CEUR-WS.org/Vol-3159/T1-38.pdf

GATED MULTI-TASK LEARNING FRAMEWORK FOR TEXT CLASSIFICATION

Suyash Sangwan[1], Lipika Dey[2] and Mohammad Shakir[3]

Tata Consultancy Services Limited, Block-C, Kings Canyon,ASF Insignia, Gurgaon, Gwal Pahari, Gurgaon 12203, Haryana,India

Abstract

Related tasks are generally dependent on each other and thereby perform better when solved in a joint framework. Considering the same fact, our team named 'TCS Research Lab Gurgaon' presents a deep learning-based multi-task framework that jointly identifies the presence of 'Hate and Offensive' *(HOF)* speech and further classifies the type of 'HOF' speech present *(i.e. hate or offensive or profane)*. For each tweet, we extract three forms of feature sets *(i.e. word embeddings, topical distribution, and TF-IDF score)* to convey diverse and distinctive information. As we know all these feature sets usually don't have equal contribution in final decision-making, therefore we use the 'gated' mechanism to assign weights to these feature sets based on their importance in the final prediction. We have evaluated and validated our proposed approach on subtask1A and subtask1B [1] of HASOC-2021 challenge [2]. Evaluation results suggest and show that the multi-task learning *('MTL')* framework provides better results as compared to the single-task learning *('STL')* framework *(i.e. solving both the subtasks independently)*.

Keywords

Deep Learning, Multi-task framework, Text classification, Twitter, Hate speech, Offensive speech, Profane speech, Social media.

1. Introduction

Social media platforms *(like Twitter, Facebook, Instagram, etc.)* provide people with an opportunity to express their views, experiences, knowledge, and emotions freely. People use these platforms to exhibit their likes or dislikes towards something, their reaction to a government decision, or how they felt about a situation, and so on. But it becomes troublesome when these social-media platforms rather become a platform for abusive comments, remarks, and conversations. The uncontrolled spread of such data has the potential to gravely damage our society, and severely harm youth and marginalized people or groups, emphasizing the need to detect and classify such content.

The challenge of wrangling hate, offensive, and profane data is an ancient one, but the scale at which this data is generated today is a uniquely modern dilemma. Hatred is a very disgusting or angry emotional response to certain people or ideas which is deemed to be harmful based on defined 'protected attributes' such as race, disability, sexuality, etc. This class of data is threatening to another, and sometimes includes a call for violence. *For Example: "It's an international shame for India as our PM care fund means care for PM"*. On the other hand

Forum for Information Retrieval Evaluation, December 13-17, 2021, India
suyash.sangwan@tcs.com (S. Sangwan); lipika.dey@tcs.com (L. Dey); m.shakir@tcs.com (M. Shakir)
© 2021 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).
CEUR Workshop Proceedings (CEUR-WS.org)

offensive data is the data that offends someone. It does not have to be threatening or hateful to be considered offensive. It is simply the data that upsets someone and has nothing with the legality of the speech. *For Example: "I get turned on by 13 years old".* On the other hand 'Profanity' is a class of data that includes dirty words and ideas. Swear words, vulgar language, obscene gestures, and naughty jokes are all considered 'Profane'.

Now there are many layers to the difficulty of automatically detecting hateful and/or offensive speech, particularly in social media. The first one is 'subjectivity'. A seemingly neutral sentence can be offensive for one person and not bother another. Similarly hate data has no legal definition. For this reason, what is and is not hate/offensive data is open to interpretation. A lot depends on the domain and the context. Even the usage of these slang or insulting words vary in usage to express contempt, the difference of opinions, and in some cases, people use these words in humor. Another big challenge is the use of "Hinglish" *(which is a blend of Hindi and English i.e. Hindi written in the Roman script instead of the native Devanagari)* in the HASOC [3] dataset. As the dataset is collected from Twitter India so being a multi-lingual society people tend to use code-mixed patterns.

Therefore keeping all these challenges in mind, we propose a multi-task deep learning-based gated model where we aim to leverage the inter-dependence of given two subtasks to increase the confidence of individual subtask in prediction. *For Example: if subtask 1A says "NOT", then for the subtask 1B it's always "NONE".* For each tweet, we utilize three feature sets *(i.e. RNN based word embedding, topical distribution, and TF-IDF scores of words in the tweet).* Our main intuition behind using different feature sets is to utilize the advantages of different feature extraction methods in a single framework. For tweets having out of vocabulary words *(like hashtags)*, the TF-IDF component will be of utmost importance. Similarly, the intensity and combination of different topics present within a tweet also helps in providing additional context.

The main contributions of this paper are:

1. We propose a multi-task framework to leverage the inter-dependence of two related tasks *(i.e. HOF/NOT and type of HOF)* in improving each other's performance.
2. We utilize different feature extraction techniques in a single framework.
3. We apply a gated module to refine the feature sets *(i.e assign weights to feature sets)* based on their role in the final prediction.

The remainder of this paper is organized as follows. In section 2 we describe our problem definition. In section 3 we describe some previous work. Our architecture is introduced in section 4, followed by an outline of dataset, experimental results, and analysis in section 5. Finally we conclude with a discussion in section 6.

2. Problem Definition

Our first subtask (1A) focuses on Hate speech and Offensive language identification. It is a coarse-grained binary classification where we have to classify the tweets into two classes, namely: Hate and Offensive *(HOF)* and Non-Hate and offensive *(NOT)*. The first category of data i.e. 'HOF' contains posts having hate, offensive, and profane content whereas the second category of data i.e. 'NOT' contains posts that do not have any hate, profane, offensive content.

The other sub-task (1B) is a fine-grained classification where 'HOF' posts from the above sub-task are further classified into three categories: 1). Hate *(HATE)* – posts under this category contain hate speech content. 2). Offensive *(OFFN)* – posts under this category contain offensive content. 3). Profane *(PRFN)* – posts under this category contain profane content.

3. Related Work

The interest in detecting hate speech and bullying data, particularly on social media, has attracted attention from researchers interested in sociological and linguistic features. In this section we review a number of studies and briefly discuss their findings. In 2012, Xu et al. [4] applied sentiment analysis to detect bullying roles in the tweets. Here the authors have used LDA for identifying relevant topics and formulated the task as a binary classification task *i.e.* text is classified as an instance of bullying or not. Further in 2015, Burnap et al. [5] used a supervised machine learning text classifier that distinguishes between hateful and/or antagonistic responses with a focus on race, ethnicity, or religion; and more general responses. They derived the classification features like grammatical dependencies between words and used this model to forecast the likely spread of cyber hate in a sample of Twitter data. In 2016, Nobata et al. [6] developed a machine learning based method to detect hate speech on online user comments by developing a corpus of user comments annotated for abusive language. As we know lexical detection methods tend to have low precision because they classify all messages containing particular terms as hate speech and previous work using supervised learning has failed to distinguish between different categories. Therefore in 2017, Davidson et al. [7] used a crowd-sourced hate speech lexicon to collect tweets containing hate speech keywords. Their findings revealed that racist and homophobic tweets are more likely to be classified as hate speech but sexist tweets are generally classified as offensive and tweets without explicit hate keywords are also more difficult to classify. Our model is different in the sense that we utilize different feature sets to identify both presence and type of 'HOF' data in a single framework. We also hypothesize that applying the gating mechanism to assign weights to different feature sets may assist the network in a better way.

4. Proposed Architecture

In this section, we describe our proposed architecture as shown in Figure1, where we aim to leverage different feature sets and gating mechanism for solving multiple tasks together. The proposed framework takes given tweet as an input and pre-processes it. For each pre-processed tweet we provide three types of input *(or feature sets).* First, pre-trained word embeddings which are further processed through bi-directional Gated Recurrent Units *(GRUs)* [8] for capturing the contextual information. Second, topical distribution *(i.e. percentage of topics present in the tweet).* Third, 'TF-IDF' score of the tweet words. All three feature sets are then applied to a fully connected layer to make the output dimension same. Now the key challenge here is to utilize and fuse the relevant information for the prediction. For that, we employ a gated mechanism to

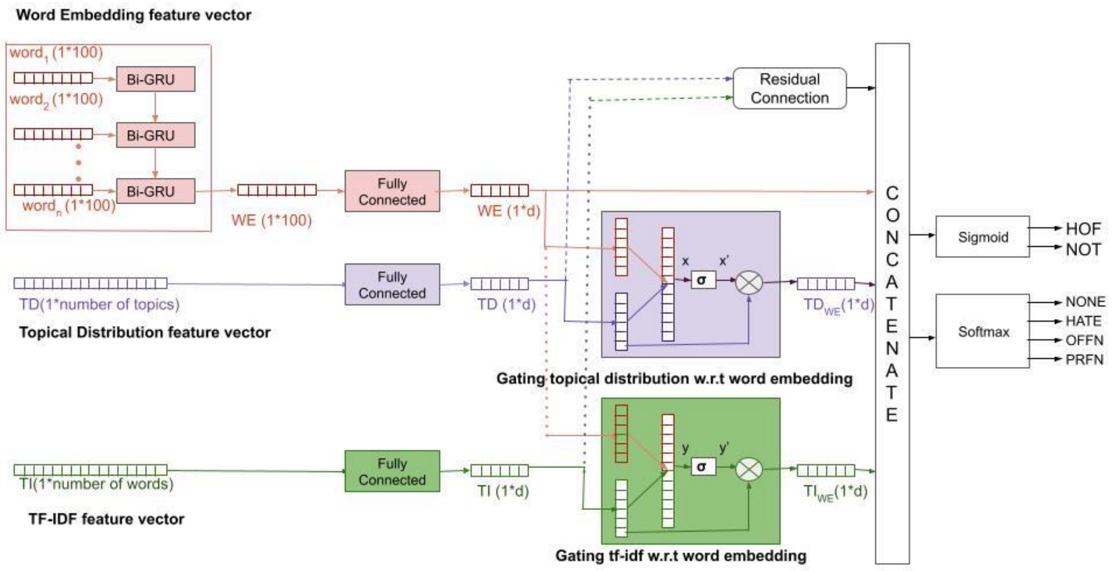

Figure 1: Overall Architecture of the proposed Multi-task Gated Framework where 'WE': Word Embedding vector, 'TD': Topical Distribution vector, 'TI': TF-IDF vector

refine the feature sets *(i.e. assign weights to the feature sets)*. The gating mechanism evaluates the importance of an individual feature set based on its role in final prediction.

1. **Pre-Processing:**The first step before building any machine learning model is to pre-process the data. If the data is fairly pre-processed, the results would also be reliable. So at first we remove all the punctuations and convert the tokenized words into lower-case format. Then to clean the noise present in the dataset we use framework [9] to automatically improve the quality of the dataset. Since HASOC [3] dataset consists of tweets, hence it is prone to spelling mistakes and people usually misspell abusive words to avoid getting detected by the auto blocking. Through this framework, we detect misspelled words and replace them with the correctly spelled words to improve the quality of the classifier.

2. **Feature Extraction:** The pre-processed text needs to be transformed into vectors so that the algorithms will be able to learn and make predictions. In the following, we give a brief description of the three feature sets used in our proposed multi-task framework.

    a) **Word Embedding *(WE)*:** Here each word is represented as a 100-dimensional feature vector using pre-trained Word2Vec [10] embeddings. Each of these wordwise embeddings is then applied to a separate bi-directional Gated Recurrent Unit *(GRU)* [8] for capturing the contextual information and only one output representation for all the input words is obtained at the end.

    b) **Topical Distribution *(TD)*:** Second, we use topical distribution as a feature set. Unsupervised models like topic extraction help us in clustering similar content. So we use LDA MALLET *(Latent Dirichlet Allocation MAchine Learning for LanguagE*

Toolkit [11], which is an open-source toolkit designed for NLP tasks like classification, clustering, and topic-modeling) for extracting topics from training data. Given a fixed number of topics *(which we decide by maximizing intra-topic coherence scores)*, say *'k'*, LDA MALLET uses an optimized Gibbs sampling algorithm to calculate the probability of each word in a document to belong to a particular topic, and thereafter the distribution of all *'k'* topics in each text document. The probability score of each word in the vocabulary belonging to a topic is also obtained. Each topic can therefore be represented by the top *'n'* words that have the highest probability of belonging to it. These probability scores are further used to compute the topical distribution in the test data. Our main intuition behind using topical distribution as a feature set is that the combination and intensity of different topics present in the tweet may help in predicting the final class.

For Example: Top topic words of *topic2* 2 are *'time', 'death', 'money', 'virus', 'lockdown', 'Chinese'*, which show the topic contains general discussion on Coronavirus. So tweets having higher percentage of *topic2* are more probable of belonging to the *'NOT'* class. Similarly, top topic-words of *topic5* 2 are *'bitch', 'ass', 'fuck', 'shit', 'whore', 'dumb'*. So tweets having a higher percentage of *topic5* are more probable of belonging to the 'Profane' *(PRFN)* class.

c) **TF-IDF *(TI)*:** Third, we use 'TF-IDF' as a feature set. Here we use Term Frequency – Inverse Document Frequency *('TF-IDF')* weight to evaluate how important a word is to a class *(HOF or NOT or OFFN or PRFN)*. 'TF' summarizes how often a given word appears within a given class, whereas 'IDF' downscales words that appear a lot across classes. A word has a high 'IDF' score if it appears in few classes. Conversely, if the word is very common among classes *(i.e. 'a', 'an', 'the')*, the word would have a low 'IDF' score.

$$IDF = \log \frac{\text{(Number of total classes)}}{\text{(Number of classes the term appears in)}} \tag{1}$$

So using 'TF-IDF', we try to capture the most important words or terms per class. This approach also helps in dealing with out-of-vocabulary words or terms *(like hashtags and slang words)*.

3. **Gating Mechanism:**It is true and obvious that all the three feature sets cannot contribute equally to the final prediction. As 'topical distribution' and 'TF-IDF' approaches are completely unsupervised, we refine these features before passing them to the classifier. For refinement, we use gating mechanism where we decide the weights of these feature sets with respect to the word-embedding feature vector *('WE')*. So at first, we concatenate *'WE'* feature vector and topical distribution *('TD')* feature vector and apply a dense layer to the concatenation. Now we pass the concatenated feature vector *(X)* through a *'Sigmoid'* which gives the value *(x')*, representing the weight of *'TD'* feature vector w.r.t. *'WE'* feature vector. Finally, this weight *(x')* is multiplied by the entire feature vector *'TD'*. So weightage of topical distribution w.r.t. word embedding is TD_{WE}=x'·TD, which helps in

the refinement of the topical-distribution feature *(i.e. either passing it or suppressing it)* depending on its role in final prediction.

Equations for topical gating w.r.t. word embeddings are:

$$X = fully\_connected([WE, TD]) \tag{2}$$

$$x` = sigmoid(X) \tag{3}$$

$$TD_{\mathrm{WE}} = x` \cdot TD \tag{4}$$

Similarly to refine TF-IDF vector ('TI') w.r.t. word-embedding vector ('WE') :

$$Y = fully\_connected([WE, TI]) \tag{5}$$

$$y' = sigmoid(Y) \tag{6}$$

$$TI_{\mathrm{WE}} = y` \cdot TI \tag{7}$$

Then we concatenate these gated representations $\mathrm{TD}_{\mathrm{WE}}$ and $\mathrm{TI}_{\mathrm{WE}}$ with *'WE'* along with their residual connections for final prediction. We append residual connections of the modalities to boost the gradient flow to the lower layers.

4. **Multi-task Framework:**The multi-task learning paradigm provides an efficient platform for achieving generalization. Multiple tasks can exploit the inter-relatedness for improving individual performance through a shared representation. Overall, it provides three basic advantages over the single-task learning paradigm. 1). It helps in achieving generalization for multiple tasks. 2). Each task improves its performance in association with the other participating tasks. 3). It offers reduced complexity because a single system can handle multiple problems or tasks at the same time. So after gating, the concatenated representation is shared across the two branches of our proposed multi-task framework-corresponding to the two tasks, i.e. *'HOF'* or *'NOT'* and 'type of *'HOF'*.The shared representation will receive gradients of error from both the branches and accordingly adjust the weights of the models. Thus, the shared representation will not be biased to any particular task, and it will assist the model in achieving generalization for multiple tasks.

5. Dataset, Experimental Results, and Analysis

In this section, we describe the dataset used for our experiments, hyper-parameters used, results, error analysis, and finally other models that we use to compare our results with.

1. **Dataset:** We use the English dataset of HASOC-Challenge 2021 [2] to evaluate and validate our proposed approach. The training and test set consists of 3,843 and 1,281 tweets respectively.

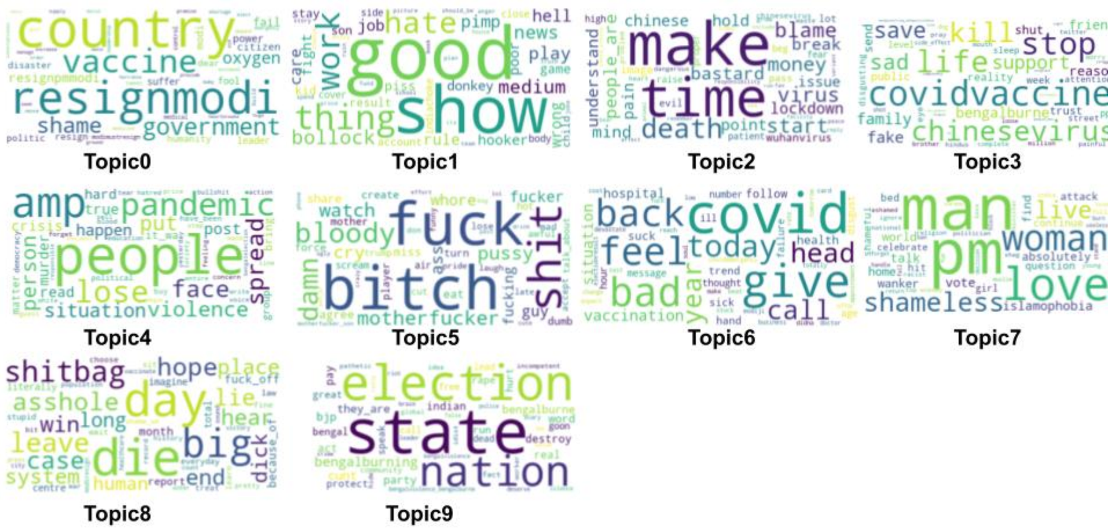

Figure 2: Optimal topics extracted from training set

2. **Experiments:** We use the python based Keras [1] library for its implementation. For the experiments, we perform 5-fold cross-validation on the training data, as the test data is unlabeled. For evaluation, we compute *macro f1-score*, and *weighted f1-score* to measure the performance of the model. We choose *weighted f1-score* as a metric because samples are unbalanced across various classes. We use the grid search to find the optimal hyper-parameters for our experiments. We use Bi-GRU with 300 neurons each and set dropout to 0.3, batch size to 50 and the number of epochs to 5. We use *ReLu* as the activation function, *Adam* as an optimizer, and *binary cross-entropy* as the loss function. As first subtask (1A) is a binary classification problem, *'sigmoid'* is applied for final prediction where we choose a threshold value of 0.5, whereas for the other subtask (1B) we use *'softmax'*.

   The optimal number of topics extracted from different sets of cross-validation is different, but we choose the model with the highest *weighted f1-score*. So the optimal number of topics extracted is 10, as shown in fig 2.

   For extracting class-wise most important features using *TF-IDF* approach, we use a Python library named *'sklearn'* [12] and selected the features having p-value >0.95 *Some examples of class-wise top features selected:*

   a) *'NOT'* class - 'indiacovidcrisis', 'covidvaccine', 'heartbreaking', 'covid19'.

   b) *'HATE'* class - 'bjp', 'shame', 'resignmodi', 'resignpmmodi', 'politics'.

   c) *'OFFN'* class - 'narendramodi', 'modi', 'pm', 'modi ji', 'pmoindia'.

[1]https://github.com/fchollet/keras

d) *'PRFN'* class - 'fuck', 'motherfucker', 'dick', 'pussy', 'asshole', 'bollock'.

We evaluate our proposed approach with FastText [13] embeddings, pretrained BERT [14] model and Word2Vec [10] embeddings. As shown in Table 1 and 2 Word2Vec model performed better than FastText and BERT models. We therefore choose Word2Vec model for further experiments.Then we evaluate our proposed model with input combinations like, only word embeddings (WE), combination of word-embeddings and TF-IDF (WE+TI), combination of word-embeddings and topical distribution (WE+TD), and finally combination of all three (WE,TD,TI). For consistency, we use the same hyperparameters for training all the models. We obtain best results when we use gated multi-task framework with all the three feature sets *(i.e. 'WE', 'TI', and 'TD')*

Table 1
Macro F1-Scores of Single-task learning (STL) and Multi-task learning (MTL) frameworks for the proposed approach where TI: TF-IDF, WE: Word2Vec Embedding, TD: Topical Distribution

Tasks	Framework	Macro F1-Scores					
		FastText	BERT	WE	WE+TI	WE+TD	WE+TD+TI
Subtask 1A	**STL**	66.0	67.2	67.4	70.2	68.2	74.7
	MTL	-	-	70.4	76.6	76.1	81.2
Subtask 1B	**STL**	52.1	53.8	54.6	57.9	55.3	60.2
	MTL	-	-	61.3	64.0	62.6	68.9

Table 2
Weighted F1-Scores of Single-task learning (STL) and Multi-task learning (MTL) frameworks for the proposed approach where TI: TF-IDF, WE: Word2Vec Embedding, TD: Topical Distribution

Tasks	Framework	Macro F1-Scores					
		FastText	BERT	WE	WE+TI	WE+TD	WE+TD+TI
Subtask 1A	**STL**	70.2	71.6	71.9	73.8	72.4	78.1
	MTL	-	-	73.0	79.7	78.8	83.6
Subtask 1B	**STL**	54.9	56.2	56.9	59.0	57.1	64.8
	MTL	-	-	63.9	67.2	66.0	74.3

6. Conclusion

In this paper, we have proposed an RNN based gated multi-task framework that aims to reveal and utilize the inter-dependence of two related tasks i.e. presence of *'HOF'* data and type of *'HOF'* data present. Our proposed approach learns a joint-representation for both the tasks and uses weighted representation of feature sets *(using gating mechanism)*. We evaluate our proposed approach on the recently released English dataset of HASOC challenge[2]. Experimental results suggest that *('topical distribution')* and *('TFIDF')*, if refined, help *('word embeddings')* for better predictions. In the future, we would like to explore the other dimensions of our multi-task framework.

7. Citations and Bibliographies

References

[1] T. Mandl, S. Modha, G. K. Shahi, H. Madhu, S. Satapara, P. Majumder, J. Schäfer, T. Ranasinghe, M. Zampieri, D. Nandini, A. K. Jaiswal, Overview of the HASOC subtrack at FIRE 2021: Hate Speech and Offensive Content Identification in English and Indo-Aryan Languages, in: Working Notes of FIRE 2021 - Forum for Information Retrieval Evaluation, CEUR, 2021. URL: http://ceur-ws.org/.

[2] S. Modha, T. Mandl, G. K. Shahi, H. Madhu, S. Satapara, T. Ranasinghe, M. Zampieri, Overview of the HASOC Subtrack at FIRE 2021: Hate Speech and Offensive Content Identification in English and Indo-Aryan Languages and Conversational Hate Speech, in: FIRE 2021: Forum for Information Retrieval Evaluation, Virtual Event, 13th-17th December 2021, ACM, 2021.

[3] HASOC 2021 dataset, https://hasocfire.github.io/hasoc/2021/dataset.html, 2021.

[4] J.-M. Xu, K.-S. Jun, X. Zhu, A. Bellmore, Learning from bullying traces in social media, in: Proceedings of the 2012 conference of the North American chapter of the association for computational linguistics: Human language technologies, 2012, pp. 656–666.

[5] P. Burnap, M. L. Williams, Cyber hate speech on twitter: An application of machine classification and statistical modeling for policy and decision making, Policy & internet 7 (2015) 223–242.

[6] C. Nobata, J. Tetreault, A. Thomas, Y. Mehdad, Y. Chang, Abusive language detection in online user content, in: Proceedings of the 25th international conference on world wide web, 2016, pp. 145–153.

[7] T. Davidson, D. Warmsley, M. Macy, I. Weber, Automated hate speech detection and the problem of offensive language, in: Proceedings of the International AAAI Conference on Web and Social Media, volume 11, 2017.

[8] R. Dey, F. M. Salem, Gate-variants of gated recurrent unit (gru) neural networks, in: 2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS), IEEE, 2017, pp. 1597–1600.

[9] G. Roy, L. Dey, M. Shakir, T. Dasgupta, Learning domain terms-empirical methods to enhance enterprise text analytics performance, in: Proceedings of the 28th International Conference on Computational Linguistics: Industry Track, 2020, pp. 190–201.

[10] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, arXiv preprint arXiv:1301.3781 (2013).

[11] D. E. King, Dlib-ml: A machine learning toolkit, The Journal of Machine Learning Research 10 (2009) 1755–1758.

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al., Scikit-learn: Machine learning in python, Journal of machine learning research 12 (2011) 2825–2830.

[13] A. Joulin, E. Grave, P. Bojanowski, M. Douze, H. Jégou, T. Mikolov, Fasttext. zip: Compressing text classification models, arXiv preprint arXiv:1612.03651 (2016).

[14] I. Tenney, D. Das, E. Pavlick, Bert rediscovers the classical nlp pipeline, arXiv preprint arXiv:1905.05950 (2019).