Fine Grained Insincere Questions Classification using Ensembles of Bidirectional LSTM-GRU Model

Sourya Dipta Das[1]*, Ayan Basak[1], and Soumil Mandal[2]

[1] Jadavpur University, Kolkata, India
[2] SRM University, Chennai, India
{dipta.juetce, ayanbasak13, soumil.mandal}@gmail.com

Abstract. In this paper, we have described our deep learning based system for fine-grained insincere questions classification, which is the CIQ track in FIRE 2019. Our pipeline uses ensembles of bidirectional LSTM-GRU model with different word embedding techniques namely Glove, FastText, and Paragram. We have also used the checkpoint ensemble method to enhance performance alongside a combination of two different embeddings per-ensemble. Our pipeline has secured the first position in this track with an F1 score of 67.32%.

Keywords: Questions Classification · Bidirectional LSTM-GRU · Transfer Learning · FastText · Glove · Paragram

1 Introduction

In the recent past, there has been a rapid increase in the popularity of question-and-answer websites where the whole pipeline of asking, answering and editing is done by internet users. Some of the common websites with this property are Yahoo! Answers, Stack Overflow, Quora. Even though the majority of these questions are information-seeking, sometimes individuals post inappropriate questions to attack or insult a certain person or communities, spread rumors, hate speech, etc. These type of questions are classified as non-information seeking or insincere questions and authorities from respective sites need to filter such questions to maintain their non-offensive content standards. With the ever-increasing population of users in such online communities, it is really necessary to make an automated system to classify inappropriate questions. In this paper, we describe our system which performs fine-grained classification and classifies a question into one of these six classes, which are 1) Rhetorical Questions 2) Hate Speech 3) Hypothetical Questions 4) Sexual Content 5) Other 6) Sincere Questions. The primary steps in our methodology include preprocessing, converting words to their respective embeddings, training, and finally creating an ensemble model.

Our approach is designed to work well even when training data is small, as we have shown that it can still get a micro F1 score of 67.32% on test data.

2 Previous Work

Traditionally, machine learning algorithms like Naive Bayes have been used for solving text classification problems [1][2], especially after the success of Pedro et al. [3], who proved that Naive Bayes can yield surprisingly positive results on text classification tasks where the importance of the probability calculated by Naive Bayes itself is not very high. However, the performance of Naive Bayes classifier has not been so good when compared to other methods that involve statistical learning like Support Vector Machines [4], Nearest-Neighbor classifiers [5], and Boosting [6]. Modifications to the traditional Naive Bayes text classification model using Poisson distribution for text classification [7] has proven to be quite successful with a slight increase in time and space complexity. With the emergence of deep learning, neural network based architectures have proven to be quite successful in sentiment analysis of textual data. Word-level Convolution Neural Networks (CNNs) have been used extensively for text classification [8][9]. The CNN trained on top of pre-trained embeddings [11] has been shown to obtain favorable results. Character level neural networks have also proved to be quite effective for text classification [10]. However, due to their ability to capture contextual information, Recurrent Neural Networks (RNNs) have proved to be more effective in NLP problems [11]. While back-propagating the errors in the case of RNNs, several problems like vanishing and exploding gradients can occur [12]. LSTM networks have been shown to preserve long-term dependency in the text so they have been used in language modeling extensively. Bidirectional LSTMs can capture contextual information better and when coupled with the Attention mechanism, which can give extra emphasis on words that play a decisive role in text classification. A modification to the standard RNN [11], the Gated RNN, has been shown to dramatically improve the performance of its predecessor for sentiment classification in document modeling [14]. Kamnath et al [23] created the state-of-the-art model for the raw version of the TREC QA dataset where an RNN based similarity model with attention was used for answer-sentence selection. Yoon et al. [24] developed the state-of-the-art model for a clean TREC QA dataset using compare-aggregate, language modeling and latent clustering strategies. Zhang et al. [25] created a hate-speech detection model using a CNN-LSTM based deep neural net where pre-trained word-embeddings were used to set the weights of the embedding layer. They also used dropout and pooling in their architecture which empirically improved classification accuracy. They have shown that their model outperforms the previous state of the art based on both classical methods (SVM, Naive Bayes), as well as deep learning based methods.

3 Dataset Statistics

The dataset for this competition has been made available by the organizers of this task. They had provided us with 900 training samples and 100 test samples by labeling 6 fine-grained classes of insincere questions from Quora insincere questions binary classification dataset [3]. The distribution of labels in the dataset across 6 different classes in train & test set is shown in Fig 1.

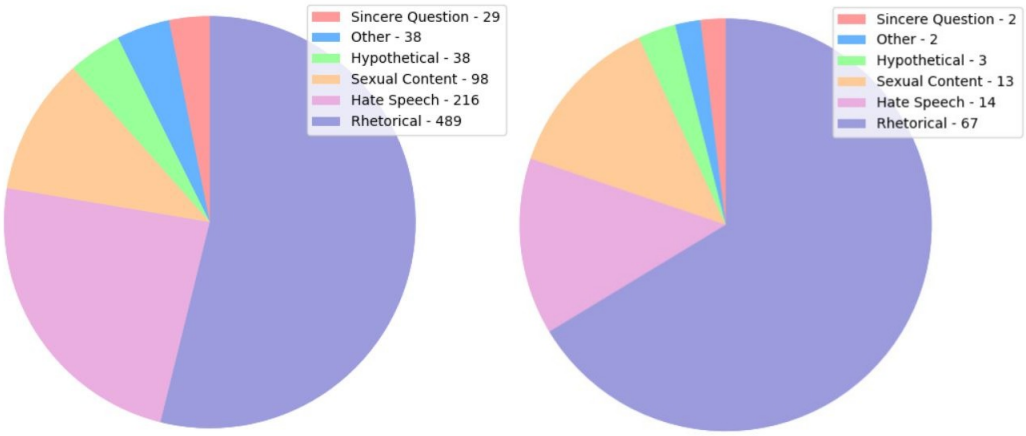


Fig. 1: Class distribution in train and test data respectively.

4 Methodology

We have approached this task as a multi-class question classification problem where each question can have only a single class, as fine-grained classes are mutually independent. Our proposed method consists mainly of three main parts, namely 1) Text Preprocessing 2) Pre-trained Word Embeddings 3) Sequential Deep Neural Network Model 4) Ensemble Model. Each of these is explained in detail in the following sections.

4.1 Text Preprocessing

Questions posted on online forums often have several spelling mistakes, repetitions of special characters, abbreviations, etc. Because of these reasons, it is quite hard to get proper word embeddings for these noisy words in a sentence. First, we tokenized our text using Spacy [4] after removing stop words and punctuations. We created our lemma dictionary with words from the text that are not

[3] www.kaggle.com/c/quora-insincere-questions-classification
[4] https://spacy.io/usage/linguistic-features

stopwords or punctuations so that they can be looked up later in the mapping of pre-trained embeddings. We checked the original version, lowercase version, uppercase version, capitalized version, stemmed version, lemmatized version and the corrected version of a word in the text, which we got by checking if the word in the text is at most two edits away from an actual word in the word embedding vocabulary, in order to look for a pre-trained embedding. For example, *"questions"* was lemmatized to *"question"*, *"appearing"* was lemmatized to *"appear"*, *"immuture"* was corrected to *"immature"*, *"khatriyas"* was corrected to *"kshatriyas"*, *"ociopath"* was corrected to *"sociopath"*, etc. Censored words are also expanded according to word embedding vocabulary, for example, *"bit*h"* was expanded to *"bitch"*. Examples of text preprocessing are shown in Table 1 and Table 2 below.

Sentence									
"Why are the doctors in Europe so inefficient?"									
token	text	is_lower	is_upper	is_stop	lemma	is_alpha	is_ascii	is_digit	is_punct
Why	'Why'	False	False	True	'Why'	True	True	False	False
are	'are'	True	False	True	'are'	True	True	False	False
the	'the'	True	False	True	'the'	True	True	False	False
doctors	'doctors'	True	False	False	'doctor'	True	True	False	False
in	'in'	True	False	True	'in'	True	True	False	False
Europe	'Europe'	False	False	False	'Europe'	True	True	False	False
so	'so'	True	False	True	'so'	True	True	False	False
inefficient	'inefficient'	True	False	False	'inefficient'	True	True	False	False
?	?'	False	False	False	'?'	True	True	False	True
Sentence									
"Is the idea of "white privilege" an Afrocentric conspiracy theory?"									
token	text	is_lower	is_upper	is_stop	lemma	is_alpha	is_ascii	is_digit	is_punct
Is	'Is'	False	False	True	'Why'	True	True	False	False
the	'the'	true	False	true	'the'	True	True	False	False
idea	'idea'	True	False	False	'idea'	True	True	False	False
of	'of'	True	False	True	'of'	True	True	False	False
"	"''	False	False	False	"''	True	True	False	True
white	'white'	True	False	False	'white'	True	True	False	False
privilege	'privilege'	True	False	False	'privilege'	True	True	False	False
"	"''	False	False	False	"''	True	True	False	True
an	'an'	True	False	True	'a'	True	True	False	False
Afrocentric	'Afrocentric'	False	False	False	'Afrocentric'	True	True	False	False
conspiracy	'conspiracy'	True	False	False	'conspiracy'	True	True	False	False
theory	'theory'	True	False	False	'theory'	True	True	False	False
?	'?'	False	False	False	'?'	True	True	False	True

Table 1: Example of token attribute extraction using Spacy.

Misspelled/Censored	Corrected
immuture	immature
WHy	Why
Khatriyas	Kshatriyas
ociopath	sociopath
discreminate	discriminate
bit*h	bitch
abnoxiously	obnoxiously
fiuta	fita

Table 2: Examples of misspelled/censored words and corrected words.

4.2 Pre-trained Word Embeddings

As there are only 900 labeled questions in the training set, training an embedding layer by using this data will lead to problems relating to underfitting. Thus, we decided to go for a transfer learning based approach using three popular word embedding techniques, namely FastText [16], Glove [17], Paragram [18]. We have used 2 set of concatenated word embeddings which are 1) Glove, FastText and 2) Glove, Paragram. We have considered Glove as the base word embedding technique. FastText was used as it can handle rare and out of vocabulary word efficiently while Paragram was used as it can generate phrase embeddings containing semantic information. We decided to go for multiple pre-trained embeddings to tackle the diverse vocabulary in the dataset and to get the combined benefits.

4.3 Sequential Deep Neural Network Model

Here we have used two different sequential neural networks, namely LSTM and GRU to create our models. Choice of GRU or LSTM generally depends on the size and diverse nature of dataset and also sequence length. To utilize the specialty of both the architectures, we have used a cascaded LSTM-GRU with max pooling layer at two different point of the cascaded network. Usage of max pooling layer significantly improves model performance since it gets activation from two different hierarchical points. The whole model architecture is shown in Fig 2.

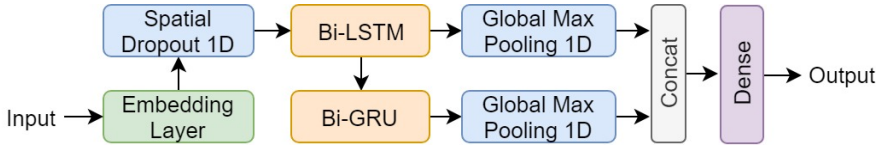


Fig. 2: Architecture overview.

4.4 Ensemble Model

Here, we have used two sets of concatenated word embeddings (Glove-FastText & Glove-Paragram) to feed into the input layer of the model. For each of these two embedding pairs, we have performed checkpoint ensembling [22] at 31^{st} and 32^{nd} epochs, thus creating a total of four classification models. As we wanted to create a weighted voting ensemble model, we empirically assigned weights to each of the models, which were 0.35 to the 31^{st} epoch models and 0.15 to the 32^{nd} epoch models. The final class label L_{final} was calculated using the equation given below.

$$\begin{aligned} L_{\text{final}} = 0.35 \times [L_{(\text{glove, fasttext, 32})} + L_{(\text{glove, paragram, 32})}] \\ + 0.15 \times [L_{(\text{glove, fasttext, 31})} + L_{(\text{glove, paragram, 31})}] \end{aligned}$$

Here, $L_{(\text{embedding 1, embedding 2, } n)}$ is the class prediction vector generated from the model trained by using concatenated word embedding of 'embedding 1' and 'embedding 2' for n epochs.

5 Experiments & Results

We have trained our model using adam [21] optimizer and categorical cross-entropy as loss function for 32 epochs with a batch-size of 128. Maximum sequence length was kept at 55, embedding size at 600, and learning rate at 0.001. For checkpoint ensemble, we have done weighted average of the models at two checkpoints which occur at the 31^{st} and 32^{nd} epoch during training period. Apart from our proposed method, we have also tried LSTM-CNN model and BERT [20] for question classification. Performance comparison between these two model with our proposed method on test data is shown in the Table 3. Class-wise F1 scores are also given along with the overall F1 scores. The performance of the individual models are shown in Table 4. From the results we can see that ensembling them resulted in an improvement of $\approx 7.22\%$ in F1 score.

Model Name	Sincere Question	Rhetorical	Sexual Content	Hate Speech	Hypothetical	Other	Micro Precision	Micro Recall	Micro F1
	F1	F1	F1	F1	F1	F1			
Our Method	0	0.812	0.545	0.296	0.222	0.500	0.6732	0.6732	**0.6732**
BERT	0	0.794	0.522	0.276	0.250	0.667	0.6534	0.6534	0.6534
Bi-LSTM-CNN	0	0.713	0.476	0.167	0.400	0.286	0.5544	0.5544	0.5544

Table 3: Model-wise performance metrics.

6 Conclusion & Discussions

In the present work we have described a series of experiments with a bidirectional LSTM based deep learning model built on top of two combinations of word

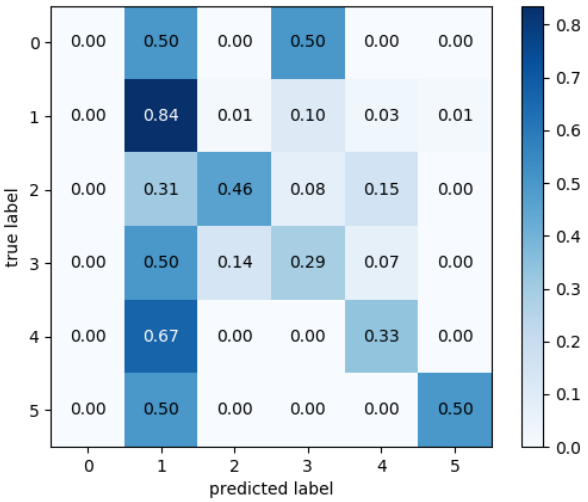
Ensemble Name	Sincere Question	Rhetorical	Sexual Content	Hate Speech	Hypothetical	Other	Micro Precision	Micro Recall	Micro F1
	F1	F1	F1	F1	F1	F1			
Glove-FastText-at-epoch-31	0	0.646	0.800	0.423	0	0.462	0.586	0.586	0.586
Glove-FastText-at-epoch-32	0	0.739	0.824	0.188	0	0.667	0.646	0.646	0.646
Glove-Para-at-epoch-31	0	0.680	0.757	0.340	0	0.500	0.586	0.586	0.586
Glove-Para-at-epoch-32	0	0.692	0.769	0.278	0	0.286	0.586	0.586	0.586

Table 4: Ensemble-model performance metrics.

embeddings: Glove-FastText and Glove-Paragram. Even with minimal hyperparameter tuning, our model with two Bi-LSTM layers was able to learn semantic sentence representations on fine-grained question classification task quite well and performs quite satisfactorily, as confirmed in our evaluation. As evidenced from the normalized confusion matrix shown in Fig 3, our model has performed remarkably well in classifying most of the rhetorical classes accurately. However, it was confused in the discrimination of some of the sexual content, hate speech, and hypothetical classes with the rhetorical class. Our model was also not able to identify the two non-insincere questions in the test set, which can be attributed to scarcity of this class in the dataset. As evident from Table 3, Bi-LSTM-CNN based models can identify the hypothetical class better, so we would try different variants and combinations of this model and our Bi-LSTM model in the future. We would also explore different variations of LSTM and GRU networks with combination of CNN layers and experiment with those models built on top of BERT embeddings. Increasing the size of our dataset by annotating more questions from the Quora dataset would be one of the goals as well.

References

1. Lewis, David D. "Naive (Bayes) at forty: The independence assumption in information retrieval." In European conference on machine learning, pp. 4-15. Springer, Berlin, Heidelberg, 1998.
2. McCallum, M., and K. Nigam. "Employing EM in pool-based active learning for text classification, 1998." In International Conference on Machine Learning (ICML).
3. Domingos, Pedro, and Michael Pazzani. "On the optimality of the simple Bayesian classifier under zero-one loss." Machine learning 29, no. 2-3 (1997): 103-130.
4. Joachims, Thorsten. "Text categorization with support vector machines: Learning with many relevant features." In European conference on machine learning, pp. 137-142. Springer, Berlin, Heidelberg, 1998.
5. Yang, Yiming, and Christopher G. Chute. "An example-based mapping method for text categorization and retrieval." ACM Transactions on Information Systems (TOIS) 12, no. 3 (1994): 252-277.
6. Schapire, Robert E., and Yoram Singer. "BoosTexter: A boosting-based system for text categorization." Machine learning 39, no. 2-3 (2000): 135-168.
7. Kim, Sang-Bum, Kyoung-Soo Han, Hae-Chang Rim, and Sung Hyon Myaeng. "Some effective techniques for naive bayes text classification." IEEE transactions on knowledge and data engineering 18, no. 11 (2006): 1457-1466.
8. Johnson, Rie, and Tong Zhang. "Effective use of word order for text categorization with convolutional neural networks." arXiv preprint arXiv:1412.1058 (2014).



h

Fig. 3: Confusion matrix, where 0) Sincere Question 1) Rhetorical 2) Sexual Content 3) Hate Speech 4) Hypothetical 5) Other.

9. Kim, Yoon. "Convolutional neural networks for sentence classification." arXiv preprint arXiv:1408.5882 (2014).
10. Zhang, Xiang, Junbo Zhao, and Yann LeCun. "Character-level convolutional networks for text classification." In Advances in neural information processing systems, pp. 649-657. 2015.
11. Liu, Pengfei, Xipeng Qiu, and Xuanjing Huang. "Recurrent neural network for text classification with multi-task learning." arXiv preprint arXiv:1605.05101 (2016).
12. Bengio, Yoshua, Patrice Simard, and Paolo Frasconi. "Learning long-term dependencies with gradient descent is difficult." IEEE transactions on neural networks 5, no. 2 (1994): 157-166.
13. Zhou, Peng, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. "Attention-based bidirectional long short-term memory networks for relation classification." In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 207-212. 2016.
14. Tang, Duyu, Bing Qin, and Ting Liu. "Document modeling with gated recurrent neural network for sentiment classification." In Proceedings of the 2015 conference on empirical methods in natural language processing, pp. 1422-1432. 2015.
15. Zhou, Chunting, Chonglin Sun, Zhiyuan Liu, and Francis Lau. "A C-LSTM neural network for text classification." arXiv preprint arXiv:1511.08630 (2015).
16. Joulin, Armand, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hrve Jgou, and Tomas Mikolov. "Fasttext. zip: Compressing text classification models." arXiv preprint arXiv:1612.03651 (2016).
17. Pennington, Jeffrey, Richard Socher, and Christopher Manning. "Glove: Global vectors for word representation." In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1532-1543. 2014.

18. Wieting, John, Mohit Bansal, Kevin Gimpel, and Karen Livescu. "From paraphrase database to compositional paraphrase model and back." Transactions of the Association for Computational Linguistics 3 (2015): 345-358.
19. Chung, Junyoung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. "Empirical evaluation of gated recurrent neural networks on sequence modeling." arXiv preprint arXiv:1412.3555 (2014).
20. Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).
21. Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).
22. Chen, Hugh, Scott Lundberg, and Su-In Lee. "Checkpoint Ensembles: Ensemble Methods from a Single Training Process." arXiv preprint arXiv:1710.03282 (2017).
23. Kamath, Sanjay, Brigitte Grau, and Yue Ma. "Predicting and Integrating Expected Answer Types into a Simple Recurrent Neural Network Model for Answer Sentence Selection." 20th International Conference on Computational Linguistics and Intelligent Text Processing. 2019.
24. Yoon, Seunghyun, Franck Dernoncourt, Doo Soon Kim, Trung Bui, and Kyomin Jung. "A Compare-Aggregate Model with Latent Clustering for Answer Selection." arXiv preprint arXiv:1905.12897 (2019).
25. Wei, Xiaocong, Hongfei Lin, Liang Yang, and Yuhai Yu. "A convolution-LSTM-based deep neural network for cross-domain MOOC forum post classification." Information 8, no. 3 (2017): 92.