Detecting Aggression and Toxicity using a Multi Dimension Capsule Network

Saurabh Srivastava
TCS Research
Noida, India
sriv.saurabh@tcs.com

Prerna Khurana
TCS Research
Noida, India
prerna.khurana2@tcs.com

Abstract

In the era of social media, hate speech, trolling and verbal abuse have become a common issue. We present an approach to automatically classify such statements, using a new deep learning architecture. Our model comprises of a Multi Dimension Capsule Network that generates the representation of sentences which we use for classification. We further provide an analysis of our model's interpretation of such statements. We compare the results of our model with state-of-art classification algorithms and demonstrate our model's ability. It also has the capability to handle comments that are written in both Hindi and English, which are provided in the TRAC dataset. We also compare results on Kaggle's Toxic comment classification dataset.

1 Introduction

Many people refrain from expressing themselves or giving opinions online for the fear of harassment and abuse. Twitter admitted that such behavior is resulting in users quitting from their platform and sometimes they are even forced to change their location. Due to this, combating hate speech and abusive behavior has become a high priority area for major companies like Facebook, Twitter, Youtube, and Microsoft. With an ever-increasing content on such platforms, it makes impossible to manually detect toxic comments or hate speech.

Earlier works in Capsule network based deep learning architecture to classify toxic comments have proved that these networks work well as compared to other deep learning architectures (Srivastava et al., 2018). In this paper, we investigate the performance of a multi-dimension Capsule network as opposed to using a fixed dimension Capsule network for capturing a sentence representation and we shall discuss how well it captures features necessary for classification of such sentences. For our experiments we have taken up two different datasets, namely, TRAC-1, which has comments in Hindi and English both scraped from Facebook and Twitter and, Kaggle's Toxic Comment Classification Challenge which is a multi-label classification task. In our experiments, we discovered that our model is capable of handling transliterated comments, which is another major challenge in this task. Since one of the datasets we used, TRAC-1, was crawled from public Facebook Pages and Twitter, mainly on Indian topics, hence there is a presence of code-mixed text. This type of data is more observed in a real-world scenario.

2 Related Work

Numerous machine learning methods for detection of inappropriate comments in online forums exist today. Traditional approaches include Naive Bayes classifier (Kwok and Wang, 2013)(Chen et al., 2012)(Dinakar et al., 2011), logistic regression (Waseem, 2016) (Davidson et al., 2017) (Wulczyn et al., 2017) (Burnap and L. Williams, 2015), support vector machines (Xu et al., 2012) (Dadvar et al., 2013) (Schofield and Davidson, 2017), and random forests. However, deep learning models, for instance, convolutional neural networks (Gambäck and Sikdar, 2017) (Potapova and Gordeev, 2016) and variants of recurrent neural networks (Pavlopoulos et al., 2017) (Gao and Huang, 2017)(Pitsilis et al., 2018) (Zhang et al., 2018), have shown promising results and achieved better accuracies. Recent works in Toxic comment classification (van Aken et al.) compared different deep learning and shallow approaches on datasets and proposed an ensemble model that outperforms all approaches. Further, work done by (Nikhil et al., 2018) (Kumar et al., 2018) proposed LSTMs with attention on TRAC dataset for bet-

Proceedings of the Third Workshop on Abusive Language Online, pages 157–162
Florence, Italy, August 1, 2019.

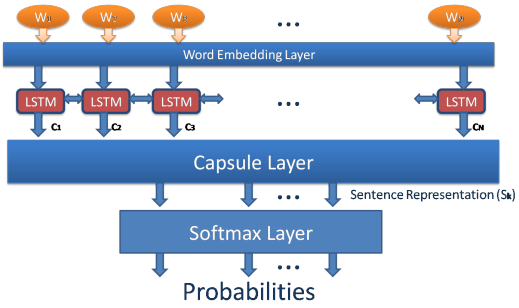

Figure 1: Multi Dimension Capsule Network

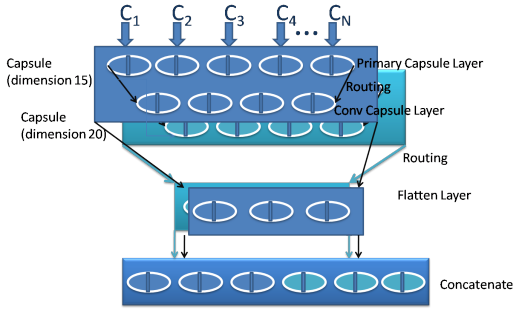

Figure 2: Capsule Layers

ter classification. Capsule networks have shown to work better on images (Sabour et al., 2017), also recently these networks have been investigated for text classification (Yang et al., 2018). (Srivastava et al., 2018) proposed a Capsule Net based classifier for both the datasets used in this study, and showed that it works better than the previous state-of-art methods. We propose to extend this work by modifying it into a multi-dimension Capsule network, taking inspiration from Multi filter CNNs (Kim, 2014a).

3 Multi Dimension Capsule Net for Classification

We describe our multi-dimension Capsule Net architecture in this section which consists primarily of 5 layers as shown in Fig 1. To get initial sentence representation, we concatenated individual word representation obtained from pretrained fastText embeddings (Joulin et al., 2016). The sentence representation is then passed through a feature extraction layer which consists of BiLSTM units to get a sentence representation. This representation is then passed through the Primary and Convolutional Capsule Layer to extract the high-level features of a sentence. Finally, the features are then passed through a classification layer to calculate the class probabilities.

Word Embedding Layer: To get initial sentence representation, we used a weight matrix $\mathbf{W} \in \mathbb{R}^{d_w} \times |V|$ where, d_w is the fixed vector dimension and $|V|$ is vocabulary size. The vector in column $\mathbf{w_i}$ of $\mathbf{W}$ represents lexical semantics of a word w_i obtained after pre-training an unsupervised model on a large corpus (Mikolov et al., 2013), (Pennington et al., 2014), (Joulin et al., 2016).

Feature Extraction Layer: This layer consists of BiLSTM units to capture the contextual information within words of a sentence. As proposed in (Schuster and Paliwal, 1997), we obtained both the forward and backward context of a sentence. The layer outputs $\mathbf{C_i} = [\overrightarrow{c_i}; \overleftarrow{c_i}] \in \mathbb{R}^{2 \times d_{sen}}$ for a word w_i where, $\overrightarrow{\mathbf{c_i}}$ and $\overleftarrow{\mathbf{c_i}}$ are forward and backward contexts (hidden activations), and d_{sen} is number of LSTM units. Finally, for all the N words, we have $\mathbf{C} = [\mathbf{C_1}, \mathbf{C_2}, ..., \mathbf{C_N}] \in \mathbb{R}^{N \times (2 \times d_{sen})}$. We have used BiLSTMs for feature extraction as opposed to CNNs which have been used as a feature extraction layer for capsules in (Yang et al., 2018) and (Sabour et al., 2017), as CNNs put forward a difficulty of choosing an optimal window size (Lai et al., 2015) which could introduce noise.

Primary Capsule Layer: In (Sabour et al., 2017) authors proposed to replace singular scalar outputs of CNNs with highly informative vectors which consist of "instantiation parameters". These parameters are supposed to capture local order of word and their semantic representation (Yang et al., 2018). We have extended the model proposed in (Srivastava et al., 2018) to capture different features from input by varying the dimension of capsules. As proposed in (Kim, 2014b). having different window size can allow us to capture N-gram features from the input, we hypothesize that by varying dimension of capsules we can capture different instantiation parameters from the input.

For context vectors $\mathbf{C_i}$, we used different shared windows refer Fig 2, $\mathbf{W_b} \in \mathbf{R}^{(2 \times d_{sen}) \times d}$ to get capsules $\mathbf{p_i}$, $\mathbf{p_i} = g(\mathbf{W^b C_i})$ where, $\mathbf{g}$ is non-linear *squash* activation (Sabour et al., 2017), d is capsule dimension and d_{sen} is the number of LSTM units used to capture input features. Factor d can be used to vary a capsule's dimension which can be used to capture different instantiation parameters. The capsules are then stacked together to create a capsule feature map, $\mathbf{P} = [\mathbf{p1}, \mathbf{p2}, \mathbf{p3}, ..., \mathbf{p_C}] \in \mathbb{R}^{(N \times C \times d)}$ consisting of total $N \times C$ capsules of dimension d.

Dynamic Routing algorithm was proposed in

(Sabour et al., 2017) to calculate agreement between capsules. The routing process introduces a **coupling effect** between the capsules of level (l) and (l+1) controlling the connection strengths between child and parent capsules. Output of a capsule is given by

$$\mathbf{s_j} = \sum_{i=1}^{m} \mathbf{c_{ij}} \hat{\mathbf{u}}_{\mathbf{j|i}}; \hat{\mathbf{u}}_{\mathbf{j|i}} = \mathbf{W^s_{ij}} \mathbf{u_i}$$

where, $\mathbf{c_{ij}}$ is the coupling coefficient between capsule i of layer l to capsule j of layer (l+1) and are determined by iterative dynamic routing, $\mathbf{W^s}$ is the shared weight matrix between the layers l and l+1. The routing process can be interpreted as computing soft attention between lower and higher level capsules.

Convolutional Capsule Layer: Similar to (Sabour et al., 2017) and (Yang et al., 2018), the capsules in this layer are connected to lower level capsules. The connection strengths are calculated by multiplying the input with a transformation matrix followed by the routing algorithm. The candidate parent capsule $\hat{\mathbf{u}}_{\mathbf{j|i}}$ is computed by $\hat{\mathbf{u}}_{\mathbf{j|i}} = \mathbf{W^s_{ij}} \mathbf{u_i}$ where, $\mathbf{u_i}$ is the child capsule and $\mathbf{W^s}$ is shared weight between capsule i and j. The coupling strength between the child-parent capsule is determined by the routing algorithm to produce the parent feature map in r iterative rounds by $\mathbf{c_{ij}} = \frac{\exp(\mathbf{b_{ij}})}{\sum_k exp(\mathbf{b_{ik}})}$. Logits $\mathbf{b_{ij}}$ which are initially same, determines how strongly the capsules j should be coupled with capsule i. The capsules are then flattened out into a single layer and then multiplied by a transformation matrix $\mathbf{W^{FC}}$ followed by routing algorithm to compute the final sentence representation $(\mathbf{s_k})$. The sentence representation is finally passed through the softmax layer to calculate the class probabilities.

4 Datasets

4.1 Kaggle Toxic Comment Classification

In 2018, Kaggle hosted a competition named Toxic Comment Classification[1]. The dataset is made of Wikipedia talk page comments and is contributed by Conversation AI. Each comment has a multi-class label, and there are a total of 6 classes, namely, toxic, severe toxic, obscene, threat, insult and identity hate. We split the data (159571 sentences) into training (90%), validation (10%) and 153164 test sentences.

[1]https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/data

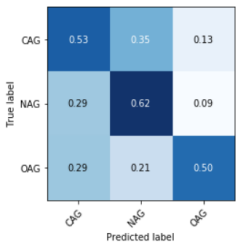

Figure 3: Confusion matrix for TRAC dataset

4.2 TRAC dataset

It is a dataset for Aggression identification[2], and contains 15,000 comments in both Hindi and English. The task is to classify the comments into the following categories, Overtly Aggressive (OAG), Covertly Aggressive (CAG), and Non-aggressive (NAG). We used the train, dev and test data as provided by the organizers of the task.

5 Experiments

As a preprocessing step, we performed case-folding of all the words and removal of punctuations. The code for tokenization was taken from (Devlin et al., 2018) which seems to properly separate the word tokens and special characters.

For training all our classification models, we have used fastText embeddings of dimension 300 trained on a common crawl. For out of vocabulary (OOV) words we initialized the embeddings randomly. For feature extraction, we used 200 LSTM units, each for capturing forward and backward contexts (total of 400). We used 20 capsules of dimension 15 and another 20 of dimension 20 for all the experiments. We kept the number of routings to be 3 as more routings could introduce overfitting. To further avoid overfitting, we adjusted the dropout values to 0.4. We used cross-entropy as the loss function and Adam as an optimizer (with default values) for all the models. We obtained all these hyperparameters values by tuning several models on the validation set and then finally selecting the model with minimum validation loss.

6 Results and Analysis

We have reported the results on a total of 3 datasets, two of which belong to TRAC-1 dataset. Our evaluation metric for TRAC-1 is F1 score, while for Kaggle dataset is ROC-AUC. We performed better for all the datasets except for TRAC Twitter data, in which our model could not beat the previous Capsule Network. We have used very

[2]https://sites.google.com/view/trac1/shared-task

Model	Kaggle Toxic Comment Classification (ROC-AUC)	TRAC Twitter English (F1-Score)	TRAC Facebook English (F1-Score)
Vanilla CNN	96.615	53.006	58.44
Bi-LSTM	97.357	54.147	61.223
Attention Networks (Raffel and Ellis, 2015)	97.425	55.67	62.404
Hierarchical CNN (Conneau et al., 2017)	97.952	53.169	58.942
Bi-LSTM with Maxpool (Lai et al., 2015)	98.209	53.391	62.02
Bi-LSTM and Logistic Regression	98.011	53.722	61.478
Pretrained LSTMs (Dai and Le, 2015)	98.05	53.166	62.9
CNN-Capsule (Yang et al., 2018)	97.888	54.82	60.09
LSTM-Capsule (Srivastava et al., 2018)	98.21	**58.6**	62.032
Our Model	**98.464**	57.953	**63.532**

Table 1: Results Of various architectures on publicly available datasets

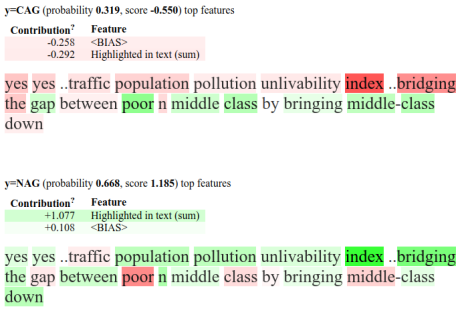

Figure 4: CAG comment predicted as NAG comment

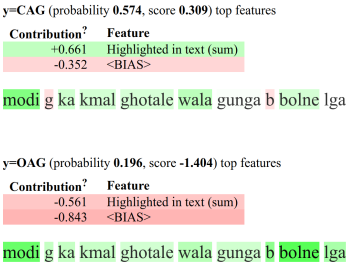

Figure 5: OAG comment predicted as CAG comment

strong and some recent baseline algorithms for comparing our results. We shall now analyze examples for which our model is making mistakes, we will pick samples from TRAC Facebook English dataset. For analysis, we use LIME (Ribeiro et al., 2016), which performs some perturbations on the input data to understand the relationship between input and the output data. It uses a local interpretable model to approximate the model in question and tries to create certain *explanations* of input data.

From the confusion matrix, we can observe that the model gets most confused by predicting CAG comments as NAG. This can be because the words used in the sentence might not sound aggressive and the model labels them as neutral sentences. However, in reality, the sentence as a whole is a sarcastic one. For example, refer to Fig 4 which goes wrong because the words it is focussing on, are all neutral words, but when combined, it is sarcasm on bridging the gap the between the poor and the middle class.

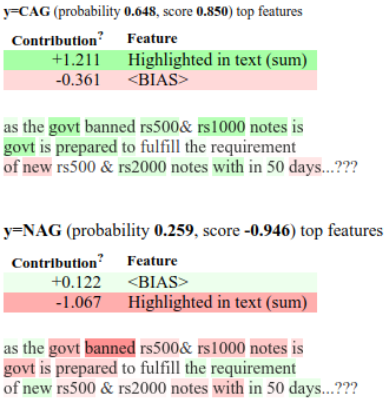

Figure 6: NAG comment predicted as OAG comment

Secondly, the model is also incorrectly predicting NAG and OAG comments as CAG equally, this is because there are certain comments against the government which are mostly present in CAG class. Refer to Fig 6 and Fig 4, in these comments, the government or some government official is being criticized, the attack is not directly pointed and there is hidden aggression.

7 Conclusion and Future Work

We reported our results on several obvious state-of-the-art deep learning architectures and reported better results on Capsule network. We also analyzed some misclassifications made by the model and tried to reason them as well using heatmap of the weights obtained from the model. For future work, as mentioned in (Sabour et al., 2017), there can be several methods to train capsules hence, we would like to explore these methods. We also want to try different loss functions like spread loss, focal loss and margin loss. We would also like to explore competency of capsules on different NLP tasks and explore their working using different investigation techniques seen in (Yang et al., 2018).

References

Betty van Aken, Julian Risch, Ralf Krestel, and Alexander Löser. Challenges for toxic comment classification: An in-depth error analysis. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*.

Pete Burnap and Matthew L. Williams. 2015. Cyber hate speech on twitter: An application of machine classification and statistical modeling for policy and decision making: Machine classification of cyber hate speech.

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.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*.

Maral Dadvar, Dolf Trieschnigg, Roeland Ordelman, and Franciska de Jong. 2013. Improving cyberbullying detection with user context. In *Proceedings of the 35th European Conference on Advances in Information Retrieval*, ECIR.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In *Advances in Neural Information Processing Systems*, pages 3079–3087.

Thomas Davidson, Dana Warmsley, Michael W. Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *ICWSM*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Karthik Dinakar, Roi Reichart, and Henry Lieberman. 2011. Modeling the detection of textual cyberbullying. In *fifth international AAAI conference on weblogs and social media*.

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*. 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*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Yoon Kim. 2014a. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar. Association for Computational Linguistics.

Yoon Kim. 2014b. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Ritesh Kumar, Guggilla Bhanodai, Rajendra Pamula, and Maheshwar Reddy Chennuru. 2018. Trac-1 shared task on aggression identification: Iit(ism)@coling'18. In *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Irene Kwok and Yuzhou Wang. 2013. Locate the hate: Detecting tweets against blacks. In *Twenty-seventh AAAI conference on artificial intelligence*.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *AAAI*, volume 333, pages 2267–2273.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Nishant Nikhil, Ramit Pahwa, Mehul Kumar Nirala, and Rohan Khilnani. 2018. Lstms with attention for aggression detection. In *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*.

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

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Georgios K. Pitsilis, Heri Ramampiaro, and Helge Langseth. 2018. Detecting offensive language in tweets using deep learning.

Rodmonga Potapova and Denis Gordeev. 2016. Detecting state of aggression in sentences using CNN.

Colin Raffel and Daniel PW Ellis. 2015. Feed-forward networks with attention can solve some long-term memory problems. *arXiv preprint arXiv:1512.08756*.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144. ACM.

Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. 2017. Dynamic routing between capsules. In *Advances in Neural Information Processing Systems 30*, pages 3856–3866.

Alexandra Schofield and Thomas Davidson. 2017. Identifying hate speech in social media.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Saurabh Srivastava, Prerna Khurana, and Vartika Tewari. 2018. Identifying aggression and toxicity in comments using capsule network. In *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, Santa Fe, New Mexico, USA. 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.

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex machina: Personal attacks seen at scale. WWW.

Jun-Ming Xu, Kwang-Sung Jun, Xiaojin Zhu, and Amy Bellmore. 2012. 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*, NAACL HLT.

Min Yang, Wei Zhao, Jianbo Ye, Zeyang Lei, Zhou Zhao, and Soufei Zhang. 2018. Investigating capsule networks with dynamic routing for text classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium. Association for Computational Linguistics.

Ziqi Zhang, D Robinson, and Jonathan Tepper. 2018. Detecting hate speech on twitter using a convolution-gru based deep neural network.