LaDiff ULMFiT: A Layer Differentiated Training Approach for ULMFiT

Mohammed Azhan[1(✉)] and Mohammad Ahmad[2]

[1] Department of Electrical Engineering, Jamia Millia Islamia, New Delhi, India
mohd178974@st.jmi.ac.in
[2] Department of Electronics and Communication Engineering, Jamia Millia Islamia, New Delhi, India
mohammad178576@st.jmi.ac.in
https://azhanmohammed.netlify.app/
https://ahmadkhan242.github.io

Abstract. In our paper we present Deep Learning models with a layer differentiated training method which were used for the SHARED TASK @ CONSTRAINT 2021 sub-tasks COVID19 Fake News Detection in English and Hostile Post Detection in Hindi. We propose a Layer Differentiated training procedure for training a pre-trained ULMFiT [8] model. We used special tokens to annotate specific parts of the tweets to improve language understanding and gain insights on the model making the tweets more interpretable. The other two submissions included a modified RoBERTa model and a simple Random Forest Classifier. The proposed approach scored a precision and f1-score of 0.96728972 and 0.967324832 respectively for sub-task COVID19 Fake News Detection in English. Also, Coarse Grained Hostility f1 Score and Weighted Fine Grained f1 score of 0.908648 and 0.533907 respectively for sub-task Hostile Post Detection in Hindi. The proposed approach ranked 61st out of 164 in the sub-task "COVID19 Fake News Detection in English" and 18th out of 45 in the sub-task "Hostile Post Detection in Hindi". The complete code implementation can be found at: GitHub Repository (https://github.com/sheikhazhanmohammed/AAAI-Constraint-Shared-Tasks-2021).

Keywords: Layer differentiated training · Text classification · Language model · Text interpretation

1 Introduction

COVID-19 was declared as a global health pandemic by the WHO, and it can be very well noticed that social media has played a very significant role much before the spread of the virus. As various countries around the world went into lockdown for long periods, it was noticed that social media became a very important platform for people to share information, post their views and emotions in short amount of texts. It has been seen that a study of these texts have resulted

© Springer Nature Switzerland AG 2021
T. Chakraborty et al. (Eds.): CONSTRAINT 2021, CCIS 1402, pp. 54–61, 2021.
https://doi.org/10.1007/978-3-030-73696-5_6

in various novel applications which are not only limited to, political opinion detection as seen in [12], stock market monitoring as seen in [2], and analysing user reviews of a product as seen in [15]. The wide usage of figurative language like hashtags, emotes, abbreviations, and slangs makes it even more difficult to comprehend the text being used on these social platforms, making Natural Language Processing a more challenging task. It has been seen that techniques like Latent Topic Clustering [10], Cultivating deep decision trees [9], performing Fine grained sentiment analysis [15], and ensemble techniques [5] have given competitive results in language understanding tasks in NLP. In this paper we present a similar Deep Learning technique which competed in AAAI SHARED TASK @ CONSTRAINT 2021 'COVID 19 Fake News Detection in English' and 'Hostile Post Detection in Hindi'. The overview of above Shared Task has been explain in this [13]. We explored differentiated layer training technique, where different sections of the layers were frozen and unfrozen during the training. This was combined with the training procedure as discussed in ULMFiT [8]. The complete training procedure is explained in the coming sections. The paper is divided into sections, the next section discusses the task at hand, details of the dataset provided and the preprocessing steps that were taken.

2 Overview

This section contains details of the given task, the dataset provided, and the preprocessing steps taken to clean the dataset.

2.1 Task Description and Dataset

Task Definition Sub-task 1. This subtask focuses on the detection of COVID19-related fake news in English. The sources of data are various social-media platforms such as Twitter, Facebook, Instagram, etc. Given a social media post, the objective of the shared task is to classify it into either fake or real news. The dataset provided for the task is discussed in [14]. The dataset contains a total of 6420 labeled tweets for training, 2140 labeled tweets for validation and 2140 unlabeled tweets were given during the test phase. The complete class distribution for the dataset is shown in Fig. 1(a). The image shows that the distribution of the classes was almost balanced, hence no under-sampling or over-sampling techniques were used during the preprocessing to balance the dataset.

Task Definition Sub-task 2. This subtask focuses on a variety of hostile posts in Hindi Devanagari script collected from Twitter and Facebook. The set of valid categories are fake news, hate speech, offensive, defamation, and non-hostile posts. It is a multi-label multi-class classification problem where each post can belong to one or more of these hostile classes. The dataset for this sub-task covers four hostility dimentions: fake news, hate speech, offensive, and defamation posts, along with a non-hostile label. Dataset is multi labelled due to overlap of different hostility classes. The dataset is further described here [6]. The dataset provided 5728 labeled posts for training, 811 labeled post for

validation, and 1653 unlabeled for test phase. The labeled distribution for train set is shown in Fig. 1(b).

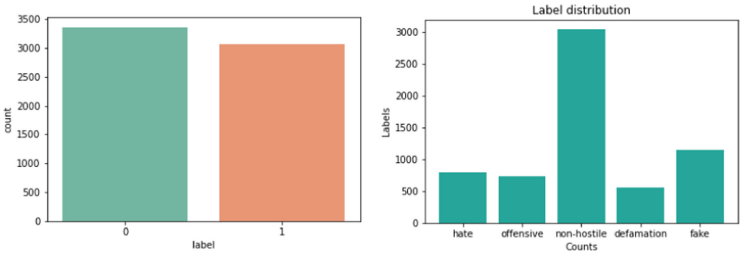

Fig. 1. Label distribution for training dataset (a) "COVID19 Fake News Detection" (b) "Hostile Post Detection in Hindi"

2.2 Preprocesing

The various steps used during the preprocessing of the dataset are mentioned below.

Replacing Emojis. Since tweets from twitter are mostly accompanied with graphics (emojis) which are supposed to help a user express his thoughts, our first task was to replace these emojis with their text counterpart. While a machine cannot understand the emoji, it's text counterpart can easily be interpreted as discussed in [1] and [4]. We used the emoji library[1] for converting emojis to their English textual meanings. For the Hindi dataset we created our own library 'Emot Hindi'[2] similar to the emoji library discussed above which contains emojis and their Hindi textual meanings. This was a common step for both sub-tasks. A few examples of sample emojis and their meanings are shown in Fig. 2.

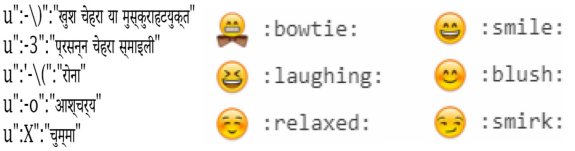

Fig. 2. Example: Emoji and text counterpart (a) Emoji to Hindi (b) Emoji to English

[1] https://pypi.org/project/emoji/.
[2] https://github.com/ahmadkhan242/emot_hindi.

Addressing Hashtags. Hashtags are word or phases preceded by a hash sign '#' which are used to identify texts regarding a specific topic of discussion. It has been seen that the attached hashtags to a post or tweet tell what the text is relevant to, this has been discussed in [4] and [3]. For the given tweets a white space was added between the hash symbol and the following word for the model to comprehend it easily. This was also a common step for both sub-tasks.

Adding Special Tokens. We replaced specific parts of the text with special tokens as discussed in the fastai library[3]. The special tokens and their usage are mentioned in the list below.

- **{**TK_REP} This token was used to replace characters that were occurring more than thrice repeatedly. This special token was used for both sub-tasks. For example 'This was a verrrryyyyyyy tiring trip' will be replaced with 'This was a ve{TK_WREP} 4 r {TK_WREP} 7 y tiring trip'.
- **{TK_WREP}** This token was used to replace words occuring three or more times consecutively. This special token was used for both sub-tasks. For example 'This is a very very very very very sad news' will be replaced with 'This is a {TK_WRPEP} 5 very sad news'.
- **{TK_UP}** This token was used to replace words using all caps. Since the Devnagri script used for Hindi has no uppercase alphabetsm this special token was used for the English sub-task only. For example 'I AM SHOUTING' becomes '{TK_UP} i {TK_UP} am {TK_UP} shouting'.
- **{TK_MAJ}** Used to replace characters in words which started with an upper case except for when it is the starting of a sentence. Again, this special token was used for the English subtask only. For example, 'I am Kaleen Bhaiya' becomes 'i am {TK_MAJ} kaleen {TK_MAJ} bhaiya'.

Normalization. These steps included removing extra spacing between words, correcting hmtl format from texts if any, adding white space between special characters and alphabets, and replacing texts with lower case. The above preprocessing steps were taken for both subtasks.

Tokenization. Once the preprocessing of the dataset was complete, we performed tokenization. For the ULMFiT training the ULMFiT tokenizer was used, similarly the text for the customized RoBERTa model was tokenized using RoBERTa tokenizer, and for the Random Forest Classifier (English and Hindi sub-task) and Linear Regression (Hindi sub-task) the text was tokenized using the nltk library for both the languages.

3 Model Description

Next, we provide an in detail description of the training strategies that were used to achieve the results. The test results obtained using each technique is

[3] https://docs.fast.ai/text.core.html.

mentioned in the results section. Each technique is discussed in the coming subsections.

3.1 Layer Differentiated ULMFiT Training

As discussed in [8] inductive training has shown incredible performance in Computer Vision tasks where the model is first pretrained on large datasets like ImageNet, MS-COCO, and others. The same idea was implemented during the training of the ULMFiT model, only it was modified using a pretrained language model. Traditional transfer learning language models used to pretrain the language model on a relatively larger dataset, this language model was then used to create the classifier model which will again pretrain on the large dataset, at the final step the classifier model was fine-tuned on the target dataset. ULMFiT introduced LM Pretraining and Fine-tuning to make sure that the language model used to pretrain the classifier consisted of extracted features from the target domain. This part of the training procedure is exactly same as discussed in [8]. The image below shows the training of both Language model and classifier as in [8]. We introduced a layer differentiated training procedure, which gradually unfreezed the layers for training them. This differentiated training procedure was implemented for training both, the language model and the classifier model for both of the sub-tasks. Figure 4 shows a plot between the training and validation losses as the training progressed for the English sub-task. The graph shows a spike after every 100 batches which is then followed by a sharp decline. These spikes are the parts where the layers were unfreezed. As the layers were unfreezed, the untrained layers led to an increase in the training loss, which gradually decreased as the training progressed. This also made sure that the final layers were trained longer as compared to initial layers so that the initial layers dont́ start overfitting and the model doesn't drops out any important features. This concludes our discussion for the LaDiff ULMFiT training. We now move forward with our next technique (Fig. 3).

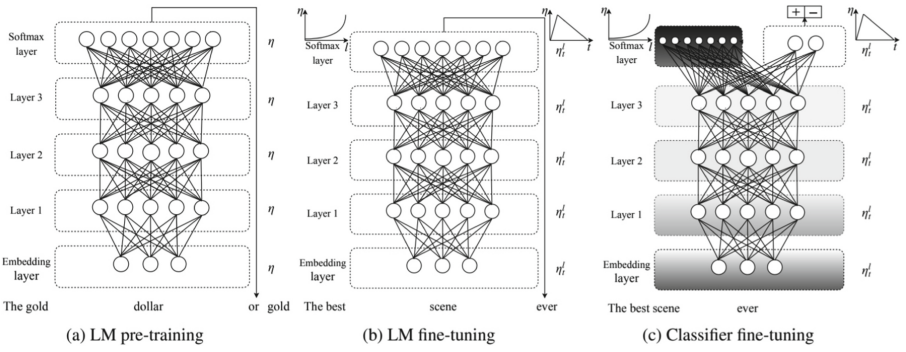

Fig. 3. ULMFiT traditional training

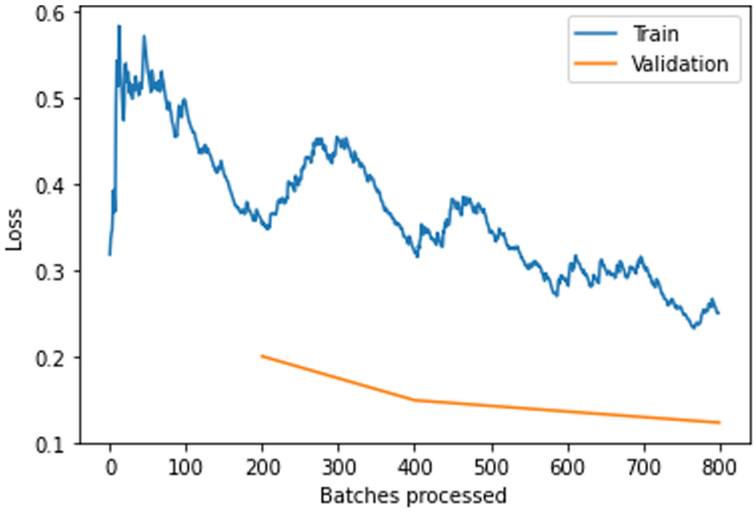

Fig. 4. Loss vs batches progressed: LaDiff ULMFiT

3.2 Customized RoBERTa

RoBERTa [11] is a robustly optimized pretraining approach designed for BERT [7]. BERT stands for Bidirectional Encoder Representations from Transformers, and it introduced the use of transformers for language training tasks. RoBERTa aimed at improvising the training methodology as introduced in BERT using dynamic masking, provising full sentences rather than using next sentence prediction, training with a large number of batches having small sizes and a larger byte-level Byte-Pair Encoding. For our customized model, we used the RoBERTa uncased model pre-trained on various larger twitter dataset. We then added a few customized layers to the model. This training procedure was implemented on the English sub-task only.

3.3 Random Forest Classifiers and Logistic Regression

While the above two approaches have shown how language modelling and using text transformers give exceptionally high performance, our idea behind trying these approach was to understand where do simple language classifiers lack as compared to deep neural networks. While the baseline results as presented in the English dataset paper [14] and Hindi dataset paper [6] use an SVM Classifier, we decided to use various Machine Learning techniques, and submit the one which has the highest score in the validation set. In our case, we achieved the best results using a Random Forest Classifier, having n_estimators set as 1000, min_samples_split as 15 and a random_state of 42. The same classifier hyper-parameters were passed to both of the classification models and trained separately. The Logistic Regression Classifier was used only for the Hindi subtask. This brings an end to our discussion for the various approaches used. We

now move forward to the results obtained and compare them with the available baseline results [6,14].

4 Results

We first present the results obtained for the English sub-task "COVID19 Fake News Detection in English".The table given below gives the accuracy, precision, recall and f1-score of our approaches and compares them with the available baseline results. Our best approach, LaDiff-ULMFiT ranked 61st out of 167 submissions on the final leaderboard (Table 1).

We now present our results for the Hindi sub-task "Hostile Post Detection in Hindi" shown in Table 2. The results ranked 18th for the Coarse Grained f1 Score and 25th for the Fine Grained f1 Score. We now proceed with our conclusions.

Table 1. Comparison results on test set: LaDiff ULMFiT vs customized RoBERTa vs Random Forest Classifier vs baseline model - sub-task 1

Model	Accuracy	Precision	Recall	f1-score
LaDiff ULMFiT	0.96728972	0.967908486	0.96728972	0.967324832
Baseline Model	93.46	93.46	93.46	93.46
Customized RoBERTa	0.929906542	0.929906542	0.929906542	0.929906542
Random Forest Classifier	0.91728972	0.917382499	0.91728972	0.917311831

Table 2. Comparison results on test set: LaDiff ULMFiT vs Logistic Regression vs Random Forest Classifier vs baseline results - sub-task 2

Model	Coarse grained hostility f1 score	Defamation f1 score	Fake f1 score	Hate f1 score	Offensive f1 score	Weighted fine grained f1 score
LaDiff ULMFiT	**90.87**	27.31	**73.83**	44.93	**51.39**	**0.53**
Baseline results	84.11	**43.57**	68.15	**47.49**	41.98	
Logistic regression	76.56	24.8	54.71	40.65	40.58	42.74
Random Forest Classifier	76.56	24.8	54.71	40.65	40.58	42.74

5 Conclusions

From the achieved results as shown in Table 2, the following conclusions can be drawn:

- Fine-tuned language model used with a simple classifier (LaDiff-ULMFiT) outperforms transformers used with sophisticated networks (Customized RoBERTa).
- The losses trend seen in Fig. 4 also signifies the fact that target domain fine tuned on a pre-trained model done at when trained at gradual steps leads to faster decrease in losses.

– We also conclude that, tweets containing hashtags and short texts can also be confidently classified using Machine Learning techniques.

Finally, we make all our approaches and their source codes completely available for the open source community, to reproduce the results and facilitate further experimentation in the field.

References

1. Ashish, V.C., Somashekar, R., Sundeep Kumar, K.: Keyword based emotion word ontology approach for detecting emotion class from text. Int. J. Sci. Res. (IJSR) **5**(5), 1636–1639 (2016)
2. Abdullah, S.S., Rahaman, M.S., Rahman, M.S.: Analysis of stock market using text mining and natural language processing. In: 2013 International Conference on Informatics, Electronics and Vision (ICIEV). IEEE, May 2013
3. Alfina, I., Sigmawaty, D., Nurhidayati, F., Hidayanto, A.N.: Utilizing hashtags for sentiment analysis of tweets in the political domain. In: Proceedings of the 9th International Conference on Machine Learning and Computing - ICMLC 2017. ACM Press (2017)
4. Azhan, M., Ahmad, M., Jafri, M.S.: MeToo: sentiment analysis using neural networks (grand challenge). In: 2020 IEEE Sixth International Conference on Multimedia Big Data (BigMM). IEEE, September 2020
5. Balikas, G., Moura, S., Amini, M.-R.: Multitask learning for fine-grained twitter sentiment analysis. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, August 2017
6. Bhardwaj, M., Akhtar, M.S., Ekbal, A., Das, A., Chakraborty, T.: Hostility detection dataset in Hindi. arXiv preprint arXiv:2011.03588 (2020)
7. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. CoRR, abs/1810.04805 (2018)
8. Howard, J., Ruder, S.: Universal language model fine-tuning for text classification (2018)
9. Ignatov, D., Ignatov, A.: Decision stream: cultivating deep decision trees. In: 2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI). IEEE, November 2017
10. Lee, Y., Yoon, S., Jung, K.: Comparative studies of detecting abusive language on Twitter. In: Proceedings of the 2nd Workshop on Abusive Language Online (ALW2). Association for Computational Linguistics (2018)
11. Liu, Y., et al.: RoBERTa: a robustly optimized BERT pretraining approach. CoRR, abs/1907.11692 (2019)
12. Maynard, D., Funk, A.: Automatic detection of political opinions in tweets. In: García-Castro, R., Fensel, D., Antoniou, G. (eds.) ESWC 2011. LNCS, vol. 7117, pp. 88–99. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-25953-1_8
13. Patwa, P., et al.: Overview of constraint 2021 shared tasks: detecting English COVID-19 fake news and Hindi hostile posts. In: Proceedings of the First Workshop on Combating Online Hostile Posts in Regional Languages during Emergency Situation (CONSTRAINT). Springer (2021)
14. Patwa, P., et al.: Fighting an infodemic: COVID-19 fake news dataset. arXiv preprint arXiv:2011.03327 (2020)
15. Shrestha, N., Nasoz, F.: Deep learning sentiment analysis of amazon.com reviews and ratings. CoRR, abs/1904.04096 (2019)