Soft Computing manuscript No.
(will be inserted by the editor)

Bilingual Cyber-aggression Detection on Social Media using LSTM Autoencoder

Kirti Kumari · Jyoti Prakash Singh · Yogesh Kumar Dwivedi · Nripendra Pratap Rana

Received: date / Accepted: date

Abstract Cyber-aggression is an offensive behaviour attacking people based on race, ethnicity, religion, gender, sexual orientation, and other traits. It has become a major issue plaguing the online social media. In this research, we have developed a deep learning-based model to identify different levels of aggression (direct, indirect and no aggression) in a social media post in a bilingual scenario. The model is an autoencoder built using the LSTM network and trained with non-aggressive comments only. Any aggressive comment (direct or indirect) will be regarded as an anomaly to the system and will be marked as Overtly (direct) or Covertly (indirect) aggressive comment depending on the reconstruction loss by the autoencoder. The validation results on the dataset from two popular social media sites: Facebook and Twitter with bilingual (English and Hindi) data outperformed the current state-of-the-art models with improvements of more than 11% on the test sets of the English dataset and more than 6% on the test sets of the Hindi dataset.

Keywords Online Social Networks · Cyber-aggression · Cyberbullying · Autoencoder · Long Short Term Memory

Kirti Kumari
National Institute of Technology Patna, Patna, India
E-mail: kirti.cse15@nitp.ac.in

Jyoti Prakash Singh
National Institute of Technology Patna, Patna, India
E-mail: jps@nitp.ac.in

Yogesh Kumar Dwivedi
School of Management, Swansea University, Swansea, UK
E-mail: ykdwivedi@gmail.com

Nripendra Pratap Rana
School of Management, University of Bradford, Bradford, UK
E-mail: nrananp@gmail.com

1 Introduction

Online Social Network (OSN) is a platform that allows individuals to present themselves, articulate their social status and establish or maintain connections with others with the help of the Internet. Facebook[1], Twitter[2] and Instagram[3]; are popular examples of OSN used by millions of users to share their ideas, convey daily greetings and chat with friends. OSN has numerous positive usages like getting help from other online friends, gaining knowledge, entertainment and many more. On the other hand, these platforms have some negative usages such as impersonation, hate speech (Kumari and Singh, 2019), cyber-aggression (Chatzakou et al., 2017) and cyberbullying (Kumari et al., 2019b; Kumari and Singh, 2020). Cyber-aggression is an aggressive online behaviour using electronic media to harass others, which have become a serious problem in society. It is usually done by writing offensive comments or by sharing embarrassing images and/or videos.

The victim[4] of Cyber-aggression are found to show low self-esteem, depression, hopelessness, frustration, anxiety and sleeping-related issues (Modecki et al., 2013). Given the harmful effects of Cyber-aggression on its victims and their subsequent development among Internet users (especially on kids and adolescents), it is essential to identify such behaviour as early as possible to prohibit it from ramping up to a critical level. The nature of the problem requires urgent technological attention as manual identification and moderation of post is not feasible due to its huge volume. This motivated us to develop a system that can identify whether a social media post is directly aggressive, indirectly aggressive or non-aggressive.

Most of the earlier researchers (Aroyehun and Gelbukh, 2018; Arroyo-Fernández et al., 2018; Chatzakou et al., 2017; Modha et al., 2018; Raiyani et al., 2018; Samghabadi et al., 2018) applied supervised machine and deep learning algorithms to classify the social media comments into different classes of aggression. However, they achieved very limited success on bilingual and cross-domain social media posts. An unsupervised approach was taken by (Zhao and Mao, 2017) using denoising autoencoder to detect cyberbullying cases. They trained their autoencoder with bullying posts only. One of the limitations of their model was that a different combination of bullying words, not present during the training of the encoder, may be predicted as a non-bullying post. A K-competitive autoencoder based approach was proposed by (Maitra and Sarkhel, 2018) to detect aggressive comments. They, however, achieved very limited results for bilingual Hindi and English datasets.

Motivated by the need of identifying Cyber-aggressive comments and earlier results, we proposed an LSTM autoencoder based approach solve to this problem. We developed an LSTM autoencoder followed by a decision tree classifier to classify the comments among Non-aggressive, Covertly Aggressive and Overtly Aggressive class. Our proposed work differs from the earlier autoencoder based approaches (Maitra and Sarkhel, 2018; Zhao and Mao, 2017) as we used Long Short Term Memory (LSTM) network to build our autoencoder to process the text comments. The proposed LSTM autoencoder was trained with normal (non-

[1] www.facebook.com

[2] www.twitter.com

[3] www.instagram.com

[4] The person who intentionally tried to harm others through electronic media is called Bully or Perpetrator, while the targeted one by the Bully is called Victim.

aggressive) comments only without using the class label of the dataset. These normal comments can be easily obtained from any OSN. The LSTM autoencoder preserves the long-term dependency of the textual comments. The autoencoder yields losses of each post during testing which is used by a decision tree classifier to classify the posts into (i) Non-aggressive, (ii) Covertly Aggressive and (iii) Overtly Aggressive classes. The current model is trained and tested with bilingual text corpora and is found to perform better than the existing methods.

The major contributions of our research can be listed as follows:

- Developed an LSTM-based autoencoder to classify social media comments among Non-aggressive, Covertly Aggressive and Overtly Aggressive classes.
- Experimentally, it is found that only one autoencoder trained with normal comments is sufficient to do the classification.
- Experimentally, it is also found that Randomly-initialized embedding is performing better than pre-trained embeddings.
- Decision Tree classifier is found to perform better than other classifiers with reconstruction loss of LSTM autoencoder.
- Performance of the developed model is found almost similar for English as well as for Hindi comments across Facebook and Twitter posts.

The remainder of the paper is structured as follows. Similar studies are described in Section 2. Section 3 introduces our conceptual framework for detecting Cyber-aggression. Experimental setup and evaluation results are presented in Section 4. Section 5 discusses about the results and implications of the current research. Finally, we have concluded our work with a discussion on possible future work in Section 6.

2 Related Works

In the last couple of years, the detection of hate speech has attracted a lot of attention from the research community (Badjatiya et al., 2017; Bohra et al., 2018; Burnap and Williams, 2015; Davidson et al., 2017; Nobata et al., 2016; Waseem and Hovy, 2016). The majority of the past works (Burnap and Williams, 2015; Nobata et al., 2016; Waseem and Hovy, 2016) have represented text using a bag-of-words (BoW) model to use the word and character *n-grams* as features for their experiments. A BoW model uses words as the main features and overlooks the sequential, syntactical and semantic information present in the text. Djuric et al. (2015) used a Continous Bags-of-Words (CBOW) with a Logistic Regression classifier to detect hate comments on the Yahoo Finance website. Davidson et al. (2017) found that Logistic Regression and Support Vector Machine (SVM) classifiers as better models for hate speech detection. Bohra et al. (2018) extended the earlier research of hate speech detection for code mixed tweets of Hindi and English. Chatzakou et al. (2017) utilised the user and network-based features along with the bag of words features from the text to identify bullying, aggressive, spam and normal tweets. They found that a combination of user, network and tweet-text-based features gave better accuracy with overall recall and precision of 0.73 and 0.72, respectively.

The success of neural networks in other domains such as brain signal classification, nonlinear system identification and credit card fraud detection (Giap

et al., 2018; Fiore et al., 2019; Qiu et al., 2016, 2017, 2019; Rubio et al., 2019; Rubio de Jesús J., 2009; Rubio et al., 2018) have attracted researchers to use them for hate speech and Cyber-aggression detection also. Mehdad and Tetreault (2016) proposed a Recurrent Neural Network (RNN)-based model along with the machine-learning classifiers for hate speech detection and reported that character *n-gram* features gave better accuracy than word *n-gram* features. An autoencoder based model was proposed by Zhao and Mao (2017) to detect bullying instances on MySpace and Twitter. The model learned the bullying patterns by training the autoencoder with bullying comments. They achieved a weighted F1-score of 0.72 and 0.77 with Twitter and Myspace datasets respectively. Gambäck and Sikdar (2017) developed a Convolutional Neural Network (CNN) to detect hate speech and reported an F1-score of 0.78. Chen et al. (2018) utilised CNN along with sentiment analysis to detect aggressive tweets. Badjatiya et al. (2017) compared the deep-learning approach using Long Short Term Memory (LSTM) and Gradient Boosted Decision Trees (GBDT) for hate speech detection and reported that deep-learning-based approach was giving a better accuracy compared to traditional machine-learning approaches. Malte and Ratadiya (2019) used bi-directional transformer architecture for a bilingual and cross-domain dataset. They, however, reported only marginal improvements over earlier results.

Considering the detrimental effects of online aggression, the organisers of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-1) workshop launched a shared task to get some real-life solutions for Cyber-aggression. They released bilingual datasets (Kumar et al., 2018b) containing Non-aggressive, Covertly aggressive and Overtly aggressive comments in English and Hindi from Facebook and Twitter to train and test the proposed models. The shared task collected 18 system description papers and 5 regular papers using these datasets. In the following paragraph, we briefly describe some of these studies having better results.

A group of participants (Arroyo-Fernández et al., 2018; Orasan, 2018; Samghabadi et al., 2018) used machine learning-based models with manually extracted features. In these models, SVM, Logistic Regression and Random Forest classifiers were mainly used. Most of the other participants (Aroyehun and Gelbukh, 2018; Galery et al., 2018; Kumar et al., 2018a; Madisetty and Desarkar, 2018; Modha et al., 2018; Nikhil et al., 2018; Orabi et al., 2018; Ramiandrisoa and Mothe, 2018; Risch and Krestel, 2018; Srivastava et al., 2018) experimented with various deep learning models such as LSTM, CNN, Bi-directional LSTM, Gated Recurrent Neural Network (GRU). Few other participants (Raiyani et al., 2018; Tommasel et al., 2018) used hybrid models of deep learning and machine learning. Raiyani et al. (2018) compared dense neural network and FastText classification model to find that simple three-layers dense neural network was performing better than the other models. Tommasel et al. (2018) used combination of neural network model and SVM classifier with GloVe, sentiment, *n-gram* and tf-idf (Term Frequency-Inverse Document Frequency) features. Maitra and Sarkhel (2018) used unsupervised technique using K-Competitive autoencoder to identify aggressive comments. Their model, however, was not predicting all the three classes effectively. The best-reported results in terms of weighted F1-score on the TRAC-1 datasets were in the range of 0.50 - 0.66 only requiring the more advanced model to improve on these results.

A bulk of the earlier works in aggression and hate speech detection have been developed and tested only for the English language and a specific social media platform. A few works have tried to address the issues of multi-linguality and

Table 1 Description of bilingual Cyber-aggression datasets

Corpus		Number of comments			
		Training data	Validation data	Testing data	
				Facebook	Twitter
English	OAG	2708	711	144	361
	CAG	4240	1057	142	413
	NAG	5051	1233	630	483
	Total	11,999	3001	916	1257
Hindi	OAG	4856	1217	362	459
	CAG	4869	1246	413	381
	NAG	2275	538	195	354
	Total	12,000	3001	970	1194

style variations of different platforms. In the current work, we have tried to address these issues by developing an LSTM autoencoder followed by a machine learning classifier to classify the social media comments into different classes of aggression.

3 Methodology

To detect Cyber-aggression automatically in Online Social Network (OSN), we propose a Long Short Term Memory autoencoder (LSTM autoencoder) followed by a machine-learning classifier. In the following Section 3.1, we have described the details of the datasets and the pre-processing done on these. The LSTM autoencoder based model is described next in Section 3.2.

3.1 Data Description and Pre-processing

The datasets used in the current research is taken from the released datasets (Kumar et al., 2018b) by the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-1) at COLING-2018. The datasets were created to develop a system to classify textual comments of social media among Overtly Aggressive (OAG), Covertly Aggressive (CAG) and Non-aggressive (NAG) classes. OAG comments are the comments which contain direct aggression, CAG comments are the comments which contain an indirect expression of aggression, and NAG comments are those which do not contain any direct or indirect aggression. The datasets collected from Facebook and Twitter were bilingual containing text comments in English and Hindi. Hindi text comment is written both in Devanagari as well as in the Roman script. The description of the datasets is presented in Table 1.

As a preprocessing step, we replaced informal abbreviations to the standard word for English text, emoji to symbolic word i.e. smile emoji was replaced with em_smile, similarly, sad emoji was replaced with em_sad and so on. As the final step of preprocessing, we converted Devanagari script of Hindi comments into Roman script using transliteration technique.

3.2 Proposed Model

The proposed model is an autoencoder built using Long Short Term Memory (LSTM) nodes. A schematic diagram of the proposed model is shown in Figure 1.

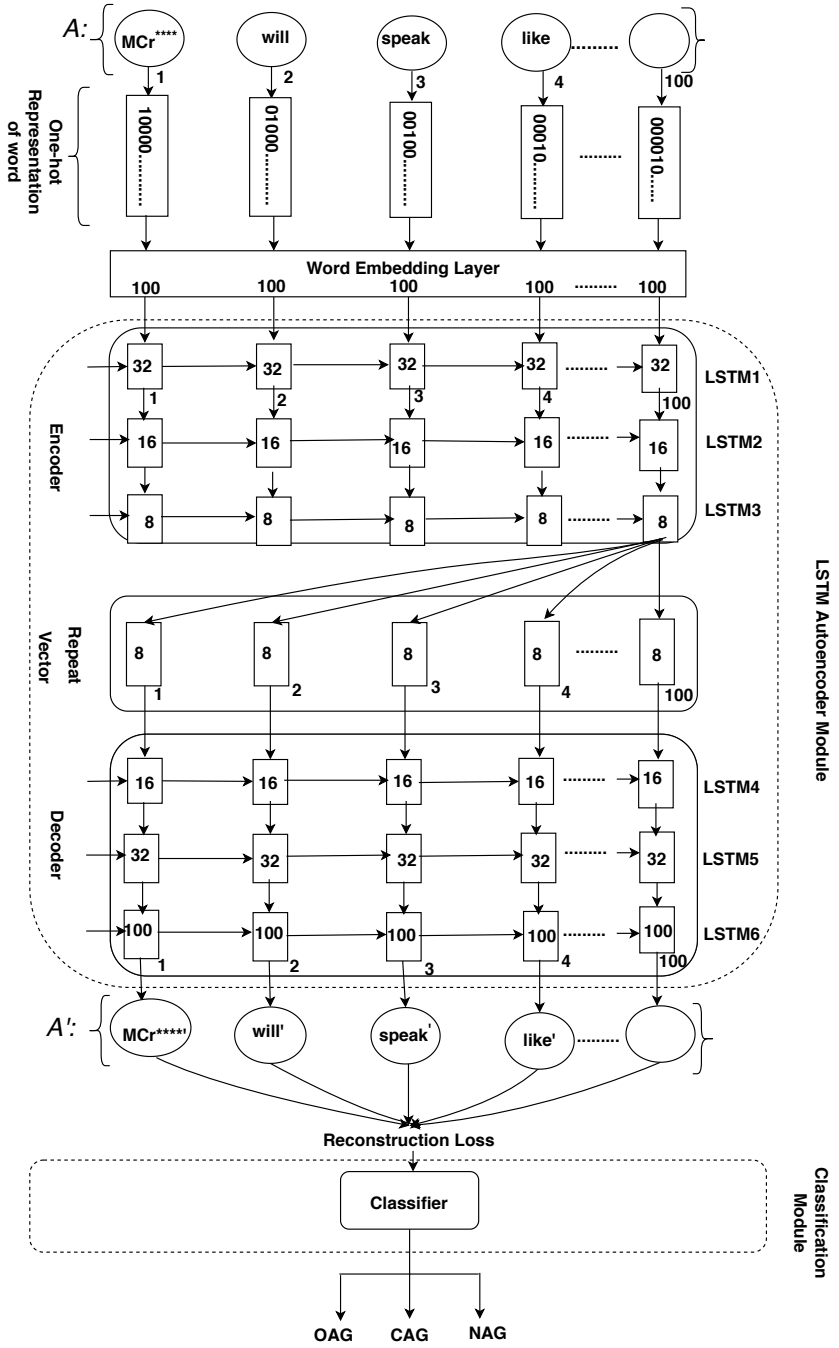

Fig. 1 Overview of the proposed stacked LSTM autoencoder architecture

Table 2 Description of layer-wise dimension of LSTM autoencoder

Layer number	Layer name	Input dimension	Time step	Output dimension
1	Input layer	100	100	-
2	LSTM Layer1	100	100	32
3	LSTM Layer2	32	100	16
4	LSTM Layer3	16	1	8
5	Repeat Vector	8	100	8
6	LSTM Layer4	8	100	16
7	LSTM Layer5	16	100	32
8	LSTM Layer6	32	100	100

The upper part of the diagram represents the input and word embedding layers. The module below word embedding is the LSTM autoencoder module. It consists of three major units: (i) Encoder, (ii) Repeat Vector and (iii) Decoder. The Encoder submodule consists of three LSTM layers, named LSTM1, LSTM2 and LSTM3 for the first, second and third layers, respectively. The Decoder submodule also consists of three LSTM layers denoted by LSTM4, LSTM5 and LSTM6, for the fourth, fifth and sixth layers respectively. The Decoder and the Encoder submodules are identical. The repeat vector is used to replicate the encoder output so that the same representation can be fed to each LSTM node of the decoder layer. We fixed the maximum length of any comment to 100 words. The input dimension to the embedding layer was set to the vocabulary size of the dataset and the output dimension was set to 100. The embedded word vector with 100 dimensions at the input layer was reduced to 32 dimensions in the LSTM1, 16 dimensions in the LSTM2 and 8 dimensions in the LSTM3. In the decoder phase, the 8-dimensional vector was given as input to the LSTM4, which was expanded to a 16-dimensional vector. It was further expanded into 32 and 100 dimensions by LSTM5 and LSTM6, respectively. The output of the LSTM6 was used to calculate the loss between the input (A) and output(A'). The input and output dimensions used at each layer of LSTM encoder and decoder are presented in Table 2. A detailed description of the stacking of LSTM can be seen in Pascanu et al. (2014).

We used an autoencoder network that consists of mainly two parts called the encoder and the decoder. The used functions are represented as follows:

$$Encoder function(\psi) : A \rightarrow S \tag{1}$$

$$Decoder function(\chi) : S \rightarrow A \tag{2}$$

The *Encoder function* (ψ) maps the original data A to the latent space S. The *Decoder function* (χ) maps the data from the latent space S to the output A. In this case, the output is the same as the input A. So, we are trying to recreate the original text after some generalised non-linear compression.

The encoding network is represented by the following function of the standard neural network passed through a sigmoid activation function.

$$k = \sigma(Ua + B) \tag{3}$$

Here, k, σ, U and B stand for the latent dimension, a non-linear activation function called Sigmoid activation function, the weight matrix and the bias. Similarly, the decoding network can be represented, but with different weight (U') and bias (B'). The equation is represented as follows:

$$a = \sigma(U'k + B') \tag{4}$$

We used *MSE* (Mean Square Error) as the loss function to train the neural network which can be represented in equation 5. The autoencoder minimizes reconstruction losses between the input A $(a_1, a_2, ..., a_n)$ and the output A' $(a'_1, a'_2, ..., a'_n)$ where n is the number of sequence in the input A.

$$MSE = \frac{1}{n}\sum_{i=1}^{n} ||a_i - a'_i||^2 \tag{5}$$

As the model is an autoencoder, its input and output are the same during training. The model encodes the input patterns to recreate it later on. Since the input is text, the LSTM model encodes the semantic information of the input text in the autoencoder. The output from the autoencoder is used to calculate the reconstruction loss. During the training process, this loss is reduced by updating the weights of the LSTM network through back-propagation. During validation and testing, this loss is used to classify the input into different classes. If the input is similar to the learned pattern, the loss will be minimal. On the contrary, if the input pattern is different from the learned pattern, the loss will be on the higher side. We applied Support Vector Machine, Logistic Regression, Naive Bayes and Decision Tree classifiers on these losses to classify the original comment in different categories of aggressive classes present in the datasets.

4 Results

This section describes the experimental setup, input data preparation, hyperparameter settings and obtained results. The experimental settings and results are discussed, respectively, in subsections 4.1 and 4.2.

4.1 Experimental Settings

For experimental purposes, the text comments were represented as embedded vectors using Randomly-initialized matrix and pre-trained matrices such as GloVe and FastText. The maximum length of each comment was fixed to 100 words. The comments having less than 100 words were pre-padded with zeros and longer comments were truncated after 100 words. As a first step in the experiment, we evaluated the performance of the three: pre-trained GloVe, pre-trained FastText and Randomly-initialized embeddings. The performance of the classifier was very poor for pre-trained embedding techniques, GloVe and FastText. Hence, for further experiments, we used a one-hot representation of text with Randomly-initialized embedding. The autoencoder with LSTM network is built using Keras[5] library with Tensorflow as back end. The LSTM autoencoder model was trained for 100 epochs with a batch size of 100, Adam as the optimizer function and the Mean Square Error (*MSE*) as the loss function.

[5] https://keras.io/

The proposed LSTM autoencoder was trained with non-labelled comments. Once the LSTM autoencoder was trained, the reconstruction losses were calculated for all comments of the datasets. The reconstruction losses of the comments were used to classify the comments into the Non-aggressive (NAG), Covertly Aggressive (CAG) and Overtly Aggressive (OAG) classes using these four different machine-learning classifiers: (i) Support Vector Machine (SVM), (ii) Logistic Regression (LR), (iii) Naive Bayes (NB), and (iv) Decision Tree (DT). The classifiers were implemented using scikit-learn tool[6] in Python.

4.2 Experimental Results

The results of the current research are presented in the following four subsections. Our first result is about the selection of suitable number of Autoencoders and proper data-subset to train the autoencoder. The second part of the results is about the selection of proper embedding technique for comments' representation. The third part of the result deals with the selection of proper classifier. The last result is the classification result when the problem is viewed as a two class problem, merging CAG and OAG comments into one class.

4.2.1 Evaluating the number of autoencoder and training class of autoencoder

One of the primary design decisions was to decide the number of autoencoders for the proper classification of the comments. We started with three LSTM autoencoders by training each with a unique class (NAG, CAG and OAG) text comments without labels. Every autoencoder was then tested with complete testing data. The obtained result is shown in Table 3. As can be seen in Table 3, when the autoencoder was trained with NAG comments, the classifier was able to classify the comments among all the three classes with a good performance measure. But when an autoencoder was trained with the CAG class dataset, the classifier was unable to predict anything for the class NAG and OAG. A similar case happened with the autoencoder trained with the OAG class dataset. The prediction performance of the CAG class was very poor, especially for other domains. These results suggest that only one autoencoder trained with NAG comments is more suitable to classify the comments among the three classes. Hence, for our further experiments, we have used only one autoencoder trained with NAG comments.

4.2.2 Evaluating the proper embedding

Our second design issue was to find out the best embedding technique for the said task. We evaluated three popular embedding techniques: Randomly-initialized embedding, GloVe embedding (Pennington et al., 2014) and FastText embedding (Joulin et al., 2016) for English dataset. However, for the Hindi dataset, the experiment was done with the Randomly-initialised embedding and FastText embedding because GloVe embedding is not available for Hindi text. The distribution of autoencoder reconstruction loss for validation datasets with Randomly-initialised, GloVe and FastText embeddings are shown in Figures 2, 3 and 4, respectively.

[6] https://scikit-learn.org/stable/

Table 3 Classification result of three LSTM Autoencoders with their training class

Corpus	Training class of autoencoder	Comment class	Results					
			Facebook test set			Twitter test set		
			Precision	Recall	F1-score	Precision	Recall	F1-score
English	NAG	NAG	0.89	0.86	**0.87**	0.75	0.71	**0.73**
		CAG	0.85	0.62	**0.72**	0.76	0.66	**0.70**
		OAG	0.56	0.78	**0.65**	0.61	0.76	**0.68**
		Weighted average	0.83	0.81	**0.81**	0.72	0.71	**0.71**
	CAG	NAG	0.00	0.00	0.00	0.00	0.00	0.00
		CAG	0.16	1.00	0.27	0.33	1.00	0.49
		OAG	0.00	0.00	0.00	0.00	0.00	0.00
		Weighted average	0.02	0.16	0.04	0.11	0.33	0.16
	OAG	NAG	0.65	0.44	0.52	0.59	0.10	0.16
		CAG	0.15	0.30	0.20	0.00	0.00	0.00
		OAG	0.13	0.19	0.15	0.30	0.97	0.46
		Weighted average	0.49	0.38	0.48	0.31	0.32	0.19
Hindi	NAG	NAG	0.59	0.68	0.63	1.00	1.00	1.00
		CAG	0.75	0.94	0.84	0.26	0.04	0.08
		OAG	0.81	0.52	0.64	0.53	0.90	0.67
		Weighted average	0.74	0.73	0.72	0.58	0.65	0.58
	CAG	NAG	0.00	0.00	0.00	0.24	0.00	0.09
		CAG	0.53	0.93	0.67	0.32	0.89	0.47
		OAG	0.40	0.05	0.09	0.52	0.07	0.13
		Weighted average	0.37	0.41	0.32	0.37	0.33	0.23
	OAG	NAG	0.17	0.33	0.22	0.31	0.54	0.40
		CAG	0.40	0.00	0.01	0.00	0.00	0.00
		OAG	0.37	0.61	0.46	0.38	0.48	0.42
		Weighted average	0.34	0.29	0.22	0.24	0.34	0.28

Table 4 Statistics of normalized LSTM autoencoder reconstruction loss

Corpus	Embedding	Comment class	Normalized reconstruction loss		
			mean	var	std
English	Randomly-initialized	NAG	**1.86**	2.12	1.46
		CAG	**23.89**	1.38	1.17
		OAG	**28.83**	1.59	1.26
	GloVe	NAG	0.31	4.89	2.21
		CAG	0.69	13.60	3.69
		OAG	0.83	14.74	3.84
	FastText	NAG	0.31	3.07	1.75
		CAG	0.24	1.81	1.35
		OAG	0.31	2.31	1.52
Hindi	Randomly-initialized	NAG	2.05	4.29	2.07
		CAG	27.27	0.43	0.66
		OAG	25.62	0.05	0.22
	FastText	NAG	32.54	2024.49	44.99
		CAG	32.93	1252.18	35.38
		OAG	33.14	1383.55	37.19

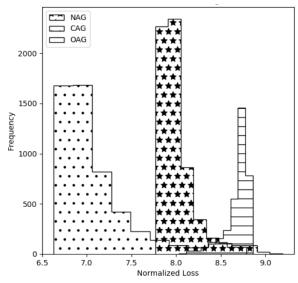

Fig. 2 Loss distribution of LSTM autoencoder in Randomly-initialized embedding

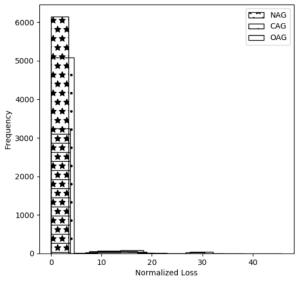

Fig. 3 Loss distribution of LSTM autoencoder in GloVe embedding

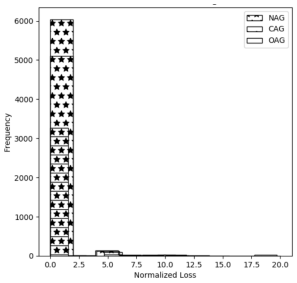

Fig. 4 Loss distribution of LSTM autoencoder in FastText embedding

The statistics of the losses are presented in Table 4. It can be seen from these figures that the losses of different comments using Randomly-initialised embedding are easily distinguishable into three groups (highlighted in bold in Table 4). However, the losses of different comments are mostly overlapped for GloVe and FastText embeddings which can be seen from Figures 3 and 4, respectively. The statistics such as mean, variance (var) and standard deviation (std) of the losses of different comments for all the embedding techniques are shown in Table 4. As we can observe from Table 4, the difference between the means of losses is large for any two classes in the case of Randomly-initialised embedding compared to pre-trained GloVe and FastText embeddings.

4.2.3 Evaluating the best classifier

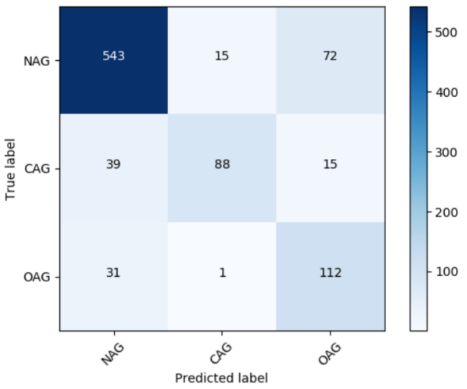

Fig. 5 Confusion matrix for the English Facebook test set

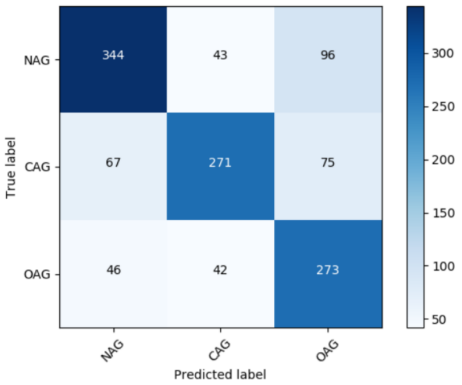

Fig. 6 Confusion matrix for the English Twitter test set

The last design issue was to select a classifier that can classify the obtained losses into different classes with maximum accuracy. For this, we evaluated the best classifier for the problem with the obtained losses as input. We evaluated four

Table 5 Results of our proposed classification model on English and Hindi corpus

Corpus	Classifier	Comment Class	Results					
			Facebook test set			Twitter test set		
			Precision	Recall	F1-score	Precision	Recall	F1-score
English	**DT**	NAG	0.89	0.86	*0.87*	0.75	0.71	*0.73*
		CAG	0.85	0.62	*0.72*	0.76	0.66	*0.70*
		OAG	0.56	0.78	*0.65*	0.61	0.76	*0.68*
		Weighted average	**0.83**	**0.81**	**0.81**	**0.72**	**0.71**	**0.71**
	NB	NAG	0.68	0.80	0.73	0.40	0.00	0.01
		CAG	0.00	0.00	0.00	0.00	0.00	0.00
		OAG	0.26	0.31	0.29	0.29	0.99	0.44
		Weighted average	0.51	0.60	0.55	0.24	0.29	0.13
	LR	NAG	0.08	0.02	0.03	0.00	0.00	0.00
		CAG	0.00	0.00	0.00	0.00	0.00	0.00
		OAG	0.19	0.99	0.32	0.29	0.99	0.44
		Weighted average	0.08	0.17	0.07	0.08	0.29	0.13
	SVM	NAG	0.68	0.87	0.76	0.31	0.14	0.19
		CAG	0.00	0.00	0.00	0.00	0.00	0.00
		OAG	0.22	0.17	0.56	0.29	0.84	0.43
		Weighted average	0.50	0.62	0.53	0.20	0.30	0.20
Hindi	**DT**	NAG	0.59	0.68	*0.63*	1.00	1.00	*1.00*
		CAG	0.75	0.94	*0.84*	0.26	0.04	*0.08*
		OAG	0.81	0.52	*0.64*	0.53	0.90	*0.67*
		Weighted average	**0.74**	**0.73**	**0.72**	**0.58**	**0.65**	**0.58**
	NB	NAG	0.19	0.44	0.26	1.00	1.00	1.00
		CAG	0.78	0.97	0.87	0.45	1.00	0.62
		OAG	0.00	0.00	0.00	0.00	0.00	0.00
		Weighted average	0.37	0.50	0.42	0.44	0.62	0.50
	LR	NAG	0.00	0.00	0.00	1.00	1.00	1.00
		CAG	0.43	1.00	0.61	0.45	1.00	0.62
		OAG	0.92	0.03	0.06	0.00	0.00	0.00
		Weighted average	0.53	0.44	0.28	0.44	0.62	0.50
	SVM	NAG	0.34	1.00	0.50	1.00	1.00	1.00
		CAG	1.00	0.93	0.97	0.45	1.00	0.62
		OAG	0.00	0.00	0.00	0.00	0.00	0.00
		Weighted average	0.49	0.60	0.51	0.44	0.62	0.50

Table 6 Results of classification model on non-aggressive versus aggressive class

Corpus	Comment class	Result					
		Facebook test set			Twitter test set		
		Precision	Recall	F1-score	Precision	Recall	F1-score
English	NAG	0.93	0.97	0.95	0.87	0.88	0.87
	AG	0.92	0.84	0.88	0.92	0.92	0.92
	Weighted average	0.93	0.93	**0.93**	0.90	0.90	**0.90**
Hindi	NAG	0.61	0.88	0.72	0.43	0.68	0.53
	AG	0.97	0.86	0.91	0.91	0.77	0.84
	Weighted average	0.90	0.86	**0.87**	0.81	0.76	**0.77**

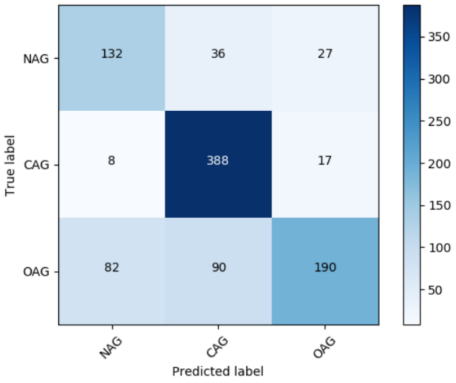

Fig. 7 Confusion matrix for the Hindi Facebook test set

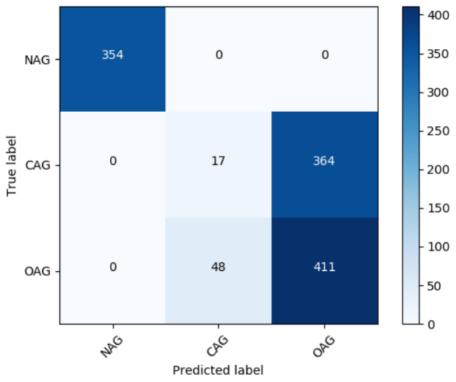

Fig. 8 Confusion matrix for the Hindi Twitter test set

different classifiers: (i) Decision Tree (DT), (ii) Naive Bayes (NB), (iii) Logistic Regression (LR) and (iv) Support Vector Machine (SVM)) listed in Table 5. It was found that the Decision Tree classifier is performing best among the evaluated classifiers. For the English Facebook test set, the proposed model with Decision Tree classifier achieved an F1-score of 87% for NAG comments, 72% for CAG comments and 65% for OAG comments as shown in italic in Table 5. For the English Twitter test set, the Decision Tree classifier achieved an F1-score 73% for NAG comments, 70% for CAG comments and 68% of OAG comments presented in italic in Table 5. For the Hindi Facebook test set, the Decision Tree classifier yielded an F1-score of 63% for NAG, 84% for CAG and 64% for OAG class. For the Hindi Twitter test set, the Decision Tree classifier yielded an F1-score of 100% for NAG, 8% for CAG and 67% for OAG class. The best results had a weighted F1-score of 81% and 72% for English and Hindi corpus for the Facebook test sets, respectively, as shown in bold in Table 5. The best results on the Twitter test sets were a weighted F1-score of 71% and 58% for English and Hindi corpus, respectively, as listed in bold in Table 5. The confusion matrix of the English Facebook test set, English Twitter test set, Hindi Facebook test set and Hindi Twitter test set are shown in Figures 5, 6, 7 and 8, respectively.

4.2.4 Aggressive versus Non-aggressive classification

We further tested the system for the problem of identifying Aggressive (AG) versus Non-aggressive(NAG), by merging the OAG and CAG classes to one Aggressive class only and hence making it a two-class problem. The results of the two-class problem are shown in Table 6. The model achieved very good results with a weighted F1-score of 93% and 87% for English and Hindi corpus, respectively, on Facebook test sets, as shown in bold in Table 6. The result on the Twitter test sets were 90% and 77% for English and Hindi corpus, respectively, as presented in bold in Table 6.

5 Discussion

The major finding of the current research is that a single LSTM autoencoder trained with NAG comments is a better model to classify social media comments into NAG, CAG and OAG classes. Another finding is that the Randomly-initialised embedding is better than pre-trained GloVe and FastText embeddings to be used with LSTM autoencoder for the given datasets. One more finding of our research is that the Decision Tree classifier is better compared to the other classifiers to classify the comments into NAG, CAG and OAG using losses obtained from the LSTM autoencoder.

The Randomly-initialised embedding performs better than the pre-trained GloVe and FastText embedding with LSTM autoencoder because the one-hot representation with Random-initialised weights preserves the local context of comments rather than a global context. An embedded vector with GloVe and FastText returns almost similar embedded input in each case. But, the Randomly-initialised embedding results in different vectors for OAG, CAG and NAG comments as it only considers the local context of the word.

A Decision Tree classifier is found to perform better than the other classifiers (Support Vector Machine, Naive Bayes and Logistic Regression classifiers) as the input (reconstruction loss) to the classifier was a single scalar value.

The proposed model performs better for Facebook data than Twitter data because the users of Facebook are more vocal and overtly aggressive in comparison to Twitter, where the users are more subtle and covert. The other reason is that the Facebook comments, being lengthier than the Twitter comments, capture better semantic information.

The proposed model was found to differentiate the NAG with AG (combined CAG and OAG) class very well. The current model performed best for NAG comment because LSTM autoencoder was trained on these comments and could classify those comments. In most of the cases, the current model could differentiate between Non-aggressive (NAG) and Aggressive (AG) classes due to the presence of a different set of words in the comments of these classes.

The proposed model outperformed the other reported works on the same dataset (Kumar et al., 2018b). A comparative study of the approaches, features and results of these works on TRAC-1 datasets are presented in Table 7. The previous best results reported for Facebook datasets were a weighted F1-score of 64% (Aroyehun and Gelbukh, 2018) and 66% (Malte and Ratadiya, 2019) for English and Hindi corpus, respectively. The best results on the Twitter test set were 60% (Raiyani et al., 2018) for the English and 50% (Modha et al., 2018) for Hindi. Our proposed model achieved an increase of 17% and 11% with English test sets, respectively on Facebook and Twitter. We achieved an increase of 6% and 8% with Hindi test sets, respectively on Facebook and Twitter compared to the previous best results.

One of the limitations of our work is that only the text part of the post is used for the current study. The social media posts also contain a lot of images, audio, URLs and video clips having aggressive content. The other limitation of the current work is that it is only tested with comments collected from Facebook and Twitter. The comments collected from other social media sites such as Reddit, Instagram and Formspring can also be tested.

Table 7 Comparison of our work with past works of aggression detection on bilingual datasets (Kumar et al., 2018b)

Source	**Approaches**	**Features / Embedding**	**Results (F1-score)**			
			English corpus		**Hindi corpus**	
			Facebook	**Twitter**	**Facebook**	**Twitter**
Kumar et al. (2018a)	LSTM	DeepMoji embedding	0.36	0.19	x	x
Galery et al. (2018)	GRU	FastText embedding	0.53	0.44	x	x
Maitra and Sarkhel (2018)	autoencoder	Word2Vec embedding, word count, sentiment score, capitalization and hashtag analyzer features	0.57	0.34	0.42	0.31
Ramiandrisoa and Mothe (2018)	CNN, LSTM, Machine-learning	Word2Vec, emoticons and exclamation mark features	0.57	0.51	x	x
Nikhil et al. (2018)	LSTM with attention unit and Random Forest classifier	Word2Vec embedding	0.57	0.55	0.60	0.47
Orasan (2018)	SVM, Random Forest	GloVe embedding, sentiment feature	0.58	0.51	x	x
Raiyani et al. (2018)	Dense architecture	One-hot encoding	0.58	*0.60*	0.59	0.48
Tommasel et al. (2018)	Combination of neural network and SVM	GloVe, sentiment, *n-gram* TF-IDF features	0.59	0.55	x	x
Samghabadi et al. (2018)	Logistic regresssion, SVM	GloVe embedding, word *n-gram*, character *n-gram*, TF-IDF features	0.59	0.56	0.63	0.48
Orabi et al. (2018)	CNN	GloVe embedding	0.59	0.57	x	x
Madisetty and Desarkar (2018)	CNN, LSTM, Bi-LSTM, Ensemble learning	GloVe embedding	0.60	0.51	x	x
Risch and Krestel (2018)	Augmented training and Ensemble learning	Word *n-gram*, character *n-gram*, TF-IDF features	0.61	0.60	0.63	0.38
Modha et al. (2018)	LSTM, CNN	FastText embedding	0.62	0.55	0.61	*0.50*
Arroyo-Fernández et al. (2018)	Ensemble of Passive-Aggressive and SVM classifier	FastText embedding, TF-IDF feature	0.63	0.57	x	x
Srivastava et al. (2018)	CNN, LSTM	One-hot encoding	0.63	0.59	x	x
Aroyehun and Gelbukh (2018)	LSTM	FastText embedding	*0.64*	0.59	x	x
Malte and Ratadiya (2019)	Bidirectional Transformer-based BERT	Attention-based features	0.62	x	*0.66*	x
Proposed model	LSTM autoencoder and DT	Randomly-initialized embedding	**0.81**	**0.71**	**0.72**	**0.58**

6 Conclusion and Future works

Cyber-aggression is becoming a serious issue on online social networks. We have developed an LSTM autoencoder based model to classify different classes of aggressive comments. We have used one autoencoder and trained it with non-aggressive comments only. The proposed model achieved an increase of 17% and 11% on the Facebook test sets and the Twitter test sets, respectively, for the English corpus with respect to state-of-the-art results. A similar improvement of 6% and 8% on the Facebook and Twitter test sets, respectively, were achieved compared to state-of-the-art results on the Hindi corpus.

The current research can be extended by including image, audio, video and URLs. Future works may include user and network-based features in the Cyber-aggression detection system. In future, other code-mixed languages and other social media comments may also be explored for the detection of Cyber-aggression. The current system can be made completely unsupervised by replacing the classification part with suitable clustering techniques. In future, generative models may be tried to reduce the need for labelled datasets. More research is needed for code-mixed languages to convert them to a universal common language so that the existing models may be properly utilised.

Acknowledgements

The first author would like to acknowledge the Ministry of Electronics and Information Technology (MeitY), Government of India for the financial support provided to her during the research work through Visvesvaraya Ph.D. Scheme for Electronics and IT.

Compliance with Ethical Standards

Conflict of Interest: All the authors declare that they have no conflict of interest.

Ethical approval: This article does not include any studies conducted by any of the authors on human respondents or animals.

References

Aroyehun ST, Gelbukh A (2018) Aggression Detection in Social Media: Using Deep Neural Networks, Data Augmentation and Pseudo Labeling. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018), pp 90–97

Arroyo-Fernández I, Forest D, Torres-Moreno JM, Carrasco-Ruiz M, Legeleux T, Joannette K (2018) Cyberbullying Detection Task: The EBSI-LIA-UNAM system (ELU) at COLING'18 TRAC-1. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018), pp 140–149

Badjatiya P, Gupta S, Gupta M, Varma V (2017) Deep Learning for Hate Speech Detection in Tweets. In: Proceedings of the 26th International Conference on World Wide Web Companion, pp 759–760

Bohra A, Vijay D, Singh V, Akhtar SS, Shrivastava M (2018) A Dataset of Hindi-English Code-Mixed Social Media Text for Hate Speech Detection. In: Proceedings of the Second Workshop on Computational Modeling of People's Opinions, Personality and Emotions in Social Media, pp 36–41

Burnap P, Williams ML (2015) Cyber Hate Speech on Twitter: An Application of Machine Classification and Statistical Modeling for Policy and Decision Making. Policy & Internet 7(2):223–242

Chatzakou D, Kourtellis N, Blackburn J, De Cristofaro E, Stringhini G, Vakali A (2017) Mean Birds: Detecting Aggression and Bullying on Twitter. In: Proceedings of the 2017 ACM on web science conference, ACM, pp 13–22

Chen J, Yan S, Wong KC (2018) Verbal aggression detection on Twitter comments: convolutional neural network for short-text sentiment analysis. Neural Computing and Applications pp 1–10

Davidson T, Warmsley D, Macy M, Weber I (2017) Automated Hate Speech Detection and the Problem of Offensive Language. In: Eleventh International AAAI Conference on Web and Social Media, pp 512–515

Djuric N, Zhou J, Morris R, Grbovic M, Radosavljevic V, Bhamidipati N (2015) Hate Speech Detection with Comment Embeddings. In: Proceedings of the 24th international conference on world wide web, ACM, pp 29–30

Fiore U, De Santis A, Perla F, Zanetti P, Palmieri F (2019) Using generative adversarial networks for improving classification effectiveness in credit card fraud detection. Information Sciences 479:448–455

Galery T, Charitos E, Tian Y (2018) Aggression Identification and Multi-Lingual Word Embeddings. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018), pp 74–79

Gambäck B, Sikdar UK (2017) Using Convolutional Neural Networks to Classify Hate-Speech. In: Proceedings of the First Workshop on Abusive Language Online, pp 85–90

Giap CN, Son LH, Chiclana F (2018) Dynamic structural neural network. Journal of Intelligent & Fuzzy Systems 34(4):2479–2490

Joulin A, Grave E, Bojanowski P, Douze M, Jégou H, Mikolov T (2016) FastText.zip: Compressing text classification models. CoRR abs/1612.03651:1–13, http://arxiv.org/abs/1612.03651, 1612.03651

Kumar R, Bhanodai G, Pamula R, Chennuru MR (2018a) TRAC-1 Shared Task on Aggression Identification: IIT(ISM)@COLING'18. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018), pp 58–65

Kumar R, Reganti AN, Bhatia A, Maheshwari T (2018b) Aggression-annotated Corpus of Hindi-English Code-mixed Data. In: Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018), pp 1425–1431

Kumari K, Singh JP (2019) AI_ML_NIT Patna at HASOC 2019: Deep Learning Approach for Identification of Abusive Content. In: Proceedings of the 11th annual meeting of the Forum for Information Retrieval Evaluation (December 2019), pp 328–335

Kumari K, Singh JP, Dwivedi YK, Rana NP (2019a) Aggression Detection in Social Media using Deep Neural Networks. In: Conference on e-Business, e-Services and e-Society, Springer, pp 415–424

Kumari K, Singh JP, Dwivedi YK, Rana NP (2019b) Towards Cyberbullying-free social media in smart cities: a unified multi-modal approach. Soft Computing

DOI :10.1007/s00500-019-04550-x
Kumari K, Singh JP (2020) Identification of Cyberbullying on multi-modal social media posts using genetic algorithm. Transactions on Emerging Telecommunications Technologies DOI :10.1002/ETT.3907
Madisetty S, Desarkar MS (2018) Aggression Detection in Social Media using Deep Neural Networks. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018), pp 120–127
Maitra P, Sarkhel R (2018) A K-Competitive Autoencoder for Aggression Detection in Social Media text. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018), pp 80–89
Malte A, Ratadiya P (2019) Multi-lingual Cyber Abuse Detection using Advanced Transformer Architecture. In: TENCON 2019-2019 IEEE Region 10 Conference (TENCON), IEEE, pp 784–789
Mehdad Y, Tetreault J (2016) Do Characters Abuse More Than Words? In: Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue, pp 299–303
Modecki KL, Barber BL, Vernon L (2013) Mapping Developmental Precursors of Cyber-Aggression: Trajectories of Risk Predict Perpetration and Victimization. Journal of Youth and Adolescence 42(5):651–661
Modha S, Majumder P, Mandl T (2018) Filtering Aggression from the Multilingual Social Media Feed. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018), pp 199–207
Nikhil N, Pahwa R, Nirala MK, Khilnani R (2018) LSTMs with Attention for Aggression Detection. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018), pp 52–57
Nobata C, Tetreault J, Thomas A, Mehdad Y, Chang Y (2016) Abusive Language Detection in Online User Content. In: Proceedings of the 25th international conference on world wide web, pp 145–153
Orabi AH, Orabi MH, Huang Q, Inkpen D, Van Bruwaene D (2018) Cyber-aggression Detection using Cross Segment-and-Concatenate Multi-Task Learning from Text. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018), pp 159–165
Orasan C (2018) Aggressive language identification using word embeddings and sentiment features. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018), pp 113–119
Pascanu R, Gulcehre C, Cho K, Bengio Y (2014) How to Construct Deep Recurrent Neural Networks. In: 2nd International Conference on Learning Representations, ICLR 2014, pp 1–13
Pennington J, Socher R, Manning C (2014) Glove: Global Vectors for Word Representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp 1532–1543
Qiu J, Wei Y, Wu L (2016) A Novel Approach to Reliable Control of Piecewise Affine Systems with Actuator Faults. IEEE Transactions on Circuits and Systems II: Express Briefs 64(8):957–961
Qiu J, Wei Y, Karimi HR, Gao H (2017) Reliable Control of Discrete-Time Piecewise-Affine Time-Delay Systems via Output Feedback. IEEE Transactions on Reliability 67(1):79–91
Qiu J, Sun K, Wang T, Gao H (2019) Observer-Based Fuzzy Adaptive Event-Triggered Control for Pure-Feedback Nonlinear Systems With Prescribed Per-

formance. IEEE Transactions on Fuzzy Systems 27(11):2152–2162
Raiyani K, Gonçalves T, Quaresma P, Nogueira VB (2018) Fully Connected Neural Network with Advance Preprocessor to Identify Aggression over Facebook and Twitter. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018), pp 28–41
Ramiandrisoa F, Mothe J (2018) IRIT at TRAC 2018. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018), pp 19–27
Risch J, Krestel R (2018) Aggression Identification Using Deep Learning and Data Augmentation. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018), pp 150–158
Rubio de Jesús, J., D. Ricardo Cruz, I. Elias, G. Ochoa, R. Balcazarand and A. Aguilar (2019) ANFIS system for classification of brain signals. Journal of Intelligent & Fuzzy Systems 37(3):4033–4041
Rubio de Jesús, J., E. Lughofer, J. A. Meda-Campaña, L. A. Páramo, J. F. Novoa and J. Pacheco (2018) Neural network updating via argument Kalman filter for modeling of Takagi-Sugeno fuzzy models. Journal of Intelligent & Fuzzy Systems 35(2):2585–2596
Rubio de Jesús (2009) SOFMLS: Online Self-Organizing Fuzzy Modified Least-Squares Network. IEEE Transactions on Fuzzy Systems 17(6):1296–1309
Samghabadi NS, Mave D, Kar S, Solorio T (2018) RiTUAL-UH at TRAC 2018 Shared Task: Aggression Identification. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018), pp 12–18
Srivastava S, Khurana P, Tewari V (2018) Identifying Aggression and Toxicity in Comments using Capsule Network. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018), pp 98–105
Tommasel A, Rodriguez JM, Godoy D (2018) Aggression Detection through Deep Learning. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018), pp 177–187
Waseem Z, Hovy D (2016) Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter. In: Proceedings of the NAACL student research workshop, pp 88–93
Zhao R, Mao K (2017) Cyberbullying Detection Based on Semantic-Enhanced Marginalized Denoising Auto-Encoder. IEEE Transactions on Affective Computing 8(3):328–339