Aggression Identification Using Deep Learning and Data Augmentation

Julian Risch
Hasso Plattner Institute
University of Potsdam
Prof.-Dr.-Helmert-Str. 2-3
14482 Potsdam, Germany
julian.risch@hpi.de

Ralf Krestel
Hasso Plattner Institute
University of Potsdam
Prof.-Dr.-Helmert-Str. 2-3
14482 Potsdam, Germany
ralf.krestel@hpi.de

Abstract

Social media platforms allow users to share and discuss their opinions online. However, a minority of user posts is aggressive, thereby hinders respectful discussion, and — at an extreme level — is liable to prosecution. The automatic identification of such harmful posts is important, because it can support the costly manual moderation of online discussions. Further, the automation allows unprecedented analyses of discussion datasets that contain millions of posts.

This system description paper presents our submission to the First Shared Task on Aggression Identification. We propose to augment the provided dataset to increase the number of labeled comments from 15,000 to 60,000. Thereby, we introduce linguistic variety into the dataset. As a consequence of the larger amount of training data, we are able to train a special deep neural net, which generalizes especially well to unseen data. To further boost the performance, we combine this neural net with three logistic regression classifiers trained on character and word n-grams, and hand-picked syntactic features. This ensemble is more robust than the individual single models. Our team named "Julian" achieves an F1-score of 60% on both English datasets, 63% on the Hindi Facebook dataset, and 38% on the Hindi Twitter dataset.

1 Introduction

Social media platforms, such as Facebook, YouTube, Twitter, and Instagram, enable millions to publicly share user-generated content. Regardless of different content types, such as text, photos, videos, and events, a crucial point of these platforms is that users can discuss content. The opportunity to articulate opinions and ideas online is a valuable good: It is part of the freedom of expression, which is declared in the Universal Declaration of Human Rights. However, aggressive and/or hateful posts can disrupt otherwise respectful discussions. Such posts are called "toxic", because they poison a conversation so that other users abandon it. Toxicity is manifold and comprises, for example, obscene language, insults, threats, and identity hate. Such statements are not covered by the freedom of expression, because they harm others. The boundaries of the freedom of expression are a controversial topic. In moderated online discussions, it is the task of human moderators to identify toxic comments and potentially delete them.

An automatic identification of toxic posts could support (or even to some extent replace) the costly manual moderation of online discussions. For example, it could draw the attention of moderators to posts that have been automatically identified as toxic. Another advantage of the automatic identification of toxic posts is that it allows the analysis of much larger datasets. For example, classifiers that were trained on a rather small hand-labeled dataset have been successfully used to machine-label and analyze datasets with tens of millions of posts (Wulczyn et al., 2017).

The First Shared Task on Aggression Identification (Kumar et al., 2018a) deals with the classification of the aggression level of user posts at different social media platforms. It is part of the First Workshop on Trolling, Aggression and Cyberbullying at the 27th International Conference of Computational Linguistics (COLING 2018). The task is a three-way classification problem with the three classes "overtly aggressive" (OAG), "covertly aggressive" (CAG), and "non-aggressive" (NAG). The training data consists

of 15,000 aggression-annotated Facebook posts each in Hindi and English. Weighted macro-averaged F1-scores serve as the evaluation metric: The individual F1-score of each class is weighted by the proportion of the concerned class in the test set and the final F1-score is the average of these individual F-scores of each class.

In this system description paper, we introduce our implemented system as submitted for the shared task. Our approach is based on a recurrent neural network, more specifically, a bi-directional gated recurrent unit (GRU) layer with max pooling and average pooling. However, the relatively small size of the training dataset is a strong limitation for deep learning approaches. Therefore, we propose to augment the training dataset: We use machine translation to translate each English comment into a foreign language, e.g. French, and translate it back into English afterwards. The result of these translations is another English comment that typically uses slightly different words compared to the initial comment. Examples for these translations are given in Section 3.1 and the augmented dataset together with our implementation is published online[1].

Our contributions can be summarized as providing a:

1. data augmentation method that triples the dataset size,
2. neural network architecture based on a GRU layer for the task of Aggression Identification,
3. comparison of our results at the different subtasks and an error analysis of our approach.

Section 2 gives an overview of related work. Section 3 explains our approach and goes into detail about the proposed data augmentation method and the neural network architecture. We lists our results at the different subtasks in Section 4 and discuss how the proposed approach generalizes to unseen data. Finally, we conclude and outline possible paths for future work in Section 5.

2 Related Work

While the shared task focuses on aggression identification, related work considers the broader field of hate speech, offensive, and abusive language identification. First approaches to the problem of classifying insulting messages use a decision tree and go back to Spertus (1997). A hand-written set of rules with syntactic and semantic text features are the basis of this model. For a slightly different task, harassment detection in the web, Yin et al. (2009) introduce contextual features, which consider a user's previous and succeeding posts as a context. The survey by Schmidt and Wiegand (2017) points out that bag-of-word models are good features for hate speech detection, although they ignore word order and sentence syntax. Further, the authors propose to generalize from particular words to clusters of words. To capture the semantic similarity of words our approach uses the fastText model by Bojanowski et al. (2017). According to the survey by Schmidt et al., positive, negative, and neutral words are promising features for hate speech detection. Polarity classifiers for short texts, such as SentiStrength (Thelwall et al., 2010), are suited to extract such words of polarity and we make use of a polarity classifier in our approach.

Previous work agrees that word n-grams are well-performing features for hate speech detection (Nobata et al., 2016; Badjatiya et al., 2017; Warner and Hirschberg, 2012; Davidson et al., 2017; Schmidt and Wiegand, 2017). Davidson et al. (2017) compared logistic regression, naive Bayes, decision trees, random forests, and support vector machines, and conclude that logistic regression and support vector machines are the best performing classifiers for hate speech detection. Based on these insights, we include word n-grams and logistic regression in our approach. Further, a combination of different models, as in the multi-level approach by Razavi et al. (2010), can make use of each model's strengths. Therefore, we combine four different models in an ensemble.

One of the main limitation for research progress in the field of toxic comment classification is the low amount of available accurately labeled data (Kennedy et al., 2017). Wulczyn et al. (2017) address this issue by training a classifier on a small set of human-labeled comments in order to afterwards generate a larger machine-labeled dataset. Recently, Kaggle's Toxic Comment Classification[2] made available more

[1] https://hpi.de/naumann/projects/repeatability/text%2Dmining.html

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

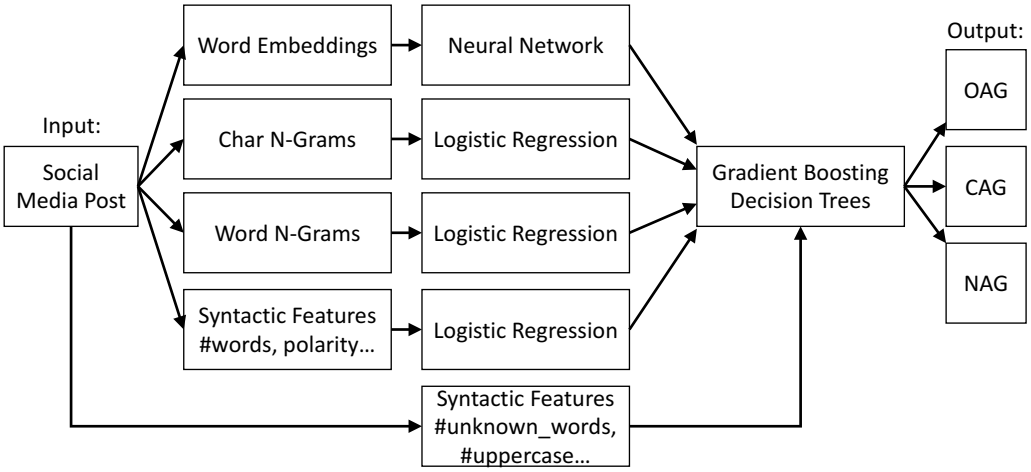


Figure 1: Given a social media post, we apply four different models and combine their predictions in an ensemble to identify the aggression level of the post (OAG, CAG, NAG).

than 150,000 English, hand-labeled comments. Other shared tasks provide non-English hand-labeled data, such as German tweets[3]. Besides this positive recent development of data sharing, fine-grained labeling becomes a topic of broader interest, for example in the work of Van Hee et al. (2015). A discussion on the challenges of identifying profanity vs. hate speech is given by Malmasi and Zampieri (2018). The results demonstrate that it can be hard to distinguish between overt and covert aggression in social media. The finer-grained labeling coincides with the demand for explanations of a classifier's decision. A human moderator that decides whether to delete an automatically identified toxic comment might want to know a fine-grained reason, such as whether the comment contains an insult, a threat, or identity hate.

3 Methodology and Data

Our approach is based on two main ideas: (1) increasing the amount of available training data by data augmentation in Section 3.1 and (2) leveraging the larger amount of training data with a deep learning approach in Section 3.3. In addition to that, we propose three other models besides the deep learning approach. Figure 1 is a system overview that shows how the four models are combined in an ensemble. We publish the augmented dataset and the implementation of our approach in python with Keras and Tensorflow online[4].

3.1 Data Augmentation Based on Machine Translation

The training dataset consists of 15,000 posts for the English task and 15,000 posts for the Hindi task. The data collection methods that were used to compile the dataset are described in Kumar et al. (2018b). We refer to the dataset with published training data as Facebook dataset, because it contains Facebook posts. The other dataset, which has been provided only as a test dataset without training labels, is referred to as Twitter dataset, because it contains tweets.

In the following, we present our data augmentation method, which is based on the following insight: Machine translating a user comment into a foreign language and then translating it back to the initial language preserves its meaning but results in different wording. This change in wording is essential for our approach: If the translation did not change the wording, it could not augment our dataset, because the dataset already contains the exact same comment. However, because the wording is different, the translated comment adds to our dataset. Only if the meaning is preserved, we can assume that the label (non-aggressive, covertly aggressive, overtly aggressive) of the initial comment also holds for the translated comment. Thanks to the recent advances of neural machine translation and its continuously

[3] https://projects.cai.fbi.h%2Dda.de/iggsa/germeval/

[4] https://hpi.de/naumann/projects/repeatability/text%2Dmining.html

improving accuracy, we can assume that machine translation preserves the meaning of posts. We give two examples for the data augmentation. The first example shows that different translations use different words, such as "health", "wealth", "(economic) growth", and "prosperity".

1. Initial English post: "Happy Diwali.!!let's wish the next one year health,wealth n growth to our Indian economy."
2. English to French to English: "Happy Diwali., Wish the next year health, wealth and growth to our Indian economy."
3. English to German to English: "Happy Diwali, let us wish the next year health, prosperity and growth of our Indian economy."
4. English to Spanish to English: "Happy Diwali We wish the health, economic growth and health of our next year!"

The second example is a rather short post and all its translations are similar. However, for example, the abbreviation "'u" for the word "you" is resolved.

1. Initial English post: "AAP dont need the monsters like u"
2. English to French to English: "AAP does not need monsters like you"
3. English to German to English: "AAP does not need the monsters like you"
4. English to Spanish to English: "AAP does not need monsters like you"

We applied this data augmentation method also to the Hindi dataset. Therefore, each Hindi post was machine-translated into English and afterwards translated back to Hindi. For Hindi, this method did not work as well. Often already the intermediate step of translating to English failed in preserving the meaning of the initial Hindi post. As a consequence, the meaning of the translated posts did not match with the initial labels and the translated posts could not be used for training. We assume that the quality of machine translations from Hindi to other languages is comparably worse due to a lower amount of training data.

3.2 Data Pre-Processing

We propose a special tokenization method for hashtags and user mentions. Many hashtags in the dataset are concatenations of multiple words. For example, the meaning of "#realsurgicalstrike", "#deathtoPakistan", and "#saysiamaproudchutiyawhodoentknowshitabouthistory" can only be understood if the words are split correctly. In some cases, the full post contains only a hashtag, such as the post "#THANKY0UTAKER" which is labeled as covertly aggressive. We propose to split the strings after "#" and "@" symbols into their original words with a dynamic programming approach. Our assumption is that the best splitting is the one that maximizes the product of each word's individual probability of occurrence. For example, the splitting "real surgical strike" is to prefer over the splitting "real surgicals trike", because the probability of "surgical" is higher than the probability of "surgicals" and the probability of "trike" is higher than the probability of "trike". A word's probability of occurrence can be inferred from a large corpus of natural language, such as Wikipedia. Another splitting strategy would be to split at capitalized letters, but we did not explore this idea.

Our deep learning approach uses 300-dimensional, pre-trained word embeddings by fastText (Bojanowski et al., 2017). More specifically, we use the common crawl embeddings[5] for the English tasks and the Hindi Wikipedia embeddings[6] for the Hindi tasks (Grave et al., 2018). In comparison to other embedding methods, fastText embeddings do not suffer from out-of-vocabulary problems. A word that has never been seen during training time is represented by embeddings of character n-grams.

[5] https://fasttext.cc/docs/en/english%2Dvectors.html
[6] https://fasttext.cc/docs/en/crawl%2Dvectors.html

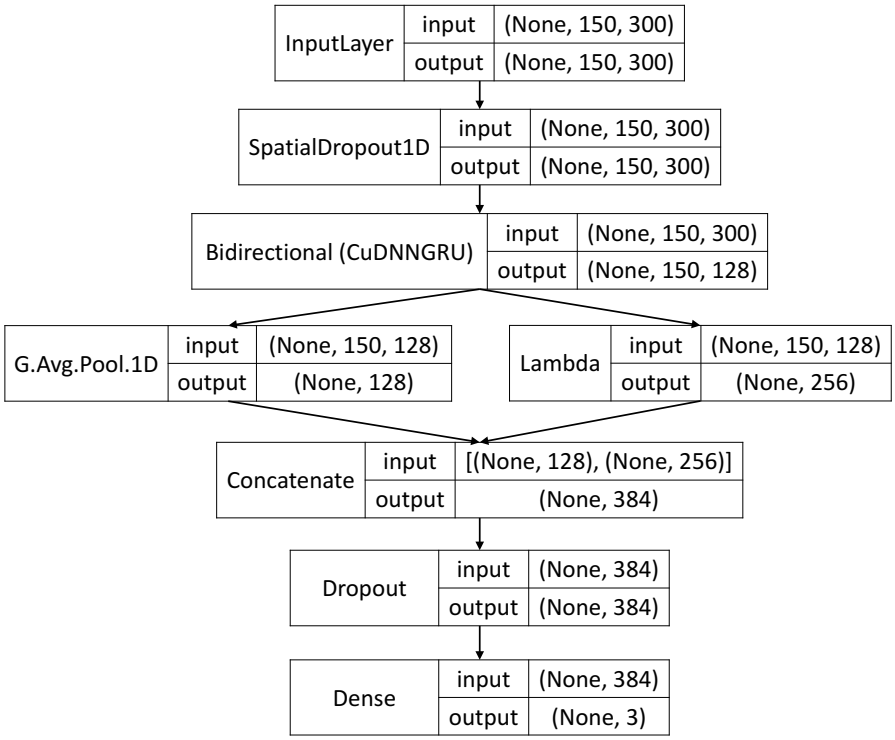


Figure 2: The neural network uses GRUs, average pooling, and k-maximum pooling (Lambda layer).

3.3 GRUs for Aggression Identification

We propose a recurrent neural network based on GRUs. The network architecture is visualized in Figure 2. We use pre-trained, fixed fastText embeddings, obtained as described in the previous Section 3.2. We pad all posts to sequences of 150 words. The word embeddings are input to a spatial dropout, which blocks the embeddings of 10% randomly chosen input words. A bi-directional layer of 64 GRUs processes the remaining 90% of input. The next layer performs global average pooling and global k-maximum pooling independently on the sequence of the outputs of all GRUs. A k-maximum pooling with $k = 2$ extracts not only the largest, but also the second-largest element of the previous layer. A Lambda layer implements this non-standard pooling technique. The average, the maximum, and the second-largest element are concatenated into one vector and a dropout of 10% is added to prevent overfitting. Finally, a dense layer with softmax activation outputs 3 class probabilities. The model is trained for 2 epochs with a batch size of 32. We observe that the data augmentation slightly reduces the number of epochs until overfitting starts.

3.4 Tf-idf on Character and Word N-Grams

We extract character n-grams of length 2 to 6 and limit the set to the 50,000 most frequent character n-grams. We extract word n-grams of length 1 to 2 and filter English stopwords, but also all words that occur in more than 50% of all documents or in less than 2 documents. We normalize the frequency of occurrence of all n-grams using tf-idf. As the classifier, we choose logistic regression for both, word and character n-grams. Based on the extracted n-grams, we train logistic regression models according to the one-vs.-rest strategy: We train one classifier per aggression level. For each aggression level, all posts of that level are positive and all other posts are negative training examples.

3.5 Hand-Picked Features

A set of hand-picked features captures various properties, such as punctuation and capitalization, but also emoticons. Overall, a combination of 35 extracted features serves as input to three logistic regressions, which are trained according to the one-vs.-rest strategy for each level of aggression. 25 of these features capture emoticons with regular expressions for sad, happy, and neutral faces. The remaining 10 features

capture, for example, the number of words, the proportion of uppercase characters to lowercase characters, the number of negation words, and also the polarity of the post. We apply the VADER sentiment analyzer to extract polarity scores (Gilbert, 2014).

3.6 Ensembling

Word embeddings, word n-grams, character n-grams, and hand-picked features capture different properties of user posts and therefore have different strengths and weaknesses. For example, word n-grams suffer from out-of-vocabulary problems, which makes them sensitive to obfuscated words. The dataset contains posts that make extensive use of obfuscation, such as "Son of a B****", "***k them!!!!". Word embeddings and word n-grams cannot capture the meaning of these obfuscated posts, but character n-grams or the number of asterisks and exclamation marks as hand-picked features can.

For the four models, we analyze the pairwise Pearson correlation of their predictions as listed in Table 1. The word n-gram and the character n-gram models have the highest correlation. In contrast, the recurrent neural network and the word n-gram model have a rather low correlation. Their low correlation motivates to combine their predictions, because we can assume that they complement each other well. If they both have a similarly high F1-score, their combination outperforms the single models.

Class	RNN-Word	RNN-Char	RNN-Hand	Word-Char	Word-Hand	Char-Hand
OAG	0.7271	0.7489	0.4548	0.8417	0.4132	0.4229
NAG	0.8070	0.8241	0.4163	0.8844	0.3961	0.4213
CAG	0.6240	0.6516	0.1779	0.7687	0.1499	0.2102

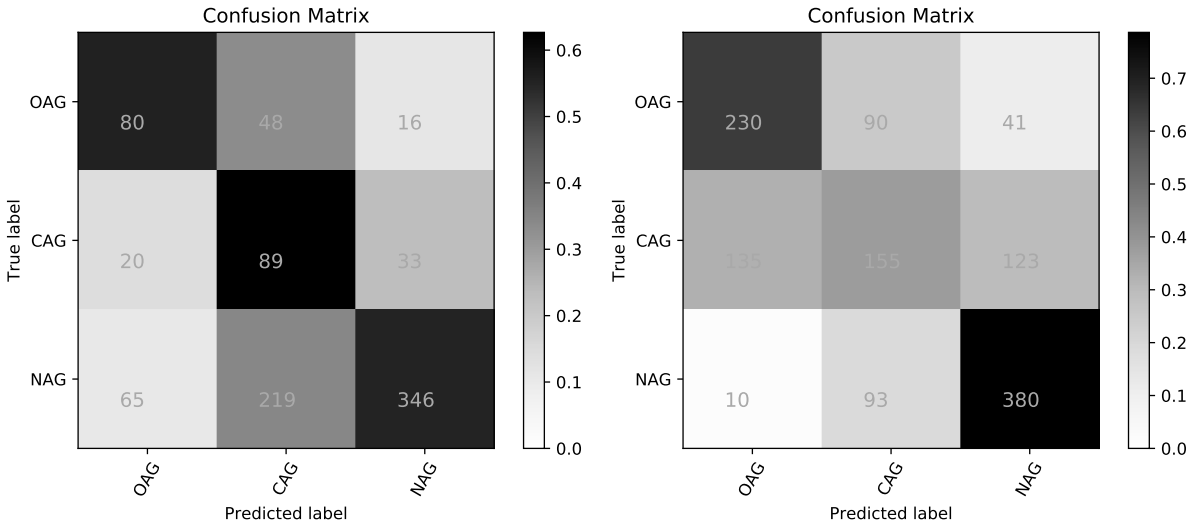
Table 1: Pearson correlation of the different models and classes (RNN: recurrent neural network, Word: word n-grams, Char: character n-grams, Hand: hand-picked features).

The different strengths and weaknesses of the proposed four models motivate their combination in an ensemble. For each of the four models, we run 10-fold cross-validation and create out-of-fold predictions. For each of the 10 runs, we also make predictions for the test set and average all 10 predictions per model. We use the out-of-fold predictions to learn, what combination of the single models performs best. Instead of a simple weighted average of the different models, we propose a stacking approach: Given a comment, we extract features and based on these features, decide how to weight the different models' predictions for this particular comment.

For each comment, we extract: (1) the length (number of characters), (2) the relative number of uppercase characters (number of uppercase characters divided by the total number of characters), (3) the relative number of non-alpha characters (number of non-alpha characters divided by the total number of characters), and (4) the relative number of exclamation marks (number of exclamation marks divided by the total number of characters). For english-language comments, we also extract the relative number of how many words in the comment have a GloVe embedding (Pennington et al., 2014). Although we use fasttext embeddings, which do not suffer from out-of-vocabulary problems, we include this feature to measure how many uncommon words are used in the comment. Based on the extracted features, we train a stacker on the out-of-fold predictions and combine all approaches in an ensemble. The stacker uses gradient boosting trees. More precisely, we use 75 trees with a depth of 3, a bagging fraction of 0.8 and a feature fraction of 0.45.

4 Results

Table 2 lists our cross-validation results. The RNN, word n-grams and character n-grams perform equally well on the English data. The data augmentation makes only a small difference overall. However, it improves the F1-score of the RNN from 57.2% to 58.5%. On the Hindi data, character n-grams clearly outperform all other models. We assume that the performance of the RNN could be improved with better word embeddings, such as embeddings trained on Hindi social media posts. The hand-picked feature selection is superior to the random baseline but inferior to all other models for both languages.



(a) Facebook

(b) Twitter

Figure 3: Confusion matrices for the ensemble appproach on the English test datasets.

Table 3 lists our test set results. Our results on both English test sets are the most stable results across all participating teams: We achieve an F1-score of 60.0% on both datasets, the Facebook dataset and the Twitter dataset. These results show that our approach does not suffer from overfitting to the training dataset and generalizes well to other datasets. Our approach achieves rank 6 out of 30 on the Facebook dataset with an F1-score of 60.0% (F1-score of the top team: 64.2%). On the Twitter dataset, it achieves rank 2 out of 30 with an F1-score of 60.0% (F1-score of the top team: 60.1%).

Our results on the Hindi test sets differ for the Facebook dataset and the Twitter dataset: Our approach achieves rank 4 out of 15 on the Facebook dataset with an F1-score of 63.1% (F1-score of the top team: 64.5%). On the Twitter dataset, it achieves rank 8 out of 15 with an F1-score of 38.3% (F1-score of the top team: 49.9%). The F1-scores of each team differ between the Facebook dataset and the Twitter dataset by 13.1% on average. Therefore, we assume that the differences in classification performance are inherent to the datasets.

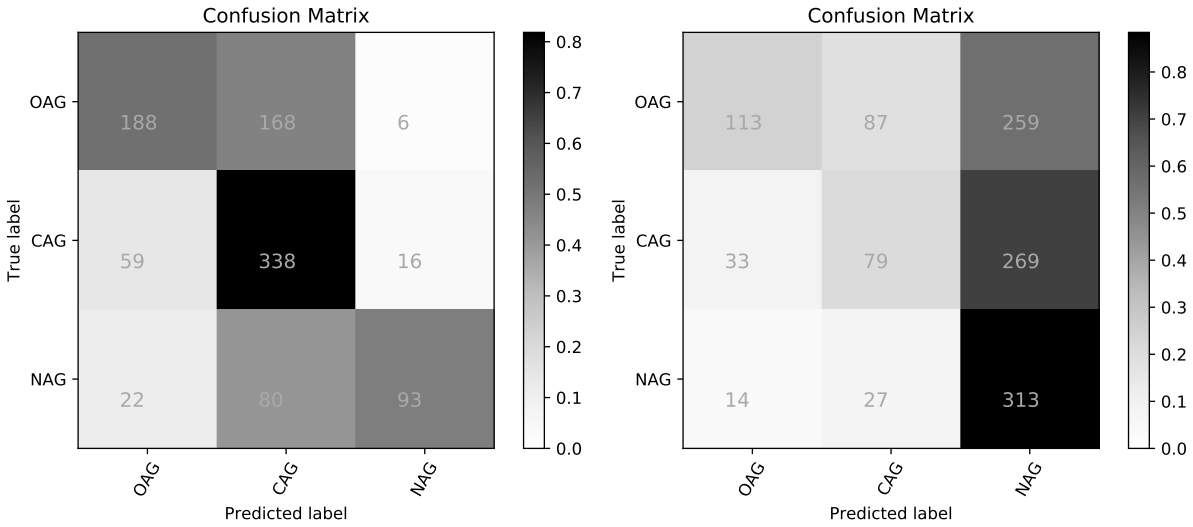
System	F1 En FB	F1 En FB augm.	F1 Hi FB
Random Baseline	0.3324	0.3395	0.3425
RNN	0.5722	0.5846	0.5413
Word N-Grams	0.5764	0.5766	0.5883
Char N-Grams	0.5803	0.5791	0.6103
Feature Selection	0.3966	0.3871	0.3701
Ensemble	**0.6060**	**0.6084**	**0.6292**

Table 2: F1-scores with 10-fold cross-validation on English Facebook dataset (En FB) and Hindi Facebook dataset (Hi FB). F1-score of the RNN approach is improved on the augmented (augm.) dataset.

System	F1 En FB	F1 En SM	F1 Hi FB	F1 Hi SM
Random Baseline	0.3535	0.3477	0.3571	0.3206
Ensemble	**0.6011**	**0.5995**	**0.6311**	**0.3835**

Table 3: F1-scores on the test set. The ensemble achieves higher F1-scores on the test set than the single models in the cross-validation.

Figure 3 shows the confusion matrix of the ensemble submission for the English test datasets. Fig-



(a) Facebook

(b) Twitter

Figure 4: Confusion matrices for the ensemble approach on the Hindi test datasets.

ure 3b shows that the classifier works equally well for all three classes: overtly aggressive, covertly aggressive, and non-aggressive. As to expect, the non-aggressive class is more often confused with the covertly aggressive class than the overtly aggressive class. Similarly, the overtly aggressive class is more often confused with the covertly aggressive class than the non-aggressive class. While on the English Twitter dataset the classifier works well for non-aggressive posts and for overtly aggressive posts, it is only slightly better than a random baseline for covertly aggressive posts. Covertly aggressive posts are often misclassified as either non-aggressive or overtly aggressive. Figure 4a and 4b show the confusion matrices of the ensemble submission for the Hindi test datasets. It is hard for the classifier to distinguish overtly aggressive from covertly aggressive Facebook posts and covertly aggressive from non-aggressive ones. For the Twitter dataset, the majority of the posts is misclassified as non-aggressive.

5 Conclusion

In this paper we considered the problem of aggression identification in social media posts. We presented our submitted system for the First Shared Task on Aggression Identification (Kumar et al., 2018a) as part of the First Workshop on Trolling, Aggression and Cyberbullying at the 27th International Conference of Computational Linguistics (COLING 2018). Our approach leverages machine translation to augment the training dataset and includes a GRU-based deep neural network for the classification. A combination of four models makes our approach more robust and improves its ability to generalize to unseen data.

Across all participating teams, our approach achieves the most stable results: On the English dataset, we achieve rank 6 out of 30 on the Facebook dataset with an F1-score of 60.0% and rank 2 out of 30 with an F1-score of 60.0% (F1-score of the top team: 60.1%). The results confirm the assumption that an ensemble of different models is robust against changes of the dataset. In future, our approach could be extended by augmenting also the test dataset. Further, we are confident that better suited word embeddings would improve classification performance and that more labeled training data would give the opportunity to train more complex neural network architectures.

References

Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. Deep learning for hate speech detection in tweets. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 759–760. International World Wide Web Conferences Steering Committee.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics (TACL)*, 5:135–146.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of the International Conference on Web and Social Media (ICWSM)*, pages 512–515.

CJ Hutto Eric Gilbert. 2014. VADER: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the International Conference on Web and Social Media (ICWSM)*, pages 216–225.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*.

George Kennedy, Andrew McCollough, Edward Dixon, Alexei Bastidas, John Ryan, Chris Loo, and Saurav Sahay. 2017. Technology solutions to combat online harassment. In *Proceedings of the Workshop on Abusive Language Online*, pages 73–77. ACL.

Ritesh Kumar, Atul Kr. Ojha, Shervin Malmasi, and Marcos Zampieri. 2018a. Benchmarking Aggression Identification in Social Media. In *Proceedings of the Workshop on Trolling, Aggression and Cyberbulling (TRAC)*.

Ritesh Kumar, Aishwarya N. Reganti, Akshit Bhatia, and Tushar Maheshwari. 2018b. Aggression-annotated Corpus of Hindi-English Code-mixed Data. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*.

Shervin Malmasi and Marcos Zampieri. 2018. Challenges in Discriminating Profanity from Hate Speech. *Journal of Experimental & Theoretical Artificial Intelligence*, 30:1–16.

Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive Language Detection in Online User Content. In *Proceedings of the International Conference on World Wide Web (WWW)*, 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 *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. ACL.

Amir H. Razavi, Diana Inkpen, Sasha Uritsky, and Stan Matwin. 2010. Offensive language detection using multi-level classification. In *Proceedings of the Canadian Conference on Advances in Artificial Intelligence (Canadian AI)*, pages 16–27. Springer-Verlag.

Anna Schmidt and Michael Wiegand. 2017. A Survey on Hate Speech Detection Using Natural Language Processing. In *Proceedings of the International Workshop on Natural Language Processing for Social Media (SocialNLP)*, pages 1–10. ACL.

Ellen Spertus. 1997. Smokey: Automatic recognition of hostile messages. In *Proceedings of the National Conference on Artificial Intelligence and Conference on Innovative Applications of Artificial Intelligence (AAAI /IAAI)*, pages 1058–1065. AAAI.

Mike Thelwall, Kevan Buckley, Georgios Paltoglou, Di Cai, and Arvid Kappas. 2010. Sentiment strength detection in short informal text. *Journal of the Association for Information Science and Technology*, 61(12):2544–2558.

Cynthia Van Hee, Els Lefever, Ben Verhoeven, Julie Mennes, Bart Desmet, Guy De Pauw, Walter Daelemans, and Véronique Hoste. 2015. Detection and fine-grained classification of cyberbullying events. In *Proceedings of the International Conference Recent Advances in Natural Language Processing (RANLP)*, pages 672–680.

William Warner and Julia Hirschberg. 2012. Detecting hate speech on the world wide web. In *Proceedings of the Workshop on Language in Social Media (LSM)*, pages 19–26. ACL.

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex machina: Personal attacks seen at scale. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 1391–1399. International World Wide Web Conferences Steering Committee.

Dawei Yin, Zhenzhen Xue, Liangjie Hong, Brian D Davison, April Kontostathis, and Lynne Edwards. 2009. Detection of harassment on web 2.0. *Proceedings of the Content Analysis in the WEB*, 2:1–7.