Journal of International Multidisciplinary Research

Preliminary Evaluation of Gaussian Naive Bayes for Multi-Label Hate Speech and Abusive Language Detection on Indonesian Twitter

Tri Pratiwi Handayani[1], Wahyudin Hasyim[2], Nursetia Wati[3]
Universitas Muhammadiyah Gorontalo [1], Gorontalo, Indonesia
Politeknik Negeri Gorontalo [3], Gorontalo, Indonesia
tripratiwi@umgo.ac.id

Informasi Artikel
E-ISSN : 3026-6874
Vol: 1 No: 2 Nov 2023
Halaman : 159-165

Keywords:
Gaussian Naïve Bayes
Hate speech
Cyberbulling
TF-IDF, BERT

Abstract

Automatic detection of hate speech and abusive language is crucial for combating online toxicity. This study explores Gaussian Naive Bayes for multi-label classification of hate speech on Indonesian Twitter, including target, category, and level. We combined TF-IDF features with contextual BERT embeddings. The model achieved balanced performance for general hate speech and good non-abusive language detection. However, it exhibited limitations with imbalanced data and specific hate speech types. The classifier consistently favored the majority class (non-hateful/non-abusive) across labels, particularly struggling with HS_Gender, HS_Physical, etc. This suggests difficulty detecting less frequent but potentially severe hate speech, likely due to limited training data. Overall accuracy and F1-scores confirm that while Gaussian Naive Bayes is efficient, it lacks robustness for nuanced multi-label classification with imbalanced datasets. This necessitates exploring alternative approaches for effectively detecting specific and less frequent hate speech.

Abstract

Ujaran kebencian di media sosial perlu dideteksi secara otomatis untuk menjaga interaksi sosial mejadi lebih aman. Ujaran kebencian memiliki target, kategori, dan tingkatan yang perlu dideteksi untuk membantu pihak berwenang memprioritaskan ujaran kebencian mana yang harus segera ditangani. Penelitian ini membahas penggunaan Gaussian Naive Bayes untuk klasifikasi teks multi-label dalam mendeteksi ujaran kebencian yang memiliki kategori ujaran kebencian, termasuk target, kategori, dan tingkat ujaran kebencian yang bersumber data dari Twitter Indonesia. Proses ekstraksi pada penelitian ini menggunakan fitur hibrida, yang menggabungkan fitur Term Frequency-Inverse Document Frequency (TF-IDF) tradisional dengan embeddings kontekstual yang dihasilkan oleh BERT (Bidirectional Encoder Representations from Transformers). Hasil eksperimen kami, berdasarkan dataset 5000 sampel, menunjukkan bahwa meskipun Gaussian Naive Bayes efisien dan mudah diimplementasikan, algoritma ini mengalami kesulitan yang signifikan pada dataset yang tidak seimbang, menyebabkan kinerja buruk untuk kategori yang jumlahnya sedikit. Secara khusus, klasifikasi mencapai presisi dan recall tinggi untuk kategori mayoritas tetapi menunjukkan presisi rendah dan skor F1 untuk kategori minoritas. Temuan ini menunjukkan bahwa Gaussian Naive Bayes tidak disarankan sebagai model utama untuk tugas ini, dan pendekatan alternatif sebaiknya

Kata Kunci : Gaussian Naïve Bayes, Hate speech, Cyberbulling, TF-IDF, BERT

INTRODUCTION

Hate speech and abusive language on social media platforms can lead to serious societal issues. Automated detection systems are essential to mitigate these risks by identifying and categorizing such harmful content. This research aims to implement and evaluate Gaussian Naive Bayes to detect multi-label hate speech and abusive language on Indonesian Twitter. The detection of hate speech and abusive language on social media has received considerable attention due to its potential to incite violence and social unrest. Traditional approaches such as those by (Davidson et al., 2017; Badjatiya et al., 2017) employed logistic regression and deep learning techniques respectively for binary classification of hate speech. While these methods were effective, they often struggled with the nuanced differences between offensive language and hate speech.

In contrast (Ibrohim & Budi, 2019) focused on multi-label classification using Support Vector Machine (SVM), Naive Bayes (NB), and Random Forest Decision Tree (RFDT) classifiers. They used Binary Relevance (BR), Label Power-set (LP), and Classifier Chains (CC) for data transformation, finding that RFDT with LP provided the best accuracy. Feature engineering has proven crucial in enhancing classifier performance. (Waseem & Hovy, 2016) demonstrated the importance of n-grams, part-of-speech tags, and user demographics, while (Zhang & Luo, 2019) showed that contextual embeddings like BERT and ELMo significantly improve detection by capturing semantic context. Additionally, (Chen et al., 2018) integrated sentiment analysis and lexicon-based features, further enhancing model performance.

Hybrid machine learning methods, particularly ensemble techniques, have shown promise in improving classification accuracy. (Xu et al., 2012) and (Wang et al., 2017) highlighted the benefits of combining multiple classifiers through methods like stacking, boosting, and bagging, which leverage the strengths of individual models.

Despite these advancements, gaps remain in multi-label classification and the application of ensemble methods in hate speech detection. This research aims to address these gaps by implementing Gaussian Naive Bayes with advanced feature engineering. Additionally, ensemble methods such as stacking will be employed to improve overall detection performance and robustness. By building upon the work of Ibrohim and Budi (2019) and integrating advanced methodologies, this study seeks to provide a preliminary evaluation of Gaussian Naïve Bayes for multi-label hate speech and abusive language detection on Indonesian

METHOD

Before diving into the modeling and evaluation process, it is essential to prepare the dataset properly to ensure accurate and reliable results. The preliminary steps include data exploration and preprocessing, which lay the foundation for building robust models.

Step 1: Data Exploration and Preprocessing

The first step involves loading the dataset to understand its structure and contents, which helps in identifying the various features and labels present in the data. This process includes several key steps:

1. Loading the Data: Load the dataset (`data.csv`) to understand its structure and contents.
2. 2.Text Normalization: Normalize slang and typo words using `new_kamusalay.csv`. This step ensures that informal and incorrectly spelled words are converted to their standard forms, improving the consistency of the text data.
3. Data Cleaning: Remove or replace usernames and URLs with placeholders to maintain privacy and focus on the textual content. This step helps in generalizing the text data and removing any specific identifiers.

Step 2: Feature Extraction

Text Vectorization is implemented using Term Frequency (TF) and Term Frequency-Inverse Document Frequency (TF-IDF) vectorization. The aim is to convert the text data into numerical features that can be used by machine learning algorithms. Additionally, BERT embeddings are generated to capture the contextual semantics of the text. These embeddings are combined with TF-IDF features to enhance the feature set.

Step 3: Model Training and Evaluation

The next step is to split the data into training and testing sets to evaluate the model's performance. The training set is used to train the model, while the testing set is used to evaluate its performance.

Training and Evaluating Classifiers with Combined Features consist of three steps:

1. Split the Data: Split the dataset into training and testing sets to evaluate the model's performance effectively.
2. Train Classifiers: Train Gaussian Naive Bayes classifier on the training data. This classifier is selected for its efficiency and ability to handle high-dimensional data.
3. Evaluate Classifiers: Evaluate the performance of the classifier using metrics such as precision, recall, and F1-score. These metrics provide a comprehensive understanding of the model's ability to correctly identify instances of hate speech and abusive language.

Step 4: Enhanced Feature Engineering with Contextual Embeddings

In the context of this research, incorporating contextual embeddings such as those generated by BERT (Bidirectional Encoder Representations from Transformers) plays a crucial role in enhancing the model's understanding and detection capabilities of hate speech and abusive language. Contextual embeddings represent a significant advancement over traditional text representation methods, enabling the model to capture the meaning of words within their specific context, thereby improving accuracy and robustness.

The model was run in Google Colab with a configuration set to Python 4 and a T4 GPU runtime. The code is described in Appendix 1.

RESULT AND DISCUSSION

This study enhanced the feature extraction process by incorporating contextual embeddings using BERT alongside traditional TF-IDF features. The combination of these features provided a richer representation of the text data, capturing both the context-dependent meanings of words and their frequency-based importance. Table 1 shows the result of evaluating the Gaussian Naive Bayes classifier on the dataset of 5000 data points.

Tabel 1. Evaluation Matrix

Label	Precision (Class 0)	Precision (Class 1)	Recall (Class 0)	Recall (Class 1)	F1-score (Class 0)	F1-score (Class 1)	Support (Class 0)
HS (Hate Speech)	0.73	0.54	0.62	0.67	0.67	0.6	0.594
Abusive	0.78	0.6	0.7	0.69	0.74	0.64	0.605
HS_Individual	0.81	0.39	0.73	0.51	0.77	0.44	0.745
HS_Group	0.86	0.22	0.82	0.27	0.84	0.24	0.849
HS_Religion	0.95	0.17	0.95	0.2	0.95	0.19	0.946
HS_Race	0.96	0.18	0.97	0.15	0.96	0.16	0.953
HS_Physical	0.98	0.6	0.97	0.1	0.97	0.7	0.979
HS_Gender	0.98	0.9	0.97	0.16	0.98	0.11	0.981
HS_Other	0.8	0.42	0.74	0.5	0.77	0.46	0.725

HS_Weak	82	38	75	48	78	43	755
HS_Moderate	88	19	83	25	86	21	868
HS_Strong	98	25	98	17	98	2	971

The model performance metrics for detecting hate speech (HS) and abusive language demonstrate varying levels of effectiveness across different categories. For general hate speech, the model shows balanced performance with a precision of 0.73 for non-hate speech (Class 0) and 0.54 for hate speech (Class 1), and recall values of 0.62 for Class 0 and 0.67 for Class 1. This indicates the model is somewhat more effective at detecting hate speech but tends to produce some false positives. In detecting abusive language, the model achieves higher precision for non-abusive content (0.78) compared to abusive content (0.60), with balanced recall values of 0.70 for non-abusive and 0.69 for abusive instances, suggesting relatively good performance in identifying abusive language.

The classifier consistently shows high performance for the majority class (Class 0) but struggles significantly with the minority class (Class 1), an imbalance evident across most labels. Certain labels, such as HS_Gender, HS_Physical, HS_Religion, HS_Race, and HS_Strong, exhibit very poor performance for Class 1, indicating that the model fails to detect these specific types of hate speech. Notably, the model has significant difficulty in detecting hate speech related to physical attributes and gender, likely due to very low support for these classes. The overall accuracy and F1-scores indicate that while Gaussian Naive Bayes can provide a quick baseline, it is not robust enough for nuanced multi-label classification tasks involving imbalanced data. Overall, while the model performs well in certain areas, there is room for improvement in detecting specific and less frequent types of hate speech.

CONCLUSION

Gaussian Naive Bayes shows limitations in handling imbalanced datasets and fails to detect underrepresented classes effectively. While it provides a quick baseline, alternative approaches should be considered for more robust and accurate multi-label hate speech and abusive language detection on social media platforms.

RECOMMENDATION

To address the challenges of class imbalance and improve the detection of minority classes in hate speech and abusive language classification, several strategies can be implemented. First, techniques like SMOTE (Synthetic Minority Over-sampling Technique) or ADASYN (Adaptive Synthetic Sampling) can be used to balance the dataset and enhance minority class detection. Second, exploring more advanced classifiers such as Random Forest, SVM, or neural networks could provide better handling of class imbalance. Third, employing ensemble methods like stacking, boosting, or bagging can leverage the strengths of multiple classifiers to improve overall performance. Lastly, continuing to enhance feature extraction methods by incorporating additional contextual or semantic features could further improve the model's accuracy and robustness in detecting nuanced and less frequent types of hate speech and abusive language.

REFERENCES

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

Chen, Z., Zhou, Y., & Zou, Y. (2018). Integrating Sentiment Features and Word Embeddings for Sentiment Analysis. Journal of Information Science and Engineering, 34(5), 1237–1250.

Davidson, T., Warmsley, D., Macy, M., & Weber, I. (2017). Automated Hate Speech Detection and the Problem of Offensive Language. Proceedings of the 11th International AAAI Conference on Web and Social Media (ICWSM).

Ibrohim, M. O., & Budi, I. (2019). Multi-label Hate Speech and Abusive Language Detection in Indonesian Twitter. ALW3: 3rd Workshop on Abusive Language Online, 46–57. https://www.aclweb.org/anthology/W19-3506.pdf

Wang, B., Peng, T., Yang, J., & Sun, H. (2017). Stacking-Based Ensemble Learning for Sentiment Classification of Chinese Microblogs. Neurocomputing, 214, 708–718.

Waseem, Z., & Hovy, D. (2016). Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter. Proceedings of the NAACL Student Research Workshop.

Xu, W., Liu, X., & Gong, Y. (2012). Document Clustering Based on Non-negative Matrix Factorization. Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

Zhang, Z., & Luo, L. (2019). Hate Speech Detection: A Solved Problem? The Challenging Case of Long Tail on Twitter. Semantic Web, 10(5), 925–945.

Appendix 1

```
# Install Required Libraries
!pip install transformers torch pandas scikit-learn
# Import Libraries
import pandas as pd
import numpy as np
import re
from sklearn.feature_extraction.text import TfidfVectorizer
from transformers import BertTokenizer, BertModel
from sklearn.model_selection import train_test_split
from sklearn.multioutput import MultiOutputClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import classification_report
import torch
# Upload your data
from google.colab import files
uploaded = files.upload()
# Load the Data with specified encoding
data = pd.read_csv('data.csv', encoding='latin1')
# Sample 50 data points from the dataset
data_sampled = data.sample(n=5000, random_state=42)
# Load the normalization dictionary with specified encoding
kamusalay = pd.read_csv('new_kamusalay.csv', encoding='latin1', header=None, names=['slang',
'formal'])
kamusalay_dict = pd.Series(kamusalay.formal.values, index=kamusalay.slang).to_dict()
def normalize_text(text):
  words = text.split()
```

```
    normalized_words = [kamusalay_dict.get(word, word) for word in words]
    return ' '.join(normalized_words)

data_sampled['Normalized_Tweet'] = data_sampled['Tweet'].apply(normalize_text)

# Data Cleaning
def clean_text(text):
    text = re.sub(r'USER', '<USER>', text)
    text = re.sub(r'URL', '<URL>', text)
    return text

data_sampled['Cleaned_Tweet'] = data_sampled['Normalized_Tweet'].apply(clean_text)

# Text Vectorization
tfidf_vectorizer = TfidfVectorizer(max_features=5000)  # Adjust the number of features as needed
tfidf_features = tfidf_vectorizer.fit_transform(data_sampled['Cleaned_Tweet'])

# BERT embeddings
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
model = BertModel.from_pretrained('bert-base-uncased')

def get_bert_embeddings(text):
    inputs = tokenizer(text, return_tensors='pt', max_length=512, truncation=True, padding=True)
    outputs = model(**inputs)
    return outputs.last_hidden_state.mean(dim=1).detach().numpy()

bert_embeddings = np.vstack([get_bert_embeddings(tweet) for tweet in
data_sampled['Cleaned_Tweet']])

# Combine TF-IDF and BERT embeddings
combined_features = np.hstack((tfidf_features.toarray(), bert_embeddings))

# Split data into training and testing sets
target_labels = ['HS', 'Abusive', 'HS_Individual', 'HS_Group', 'HS_Religion', 'HS_Race', 'HS_Physical',
'HS_Gender', 'HS_Other', 'HS_Weak', 'HS_Moderate', 'HS_Strong']
X_train, X_test, y_train, y_test = train_test_split(combined_features, data_sampled[target_labels],
test_size=0.2, random_state=42)

# Train Gaussian Naive Bayes classifier with MultiOutputClassifier for multi-label classification
gnb_classifier = MultiOutputClassifier(GaussianNB())
gnb_classifier.fit(X_train, y_train)

# Evaluate Gaussian Naive Bayes classifier
gnb_predictions = gnb_classifier.predict(X_test)

for i, label in enumerate(target_labels):
    print(f"Label: {label}")
```

```
    print("Gaussian Naive Bayes Classification Report:\n", classification_report(y_test.iloc[:, i],
gnb_predictions[:, i], zero_division=0))
```

The source of data set can be download ini https://github.com/okkyibrohim/id-multi-label-hate-speech-and-abusive-language-detection