IRJSE International Research Journal of Science & Engineering
Int. Res. J. of Science & Engineering, Volume 12 (3) 2024
ISSN: 2322-0015
SJIF Impact Factor 6.70
https://doi.org/10.5281/zenodo.11441057
RESEARCH ARTICLE
OPEN ACCESS

Using Python to Implement Aggression Detection for Multilingual Social Networking Text

Arjit Manoj Tiwari[1] and Mohammad Atique[2*]

[1]Research Scholar, Dept. of Computer Science, S.G.B.A.U., Amravati M.S., India
[2]Prof. & Head, Dept. of Computer Science, S.G.B.A.U., Amravati, M.S., India
*Corresponding author Email: arjitmt@gmail.com[1] | mohammadatique@sgbau.ac.in[2]

Manuscript Details
Received :15.05.2024
Accepted: 29.05.2024
Published: 03.06.2024

Available online on https://www.irjse.in
ISSN: 2322-0015

Cite this article as:
Arjit Manoj Tiwari and Mohammad Atique. Using Python to Implement Aggression Detection for Multilingual Social Networking Text, *Int. Res. Journal of Science & Engineering*, 2024, Volume 12(3): 101-108. https://doi.org/10.5281/zenodo.11441057

Abstract

Online communication platforms have transformed how we interact, but this connectivity has also amplified the prevalence of aggressive and harmful behavior. To mitigate these negative impacts, this paper implements a methodology for Using machine learning algorithms and a range of linguistic characteristics to identify violence in multilingual social media posts.We utilize a dataset comprising posts labeled as overtly aggressive, covertly aggressive, and non-aggressive. We leverage Support Vector Machine (SVM) algorithm alongside comprehensive pre-processing pipeline that includes sentiment analysis, emoji detection, and an expanded dictionary of aggressive words. Our model achieves promising results, with Gradient Boosting Machines (GBM) demonstrating the best overall performance (F1-score: 0.5735) compared to SVM and XGBoost. This work contributes to the development of automated content moderation systems for fostering safer online spaces.

Keywords: Aggression Detection, Multilingual Text, Machine Learning, social media, Linguistic Features, Support Vector Machine (SVM), Sentiment analysis, Natural Language Processing (NLP), Gradient Boosting Machine (GBM)

1. Introduction

The widespread use of digital communication platforms has changed how people interact, allowing instant sharing of information across various contexts. However, this increased connectivity has also led to more aggressive and harmful online behavior, creating challenges for maintaining a safe digital environment. Detecting and categorizing online aggression is crucial for reducing its negative impact. In this paper, we implement methods for detecting and classifying online aggression in

text using machine learning. Our approach includes data preprocessing, feature extraction, model training, and evaluation stages. To create a functional system, we use machine learning algorithms such as XGBoost, GBM, and SVM in addition to natural language processing techniques. Our system focuses on SVM for performance testing, supplemented by GBM and XGBoost. We employ grid search with cross- validation to optimize hyperparameters. The dataset undergoes meticulous preprocessing to remove noise, and we extract meaningful features using TF-IDF vectorization. We also consider sentiment and emoji usage as additional features. A unique aspect of our approach is the integration of an extensive dictionary of aggressive words, supplemented by external sources, to enhance detection accuracy. We categorize aggression into three levels: no aggression, covered (indirect) aggression, and direct aggression, providing detailed insights. Our approach's efficacy is demonstrated by experimental findings on a labeled dataset, where the model's performance can be seen through specific measures like accuracy, precision, recall, and F1 score. The system includes an interactive component for real-time user input analysis, beneficial for content moderation. Our work contributes to creating safer online spaces by improving online aggression detection. The methodologies presented here can guide future research in automated content moderation, applicable to various digital communication platforms.

1. Related Work

The detection of aggression in text has received significant attention from researchers due to its crucial role in ensuring safe and constructive online communication. Many machine learning methods have been investigated for their efficacy in this field, including Gradient Boosting Machine, Extreme Gradient Boosting, and Support Vector Machine Orasan [1]. Laura and Sara have also created a fuzzy method using the F1 score [2].

SVM's versatility in handling high-dimensional feature spaces makes it a popular choice for text categorization applications, Dadvar et al. [3] utilized SVM for detecting cyberbullying in social media texts, achieving improved accuracy by incorporating user-related features. Pawar and Pawar [4] employed SVM to classify online harassment in social media posts, achieving high precision and recall with the use of n-grams and TF-IDF for feature extraction.

XGBoost, known for its superior performance, has been applied to detect toxic comments in online forums by Zhao et al. [5], outperforming traditional algorithms with a combination of word embeddings and hand-crafted features. Rizwan et al. [6] leveraged XGBoost for classifying abusive language in social media, emphasizing the importance of feature engineering and hyperparameter tuning for high accuracy and F1 scores. GBM, another powerful method, focuses on correcting errors iteratively and has been applied effectively in aggression detection. Nandhini and Sheeba [7] used GBM to identify hate speech in Twitter data, showcasing its ability, along with linguistic features and sentiment analysis, to differentiate between aggressive and non-aggressive tweets. Park and Fung [8] explored GBM for detecting cyberbullying, incorporating user behavior features and temporal patterns to capture the nuanced characteristics of aggressive behavior over time. Nobata et al. [9] works on abusive language detection and use syntactic and embedding features.

Si et al. [10] works on features like word vectors, manual dictionary of aggressive word sentiment scores. Based on this we done implementation in python.

2. Methodology

Dataset

The TRAC-1 dataset, which was the basis for our dataset, included social media posts classified as Overtly Aggressive (OAG), Covertly Aggressive (CAG), and Non-Aggressive (NAG). Annotated data used to detect aggressiveness and associated phenomena in text is collected in the TRAC-1 (Trolling, aggressiveness, and Cyberbullying) dataset. It was presented by Ritesh Kumar et al. as a component of the TRAC-1 shared task. [11][12].

The dataset consists of comments in English and Hindi, labeled for different types of aggression. The data was annotated by multiple annotators to ensure reliability. Each comment received a label based on its content, with particular attention to the presence and type of aggressive language. The TRAC-1 dataset has been widely used in research related to:

- Detection of aggressive and abusive language
- Cyberbullying detection
- Sentiment analysis
- Natural Language Processing (NLP) in chats.

Example Data

Here is a small example of what the dataset might look like.

Sr.no	Text	Label
1	This is an example of an aggressive comment.	OAG
2	I think this could be improved a bit.	CAG
3	What a beautiful day!	NAG

The dataset is obtainable through a number of academic and research channels and is freely available for use for research. Researchers are encouraged to use this dataset to develop and evaluate models for aggression detection and related tasks.

Preprocessing Steps:

In our work, we focused on preprocessing the text data to prepare it for machine learning models aimed at detecting aggression. Below are the detailed steps we took for preprocessing the dataset:

1. Handled missing values by filling empty strings in the 'text' column.
2. Cleaned and normalized text by:
   - Lowercasing all text.
   - Removing URLs, user mentions, hashtags, punctuation, and digits.
3. Conducted sentiment analysis using VADER by Vinodini [13] and added compound sentiment scores to the dataset.
4. Counted emojis in each text message for emotional and contextual information.
5. Mapped categorical labels (NAG, CAG, OAG) to numerical values (0, 1, 2) for model training.
6. Compiled processed features into a new Data Frame and saved it as a CSV file for further use.

IMPLEMANTATION THROUGH SVM ALGORITHM :

Support Vector Machine (SVM) Algorithm

A supervised learning algorithm called Support Vector Machine (SVM) is used to carry out tasks such as classification and regression. It is especially well-known for how well it works in high-dimensional environments and how effectively it can tackle classification issues that are both linear and non-linear.

Algorithm:

BEGIN

1. Load Dataset

2. Sample Data: IF rows > 1000 THEN sampled_df = random 1000 rows ELSE sampled_df = dataset

3. Feature Extraction: INIT TfidfVectorizer(max_features=500) TRANSFORM 'cleaned_text' with TF-IDF

4. Combine Features: features = combine TF-IDF, sentiment, emoji counts

5. Impute Missing Values: INIT SimpleImputer(strategy='mean')

6. Split Data: SPLIT data: 80% train, 20% test

7. Train & Tune Model: INIT SVM, DEFINE param_grid (C, gamma, kernel), INIT GridSearchCV(SVM, param_grid, cv=3, scoring='f1_macro'), FIT GridSearchCV on training data, best_svm = best estimator from GridSearchCV

8. Evaluate Model: PREDICT test labels with best_svm, COMPUTE classification report, confusion matrix, accuracy, precision, recall, F1, PRINT metrics in table

9. Print Fitting Info

10. Define Prediction Function: DEFINE predict_sentiment(input_chat): TRANSFORM input_chat with TF-IDF, COMBINE features with mean sentiment, emoji count, IMPUTE missing values, PREDICT sentiment with best_svm, RETURN sentiment

11. Interactive Input Loop: WHILE True: PROMPT user for chat message (or 'exit'), IF user_input == 'exit' THEN PRINT "Exiting." BREAK, result = predict_sentiment(user_input), IF result =0 PRINT "No aggression." ELSE IF result = 1 THEN PRINT "Indirect aggression." ELSE IF result = 2 PRINT "Direct aggression."

END

IMPLEMANTATION THROUGH GBM ALGORITHM :

A machine learning method called Gradient Boosting Machine (GBM) is applied to issues related to regression and classification. It gradually constructs an ensemble of weak learners (usually decision trees), with each new model fixing mistakes in the earlier ones. The model concentrates on the samples that were incorrectly categorized in the prior iterations in each iteration. The final prediction is calculated by combining the estimates of all weak learners. GBM is known for its high predictive accuracy and ability to handle complex datasets.

Algorithm:

BEGIN

1. Load Processed Dataset

2. Sample Data
   IF rows > 10000 THEN sampled_df = random 10000 rows ELSE sampled_df = dataset

3. Feature Extraction
   INIT TfidfVectorizer(max_features=500)
   TRANSFORM 'cleaned_text' with TF-IDF

4. Combine Features: TF-IDF, sentiment scores, emoji counts

5. Impute Missing Values: INIT SimpleImputer(strategy='mean')

6. Split Data: features (X), target (y)

7. Model to be trained: INIT GradientBoostingClassifier with 100 estimators, 0.1 learning rate, and 3 maximum depth Model fit to X and Y

8. Evaluate Model

PREDICT labels for training data
PRINT classification report, accuracy, precision, confusion matrix, recall, F1 score

9. Adjust Sentiment Categories

10. Print Fitting Info

11. Define Prediction Function
DEFINE predict_sentiment(input_chat):
TRANSFORM with TF-IDF, COMBINE features, IMPUTE missing values, PREDICT with model, RETURN sentiment

12. Interactive Input Loop
WHILE True:
PROMPT user for chat message (or 'exit')
IF user_input == 'exit' THEN PRINT "Exiting." BREAK
result = predict_sentiment(user_input)
IF result == 0 THEN PRINT "No aggression."
ELSE IF result == 1 THEN PRINT "Indirect aggression."
ELSE IF result == 2 THEN PRINT "Direct aggression."

END

IMPLEMENTATION OF XBOOST ALGORITHM:

XBOOST Algorithm:
The machine learning technique known as "Extreme Gradient Boosting," is a scalable and effective gradient boosting implementation. XGBoost offers various improvements over the conventional gradient boosting architecture, such as:

Regularization: To reduce overfitting and enhance model generalization, it incorporates L1 (Lasso) and L2 (Ridge) regularization.

Tree Pruning: XGBoost employs a more sophisticated tree pruning method (max_depth parameter) than standard boosting, stopping the growth of trees when no further improvements can be made.

Missing Value Handling: XGBoost is robust for datasets with partial data since it has internal mechanisms for handling missing values.

Algorithm:
BEGIN
1. Load Processed Dataset into DataFrame df

2. Handle Missing Values: IF 'cleaned_text' has missing values THEN FILL with null

3. Sample Data: IF rows in df > 10000 THEN sampled_df = random sample of 10000 rows ELSE sampled_df = df

4. Feature Extraction: INIT TfidfVectorizer(max_features=500) TRANSFORM 'cleaned_text' to X_tfidf

5. Combine Features: COMBINE X_tfidf with 'sentiment' and 'emoji_count' from sampled_df to get features

6. Impute Missing Values: INIT SimpleImputer(strategy='mean')

7. Split Data: Sampled_df['label_value'] and X are split into X_train, X_test, y_train, and y_test (test_size=0.2, random_state=42).

8. Train Model: INIT XGBoost Classifier xg_clf (objective='multi:softmax', num_class=3) FIT xg_clf to X_train, y_train

9. Model Evaluation: PREDICT y_pred for X_test COMPUTE AND PRINT classification report, confusion matrix, accuracy, precision, recall, F1 score

10. Define Prediction Function: DEFINE predict_sentiment(input_chat): TRANSFORM input_chat, CREATE input_features_df, ADD 'sentiment' and 'emoji_count', IMPUTE missing values, PREDICT sentiment using xg_clf, RETURN predicted sentiment

11. Interactive Input Loop: WHILE True: PROMPT user for chat message (or 'exit') IF user_input == 'exit' THEN PRINT "Exiting." BREAK CALL predict_sentiment with user_input, PRINT result

END

3. Results and Discussion

Based on the Execution, here's a comparison of the performance of SVM, GBM, and XGBoost:

Metric	SVM	GBM	XGBoost
Accuracy	0.4300	0.5830	0.4885
Precision (average)	0.4256	0.6202	0.4995
Recall (average)	0.4228	0.5702	0.4910
F1 Score (average)	0.4162	0.5735	0.4839
Training Time (average)	21.5 seconds	21.5 seconds	21.5 seconds

Key observations:

- **GBM achieved the highest overall accuracy (0.5830) and the best F1 score (0.5735).** This indicates that GBM was the most effective model in capturing the relationship between the features and the target variable.
- **SVM had the lowest accuracy (0.4300) and F1 score (0.4162).** This suggests that SVM might not have

been suitable for this particular dataset or hyperparameter tuning might be needed to improve its performance.

- **XGBoost achieved a moderate accuracy (0.4885) and F1 score (0.4839).** Its performance falls between SVM and GBM.

Additional considerations:

- The training time for all three models seems to be similar (around 21.5 seconds on average).
- It's crucial to remember that the limited set of hyperparameters used for each model is the foundation for these findings. Tuning the hyperparameters could potentially improve the performance of all models.
- The optimal model selection is contingent upon the particular demands of your assignment. GBM might be the greatest option if achieving high accuracy is the top priority. If interpretability is important, then SVM might be preferred (although XGBoost can also be somewhat interpretable).
- Overall, GBM appears to be the strongest performer in this case. However, depending on your specific needs and the characteristics of your data, a different model might be more suitable.

CONCLUSION AND FUTURE SCOPE

The study shows how machine learning approaches can be used to detect aggressiveness in multilingual social media texts.Using machine learning algorithms SVM, GBM, and XGBoost algorithms, the study found that GBM outperformed other models, achieving the highest F1 score of 0.5735. The implementation emphasizes the importance of feature engineering and supplementary linguistic features to enhance model performance. The successful implementation of these models can contribute to the development of automated content moderation systems, fostering safer online environments. Improved feature development, incorporating user behavior analysis, implementing real-time systems, expanding the dataset to include more languages and cultural contexts, combining machine learning and deep learning techniques, ensuring transparency, addressing ethical and privacy concerns, and integrating detection systems across multiple platforms are all areas that future research should investigate.

Conflicts of interest: The author stated that no conflicts of interest.

References

1. Constantin Or˘asan, "Aggressive language identification using word embeddings and sentiment features," Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying, pages 113–119, Santa Fe, USA, August 25, 2018.
2. Laura P. Del Bosque and Sara E. Garza, "Aggressive text detection for cyberbullying," A. Gelbukh et al. (Eds.): MICAI 2014, Part I, LNAI 8856, pp. 221–232, 2014. Springer International Publishing Switzerland 2014.
3. Dadvar, M., Trieschnigg, D., & de Jong, F. (2013). Improving Cyberbullying Detection with User Context. In *Advances in Information Retrieval* (pp. 693-696). Springer, Berlin, Heidelberg.
4. Pawar S and Pawar P. Detection of Online Harassment on Social Sites Using SVM and KNN. In *Proceedings of the 2018 International Conference on Computing Communication Control and Automation (ICCUBEA)*, 2018, (pp. 1-5). IEEE.
5. Zhao, Y., Zhou, J., & Lee, K. Detecting Toxic Comments Using Neural Networks with Word Embeddings. In *Proceedings of the 2019 IEEE International Conference on Big Data (Big Data)* (pp. 5952-5954). IEEE, 2019.
6. Rizwan M, Ahmad M, and Qamar A. Abusive Language Detection in Social Media Using XGBoost. In *Proceedings of the 2020 3rd International Conference on Emerging Trends in Engineering, Sciences and Technology (ICEEST)*, 2020, (pp. 1-5). IEEE.
7. Nandhini M and Sheeba J. Online Social Network Bullying Detection Using Intelligence Techniques. *Procedia Computer Science*, 2015, *45*, 485-492.
8. Park JH and Fung P. One-step and Two-step Classification for Abusive Language Detection on Twitter. In *Proceedings of the First Workshop on Abusive Language Online*, 2017. (pp. 41-45).
9. Nobata C, Tetreault J, Thomas A, Mehdad Y and Chang Y. Abusive Language Detection in Online User Content." *Proceedings of the 25th International Conference on World Wide Web (WWW '16)*, 2016.
10. Si S, Datta A, Banerjee S and Naskar SK. Aggression Detection on Multilingual Social Media," in Proceedings of the 10th International Conference on Computing,

Communication and Networking Technologies (ICCCNT), Kanpur, India, July 2019, pp. 1-6.

11. Kumar R, Reganti AN, Bhatia A & Maheshwari T. Aggression-annotated Corpus of Hindi-English Code-mixed Data. In *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying*, 2018, (pp. 1-11).
12. Ritesh Kumar, Guggilla Bhanodai, Rajendra Pamula and MR. Chennuru, TRAC-1 Shared Task on Aggression Identification: IIT(ISM)@COLING'18, Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying, pages 58–65 Santa Fe, USA, August 25, 2018.
13. Vinodini S, "Analyzing Sentiments in Paulo Coelho's Literary Works Using VADER Sentiment Analysis," 2023 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES)