ABARUAH at SemEval-2019 Task 5 : Bi-directional LSTM for Hate Speech Detection

Arup Baruah
Dept. of Comp. Sc. & Engg.
IIIT Guwahati, India
arup.baruah@gmail.com

Ferdous Ahmed Barbhuiya
Dept. of Comp. Sc. & Engg.
IIIT Guwahati, India
ferdous@iiitg.ac.in

Kuntal Dey
IBM Research India
New Delhi
kuntadey@in.ibm.com

Abstract

In this paper, we present the results obtained using bi-directional long short-term memory (BiLSTM) with and without attention and Logistic Regression (LR) models for SemEval-2019 Task 5 titled ”HatEval: Multilingual Detection of Hate Speech Against Immigrants and Women in Twitter”. This paper presents the results obtained for Subtask A for English language. The results of the BiLSTM and LR models are compared for two different types of preprocessing. One with no stemming performed and no stopwords removed. The other with stemming performed and stopwords removed. The BiLSTM model without attention performed the best for the first test, while the LR model with character n-grams performed the best for the second test. The BiLSTM model obtained an F1 score of 0.51 on the test set and obtained an official ranking of 8/71.

1 Introduction

Davidson et al. (2017) has defined hate speech as “language that is used to express hatred towards a targeted group or is intended to be derogatory, to humiliate, or to insult the members of the group.” Gambäck and Sikdar (2017), Badjatiya et al. (2017), Waseem (2016) and Waseem et al. (2017) have used the term hate speech to indicate tweets having racist or sexist comments. Social media is becoming a convenient medium to spread hate speech. Hate speech spread through social media has fueled riots in Myanmar [1], Sri Lanka [2], Charlottesville (USA) [3], and many other parts of the world. Thus, it is becoming increasingly important to detect and remove hate messages from the web. It is not possible to manually moderate the vast amount of text exchanged on the web. Developing automated systems to recognize hate speech is becoming crucially important. However, detecting hate speech in a text is more than just checking for the presence of hate words. Lexicon based approaches have not been very effective in hate speech detection (Nobata et al., 2016).

As part of the 13th workshop on semantic evaluation (SemEval-2019), shared task 5 defines two subtasks with regard to detection of hate speech against immigrants and women in Twitter (Basile et al., 2019). This task was conducted for tweets in English and Spanish language. In Subtask A, it is required to determine if a tweet, with a given target, is hateful or not. In Subtask B, it is required to determine if a given hateful tweet is aggressive or not and whether it targets an individual or a group.

[1]https://www.nytimes.com/2018/10/15/technology/myanmar-facebook-genocide.html

[2]https://qz.com/1223787/sri-lanka-shut-down-facebook-whatsapp-and-instagram-to-stop-anti-muslim-violence/

[3]https://psmag.com/social-justice/how-social-media-helped-organize-and-radicalize-americas-newest-white-supremacists

2 Related Work

Nobata et al. (2016) studied the performance of different features such as character n-grams, word n-grams, word2vec, character2vec, etc. in detecting hate speech. A regression model was used in their study. Malmasi and Zampieri (2017) made a similar study to compare the performance of different features in detecting hate speech. Djuric et al. (2015) used paragraph embeddings for detecting hate speech. Wulczyn et al. (2017) worked on detecting insults in Wikipedia comments. Davidson et al. (2017) worked on detecting hate speech when hate words are not explicitly used in the text. Malmasi and Zampieri (2018) used ensemble method and combined 16 different base classifiers to detect hate speech. Serrà et al. (2017) used character-based Recurrent Neural Network (RNN) to study the use of out-of-vocabulary words in hate speech. Gao and Huang (2017) used BiLSTMs with attention mechanism to detect hate speech. Pavlopoulos et al. (2017) used Convolutional Neural Network (CNN) and RNN with attention mechanism to moderate user comments. Sax (2016) compared the performance of several deep learning techniques, LR and Support Vector Machine (SVM) models in detecting hate speech.

3 Data

Table 1 below shows the proportion of positive and negative instances of hate speech in the train, development and test data sets. As can be seen, 42% of the instances in each of the data set are hate speech. The data collected in Gao et al. (2017) had only 0.6% hateful tweets. Nobata et al. (2016) found that only 5.9% of the online comments contained hate speech. The data sets used for this task, however, are quite balanced.

Type	Not Hate Speech	Hate Speech	Total
Train	5217 (58%)	3783 (42%)	9000
Dev	573 (57.3%)	427 (42.7%)	1000
Test	1718 (57.85%)	1252 (42.15%)	2970

Table 1: Data set statistics

The models used in our study were trained and validated using the train and development sets provided as part of this task. No other external data sets were used for training or validating. However, pre-trained GloVe [4] word vectors trained using 2 billion tweets were used as features for the two BiLSTM models. The 200-dimensional word vectors were used in our experiments.

4 Experimental Settings

4.1 Preprocessing

The preprocessing performed on the text includes the following -

1. All URLs, mentions and non-alphabetic characters were removed from the tweets.
2. The tweets were then converted to lowercase.
3. Stemming was performed using NLTK's Lancaster stemmer.
4. Stopwords were removed.
5. Tokenizer was used to convert each tweet into a sequence of integers by replacing each token by its index into the vocabulary. The tweet with the maximum length had 58 tokens (when stopwords were retained). This value is later used as the length of the input sequences to the Embedding layer of the BiLSTM models. So, tweets with length less than 58 were padded with zeros, so as to make all the tweets to be of the same length.

[4]https://nlp.stanford.edu/projects/glove/

4.2 Models Used

In our study, we used a BiLSTM without attention, and a BiLSTM with attention and an LR model. The details of our models are provided below.

4.2.1 BiLSTM without attention mechanism

Figure 1 shows the architecture of the BiLSTM used in our study. Both the forward and backward Long Short-Term Memory (LSTM) layers of the bi-directional layer of this model consisted of 100 units. Pre-trained GloVe embeddings were used to train the model. An embedding layer was used to feed the word vectors to the BiLSTM layer. A dropout of 0.25 was applied to the input of LSTM and a dropout of 0.1 was used for the recurrent connections. The BiLSTM layer was set to return the hidden state for each timestep. Thus, the output shape of the BiLSTM layer was (None,58,200). A global max pooling layer was used on top of the BiLSTM layer. This resulted in an output of the shape (None,200). The output from the global max pooling layer was fed to a Dense layer having 100 units. The Rectified Linear Unit (ReLU) activation function was used for this Dense layer. The output of the Dense layer was of the shape (None,100). The output was passed through a dropout layer with the rate set to 0.25. The output from the dropout layer was then passed through another Dense layer having a single unit. The sigmoid activation function was used for this layer. The model was trained using the Adam optimizer. The loss function used was binary cross-entropy. The model was trained with a batch size set to 32. The hyperparameter values used for the model are summarized in Table 2.

4.2.2 BiLSTM with attention mechanism

This model is exactly the same as the model described in 4.2.1 except that the global max pooling layer was replaced with attention mechanism. The hyperparameter values for both the models were also same.

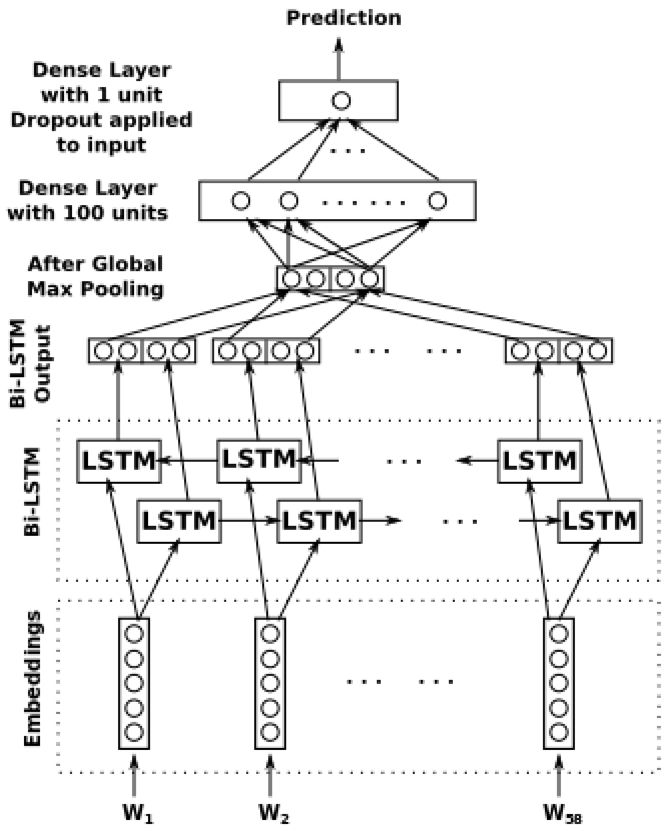

Figure 1: Architecture of the BiLSTM model used

Parameter	Value
Number of LSTM units	100
LSTM dropout	0.25
Recurrent dropout	0.10
Units in 1st Dense layer	100
Activation Function for 1st Dense layer	ReLU
Rate for dropout layer	0.25
Units in 2nd Dense layer	1
Activation Function for 2nd Dense layer	sigmoid
Optimizer	Adam
Loss Function	Binary cross-entropy

Table 2: Hyperparameters for the BiLSTM model

4.2.3 Logistic Regression

The third model we used was an LR model with L2 regularization. The class weight and C parameter were set to ‘balanced’ and 1.2 respectively. This model was trained with character n-grams (1 to 6), word n-grams (1 to 3), and a combination of both character and word n-grams. The hyperparameter values for the LR model were set as shown in Table 3.

5 Results & Discussions

The following tests were performed by us:

1. BiLSTM model trained using GloVe word embeddings as features.

2. BiLSTM model with attention mechanism trained using GloVe word embeddings as features.

3. LR model trained using character n-grams (1 to 6) only.

4. LR model trained using word n-grams (1 to 3) only.

5. LR model trained using both character and word n-grams concatenated together.

Parameter	Value
Regularization	L2
C	1.2
Class weight	balanced

Table 3: Hyperparameters for the LR model

Table 4 shows the results obtained by our models on the development set when stemming was not performed and stopwords were not removed. Table 5 shows the results when stemming was performed and stopwords were also removed.

Approach	Features	Acc	Prec	Rec	F1
BiLSTM	GloVe word embeddings	**0.748**	**0.749**	**0.748**	**0.748**
BiLSTM with attention	GloVe word embeddings	0.737	0.738	0.737	0.737
Logistic Regression	Char n-grams (1 to 6)	0.727	0.733	0.727	0.728
Logistic Regression	Word n-grams (1 to 3)	0.722	0.729	0.722	0.723
Logistic Regression	Char n-grams & Word n-grams	0.727	0.734	0.727	0.728

Table 4: Results of our models on Dev set of Task 5-Subtask A (No stemming + No Stopwords removed)

Approach	Features	Acc	Prec	Rec	F1
BiLSTM	GloVe word embeddings	0.674	0.699	0.674	0.675
BiLSTM with attention	GloVe word embeddings	0.698	0.698	0.698	0.689
Logistic Regression	Char n-grams (1 to 6)	**0.743**	**0.749**	**0.743**	**0.744**
Logistic Regression	Word n-grams (1 to 3)	0.727	0.731	0.727	0.728
Logistic Regression	Char n-grams & Word n-grams	0.739	0.746	0.739	0.740

Table 5: Results of our models on Dev set of Task 5-Subtask A (Stemming + Stopwords removed)

Approach	Accuracy	Precision	Recall	F1
SVC Baseline	0.49	0.60	0.55	0.45
MFC Baseline	**0.58**	0.29	0.50	0.37
BiLSTM	0.54	**0.64**	**0.59**	**0.51**
Best System	0.65	0.69	0.68	0.65

Table 6: Official results for Task 5-Subtask A

As can be seen from Table 4, the BiLSTM model without attention outperformed the other two models for all the metrics. However, the improvements achieved were not very significant compared to the other two models. It can also be seen that the choice of character or word n-grams did not make much difference to the performance of the LR model. Equivalent results were obtained for all the 3 tests performed using the LR model. This is surprising considering the fact that character n-grams usually performs better than word n-grams for text containing obfuscated words. Mehdad and Tetreault (2016) mentions that offenders often obfuscate the hate words in order to avoid detection by keyword-based filters. So, character n-gram features should have improved the performance of the model. One explanation for this observation could be the removal of numeric and special characters from the tweets during the data preprocessing stage. Numeric and special characters are used frequently used to obfuscate hate words. 'ass' replaced by 'a$$', 'slut' replaced by 's1ut' etc. are examples of such obfuscation. So, a test was performed without removing the numeric and special characters. However, no significant increase in the performance of character-based model was observed.

As can be seen from Table 5, the LR models performed better than the BiLSTM models when stemming was performed and stopwords were removed. The character n-gram based LR model performed the best for all the metrics considered.

The predictions obtained for the test set using the BiLSTM model without attention mechanism were submitted as the final predictions. The model that was trained with stopwords retained and stemming not performed was used to make the predictions on the test set. The official results obtained for the submission are shown in Table 6. Our official ranking is 8/71 in subtask A for the English language. As can be seen from the results, the MFC baseline had the best accuracy score. By labeling all the test instances with the most frequent label, the MFC baseline was able to obtain a better accuracy score. The MFC baseline achieved a better accuracy score at the cost of a low precision value. This resulted in a low F1 score for the baseline. Our BiLSTM model obtained a significantly higher F1 score compared to the MFC baseline. Our BiLSTM model outperformed the SVC baseline on all the metrics.

Table 7 shows the confusion matrices for the tests performed by retaining the stopwords and without performing stemming. The BiLSTM models were able to achieve better F1 score compared to the LR model by making better predictions for the benign class.

Table 8 shows the confusion matrices for the tests performed with stopwords removed and stemming performed. As can be seen, the BiLSTM models with attention and without attention show opposite tendencies. While the BiLSTM model without attention gets better in predicting the hate class, it becomes weaker in predicting the benign class. The opposite is true for the BiLSTM model with attention. The character n-gram based LR model gets better in predicting both the hate and benign classes.

6 Error Analysis

From the errors made by our system, it is evident that the model was not able to determine correctly if the hate words have really been used to express hate. For e.g., the following tweet from the test set is not a hate speech - "I can be a bitch and

	BiLSTM Predictions		**BiLSTM with Attention Predictions**		**LR (char) Predictions**		**LR (word) Predictions**		**LR (char+word) Predictions**	
	Not	**Hate**	**Not**	**Hate**	**Not**	**Hate**	**Not**	**Hate**	**Not**	**Hate**
Not	437	136	438	135	411	162	408	165	410	163
Hate	116	311	128	299	111	316	113	314	110	317

Table 7: Confusion Matrix (No Stemming + No stopword removed)

	BiLSTM Predictions		**BiLSTM with Attention Predictions**		**LR (char) Predictions**		**LR (word) Predictions**		**LR (char+word) Predictions**	
	Not	**Hate**	**Not**	**Hate**	**Not**	**Hate**	**Not**	**Hate**	**Not**	**Hate**
Not	345	228	476	97	421	152	418	155	416	157
Hate	98	329	205	222	105	322	118	309	104	323

Table 8: Confusion Matrix (Stemming + Stopwords removed)

an asshole but I will love you and care about you more than any other person you have met." Here, the speaker is attributing the word 'bitch' to himself/herself. So, the tweet cannot be a hate speech. However, our system wrongly classifies the tweet as a hate speech.

There have been many instances where our system wrongly classifies a tweet as hate speech just because of the mere presence of words such as 'bitch', '#buildthewall' etc. even when the tweet is not intended against women or immigrants. For e.g., the tweet 'He is a snake ass bitch. He is a fugly slut...' is a hate speech but it is not intended against women. But our system was not able to detect this and wrongly classifies it.

7 Conclusion

With hate speech in social media fomenting many riots in different parts of the world, it is becoming increasingly important to prevent their spread. While manual moderation is almost impossible, the need of the time is automated systems for their removal. The BiLSTM and logistic regression models used in this study have obtained some success compared to the baselines used. But there is much left to be desired. Hate words used in the benign sense and hate speech not directed at women and immigrants were wrongly getting classified. Contextual information and features such as part-of-speech (POS), dependency relations may help in classifying such instances correctly.

References

Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. Deep Learning for Hate Speech Detection in Tweets. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 759–760, Perth.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Rangel, Paolo Rosso, and Manuela Sanguinetti. 2019. Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019)*. Association for Computational Linguistics.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated Hate Speech Detection and the Problem of Offensive Language. In *Proceedings of the Eleventh International AAAI Conference on Web and Social Media (ICWSM 2017)*, pages 512–515, Montreal.

Nemanja Djuric, Jing Zhou, Robin Morris, Mihajlo Grbovic, Vladan Radosavljevic, and Narayan Bhamidipati. 2015. Hate Speech Detection with Comment Embeddings. In *International World Wide Web Conference (WWW), 2015*, pages 29–30, Florence, Italy.

Björn Gambäck and Utpal Kumar Sikdar. 2017. Using convolutional neural networks to classify hate-speech. In *Proceedings of the First Workshop on Abusive Language Online*, pages 85–90, Vancouver, BC, Canada. Association for Computational Linguistics.

Lei Gao and Ruihong Huang. 2017. Detecting online hate speech using context aware models. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 260–266, Varna, Bulgaria. INCOMA Ltd.

Lei Gao, Alexis Kuppersmith, and Ruihong Huang. 2017. Recognizing explicit and implicit hate speech using a weakly supervised two-path bootstrapping approach. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 774–782, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Shervin Malmasi and Marcos Zampieri. 2017. Detecting hate speech in social media. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 467–472, Varna, Bulgaria. INCOMA Ltd.

Shervin Malmasi and Marcos Zampieri. 2018. Challenges in discriminating profanity from hate speech. *Journal of Experimental & Theoretical Artificial Intelligence*, 30(2):187–202.

Yashar Mehdad and Joel Tetreault. 2016. Do characters abuse more than words? In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 299–303, Los Angeles. Association for Computational Linguistics.

Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive Language Detection in Online User Content. In *Proceedings of the 25th International Conference on World Wide Web*, pages 145–153, Montreal.

John Pavlopoulos, Prodromos Malakasiotis, and Ion Androutsopoulos. 2017. Deep learning for user comment moderation. In *Proceedings of the First Workshop on Abusive Language Online*, pages 25–35, Vancouver, BC, Canada. Association for Computational Linguistics.

S. Sax. 2016. Flame Wars: Automatic Insult Detection. `http://cs224d.stanford.edu/reports/Sax.pdf`, (Technical Report).

Joan Serrà, Ilias Leontiadis, Dimitris Spathis, Gianluca Stringhini, Jeremy Blackburn, and Athena Vakali. 2017. Class-based prediction errors to detect hate speech with out-of-vocabulary words. In *Proceedings of the First Workshop on Abusive Language Online*, pages 36–40, Vancouver, BC, Canada. Association for Computational Linguistics.

Zeerak Waseem. 2016. Are you a racist or am i seeing things? annotator influence on hate speech detection on twitter. In *Proceedings of the First Workshop on NLP and Computational Social Science*, pages 138–142, Austin, Texas. Association for Computational Linguistics.

Zeerak Waseem, Thomas Davidson, Dana Warmsley, and Ingmar Weber. 2017. Understanding abuse: A typology of abusive language detection subtasks. In *Proceedings of the First Workshop on Abusive Language Online*, pages 78–84, Vancouver, BC, Canada. Association for Computational Linguistics.

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex Machina: Personal Attacks Seen at Scale. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1391–1399, Perth.