YNU NLP at SemEval-2019 Task 5: Attention and Capsule Ensemble for Identifying Hate Speech

Bin Wang, Haiyan Ding*
School of Information Science and Engineering
Yunnan University, Yunnan, P.R. China
hyding@ynu.edu.cn

Abstract

This paper describes the system submitted to SemEval 2019 Task 5: Multilingual detection of hate speech against immigrants and women in Twitter (hatEval). Its main purpose is to conduct hate speech detection on Twitter, which mainly includes two specific different targets, immigrants and women. We participate in both subtask A and subtask B for English. In order to address this task, we develope an ensemble of an attention-LSTM model based on HAN and a BiGRU-capsule model. Both models use fastText pre-trained embeddings, and we use this model in both subtasks. In comparison to other participating teams, our system is ranked 16th in the Subtask A for English, and 12th in the Subtask B for English.

1 Introduction

In recent years, the popularity of social networking and microblogging sites has increased, attracting more and more users. With this huge user base, social media will continue to release a large number of user-generated content. As the use of social media has grown, other undesirable phenomena and behaviors have emerged. Social media users often abuse this freedom to spread abuse or hateful posts or comments. In many cases, these user-generated content is inherently offensive or proactive, and users may have to deal with threats such as cyber attacks or cyberbullying, as well as other undesirable phenomena (Whittaker and Kowalski, 2015). So the problem of detecting and possibly limiting the spread of hate speech is becoming more and more important.

In order to solve the problem of abuse of language in social media platforms, some related research has been published, such as cyberbullying (Dadvar et al., 2013), hate speech (Warner and Hirschberg, 2012) and abusive language (Chen et al., 2012), most methods are based on surveillance methods (Schmidt and Wiegand, 2017). There are also some (racial discrimination) bias towards specific goals. In (Waseem and Hovy, 2016), the authors proposed a series of criteria based on critical race theory to identify racism and gender discrimination, they use n-gram models for research; Tulkens et al. studied racism detection in Dutch social media (Tulkens et al., 2016). A recent discussion of the challenge of identifying hate speech was proposed by Kumar et al. (Kumar et al., 2018). The results show that it is difficult to distinguish between open and covert attacks in social media.

SemEval 2019 Task 5 is proposed to identify hate speech about immigrants and women in Twitter for English or Spanish, and classify hate speech and judge whether the target is an individual or a group (Basile et al., 2019). Hate speech is often defined as any communication that attacks an individual or group through certain characteristics (such as gender, nationality, religion, or other characteristics) in social media platforms. This task gives us some text data from Twitter, we need to classify the content through computational analysis. The task has two subtasks, in which Subtask A is Hate speech detection for immigrants and women: It's a binary classification task, the system must judge whether a tweet with a specific goal (female or immigrant) in English or Spanish is hate speech; Subtask B is Aggressive behavior and target classification: This subtask is to classify the identified hate speech based on Subtask A, to judge whether it is aggressive or non-aggressive, and then to identify the target being harassed as an individual or group.

In this paper, we developped a system stacked two different neural network models: an attention-based model with LSTMs and an Capsule-based model with BiGRUs. We make some changes to

*Corresponding author

Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019), pages 529–534
Minneapolis, Minnesota, USA, June 6–7, 2019.

Hierarchical Attention Network to make it more suitable for this task, the detailed description of the Attention-LSTM model is provided in Section 2.2. Next, we build a BiGRU-Capsule model using the latest "Capsule" model proposed by (Sabour et al., 2017), the detailed description is provided in Section 2.3. In Section 2.4, we describe the use of stacking as ensemble. In Section 3.1, some details about data preprocessing for this task are described. In Section 3.2 and Section 3.3, the hyperparameter setting and result analysis used in the whole experiment are introduced in detail.

2 Data and System Description

2.1 Data description

In this task, we only use the official training data set for training and trial data set to verify. In Subtask A, the purpose is to distinguish whether the tweet is hate speech, the data is divided into two categories(HS): 0 means non-hate speech, and conversely, 1 is hate speech. Similarly, in Subtask B, 0 is indicative of aggressiveness in categorizing hate speech(TR), 1 is non-aggressive, and in the goal of judging hate speech(AG), 0 means individual and 1 means group. In this task, we only participate in Subtask A and Subtask B in English. There are 9000 tweets in training data set, 1000 tweets in development data set, and 2971 tweets in final test data set. In the training data set, there are 5,217 labels are 0s and 3,783 labels are 1s in the label HS; in the label TR, 7659 labels are 0s and 1341 labels are 1s; and in the label AG 7440 labels are marked as 0s, 1560 are marked as 1s. Although the data of the label TR and the label AG are very unbalanced, since the ratio of 0 and 1 in the label HS is close to balance, we have not dealt with the data imbalance in this task.

2.2 Attention-LSTM Model

Here we have made some changes to HAN (Yang et al., 2017). The overall structure is shown in Figure 1. The replacement of BiGRU with LSTM (Hochreiter and Schmidhuber, 1997) is found to be significantly better than the original model for this task. The architecture of Attention-LSTM model is shown in Figure 1.

We use an LSTM to encode the sentences and to get annotations of words by summarizing information for word.

Not all words contribute equally to the expression of the emotion in the sentence. Emotion

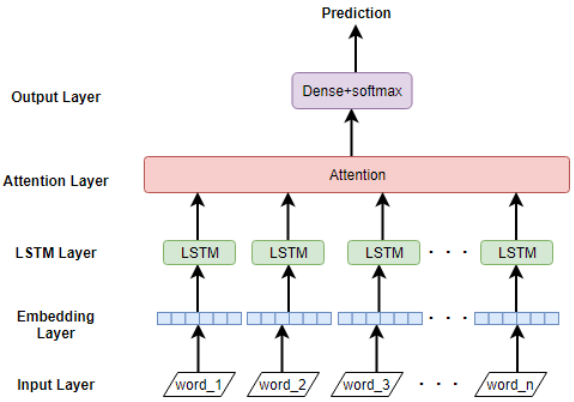

Figure 1: The architecture of Attention-LSTM Model.

greatly influences whether the sentence is hate speech, and also is helpful in identifying hate categories. There may be only a few words in a sentence that are crucial for the judgment of the goal of hate speech. So here we introduce the attention mechanism so that the system can better focus on words that are useful to identify hate speech, then it extracts those words and aggregates the representation of those important words to form a sentence vector.

First, we feed the word annotation h_i, and through a one-layer MLP to get a deeper representation u_i.

$$u_i = \tanh(W_s * h_i + b_s) \tag{1}$$

Then, we compute the similarity between u_i and word-level context vector u_s, and obtain a normalized weight α_i of importance by softmax function.

$$\alpha_i = \frac{exp(u_i^T * u_s)}{\sum_i exp(u_i^T * u_s)} \tag{2}$$

Finally, we compute the sentence vector s by a weighted sum of the word annotations h_i based on the normalized importance weights. s summarizes all the information of words in a context.

$$s = \sum_i \alpha_i * h_i \tag{3}$$

2.3 BiGRU-Capsule model

In order to improve the performance, in this system we use BiGRU and the latest capsule model (Sabour et al., 2017). The architecture of BiGRU-Capsule model is shown in Figure 2.

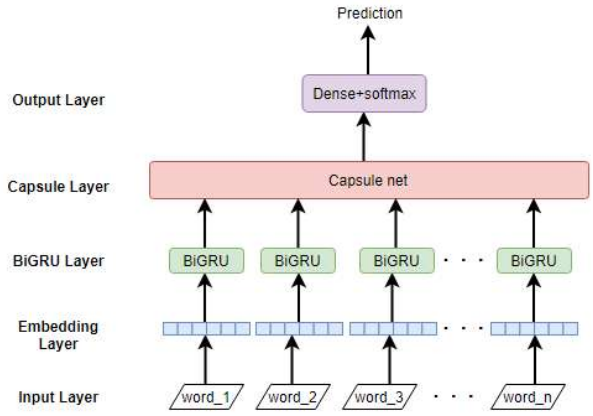

Figure 2: The architecture of BiGRU-Capsule Model.

First, we use the BiGRU layer to encode the sentences. As a variant of LSTM, GRU combines the Forget Gate and the Input Gate into a single Update Gate.

The bidirectional GRU is composed of two GRUs stacked one on top of the other. The output is determined by the state of the two GRUs.

In the capsule layer, the feature output by the previous BiGRU layer as an input to feed to the capsule network, to obtain deeper feature information. Capsule network was proposed by (Sabour et al., 2017), the main idea is to use neuron vectors instead of single neuron nodes of traditional neural networks, and finally train this new neural network by means of Dynamic Routing.

First, the "prediction vectors" $\widehat{u}_{j|i}$ are obtained by multiplying the output u_i of each capsule by a weight matrix W_{ij}.

$$\widehat{u}_{j|i} = W_{ij} * u_i \tag{4}$$

Then, all the "prediction vectors" are weighted summed to obtain the capsule s_j

$$s_j = \sum_i c_{ij} * \widehat{u}_{j|i} \tag{5}$$

where c_{ij} is the coupling coefficient between the capsules determined by "routing softmax", and the sum of the coupling coefficient of all the capsule is 1 in the layer.

Finally, we use the nonlinear "squashing" function to compress the length of the output vector of capsule between 0 and 1.

$$v_i = \frac{||s_j||^2}{1 + ||s_j||^2} \frac{s_j}{||s_j||} \tag{6}$$

where v_j is the output vector of capsule j.

2.4 Ensemble

Ensembling of several models is a widely used method to improve the performance of the overall system by combining predictions of several classifiers (Hansen and Salamon, 2002). A combination of all features leads to the best performance, they provide complementary information. Several ensembling techniques have been proposed recently: mixing experts (Jordan and Jacobs, 1991), model Stacking (Wolpert, 1992), Bagging and Boosting (Breiman, 1999) . We use Stacking in this task. The main reason is that other methods are relatively simple and may have large learning errors. Stacking is like an upgraded version of Bagging. The second layer of learning in Stacking is to find the right weight or the right combination.

The Stacking algorithm is divided into two layers. The first layer uses different algorithms to form n weak classifiers, and simultaneously generates a new data set of the same size as the original data set. This new data set and a new algorithm form the second layer classifier.

When using the Stacking strategy, we do not execute a simple logical processing of the weak learner, but add a layer of learner, that is, we will use the learning result of the Attention-LSTM model and the BiGRU-Capsule model as input, building an MLP model as second layer classifier, the MLP model has only one hidden layer, there are 200 hidden nodes in the layer, and a Dense layer as the output of the ensemble. The architecture of the ensemble model is shown in Figure 3.

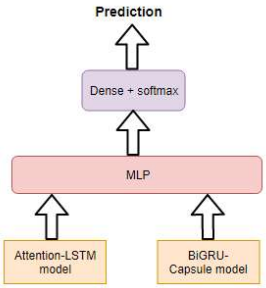

Figure 3: The architecture of the ensemble Model.

3 Experinment and Result analysis

3.1 Data Processing

The official data set is very noisy and needs to be cleaned. Preprocessing the text makes it easy for

the model to extract features and representations. We perform the following preprocessing.

- Hashtags are important markers for determining sentiment or user intention. The "#" symbol is removed and the word itself is retained. e.g.: in the sentence, "#BuildTheWall and #BuildThatWall" are marked as 1 in most cases in the training data set.
- Username mentions, e.g.: words starting with "@", generally provide no information in terms of sentiment. Hence such terms are removed completely from the tweets.
- Repeated full stops, question marks and exclamation marks are replaced with a single instance with a special token "repeat" added.
- All contractions are split into two tokens by using regular expression (e.g.: "it's" is changed to "it" and "is").
- All URLs, phone numbers and date numbers are replaced respectively as "URL", "PHONENUMBER", "NUMBER".
- Emoticons (such as, ':(', ':)', ':P' and emoji etc.) are replaced as their own meanings by emotion lexicons[1].
- Tokens are converted to lower case.

3.2 Hyperparameter setting

We select the longest sentence in all cleaned data as the maximum sentence length, which is 58 characters. The processed text is then converted to word embeddings. Converting text into word embeddings represents each word of the text with a d dimensional vector (Mikolov et al., 2013). We use available pre-trained embeddings which are trained on large data set.

In the attention-LSTM model, there is mainly one LSTM layer and one attention layer. There are 300 hidden nodes in the LSTM layer. We also use the Dropout layer with rate 0.25 between the LSTM layer and the Attention layer. The purpose is to prevent over-fitting. Finally, we also use Batch normalization with a size of 0.1 behind the Attention layer, this layer is normalized for each neuron, even only need to normalize a certain neuron, rather than normalize a whole layer of neurons. The purpose is to make the model training converge faster, and the distribution of model hidden output features is more stable, which is more conducive to model learning.

In the capsule model, we build two layers of BiGRU and one layer of Capsule. In the capsule layer, our routing size is set to 5, the number of capsules is set to 10, and the size of the capsule is set to 16. For BiGRU, we set the hidden unit to 128, and a Dropout layer with size 0.25 is added between the BiGRU layer and the Capsule layer to prevent overfitting. Finally, in all models, the loss function is $binary\ crossentropy$, and the optimizer is $adam$ (Kingma and Ba, 2014).

3.3 Result analysis

For this task, we select fastText (Joulin et al., 2017), because in this task we find that the result of fastText is much better than other word vectors such as Word2vec and Glove. Table 1 is the result of different word vectors as embedding.

Word Vector	Dim	macro-F1 Result
Word2vec	300d	0.746
Glove-twitter	200d	0.763
BPEmb	300d	0.732
fastText	**300d**	**0.761**

Table 1: The result of different word vectors as embedding in the attention-LSTM model for development data set in Subtask A.

We think that the reason why fastText works better than others is that Word2vec treats each word in the corpus as an atom, and it generates a vector for each word, which ignores the internal morphological features of the word, such as: "apple" and "apples", but fastText overcomes this problem by using character-level n-grams to represent a word; fastText may have a higher dimension than Glove-twitter, indicating more features; BPemb is based on Byte-Pair Encoding, the effect of fastText is obviously better than it.

Here we compare the effects of BiGRU, LSTM and BiLSTM and find that LSTM is superior to BiGRU and BiLSTM in this model, and the results are shown in Table 2.

We compare the results achieved by our individual approaches with the submitted ensemble system in Table 3. For brevity, we only show the macro-F1 scores on the development set.

The results of our test data set and the top three results of the official rankings are shown in Table

[1] https://emojipedia.org/

Model	macro-F1 Result
Attention-BiGRU	0.751
Attention-BiLSTM	0.742
Attention-LSTM	**0.761**

Table 2: The results of using BiGRU, LSTM, Bi-LSTM with the attention mechanism for development data set in Subtask A.

Model	macro-F1 Result
Attention-LSTM	0.761
BiGRU-Capsule	0.758
Ensemble	**0.782**

Table 3: The result of different model for development data set in Subtask A.

4 and Table 5. From the results of our model in the test data for Subtask A, its macro-F1 is only 0.498, which is 0.284 lower than the result of the training data set at training phase of 0.782, indicating that our model may have some over-fitting.

Team	macro-F1 Result
saradhix	0.651
Panaetius	0.571
YunxiaDing	0.546
Our model	**0.493**

Table 4: The results of our test data set and the top three results of the official rankings in Subtask A.

Team	EMR Result
ninab	0.570
iqraameer133	0.568
scmhl5	0.483
Our model	**0.344**

Table 5: The results of our test data set and the top three results of the official rankings in Subtask B.

4 Conclusion

In this paper, we propose a deep learning framework to classify hate speech about immigrants and women in tweets for English. The proposed approach is based on an ensemble of attention and capsule, allowing us to explore the different directions of a neural network based methodology. Each individual approach is described in detail with a view of making our experiments replicable.

In the future, we would like to experiment with handcrafted features in addition to word-vectors and lexicon features. We would also experiment with AffectiveTweets package (Mohammad and Bravo-Marquez, 2017) such as TweetToSentiStrengthFeatureVector, TweetNLP-Tokenizer etc., and try to extract the NER feature to further improve the model performance.

References

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.

Leo Breiman. 1999. Prediction games and arcing algorithms. *Neural Computation*, 11(7):1493.

Ying Chen, Yilu Zhou, Sencun Zhu, and Heng Xu. 2012. Detecting offensive language in social media to protect adolescent online safety. In *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*, pages 71–80. IEEE.

Maral Dadvar, Dolf Trieschnigg, Roeland Ordelman, and Franciska de Jong. 2013. Improving cyberbullying detection with user context. In *European Conference on Information Retrieval*, pages 693–696. Springer.

L. K Hansen and P Salamon. 2002. Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10):993–1001.

Sepp Hochreiter and Jrgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Michael I. Jordan and Robert A. Jacobs. 1991. Hierarchies of adaptive experts. In *Advances in Neural Information Processing Systems*, pages 985–992.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431. Association for Computational Linguistics.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *Computer Science*.

Ritesh Kumar, Atul Kr Ojha, Shervin Malmasi, and Marcos Zampieri. 2018. Benchmarking aggression identification in social media. In *Proceedings of the*

First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018), pages 1–11.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, 26:3111–3119.

Saif Mohammad and Felipe Bravo-Marquez. 2017. Wassa-2017 shared task on emotion intensity. In *The Workshop on Computational Approaches To Subjectivity*, pages 34–49.

Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. 2017. Dynamic routing between capsules. In *Advances in Neural Information Processing Systems*, pages 3856–3866.

Anna Schmidt and Michael Wiegand. 2017. A survey on hate speech detection using natural language processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*, pages 1–10.

Stéphan Tulkens, Lisa Hilte, Elise Lodewyckx, Ben Verhoeven, and Walter Daelemans. 2016. A dictionary-based approach to racism detection in dutch social media. *arXiv preprint arXiv:1608.08738*.

William Warner and Julia Hirschberg. 2012. Detecting hate speech on the world wide web. In *Proceedings of the Second Workshop on Language in Social Media*, pages 19–26. Association for Computational Linguistics.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL student research workshop*, pages 88–93.

Elizabeth Whittaker and Robin M Kowalski. 2015. Cyberbullying via social media. *Journal of School Violence*, 14(1):11–29.

David H Wolpert. 1992. Stacked generalization. *Neural Networks*, 5(2):241–259.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2017. Hierarchical attention networks for document classification. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489.