An Ensemble approach for Aggression Identification in English and Hindi Text

Arjun Roy, Prashant Kapil, Kingshuk Basak, Asif Ekbal
Department of Computer Science
and Engineering
Indian Institute of Technology Patna, India
(arjun.mtmc17, prashant.pcs17, kingshuk.mtcs16, asif) @iitp.ac.in

Abstract

In this paper we describe our system that we develop as part of our participation in the shared task at COLING 2018 **TRAC-1: Aggression Identification**. The objective of this task was to predict online aggression spread through online textual post or comment. The datasets were released in two languages, one for English and the other for Hindi. For each of these languages we submitted one system. Both of our systems are based on an ensemble architecture where the individual models are based on Convoluted Neural Network (CNN) and Support Vector Machine (SVM). Our system on English facebook and social media post obtains the F1 scores of 0.5151 and 0.5099, respectively where evaluation on Hindi facebook and social media obtains F1 score of 0.5599 and 0.3790, respectively.

1 Introduction

In our modern world Internet has become a very powerful tool to convey and spread our feelings, voices and intentions to a large section of people in a very short span of time. There has been a phenomenal growth in web contents due to the emergence of numerous social media networking platforms, blogs, review sites etc. There is also a growing rate of misusing these social media and other sources against individuals, groups, organization etc. by targeting them directly or indirectly. There has not been much literature that focus on machine learning applications towards building intelligent systems that could detect aggression, cyberbullying, hate speech, profanity etc. which often have overlapping characteristics. The task is more difficult when the contents are mixed with more than one language, the phenomenon which is very well-known as code mixing.

The task defined in the shared task was related to detecting aggression in two languages, namely Hindi and English. (Baron and Richardson, 1994) defined aggression as a behavior that is intended to harm another individual who does not wish to be harmed.(Buss, 1961) distinguished between physical aggression (e.g. hitting,kicking etc.), verbal aggression (e.g. yelling, screaming etc.) and relational aggression. The datasets provided in the shared task were labeled with three classes, namely "Overtly","Covertly" and "Non- aggressive".

2 Related Works

Aggression, Trolling, Cyberbullying, etc. are the problems that have attracted attention to the various stakeholders (common people, governments, researchers) as these are some of the severe issues that need urgent attention due to the abundance of information generated daily from the various social media sources. Although not much of drafted works can be found on developing an automated system to address these problems, a few works are available on the problem domains that are very closely related, such as the detection of offensive languages and hate speech. (Potapova and Gordeev, 2016) studied verbal expression of aggression and they found

Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying, pages 66–73
Santa Fe, USA, August 25, 2018.

that detection of such contents using machine learning techniques such as Random Forest (RF) and Convolutional Neural Network (CNN) combined with Part-of-Speech (PoS) information can produce good results. (Chu et al., 2017)showed that character embedding performed better than word embedding for CNN in classifying the contents into two classes: personal attack and not-personal attack. (Chen et al., 2012) proposed the Lexical Syntactic Feature (LSF) architecture to detect offensive content and identify the potential offensive users in social media. (Gao and Huang, 2017) proposed two types of hate speech detection models that incorporate context information, a logistic regression model with context features and a neural network model with learning components for context. Their evaluation showed that both the models outperform a strong baseline by around 3% to 4% in F1 score and combining these two models further improved the performance by another 7% in F1 score. (Nobata et al., 2016) divided their feature set into 4 classes as N-grams, linguistic, syntactic and distributional semantics to distinguish between clean and abusive post. The abusive posts were further fine-grained into hate, derogatory and profanity. Their character n-grams based approach also outperformed some word-based deep learning models. (Davidson et al., 2017) discussed the problems in distinguishing the hate speech from the offensive instances. They observed that lexical methods are inaccurate at identifying hate speech and emphasized to take care of social biases into context.

Although there are works available for English in these related domains, we did not find any such work on Hindi. However, there is a considerable increase in the volume of Indian language social media contents, especially in Hindi. Hence detecting trolling, cyberbullying in such languages have a very high relevance in today's scenarios. In this paper we develop an ensemble based architecture for aggression identification which tries to solve the problem in two languages, *viz.* English and Hindi.

3 Problem Definition

The problem of the shared task was on Aggression Identification. The goal is to classify the text into three classes, namely overtly, covertly and non-aggressive.
We first define aggression, and then put forward the different classes of aggression mentioned in the task with examples.

1. **Aggression:** It is a human behavior intended to harm another by verbally, physically and psychologically. **Overtly Aggressive (OAG):** This class includes the following cases.
   (a) Aggression shown openly with verbal attack directly pointed towards any group or individuals.
   (b) Attack commenced using abusive words or calling names or comparing in a derogatory manner.
   (c) By supporting false attack or supporting others comment.
   (d) Sometimes these texts also contain indirect references.
2. **Covertly Aggressive (CAG):** In these attacks aggression is generally hidden and contains sarcastic negative emotions due to its indirect nature. It can be summarized as follows.
   (a) By using metaphorical words to attack an individual,nation,religion.
   (b) Praising someone by criticizing group irrespective of being right or wrong.
   (c) Sometimes these texts also contain direct references.
3. **Non Aggressive (NAG):** These statements generally lack the intention to be aggressive and mostly used while referring to the correct facts, wishing or supporting individuals or groups on social issues.

Table 1 shows a few examples of each class.

Class	Sentence
Overtly Aggressive	1.We want to get rid of u Indians.....why don't u hear our loud cries
	2.I support his speech, RSS is divider of country, they do not want to stay peace.
	3.The U S Government of Donald Trump will do nothing
	4."गटर के कीड़े" ।
	5.एस कुत्ते को जेल मे दाल दो .देश द्रोही हैं ।
Covertly Aggressive	1.Modi ji, all the Pain & no Real Gain
	2.udhav has shown bjp its place,bravo shivsena
	3.Reservation is like another form of terrorism
	4.दंगाई के कुर्सी में बैठ जाने से चरित्र नही सुधरता ।
	5.दोनों दलाल है और इन दोनों का दलाल मिडिया है ।
Non-Aggressive	1.Sorry sir I forgot.
	2.When is work on NH-8 getting completed? Particluarly Hero Honda Chowk??
	3.I want to upgrade from my 180 CC to 400CC
	4.विदिशा से बीजेपी की सुषमा स्वराज आगे ।
	5.जन्म दिन की हार्दिक बधाई महाशया ।

Table 1: **Some examples from English and Hindi data**

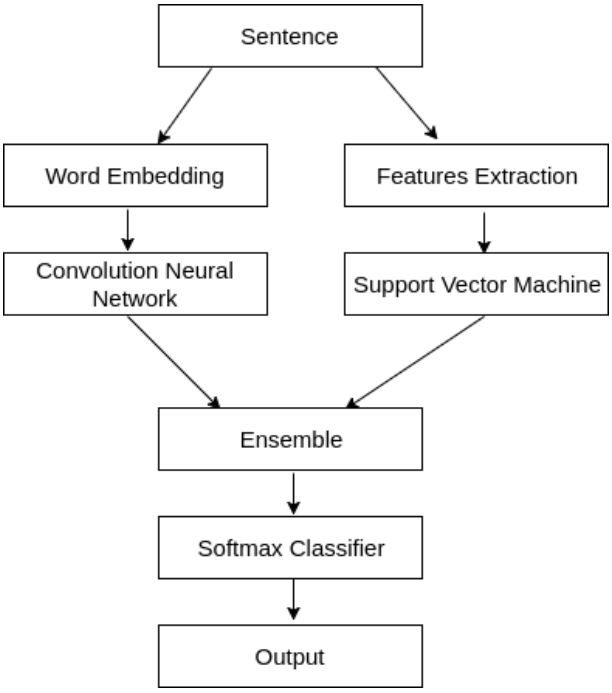

Figure 1: Block Diagram of our System

4 System Overview

We adopt an Ensemble architecture to solve the problem of offensive language detection. The base models are based on CNN and SVM. A block diagram of the system is shown in Figure 1.

4.1 Preprocessing

As the dataset was collected from social media platforms, it contains URLs, emoticons, hashtag, smiley, etc. It also includes a lot of inconsistencies in the form of typos and abbreviations. So the first step that we perform is the removal of URLs for the convenience of preprocessing. The only '#' hash signs are removed from #soldmedia, #Bhagwa,#शव etc. to preserve the meaning. After that for further preprocessing we remove punctuations and apostrophe words. These steps are performed for both Hindi and English datasets. Detailed examples are given in Table 2. The Hindi dataset further posed an additional challenge. The data has sentences mixed with actual

Hindi words, transliterated Hindi words and actual English words- commonly known as code-mixed social media text. To handle this, we convert the transliterated Hindi words and actual English words into actual Hindi words using Indic-Trans api.

1	Before	#shameonjournalism #soldmedia
	After	Shame on journalism sold media
2	Before	Which one is the best example of #Bhagwa terrorism.
	After	Which one is the best example of bhagwa terrorism
3	Before	goons and #presstitutes
	After	goons and presstitutes
4	Before	दोनों शहीदों के #शव के साथ बर्बरता
	After	दोनों शहीदों के शव के साथ बर्बरता

Table 2: **Preprocessing example**

4.2 Word Embedding

In order to fit textual data into Neural Network we need vectorization of texts. This provides useful evidence in capturing semantic property of a word. For Hindi we use the pre-trained model of Wikipedia (Bojanowski et al., 2016). Each word is represented as a vector of 300 dimension. For English, we use the pre-trained Glove(Pennington et al., 2014) vectors. After preprocessing the data, the English dataset has an average sentence length of 20 words with maximum sentence length of 50 words, while the Hindi dataset has an average sentence length of 21 words with maximum sentence length of 50 words [1]. We consider maximum length of a sentence to be 50-words. We use padding with zeros if the sentence length is less than 50, and prune from the last if length of the sentence is greater than 50.

4.3 Features

To train any statistical machine learning model we need a set of features to be extracted from the dataset. The features that we use to train the SVM model are:

- **Uni-grams**: An n-gram of size 1 is referred to as a "unigram". Unigrams helps us to identify which words in the corpus are important to which particular class. Top 1000 unigrams in the corpus are taken as one of the features for training our system.
- **Tf-Idf vectors**: We compute the Term Frequency-Inverse Document Frequency (TF-IDF) and use as feature. Term Frequency refers to how many times a particular word appears in a particular class, whereas Inverse document frequency refers to how much relevant the word is to any particular class, whether the word is common or rare across all.

4.4 Methodology

In this section we describe our proposed methodologies, which are based on SVM and CNN.

- **Support Vector Machines:** Support Vector Machine (SVM) is a popular algorithm used for solving many classification problems.(Vapnik, 2013)demonstrates that abstract learning theory established conditions for generalization and understanding these conditions inspired new algorithm approach to function estimation problems. He explained the generalization ability of learning machine depends on capacity concepts. In our model we use Support Vector Machine as one of the individual model used to form the ensemble system. In particular we use an SVM trained using John Platt's sequential minimal optimization algorithm[2] to solve the quadratic programming problem.

[1] while calculating max and avg sentence for both the English and Hindi dataset we consider all the sentences of training and development dataset

[2] http://weka.sourceforge.net/doc.dev/weka/classifiers/functions/SMO.html

- **Convolutional Neural Network:** In the recent times it has been seen that the convolution and pooling functions of CNN can be successfully used in text classification problems. A convolution layer of $n \times m$ kernel size is used (where m-size of word embedding) to look at n-grams of word at a time and then a Max-pooling layer selects the largest from the convoluted inputs. In our system the convolutional layer is constructed with filter size 64 and ReLU as activation function. We capture the bi-gram features from this layer by taking kernel size as $2 \times EmbeddingSize$. In Max-pooling layer we have used pool size of 1×2.

- **Ensemble:** Classifier ensemble[3] aims at combining the predictions of different classifiers. Ensembles (Florian et al., 2003; Ekbal and Bandyopadhyay, 2008) are often seen to be much more accurate than the individual classifiers that make them up. In the system being discussed output of the max-pooling layer of CNN classifier goes to a flatten layer and then to a dense layer with ReLU activation. On the other hand output of SVM is passed through a dense layer with ReLU activation. Output tensor form both CNN and SVM are then concatenated and passed through a dense layer and a Dropout layer with hyper-parameter of 0.5. This is finally inputted to the output layer with *Softmax* as an activation function. We use *Adam* as an optimizer and *Categorical Cross Entropy* for loss function.

5 Data set

The **Hindi** (Roman and Devanagari) training dataset consists of 15000 annotated (6072 as OAG, 6115 as CAG and 2812 as NAG) instances, while the **English** training dataset consists of 12000 annotated (2708 as OAG, 4240 as CAG and 5052 as NAG) instances of Facebook posts and comments. For tuning the system, validation set of 3000 instances of annotated (1216 as OAG, 1246 as CAG and 538 as NAG) Hindi data and 3000 instances of annotated (711 as OAG, 1056 as CAG and 1233 as NAG) English data were provided. The test data has two different sets which were prepared from two different sources-one from facebook (English: 1515, Hindi: 970) and the another one from other social media (English: 1257, Hindi: 1194) posts and comments which helped in determining generic performance ability of our system. Some details are depicted in Table 4 for Hindi, and in Table 3 for English.

Class	Train	Dev	Test FB	Test SM
CAG	4240	1056	1515	1257
NAG	5052	1233		
OAG	2708	711		

Table 3: **English data description**

Class	Train	Dev	Test FB	Test SM
CAG	4869	1246	970	1194
NAG	2275	538		
OAG	4856	1216		

Table 4: **Hindi data description**

6 Experiments and Results

We perform all the experiments in Python 3 and Java Jdk8 environment. We use the following packages: Keras, NLTK, Numpy and Weka.

[3]"ensemble classifier", "classifier ensemble" and "ensemble system" are used interchangeably referring to the same system.

We use F1-score as the evaluation metric, and use development data to fine-tune the system. We first conduct our experiments with a DT (Decision Tree) model using unigrams as features. The DT model performed on the development data with a weighted F1-score of 0.4593 on Hindi data and 0.470 for English. Thereafter we conduct our experiments using a SVM model with unigrams as features. We obtain the F1-score of 0.4986 and 0.5030 for Hindi and English, respectively. We then performed experiments on the same SVM model using bigrams as features. To our surprise, the performance of the model deteriorated to a weighted F1-score of 0.3858 on Hindi data and 0.404 on English data, and predicted almost two third of the instances as NAG for both the datasets. We then used unigrams and tf-idf vectors as features and conducted our experiments on the SVM model. Performance of the model increased to the weighted F1-score of 0.5457 for Hindi and 0.556 for English. To achieve better performance, we then looked into some deep learning strategies. First, we performed our experiments using an LSTM model. The weighted F1-score given by the model is 0.4061 on Hindi data and 0.4142 on English data. Thereafter we used a CNN model that captures unigram features. The weighted F1-score of the model is 0.5422 on Hindi data and 0.5431 on English data. Then on increasing the kernel size to capture bigram features, the CNN model yields the weighted F1-score of 0.5663 for Hindi and 0.578 for English. The weighted F1-score of the final ensemble system, using CNN with kernel size $2 \times EmbeddingSize$ and SVM with unigrams and tf-idf features, on Hindi development data is **0.5987** while that on English development data is **0.6273**. Evaluation on the test data shows the F1-score of **0.5151** on English facebook data and F1-score of **0.5599** on Hindi facebook data. We obtain the F1-score of **0.5099** on English social media data and F1-score of **0.3790** on Hindi social media data. These results are depicted in details in Table 5 for English Facebook data and other social media data. Similarly, in Table 6 we report for Hindi Facebook data and the Hindi other social media data.

Class	English-FB			English-SM		
	Precision	Recall	F1-Score	Precision	Recall	F1-Score
CAG	0.2178	0.6197	0.5151	0.3900	0.4165	0.5099
NAG	0.8612	0.4333		0.5906	0.7764	
OAG	0.3795	0.5139		0.6243	0.3130	

Table 5: **Result: English test data**

Class	Hindi-FB			Hindi-SM		
	Precision	Recall	F1-Score	Precision	Recall	F1-Score
CAG	0.5180	0.7676	0.5599	0.3415	0.4777	0.3790
NAG	0.6280	0.5282		0.4393	0.5113	
OAG	0.4137	0.3729		0.6243	0.2244	

Table 6: **Result: Hindi test data**

6.1 Error analysis

For analysis we study confusion matrix for each of the test sets. It is observed that for the Hindi-FB test data depicted in Table 9, data originally labeled as OAG gets mostly confused with the class CAG, where as data originally labeled as CAG gets mostly confused with the class NAG. This pattern is also seen in case of test set for Hindi-SM (i.e. social media Hindi) data as depicted in Table 10 and in case of test set for English-SM data as depicted in Table 8. In case of English-FB test data as depicted in Table 7, OAG class gets mostly confused with CAG, NAG class also gets confused with CAG but the data originally labeled as CAG gets confused with both OAG and NAG classes almost equally. The reason for OAG getting converted to CAG was a lot of common keywords present in both the classes like killed, terrorist etc. The

reason for misclassification of CAG to NAG was because of indirect mentions and good words used in a very sarcastic way. Due to this system could not capture the exact feeling behind those sentences. The system performs better for classifying the CAG and NAG classes for all the test datasets in comparison to the classification of OAG. Some of the misclassified examples are shown in Table 11.

Class	OAG	CAG	NAG
OAG	74	51	19
CAG	29	88	25
NAG	92	265	273

Table 7: **Confusion matrix:English-FB**

Class	OAG	CAG	NAG
OAG	113	180	68
CAG	49	172	192
NAG	19	89	375

Table 8: **Confusion matrix:English-SM**

Class	OAG	CAG	NAG
OAG	135	215	12
CAG	47	317	49
NAG	12	80	103

Table 9: **Confusion matrix:Hindi-FB**

Class	OAG	CAG	NAG
OAG	103	240	116
CAG	84	182	115
NAG	62	111	181

Table 10: **Confusion matrix:Hindi-SM**

Original	Predicted	Sentence
CAG	OAG	I told you wait.7 pak killed within hours of their cowardice act.Go and weep for them .
OAG	CAG	yes we remember you are biggest terrorist country in the world you will do anything against humanity
CAG	NAG	Just get new currency to people and we will be happy
OAG	NAG	late night party enjoying life and now once he woke up he start complaining
CAG	OAG	अबे साले वह हमारे सैनिक को मार रहे हैं और तुम सब कड़ी निंदा ही कर रहे हो ।
OAG	CAG	चोरो की सरकार से आज़ादी मिल गया ।

Table 11: **Example of Sentence predicted to different class**

7 Conclusion

In this paper, we have presented the system that we developed as part of our participation in the shared task. We have developed a supervised ensemble approach for detecting the aggression. As base classification algorithm we use CNN and SVM. We evaluate the algorithms for two languages, namely English and Hindi. Datasets of two different kinds of domains were considered: textual posts and the comments made on social media platforms. Since the LSTM model did not perform well in our experiments, it would be interesting to see how the network fares when an attention mechanism will be deployed. As in many of the overtly and covertly aggressive instances named entity is often seen especially with '#hashtags', it would also be interesting to see that extracting NER feature from the data whether having any significant impact. In future, we would like to explore some other relevant linguistic features and hybrid machine learning architectures to improve the performance of the system further.

References

Robert A Baron and Deborah R Richardson. 1994. Human aggression: Perspectives in social psychology. *Nova Iorque: Plenum Press.*

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606.*

Arnold H Buss. 1961. *The psychology of aggression.* Wiley.

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

Theodora Chu, Kylie Jue, and Max Wang. 2017. Comment abuse classification with deep learning.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. *arXiv preprint arXiv:1703.04009*.

Asif Ekbal and Sivaji Bandyopadhyay. 2008. Bengali named entity recognition using support vector machine. In *Proceedings of the IJCNLP-08 Workshop on Named Entity Recognition for South and South East Asian Languages*.

Radu Florian, Abe Ittycheriah, Hongyan Jing, and Tong Zhang. 2003. Named entity recognition through classifier combination. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 168–171. Association for Computational Linguistics.

Lei Gao and Ruihong Huang. 2017. Detecting online hate speech using context aware models. *arXiv preprint arXiv:1710.07395*.

Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive language detection in online user content. In *Proceedings of the 25th international conference on world wide web*, pages 145–153. International World Wide Web Conferences Steering Committee.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Rodmonga Potapova and Denis Gordeev. 2016. Detecting state of aggression in sentences using cnn. *arXiv preprint arXiv:1604.06650*.

Vladimir Vapnik. 2013. *The nature of statistical learning theory*. Springer science & business media.