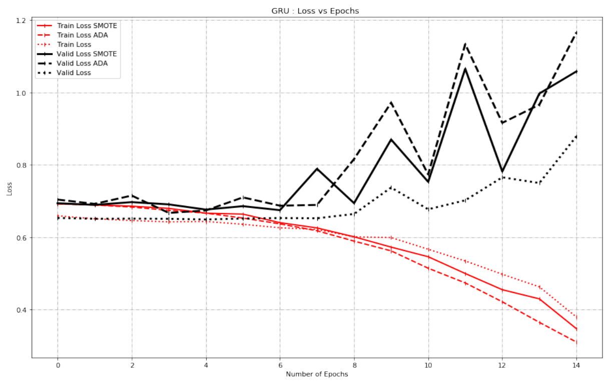

Fig. 3. Loss curves for LSTM. Dotted curve shows ADASYN, Line curve shows SMOTE. Solid line shows the normal class imbalance

2 Data Analysis

Before beginning to work on model structures, we analysed the data to gain insights that might streamline our work ahead. We began by looking for correlations within the data. For finding the correlation of tasks within themselves, each task was numbered in an ordering of intensity, eg. *Very_Negative* got a value of 0 while *Very_Positive* got a value of 5.

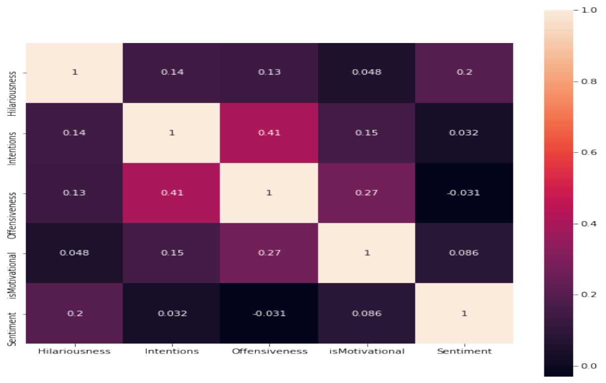

Fig. 4. Correlation Matrices for Labels of all classes.

Unfortunately, there weren't any highly correlated tasks. Sentiment and Hilariousness did share a correlation of 0.2. This showed that things that were funny were somewhat related to being positive. This meant we could look for models pre-trained on either task for our fine tuning. We also tried changing each label to a sparse vector, but the results weren't too great there either.

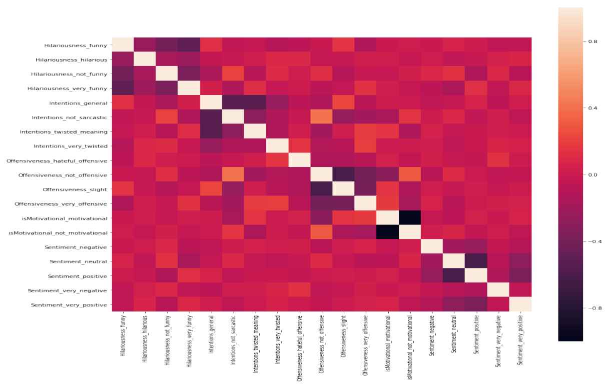

Fig. 5. Correlation Matrices for Labels of all classes converted to sparse vectors

Since memes always have a type of *template*. We figured, there could be a correlation from the template type to each of the tasks. We acquired the template type from the name of the meme (which was very consistent, perhaps because they came from one source).

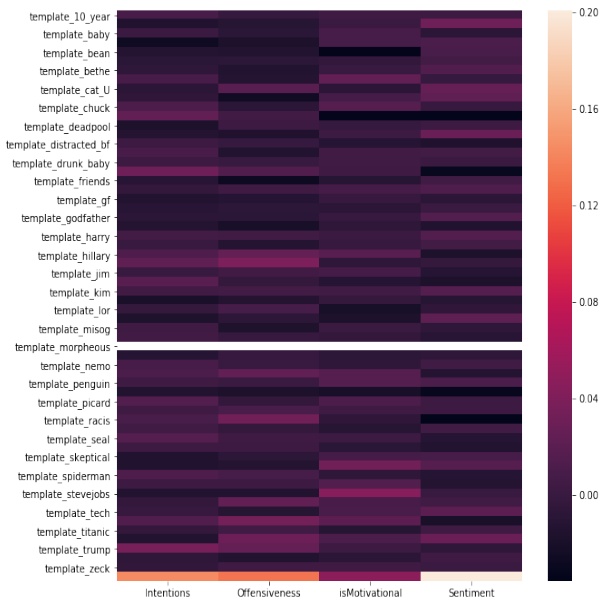

Fig. 6. Correlation Matrices for Template of the meme Vs the Tasks

The results were not very promising, with no particular template showing high correlation. Further Data analysis included finding the optimal length of sentences, which came to be either 512 words.

3 Model Selection

In this section we go over the various models that we tried for solving the tasks. Our approach was bottom-up. We started with single input models, and then performed a late fusion of 2 models. Across the board, models that were performing better on one task were performing equally well on the other tasks, so we choose to show graphs related to the motivation task only. Each model was trained on 2 optimizers, Adam [6] and Adadelta [13]. F1 score was used for evaluation with the 'micro' averaging strategy in sklearn. Some of the models here are from literature while some are just creative attempts to solve the problems.

3.1 Single-Model Attempts

Textual Input **Long Short-Term Memory Model**

A simple Long short-term Memory model seemed the most intuitive choice for classification of a sequence of text. We used a part of the model presented in [10]. Removing the dropout layers from the original model gave a better accuracy. While the model trained quickly, it was prone to getting stuck at local optimas. It also had a low accuracy and F1 score. The model also does not suit well with GPU, with a fairly large training time. So oversampling methods were not attempted here.

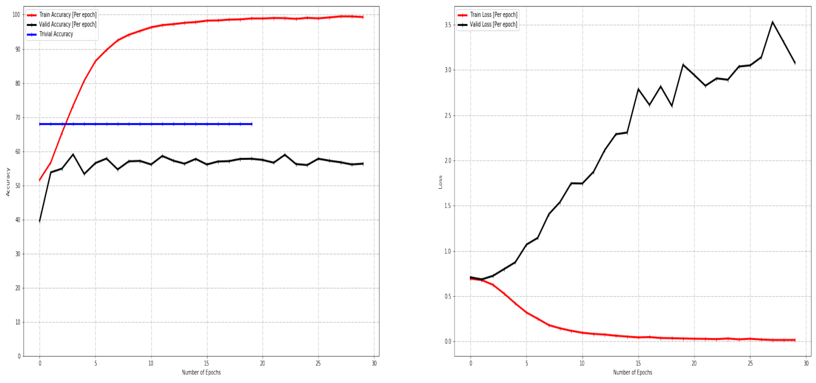

Fig. 7. Left : Accuracy Curve. Right: Loss curve.

Textual Input **1D-CNN based Model**

Inspired by use of CNNs in the following literature [2], [7], [10] and [5]. We decided to use the model in [10] for classification. We modified the model a little, the code for which is available in the repository. This model was quick to overfit, but did consider the minority class early on. Since this network trained quickly, we tried Oversampling methods as well.

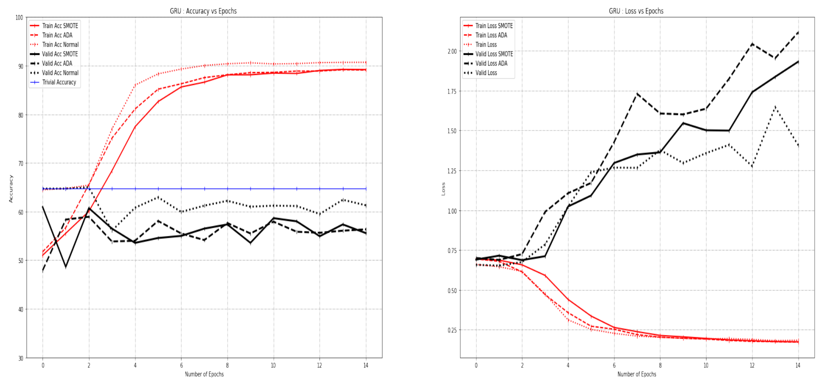

Fig. 8. Left : Accuracy Curve. Right: Loss curve. Blue line represents, the trivial accuracy for this class

This model also achieved one of the best F1 scores in all the single text-based models. For Sentiment task, it was about 0.54 and for the Motivation it was about 0.2

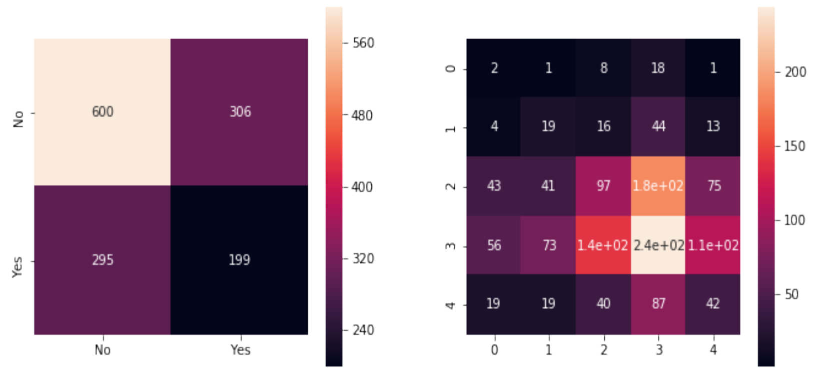

Fig. 9. Left : Confusion Matrix for Motivation. Right: Confusion Matrix for Sentiment

While SMOTE and ADASYN performed well, the best F1 score was obtained when training with the normal imbalanced dataset split in the usual 80-20 fashion. Manual learning rate optimization was also performed. The model seemed less prone to being stuck in ambiguous local minimas. Adding dropouts only caused the F1 score to decrease, so we refrained from that. In retrospect, the model is incredibly simple and we found its performance quite in line with the Occam's razor principle. The Embeddings used for this were GloVe.

Textual Input **BERT Sentence Embeddings**

Google provides BERT's sentence embeddings as a service. We experimented with this idea as it seemed plausible according to [3]. While training, we learned that oversampling was actually working better for this model. The model was however, getting stuck in local minimas during training, and was not the most optimal choice. We can't really figure out why its not working, because the sentence ebedding code is not available. It was worth a try.

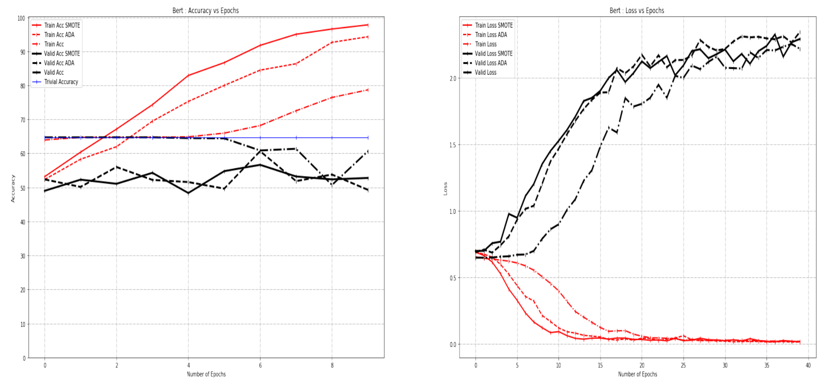

Fig. 10. Left : Accuracy Curve. Right: Loss curve

Textual Input **BERT Pre-Trained by Google**

Memes often have some context attached to them. We believed that if a model knew this context, it would be able to perform better when fine tuned over out dataset. Wikipedia, is a good source for that. Besides that, BERT is also SOTA in a huge number of Natural Language Processing tasks, so we gave it a go. Of course, we did read some literature just to make sure we're not wasting time [9]

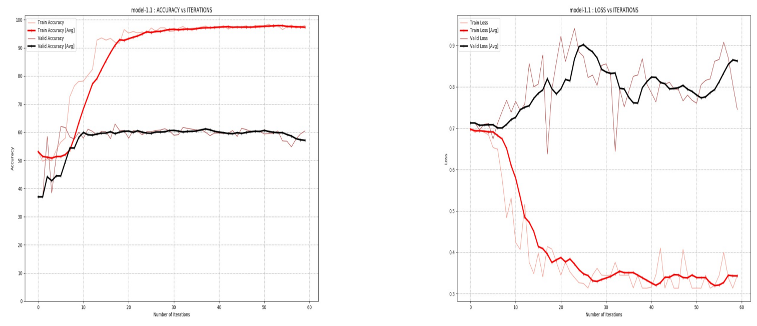

Fig. 11. Left : Accuracy Curve. Right: Loss curve

As can be seen, the model converges well, but the accuracy it not very good. We performed multiple experiments and the best results were obtained with a learning rate of 1e-5 and 2e-5. With 1e-5 being more stable and having a better accuracy in general

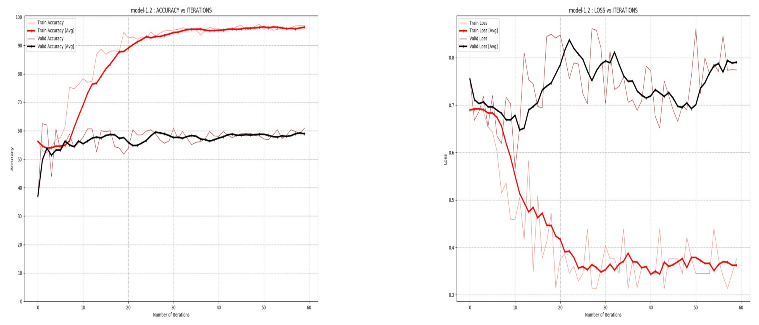

Fig. 12. Left : Accuracy Curve. Right: Loss curve

Following these attempts, we also tried some Image based single input models. Although we were not very hopeful of this pursuit, because it was hard even for us to do the same without reading the text. As a matter of fact, our three experiments on ResNet, InceptionV2, and VGG16 did not perform very well. In the interest of brevity, we omit the graphs for these attempts entirely.

3.2 Multi-Modal Attempts

Pre-Trained Bert with ResNext

As mentioned above, Bert is already SOTA in NLP, whereas a quick review of the current state of Image sentiment analysis with [8] showed that ResNet was doing well in this domain. So we tried to apply ResNet's latest successor ResNext to the task. Joining these two models was hard but, after hours of frantic typing, we were able to make these two work together in PyTorch.

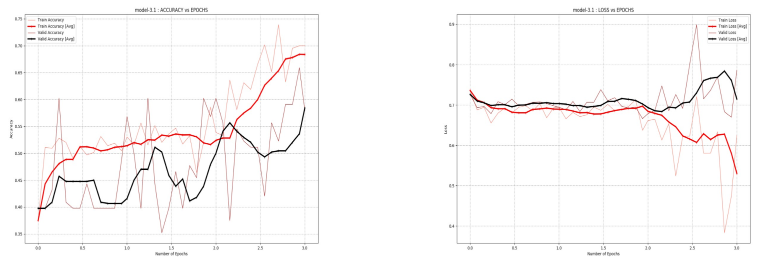

Fig. 13. Left : Accuracy Curve. Right: Loss curve

The results of this experiment were not as good as expected. The model only performed well on the task of Motivation classification, where it barely reached 60% accuracy. We believe, it is because we picked ResNext/ResNet models pre-trained on Imagenet, which is not a good baseline for sentiment analysis. However, we weren't able to find any pre-trained weights online for Sentiment although we found some literature on it [4].

LSTM with CNN Model

This model was entirely based on [10]. We removed droupouts because they werent helping in performance with such a small dataset.

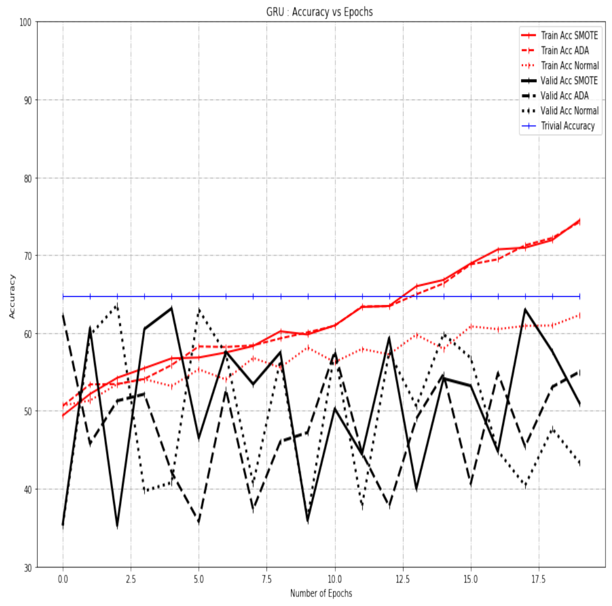

Fig. 14. Accuracy Curve for the model above

Even with SMOTE and ADASYN, the model did not cross the trivial accuracy, i.e it predicted mostly the majority class. We could not get the F1 score to an acceptable level, and in the interest of time, had to stop trying to get it to work.

4 Conclusion

After trying as many things as we could, we decided to finalize the following models for each task. These models were decided keeping the accuracy metrics in mind, which are, F1 score for Task A and B, and F2 score for Task C.

4.1 TASK A : Motivation

The multi-modal approach gave the best results for this task. We had a top F1 Score of `0.54` with this model. The 1D CNN model came very close to beating it with `0.529`. The multi-modal approach, we believe is better regardless because its variety of inputs might give it a sort of robustness.

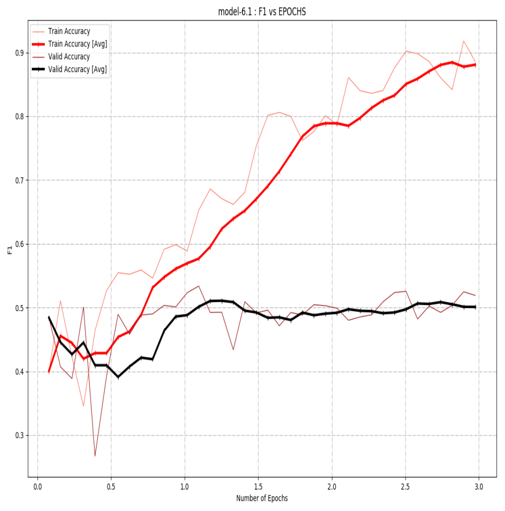

Fig. 15. F1 Curve over Step for Multi-modal approach

4.2 TASK B : Sentiment

The tables had turned for this one. We had a top F1 Score of `0.20` with the 1D CNN model whereas the Bert-ResNext model came second with `0.17`.

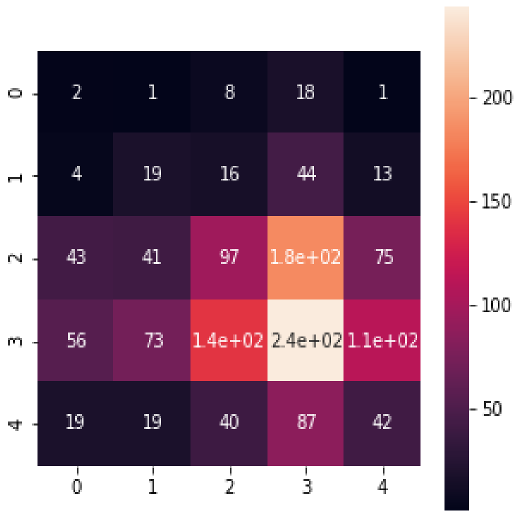

Fig. 16. Confusion Matrix for 1D CNN model

4.3 TASK C : Offensiveness

Surprisingly, neither of the models mentioned above performed the best on this task. We couldn't get either of them to converge at good minima. Interestingly, one of the Image-only approaches, the InceptionResNetV2 model worked well on this task and gave an MAE of 0.32. For training this particular model, images were augmented by applying random linear translations and random Gaussian noise. Also, to deal with class imbalance, each class was oversampled by randomly picking samples of that class and adding a duplicate, till all classes had same number of samples.

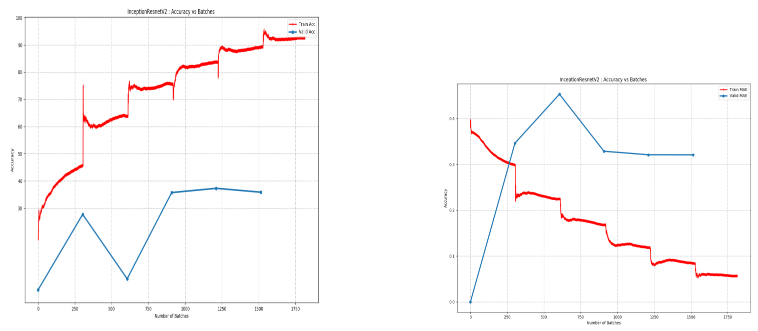

Fig. 17. Left: Accuracy Curve. Right: MAE curve

5 Further Work

There were a lot of avenues we saw but couldn't work on. These mostly included implementation heavy ideas or those where computation power and time exceeded our current capacity. Some of them are listed below.

. Progressive CNN This was an interesting paper that we could not implement in time. Good read. [12]

. XL Net This is a very new SOTA in the NLP world. We spent quite a lot of time to try and make this work. Unfortunately, the only information available is on the official GIT and changing their run_classifier.py is not easily readable. [11]

. Bert Pre-trained on NER we believe Named Entity Recognition pre-training would be a good way to make the model understand the context of many memes, and would increase its accuracy. Couldn't implement it in time. Link to one such implementation

. Oversampling text One possible way to over-sample text samples is to use Google Translate to translate text from English to any other language (say, French) and translate it back to English. This might be a worthwhile exercise because it generates a sentence with the same meaning, but with a little different sentence structure and different words used.

References

1. Ah-Pine, J., Soriano-Morales, E.P.: A study of synthetic oversampling for twitter imbalanced sentiment analysis. In: Workshop on Interactions between Data Mining and Natural Language Processing (DMNLP 2016) (2016)
2. Chen, L., Lee, C.M.: Convolutional neural network for humor recognition. arXiv preprint arXiv:1702.02584 (2017)
3. Chen, T., Xu, R., He, Y., Wang, X.: Improving sentiment analysis via sentence type classification using bilstm-crf and cnn. Expert Systems with Applications **72**, 221–230 (2017)
4. Gajarla, V., Gupta, A.: Emotion detection and sentiment analysis of images. Georgia Institute of Technology (2015)
5. Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882 (2014)
6. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
7. Majumder, N., Hazarika, D., Gelbukh, A., Cambria, E., Poria, S.: Multimodal sentiment analysis using hierarchical fusion with context modeling. Knowledge-Based Systems **161**, 124–133 (2018)
8. Ragusa, E., Cambria, E., Zunino, R., Gastaldo, P.: A survey on deep learning in image polarity detection: Balancing generalization performances and computational costs. Electronics **8**(7), 783 (2019)
9. Sun, C., Qiu, X., Xu, Y., Huang, X.: How to fine-tune bert for text classification? arXiv preprint arXiv:1905.05583 (2019)
10. Tripathi, S., Beigi, H.: Multi-modal emotion recognition on iemocap dataset using deep learning. arXiv preprint arXiv:1804.05788 (2018)
11. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., Le, Q.V.: Xlnet: Generalized autoregressive pretraining for language understanding. arXiv preprint arXiv:1906.08237 (2019)
12. You, Q., Luo, J., Jin, H., Yang, J.: Robust image sentiment analysis using progressively trained and domain transferred deep networks. In: Twenty-ninth AAAI conference on artificial intelligence (2015)
13. Zeiler, M.D.: Adadelta: an adaptive learning rate method. arXiv preprint arXiv:1212.5701 (2012)