applied sciences

MDPI

Article

Automated Text Annotation Using a Semi-Supervised Approach with Meta Vectorizer and Machine Learning Algorithms for Hate Speech Detection

Shoffan Saifullah [1,2,*], Rafał Dreżewski [1,3], Felix Andika Dwiyanto [1,4], Agus Sasmito Aribowo [2], Yuli Fauziah [2] and Nur Heri Cahyana [2]

[1] Faculty of Computer Science, AGH University of Krakow, 30-059 Krakow, Poland; drezew@agh.edu.pl (R.D.); dwiyanto@agh.edu.pl (F.A.D.)
[2] Department of Informatics, Universitas Pembangunan Nasional Veteran Yogyakarta, Yogyakarta 55283, Indonesia; sasmito.skom@upnyk.ac.id (A.S.A.); yuli.fauziah@upnyk.ac.id (Y.F.); nur.hericahyana@upnyk.ac.id (N.H.C.)
[3] Artificial Intelligence Research Group (AIRG), Informatics Department, Faculty of Industrial Technology, Universitas Ahmad Dahlan, Yogyakarta 55166, Indonesia
[4] Department of Electrical Engineering, Universitas Negeri Malang, Malang 65145, Indonesia
* Correspondence: shoffans@upnyk.ac.id or saifulla@agh.edu.pl

Abstract: Text annotation is an essential element of the natural language processing approaches. The manual annotation process performed by humans has various drawbacks, such as subjectivity, slowness, fatigue, and possibly carelessness. In addition, annotators may annotate ambiguous data. Therefore, we have developed the concept of automated annotation to get the best annotations using several machine-learning approaches. The proposed approach is based on an ensemble algorithm of meta-learners and meta-vectorizer techniques. The approach employs a semi-supervised learning technique for automated annotation to detect hate speech. This involves leveraging various machine learning algorithms, including Support Vector Machine (SVM), Decision Tree (DT), K-Nearest Neighbors (KNN), and Naive Bayes (NB), in conjunction with Word2Vec and TF-IDF text extraction methods. The annotation process is performed using 13,169 Indonesian YouTube comments data. The proposed model used a Stemming approach using data from Sastrawi and new data of 2245 words. Semi-supervised learning uses 5%, 10%, and 20% of labeled data compared to performing labeling based on 80% of the datasets. In semi-supervised learning, the model learns from the labeled data, which provides explicit information, and the unlabeled data, which offers implicit insights. This hybrid approach enables the model to generalize and make informed predictions even when limited labeled data is available (based on self-learning). Ultimately, this enhances its ability to handle real-world scenarios with scarce annotated information. In addition, the proposed method uses a variety of thresholds for matching words labeled with hate speech ranging from 0.6, 0.7, 0.8, to 0.9. The experiments indicated that the DT-TF-IDF model has the best accuracy value of 97.1% with a scenario of 5%:80%:0.9. However, several other methods have accuracy above 90%, such as SVM (TF-IDF and Word2Vec) and KNN (Word2Vec), based on both text extraction methods in several test scenarios.

Keywords: hate speech detection; machine learning; sentiment analysis; semi-supervised learning; self-learning; text mining

Citation: Saifullah, S.; Dreżewski, R.; Dwiyanto, F.A.; Aribowo, A.S.; Fauziah, Y.; Cahyana, N.H. Automated Text Annotation Using a Semi-Supervised Approach with Meta Vectorizer and Machine Learning Algorithms for Hate Speech Detection. *Appl. Sci.* **2024**, *14*, 1078. https://doi.org/10.3390/app14031078

Academic Editor: Valentino Santucci

Received: 13 November 2023
Revised: 20 January 2024
Accepted: 23 January 2024
Published: 26 January 2024

1. Introduction

Hate speech detection is based on the concept of Natural Language Processing (NLP) and sentiment analysis [1,2]. NLP uses the concept of text mining to perform processing of the specific text [3]. The process requires several stages, one of which is text annotation, so the training process requires data that has been labeled. Text annotation done manually

by humans has various disadvantages, such as being expensive, prone to errors, inconsistent labeling, subjectivity, and difficulties in processing large datasets. These factors occur because the person doing the annotation has different shortcomings, such as fatigue, pressure of circumstances and environment, the influence of workload, and the subjectivity. In addition, if many people do the annotation, they must equalize the perception (understanding of words and mindset) when annotating. This process affects the consistency of the annotated labels. These problems can be solved by applying the concept of automatic text annotation using a machine-learning approach. Thus, we developed an automatic annotation model using several machine learning methods such as SVM, DT, KNN, and NB for hate speech detection based on data from social media (YouTube). The data used can impact the learning process and classification of statements as hate speech or not.

Automatic text annotations can detect hate speech by applying machine learning methods with a semi-supervised learning approach [4,5]. Hate speech data are annotated using two categories (hate and not hate) [6–10], and using sentiment analysis methods, in which data are labeled using two or three categories, namely (positive and negative) [11,12], or (positive, negative, and neutral) [13–16]. We develop automatic annotations by utilizing a dataset with minimal labeled training data and incorporate self-learning for labels. This approach draws inspiration from AraSenCorpus modeling [17], which employs a semi-supervised framework (long short-term memory with FastText and machine learning) sentiment Analysis in Arabic. In addition, annotations can extend to sentiment and emotion detection through SVM, Random Forest (RF), and NB [17,18]. In annotating Turkish text, various methods (such as NB, convolutional neural networks (CNN), Bi-LSTM, and SVM) have demonstrated effectiveness [19].

This research aims to develop the concept of automated text annotations that apply semi-supervised learning and self-learning. The most important original contributions of this research are as follows:

1. Machine learning modeling uses the meta-vectorizer and meta-classifier methods to determine the best model;
2. Carrying out experiments with self-learning scenarios using thresholds of 0.6, 0.7, and 0.8 in the proposed model and labeled datasets with the proportion of 5%, 10%, and 20%, and with a training data to test data ratio of 80%;
3. Experimental evaluation of the machine learning methods proposed in this study and their comparisons based on the abovementioned scenarios.

This study conducted experiments to evaluate the machine learning model based on the meta-vectorizer and meta-classifier concepts proposed herein. The proposed models have different accuracy results in each of the proposed scenarios. The minimally labeled data section (5%) yielded the highest accuracy for the proposed model, indicating that this proportion is the most effective.

The research presented in this paper is organized as follows. In Section 2, we describe the dataset and machine learning methods used in the NLP-based approach for hate speech detection. Section 3 discusses the experiments and results obtained based on the proposed scenarios. Finally, Section 4 provides conclusions of the conducted research.

2. Materials and Methods

This research focuses on machine learning algorithms performing automatic annotations using a semi-supervised learning approach. The proposed approach also identifies hate speech based on text processing. The research process was generally carried out using steps presented in Figure 1.

Based on Figure 1, the semi-supervised hate speech annotation process begins by reading the expert's hate-speech annotated training data (Data) in Step 1. Step 2 is reading the data without annotation (UD). Step 3 is the text preprocessing of the Data and UD datasets. Step 4 is the meta-vectorization process. The meta-vectorization process converts the clean Data and UD datasets into two types of vectorization (TF-IDF: Term Frequency-Inverse Document Frequency and Word2Vec: Word Embeddings word to vector). Step

5 is the process of preparing the training data. Step 6 is the preparation of machine learning algorithms, namely Support Vector Machine (SVM), Decision Tree (DT), K-Nearest Neighbors (KNN), and Naive Bayes (NB). Step 7 is the creation of the meta-learning model. Meta-vector and meta-learning models will produce eight different combinations of vectorization and machine learning methods. Each combination of machine learning methods will be used for the auto-annotation process. Steps 8 and 9 form the process of preparing the dataset of vectors without annotations. In step 10, it will be checked if there are still datasets that have not been annotated and then the process will proceed to step 11, which is the process of annotating the dataset.

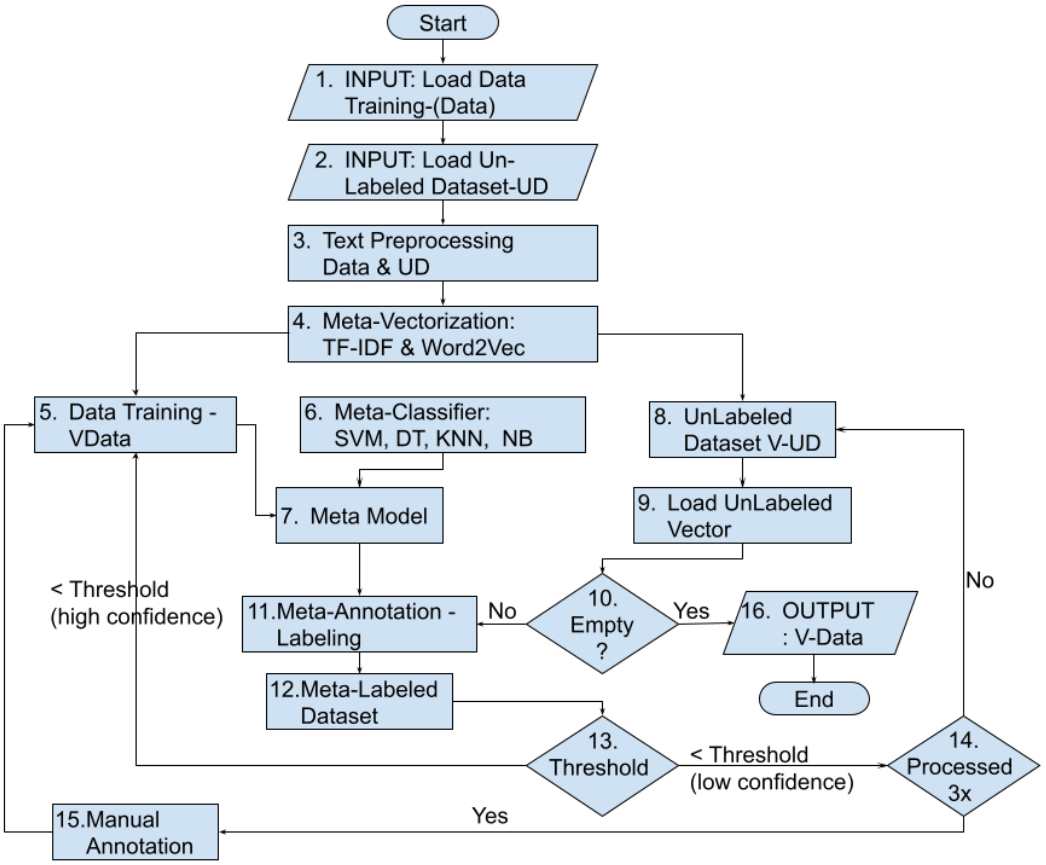


Figure 1. The model for hate speech detection using a semi-supervised learning approach based on text annotations.

The annotation process is done by predicting labels using eight different combinations of these vectors and machine learning methods. The prediction results are also validated to determine their accuracy, which results in a meta-labeled dataset from the auto-annotation process in Step 12. In Step 12, a voting process will be carried out to determine the label for each dataset record. There are two types of voting, namely the total score W (weight) of the vectorization-machine-learning model for hate-speech and non-hate speech polarity. In Step 13, the term Threshold refers to a predefined value that serves as a cutoff point for hate-speech or non-hate speech polarity scores. Specifically, if the hate-speech or non-hate speech polarity score exceeds the designated threshold, then the dataset and its annotations will be transferred to the training data. Conversely, if the score falls below the threshold, the dataset will be re-annotated in the subsequent cycle. The threshold values used in our experiments include 0.6, 0.7, 0.8, and 0.9, each representing a different level of stringency in determining hate-speech or non-hate speech polarity. In Step 14, the whole cycle will be repeated three times. Subsequently, the remaining Unlabeled Dataset will be passed to Step 15, which is manual annotation. The manual annotation results will be combined with the training data.

2.1. Datasets

This research focuses on hate speech detection using dataset [20] limited to the realm of politics and law in Indonesia. The dataset includes public opinions from YouTube comments on the presidential debate video [5], and opinions about the COVID-19 pandemic [21]. Several reasons justify considering these comments for further research:

1. Inclusive representation of hate speech, non-hate speech, and very negative hate speech;
2. The credibility of the video content from official broadcasting institutions/channels;
3. Fair representation of public opinion in YouTube comments due to the absence of the message length restrictions, censorship, and the possibility of user interactions.

Thus, this research determines that YouTube video comments related to the 2019 Indonesian presidential debate and COVID-19 are relevant dataset sources to build a semi-supervised text annotator model for hate speech. The YouTube channels used in this research are official Indonesian television channels, including Kompas TV, News Metro TV, TV-One, IDN Times, CNN Indonesia, and I-News TV. Viewer comments from videos exhibit the following characteristics:

1. Comments are free and unstructured and contain emoticons, punctuation, and special characters reflecting user sentiments;
2. Varied comments, including direct responses to video contents, replies to other comments, and revealing the user's side;
3. Ambiguous elements, such as satire, polysemy, slang words, stop-words, and metaphors.

This study focuses on six official television channels and logs the volume of opinions and comments per video using a sampling method during the data collection stage.

Population and Sampling

Given the dynamic nature of the dataset, it continuously grows with new videos and viewer opinions of official channels. This study acknowledges the impossibility of downloading all YouTube comments. This limitation arises from the unknown population size. Consequently, purposive sampling was employed for the following reasons:

1. Hate speech was observed in YouTube video comments about the 2019 Indonesian presidential debate and Covid-19, aligning with the research objectives;
2. Publicly available data from YouTube comments could be downloaded free of charge.

We collected comments from the presidential debates 1 to 5, exclusively broadcast by two official channels, to determine the sample size. This collection resulted in comments from 10 debate videos and 5 COVID-19 news videos. Additional comments were downloaded from official channel videos. While not featuring complete debates, these videos warranted significant inclusion. This decision was based on their high views (over 10,000) and comments (over 1000) on compelling topics.

Expert analysis categorized the samples for hate speech, sentiment, and emotions. These results were a reference for automatically annotating population datasets using the Semi-Supervised Text Annotator Model. The annotations' validity was verified using the prepared machine learning model, testing its ability to classify a multi-class dataset, particularly in hate speech categorization (Figure 2).

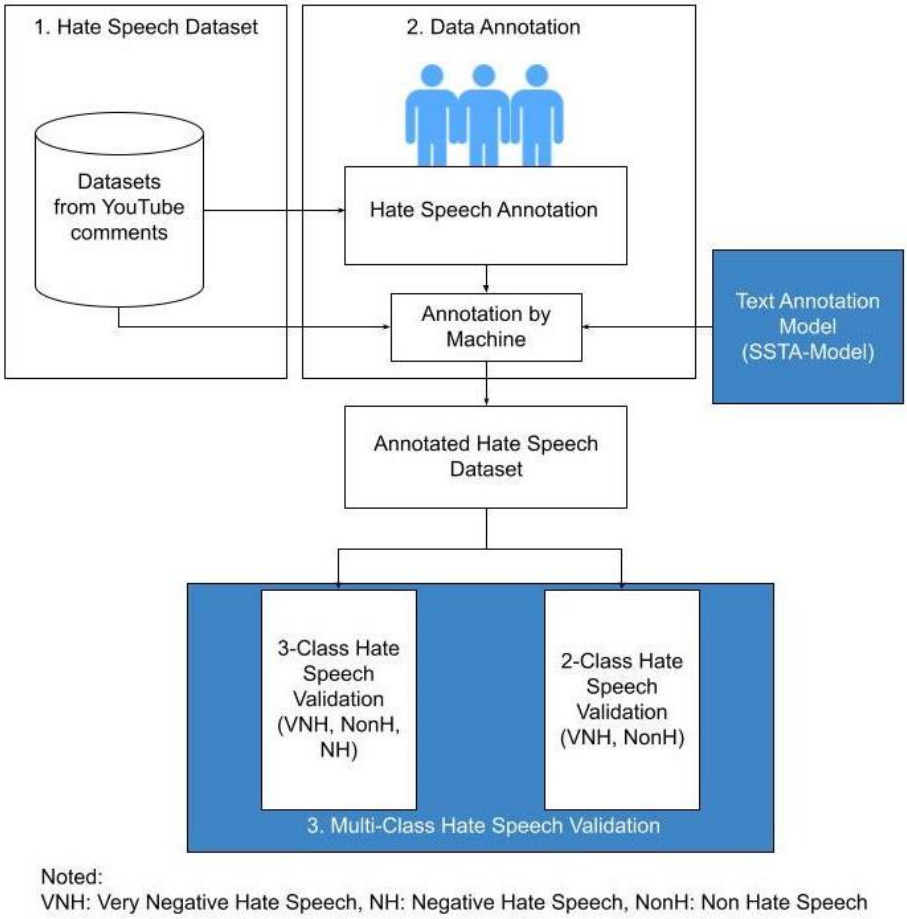


Figure 2. The proposed automatic annotation process in detecting hate speech using a semi-supervised and self-learning approach.

2.2. Pre-Processing

Based on the dataset, pre-processing is crucial to normalize the text data, as highlighted in previous research works [22–24]. Our dataset comprises unstructured data from social media sources, particularly YouTube comments. To enhance the results and structure of this data [25–28], comprehensive pre-processing steps have been applied. In natural language processing (NLP), text pre-processing involves cleaning and organizing textual data for optimal utilization in various NLP tasks. This includes removing unnecessary characters and common stop-words [29], as well as formatting to facilitate analysis by NLP algorithms. Raw text data is often noisy and difficult for algorithms to analyze, and pre-processing improves the accuracy and efficiency of NLP models [30–32]. The steps involved in text pre-processing typically include the following:

1. Cleaning the text—removing unnecessary or irrelevant characters, such as punctuation marks or special characters [33];
2. Tokenization—splitting the text into smaller units, called tokens, such as words or phrases [34];
3. Removing stop-words—common words that lack significant meaning [35], such as "the" or "and", were eliminated to reduce the text size and improve the performance of the NLP model;
4. Stemming and lemmatization—leveraging the Sastrawi library, stemming for the Indonesian language (Bahasa) was performed to reduce words to their base form [36], known as the stem or lemma;
5. Part-of-speech tagging—identifying parts of speech, such as nouns or verbs, aids certain NLP tasks [37];

6. Normalization—formatting the text consistently, converting all words to lowercase. This streamlines processing for the NLP model [38].

Overall, the goal of text pre-processing is to clean and prepare text data for use in NLP tasks to improve the accuracy and efficiency of the NLP model. The specific steps involved in text pre-processing can vary depending on the specific NLP task and the quality of the raw text data.

2.3. Meta-Vectorization Based on Text Feature Extraction

Following preprocessing, the default text serves as input for feature extraction, a pivotal step in transforming data into meaningful features [39]. Our method employs meta-vectorization as a feature extraction concept [40] integrating TF-IDF [21] and Word2Vec feature extraction [41].

2.3.1. Term Frequency-Inverse Document Frequency (TF-IDF)

TF-IDF assigns distinct weights to words [42], accentuating their significance based on the frequency within a document and across the dataset [43]. Its effectiveness in this context is supported by previous studies [24]. $TF\text{-}IDF$ is calculated by multiplying two statistics: term frequency (TF) and inverse document frequency (IDF) [44]. Term frequency indicates how often a particular word appears in a document. It is calculated by dividing the number of word occurrences by the total number of words in the document. This method can be calculated by Equations (1)–(3).

$$TF(t,d) = \frac{c(t,d)}{\Sigma_i c(t_i,d)} \tag{1}$$

$$IDF(t) = 1 + log\frac{D}{d_t} \tag{2}$$

$$TF\text{-}IDF(t,d) = TF(t,d) \cdot IDF(t) \tag{3}$$

The value of $c(t,d)$ indicates the occurrence of the t term in the document d. D represents the total number of documents, and dt is the number of documents containing the term t.

2.3.2. Word Embedding (Word2Vec)

Word2vec is a neural network-based word embedding method [45]. The method involves three layers (input, hidden, and output) and applies two models: skip-gram and Continuous Bag of Words (CBOW). In skip-grams [46], each neuron specializes in comprehending the context around a single target word, while CBOW predicts the target word from context. The activation function is linear Equation (4), and the hidden layer encodes semantic relationships between words Equation (5). The output layer employs softmax to convert outputs into probabilities for accurate prediction Equation (6).

$$h = W^T x \tag{4}$$

$$u_j = W'^T h \tag{5}$$

$$y = \frac{\exp(u_j)}{\sum_{j'=1}^{V} \exp(u'_j)} \tag{6}$$

The h is the hidden layer, and W^T is the transpose of the weight vector (x) where W' is the column. u_j is the output path to the hidden layer. The Soft-max activation function (y_i) uses the output value of u_j according to the number of vocabulary words (V).

2.4. Meta-Classification Using Machine Learning Algorithms

Based on standard machine learning hate detection, we use methods in meta-classification, including SVM, DT, KNN, and NB [47].

2.4.1. Support Vector Machine

Support Vector Machines (SVMs) are supervised learning algorithms for classification and pattern recognition. Employing the principles of structural risk minimization, SVMs utilize linear classifiers to identify hyperplanes that separate different classes. A hyperplane acts as a class separator (support vector), optimizing the hyperplane during training to maximally separate classes. SVM identifies training data points closest to the hyperplane (support vectors) to define the hyperplane and predict new data [48].

SVM implements the Mercer theorem kernel using Equation (7) to optimize quadratic programming with the constraints $a_j \geq 0$ and $\sigma_j(a_j y_j)$. The dot product Equation (8) calculates the pair's effect on the value of w (orthogonal of hyperplane) concerning vector a.

$$\arg\min_{w} \sum_{j} a_j - \frac{1}{2} \sum_{j,k} a_j, a_k y_j y_k (x_j \cdot x_k) \tag{7}$$

$$h(x) = sign\left(\sum_{j} a_j y_j (x \cdot x_j - b)\right) \tag{8}$$

j will consistently hold the value of 0 when the vector is positioned far from its nearest point. In SVM, a training and testing process identifies and annotates hate speech guided by minimal labeling in the training. The process involves addressing the constraints defined by Equation (9) while simultaneously optimizing the value of w and m as specified in Equation (10). These results determine the penalty parameter c and error value (ϵ) based on $lab_j(w.x_i) \leq 1 - \epsilon_j, j = 1, 2, 3, \ldots, n$.

$$y(x) = w \cdot x + m \tag{9}$$

$$minimize\left(\frac{1}{2}|w|^2 + c\sum \epsilon_j\right) \tag{10}$$

2.4.2. Decision Tree (DT)

Decision Tree, a machine learning and data mining algorithm [49], is a predictive model for classifying objects or predicting outcomes based on input characteristics. This hierarchical structure starts with a root node representing the entire dataset and branches representing possible outcomes linked to input feature values. The leaves signify the final classification or prediction, constructed from a relatively simple set of rules. DT algorithm is a supervised learning method for classification and regression tasks. In this method, each inner node represents a "test" attribute, branches depict the test result, and leaves represent class labels. The algorithm traverses the tree, making decisions at each internal input attribute. It concludes with a prediction determined by the leaf node reached.

DT can handle numeric and categorical data, and ensemble DTs enhance performance by combining predictions from multiple trees [50]. Ensemble methods, incorporating techniques such as bagging, boosting, and bootstrapping, use multiple algorithms to achieve better predictive performance than a single algorithm. Ensemble DTs are commonly used in various applications, such as image and text classification, regression, and anomaly detection.

2.4.3. K-Nearest Neighbors (KNN)

K-Nearest Neighbors is a machine learning method for classification and regression in pattern recognition [5]. It operates in real-time by classifying incoming data based on the nearest reference data point in the feature space [51]. It assigns labels by considering characteristics shared with nearby data points, distinguishing test and training data organized into tuples. Each tuple represents a multidimensional point in space, forming a reference set. To determine their classification, the KNN involves comparing uncertain tuples with their nearest neighbors. This process is influenced by the closest K-value neighbors.

In the KNN algorithm [52], distance metrics such as Euclidean, Mahalanobis, and Minkowski Distance assess the similarity or dissimilarity between data points. This study focuses on the Euclidean Distance (ED) formula Equation (1), which measures the dissimilarity between data points in multidimensional coordinates (x_{i_k} and x_{j_k}) within the dataset p. This ED calculation, denoted as $d(i, j)$, is pivotal in KNN-based decision-making processes, impacting various machine learning and pattern recognition applications.

$$d(i,j) = \sqrt{\sum_{k=1}^{p} (X_{i_k} - X_{j_k})^2} \tag{11}$$

Determining the optimal k value involves systematic experimentation, starting with $k = 1$, and iteratively testing to identify the optimal value of k that yields the best results. Alternatively, the meta-learner automates k value selection based on predefined criteria to maximize result accuracy by considering relationships between data points and their neighbors.

2.4.4. Naive Bayes (NB)

Naive Bayes, a family of probabilistic classifiers based on Bayes' theorem, assumes independence between features for simplicity [53]. Class probability is calculated by considering the probability of each feature occurring within that class and dividing it by the probability of the feature itself; see Equation (12). These classifiers are scalable, requiring parameters proportional to the dataset's number of variables, making them suitable for large datasets. NB is commonly used in text classification and NLP tasks, where strong independence assumptions are often met.

Class prior and feature probabilities are computed from the training data using maximum likelihood estimation for NB predictions. Bayes' theorem is then applied to find the posterior probabilities of each class for a given feature set, and the class with the highest posterior probability is chosen for prediction.

An ensemble NB classifier combines the predictions of multiple NB classifiers to improve the performance and accuracy [54]. Ensemble methods use multiple learning algorithms utilizing techniques like bagging, boosting, and bootstrapping to improve prediction performance. The Ensemble NB classifier finds applications in various tasks, including text classification and NLP [55].

$$P(class|feature) = \frac{P(feature|class) \times P(class)}{P(feature)} \tag{12}$$

3. Results

In this section, we present the experimental process and explain the results obtained in this study. The experiments were conducted using datasets from YouTube comments based on the proposed scenarios. This aligns with the goal of self-learning automated annotation processes with minimal datasets to obtain optimal accuracy according to the proposed method. The model performance was assessed using an accuracy measurement.

3.1. Experimental Scenario Setup

In this study, we designed annotation text scenarios using a self-learning approach to detect hate-speech in three-ways (very negative hate-speech, negative hate-speech, and non-hate-speech) as shown in Figure 2. The proposed model uses several machine learning methods to divide the dataset (see Table 1). This scenario aims to obtain optimal annotation predictions using minimal data. This scenario has different variations in the number of labeled datasets used and the threshold that identifies the limits of the similarity of the weights of the tested data.

Table 1. The automatic annotation process scenario uses minimal training data and threshold.

No	Training Data (%)	Unlabeled Data (%)	Threshold
1	20	80	0.6
2	20	80	0.7
3	20	80	0.8
4	20	80	0.9
5	10	90	0.6
6	10	90	0.7
7	10	90	0.8
8	10	90	0.9
9	5	95	0.6
10	5	95	0.7
11	5	95	0.8
12	5	95	0.9

Significantly, our experimental design diverges from prior studies, such as those represented by Refs. [5,56]. In the experiment detailed in Ref. [5], a common scenario was employed with a training and testing proportion of 80:20. On the other hand, the experiment documented in Ref. [56] sought to enhance the quality of the scenario outlined in Ref. [5] by adjusting the proportions to 20:80:80 for training, labeled datasets, and testing, respectively. In contrast, our proposed research introduces novel scenarios characterized by minimal training data and distinct threshold values, as meticulously detailed in Table 1. This strategic departure is motivated by the quest for a more in-depth understanding of hate speech detection, mainly when limited training data and the model's sensitivity to similarity thresholds are explored.

3.2. Experimental Results of the Machine Learning Based Approach

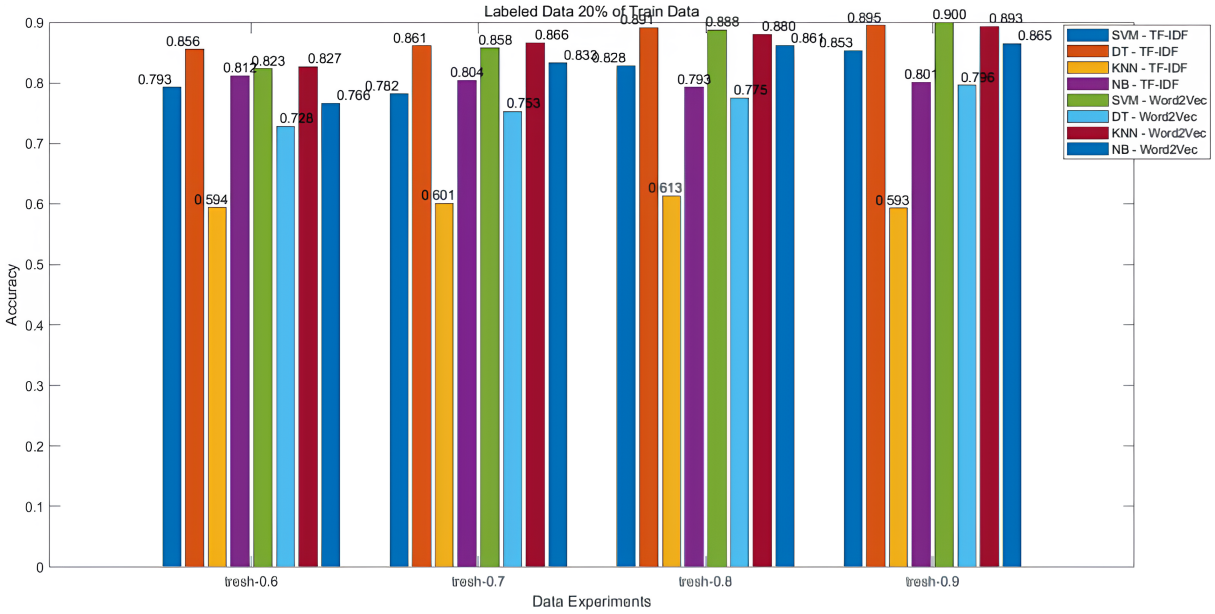
This section presents the results and analysis of machine learning methods' performance with the meta-learner concept based on feature extraction of meta vectorization. We carried out experiments using the processing scenario, as shown in Table 1. The grouping was based on the percentage of training data, namely 20%, 10%, and 5%.

The experiment aimed to determine meta-learners' performance in automatic annotations using the least amount of labeled training data to optimize the annotation process. Thus, the annotation process with a small amount of labeled training data can annotate itself using the application of self-learning. Furthermore, we employ threshold parameters in self-learning to identify and seek similarities within the training dataset. This approach ensures that annotations align closely with the training and testing phases.

The results obtained use accuracy to determine the correctness of the annotation process. Figure 3 presents the results of calculating the accuracy of each combination of machine learning algorithms (SVM, DT, KNN, and NB) with feature extraction method (TF-IDF and Word2Vec). The evaluation uses manual annotations of 20% for 13,169 data. This automatically generated annotation will be used for the initial training.

Figure 3 shows that the machine learning model with 20% initially labeled data scenario is capable of performing automatic annotations with varying accuracy, with the most considerable value being 90% at the threshold of 0.9 obtained by the SVM-Word2Vec method. The accuracy of each individual method is visible on the chart presented in Figure 3. The evaluation showed that the worst machine learning method of all the different scenarios was KNN combined with feature extraction algorithms (TF-IDF and Word2Vec). The other methods have an accuracy of ≈80% and above. The accuracy of the KNN-TF-IDF method has increased as compared to Cahyana et al. [5] from 59.68% to 61.3% (+1.62 p.p.). This enhancement reflects advancements in the preprocessing procedures from prior research and has been validated through various experimental scenarios, as shown in Table 1. In the stemming process, the Indonesian language preprocessing was carried out using the Sastrawi library for Python. In addition, we have also added several stop-words to calculate the average vector features. However, it is worth noting that these stop-words

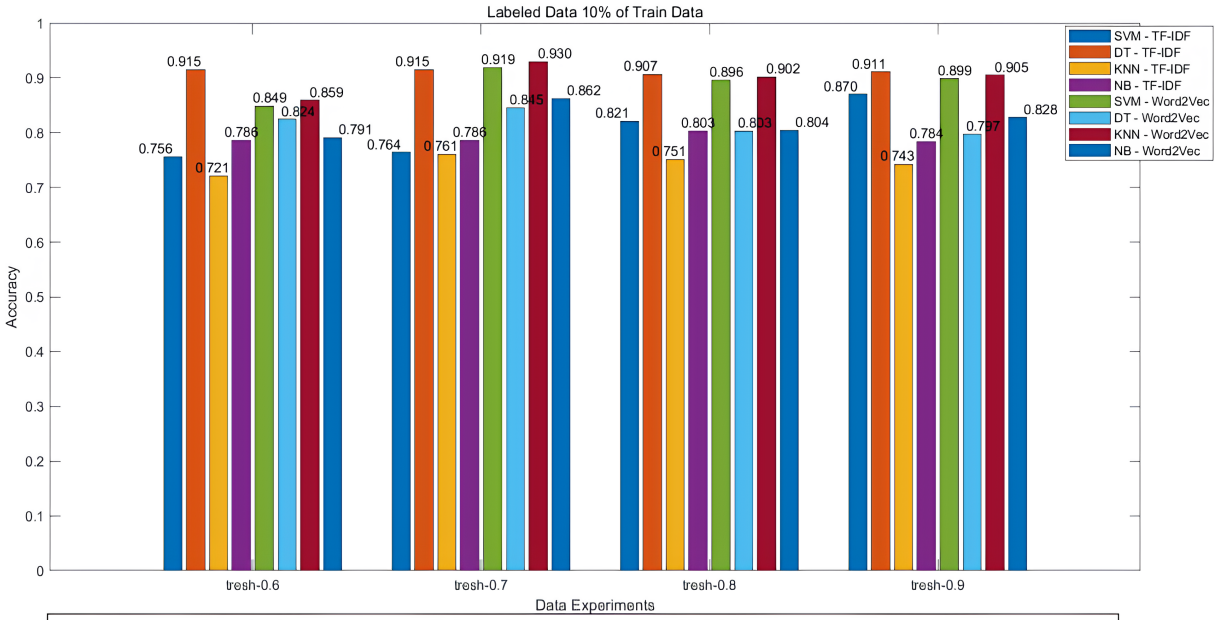
are not employed in the Word2Vec method due to the fundamental differences in the calculation process it entails. In addition, the SVM-TF-IDF and SVM-Word2Vec methods, as implemented by Saifullah et al. [56], have been assessed under similar conditions. These methods achieved consecutive accuracies of 63.3% and 89%. As a result, feature extraction using TF-IDF has increased accuracy in performing automatic annotations by 19.5%. This also influences the application of the ensemble concept and the addition of stop-word data in the preprocessing. The Word2Vec method has achieved optimal results, and when a threshold value of 0.9 is added, it leads to a 90% yield, representing a slight 1% increase.



Method	tresh-0.6	tresh-0.7	tresh-0.8	tresh-0.9
SVM - TF-IDF	0.793	0.782	0.828	0.853
DT - TF-IDF	0.856	0.861	0.891	0.895
KNN - TF-IDF	0.594	0.601	0.613	0.593
NB - TF-IDF	0.812	0.804	0.793	0.801
SVM - Word2Vec	0.823	0.858	0.888	0.900
DT - Word2Vec	0.728	0.753	0.775	0.796
KNN - Word2Vec	0.827	0.866	0.880	0.893
NB - Word2Vec	0.766	0.833	0.861	0.865

Figure 3. Accuracy of the proposed automatic annotation process in detecting hate speech using a semi-supervised and self-learning approach based on scenarios 1, 2, 3, and 4 (Table 1).

Based on the scenario results, the amount of 20% of the labeled data has not yet obtained optimal accuracy because several methods for detecting hate speech annotations have an accuracy of ≈90% [57] and even more than 95% [47]. So, we conducted a trial to reduce the amount of labeled data to 10% (Figure 4) and 5% (Figure 5) of the total YouTube comment data. We aim to optimize the automatic annotation with a minimal amount of data. The text annotation results improved in the case of several methods, such as DT-TF-IDF and KNN-Word2Vec. These methods obtained an accuracy of 90% and more for all threshold experiments (for DT-TF-IDF) and 0.7, 0.8, and 0.9 threshold experiments (for KNN-Word2Vec). In addition, the accuracy of SVM-Word2Vec also increased to 91.9% in the 10%, 90%, and 0.7 parameters of the same scenario. So, the notable accuracy improvement when reducing the labeled data from 20% to 10% data is based on Figures 3 and 4. This increase in accuracy is based on calculating the weight of the available data, where the smaller the amount of data, the higher the accuracy of the process. In addition, the self-learning performance of the meta-learner and meta-vectorizer relies on an ensemble approach to optimize the calculated weights, resulting in enhanced performance.



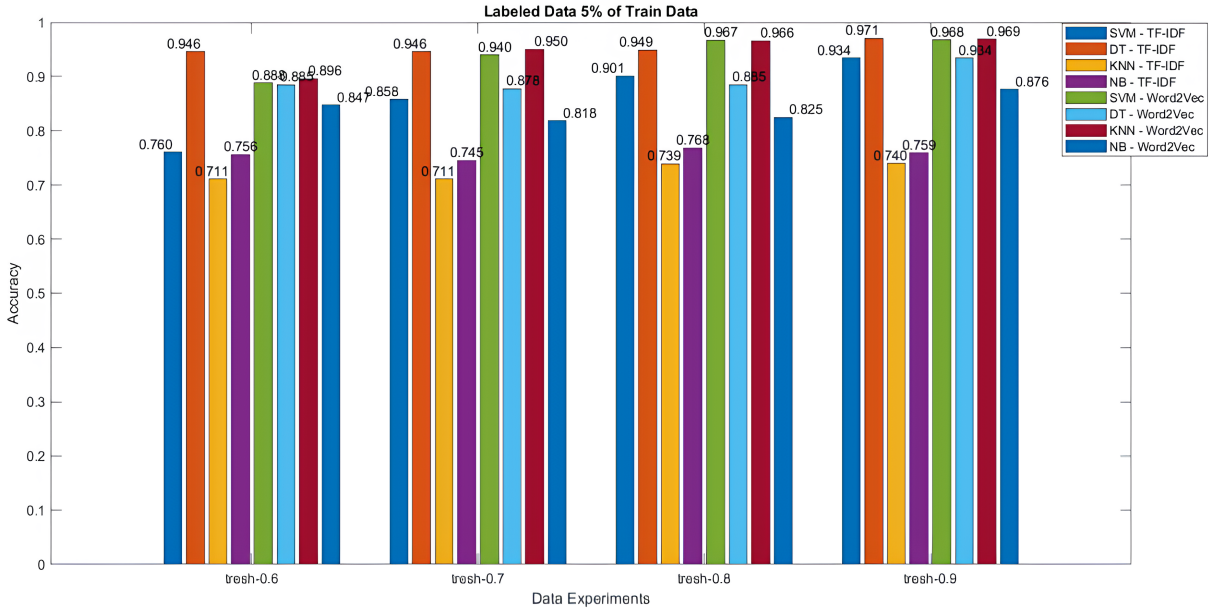
Method	tresh-0.6	tresh-0.7	tresh-0.8	tresh-0.9
SVM - TF-IDF	0.756	0.764	0.821	0.870
DT - TF-IDF	0.915	0.915	0.907	0.911
KNN - TF-IDF	0.721	0.761	0.751	0.743
NB - TF-IDF	0.786	0.786	0.803	0.784
SVM - Word2Vec	0.849	0.919	0.896	0.899
DT - Word2Vec	0.824	0.845	0.803	0.797
KNN - Word2Vec	0.859	0.930	0.902	0.905
NB - Word2Vec	0.791	0.862	0.804	0.828

Figure 4. Accuracy of the proposed automatic annotation process for detecting hate speech using a semi-supervised and self-learning approach based on scenarios 5, 6, 7, and 8 (Table 1).

The improvement of selected proposed methods from previous studies [5,56] became a reference for conducting this research. Building upon some proposed methodologies from the previous studies [5], our future work encompasses exploring various scenarios. These scenarios involve reducing labeled data to 5%, 10%, and 20%, in conjunction with adjusting the threshold values to 0.6, 0.7, 0.8, and 0.9. This study has achieved this by employing meta-vectorization and meta-learning methods to enhance the annotation process. Furthermore, applying threshold variations consistently resulted in the highest accuracy at the most significant value, specifically 0.9. This pattern was observed across all variations in the percentage of labeled data, except for the 10% labeled data scenario. This observation is attributed to the robust performance of SVM in various contexts, as documented in references [58,59], including the present research, as evidenced by Figures 3–5. These comparisons are readily evident in Figure A1 and Table A1. Notably, SVM consistently exhibits increasing accuracy. However, while the overall accuracy results demonstrate an increase, it is worth noting that certain methods experience a reduction of up to 8% in specific scenarios. The detailed comparison of the increase and decrease in method accuracy based on the scenario can be seen in Table A2.

Based on the results of the 10% labeled data scenario, the final experiment is to apply a 5% labeled data scenario for training with each existing threshold. Remarkably, each scenario demonstrates a systematic progression, encompassing thresholds ranging from minor to the most significant. The accuracy of this scenario is best, with a threshold value of 0.9. The highest accuracy was obtained for DT-TF-IDF with the value of 97.1%, followed by KNN-Word2Vec (96.9%), SVM-Word2Vec (96.8%), SVM-TF-IDF, DT-Word2Vec (93.4%), and others below 90%. These findings indicate that varying the quantity of data and adjusting the threshold value had a discernible impact on the methods' accuracy, leading to improved performance. In Figure 5, we can observe that DT-TF-IDF consistently

achieves a commendable accuracy level exceeding 94%, surpassing both KNN-Word2Vec and SVM-Word2Vec (exact accuracy values are shown in Table A1).



Method	tresh-0.6	tresh-0.7	tresh-0.8	tresh-0.9
SVM - TF-IDF	0.760	0.858	0.901	0.934
DT - TF-IDF	0.946	0.946	0.949	0.971
KNN - TF-IDF	0.711	0.711	0.739	0.740
NB - TF-IDF	0.756	0.745	0.768	0.759
SVM - Word2Vec	0.888	0.94	0.967	0.968
DT - Word2Vec	0.885	0.878	0.885	0.934
KNN - Word2Vec	0.896	0.95	0.966	0.969
NB - Word2Vec	0.847	0.818	0.825	0.876

Figure 5. Accuracy of the proposed automatic annotation process for detecting hate speech using a semi-supervised and self-learning approach based on scenarios 9, 10, 11, and 12 (Table 1).

Based on the presented scenarios and outcomes, this study adopts a semi-supervised learning approach, as illustrated in Figure 1. It involves annotating unlabeled data and substantially refining the sample dataset, ultimately leading to the automatic hate speech annotation. This process implements a self-learning mechanism, which leverages prior learning experiences to address the limitations associated with manual annotations. Regarding algorithm design, our proposed approach is shown in Listing 1. This annotation delineates an approach that employs minimal training data to maximize accuracy. This is achieved by incorporating a threshold parameter to align the training dataset with the annotation process.

Listing 1 details a novel approach to text auto-annotation. This approach employs semi-supervised and self-learning strategies. The initial steps involve dataset input. It includes an annotated dataset featuring scenarios of varied training proportions and a larger unannotated dataset. Preprocessing involves transforming both datasets into a semi-structured format. This process paves the way for subsequent feature extraction through TF-IDF and Word2Vec vectorization methods. The subsequent steps involve creating classifier models using SVM, DT, KNN, and NB. The best model is then validated through a semi-supervised approach using the Semi-Structured Annotated Dataset (SSADF). The pseudocode incorporates a thoughtful strategy of creating a model using a subset of the annotated dataset for validation, thus ensuring robust model performance. This process focuses on a semi-supervised validation. It involves converting datasets and iteratively refining the annotation results until a satisfactory threshold is achieved. The final output of SSADF reflects the culmination of this intricate process, showcasing the model's ability to autonomously annotate text data.

Listing 1. Text auto-annotation based on the semi-supervised and self-learning approach.

STEPS :

1. Dataset Input:
   a. Annotated Dataset (20%) with scenarios training 5%:10%:20%
   b. Unannotated dataset (80%)
2. Preprocessing
   a. Annotated Dataset → semi structured Annotated Dataset
   b. Unannotated dataset → semi structured Unannotated Dataset
3. Vectorization using merged semi structured annotated dataset + unannotated dataset : TF-IDF, Word2Vec
4. Put 10% from semi structured annotated dataset for sample dataset
5. Create a model classifier with 80% data training and 20% data validation. Model using SVM, DT, KNN, NB
6. Get the best model MTA (Multi-class Text Annotation): vectorizer + machine learning (with the best parameter)
   ==========
   VALIDATE SEMI SUPERVISED ANNOTATION BEST MODEL Using SSADF
   ==========
7. Convert Semi Structured Annotated Dataset → Semi Structured Annotated Dataset FEATURE : SSADF
8. Split SSADF into 80% for Data Training (SSADFT) and 20% for Data Validation (SSADFV)
9. Create MTA model using SSADFT (as Data Training)
10. VALIDATE MTA Model using SSADFV (as Data Validation)
    ==========
    SEMI SUPERVISED ANNOTATION using THE BEST MODEL
    ==========
11. Convert Semi Structured Unannotated dataset → Semi Structured UnAnnotated Dataset FEATURE : SSUDF
12. AnnotationResult = Annotate SSUDF using MTA MODEL with SSADFT
13. percentage = VALIDATE AnnotationResult using SAADFV
14. SAVE AnnotationResult_Valid to merge with SSADF
15. IF percentage < treshold THEN Go To 7.
    Output : SSADF

3.3. Discussion

Our approach strives to enhance the annotation process, focusing on speed, precision, and annotator consistency. In this section, we delve into the comparative analysis with previous studies, accuracy trends, and optimization with minimal labeled data. Additionally, we discuss the limitations of our approach.

3.3.1. Comparative Analysis with Previous Studies

Our study diverges significantly from previous works [5,56], showcasing distinct methodologies and achieving notable improvements. Ref. [5] adhered to a conventional scenario, employing KNN with TF-IDF for a 60% accuracy rate. In contrast, Ref. [56] utilized SVM with TF-IDF (92.5%) and Word2Vec (89.7%), aiming to enhance the annotation process. Innovatively, our research introduces scenarios featuring minimal training data and unique threshold values (Table 1). This departure from the traditional 80:20 training and testing proportion used in Ref. [5] sets our approach apart. It also explores hate speech detection under limited training data, emphasizing the model's sensitivity to varying similarity thresholds. This nuanced investigation contributes to a more in-depth understanding of the model's performance in scenarios with constrained training data.

Comparative results, detailed in Table A1, underscore the effectiveness of our proposed scenarios. The performance metrics reveal substantial enhancements compared to those reported in Refs. [5,56]. Table A2 quantifies these improvements, providing an apparent percentage increase over the baseline results presented in Refs. [5,56].

3.3.2. Accuracy Trends and Method Performance

This section examines the accuracy trends and performance of various methods employed in hate speech detection scenarios. Our study adopts a semi-supervised learning approach, as depicted in Figure 1. It involves annotating unlabeled data to refine the sample dataset and automate the hate speech annotation process. The systematic progression across varying scenarios and threshold values demonstrates a discernible impact on the accuracy of the methods. This impact is evident in the outcomes of our study.

SVM's Consistent Performance: One notable trend is the consistent increase in accuracy exhibited by the Support Vector Machine (SVM) across different scenarios. SVM, a robust classification algorithm, has demonstrated its efficacy in hate speech detection, consistently outperforming other methods. This aligns with the findings in Refs. [58,59]. The robust performance of SVM is highlighted in diverse contexts. The incremental accuracy observed across scenarios reinforces SVM's suitability for hate speech detection tasks.

DT's Superior Performance in the Last Scenario: In the last scenario (5%), we employed only 5% labeled data with a threshold of 0.9. The DT with TF-IDF emerged as a powerful performer, showcasing superior performance with an impressive accuracy of 97.1% (refer to Table A1). This exceptional result underscores the adaptability and strength of DT in handling hate speech detection tasks. It is especially notable under conditions of minimal labeled data and stringent similarity thresholds.

Method-Specific Variations: While SVM consistently performs well and DT excels in specific scenarios, certain methods experience a reduction of up to 8% in specific contexts. This reduction underscores the nuanced nature of hate speech detection. Method-specific variations impact performance. Analyzing these variations is crucial for understanding the strengths and limitations of each method under different conditions.

Impact of Threshold Variations: Threshold values are pivotal in influencing the accuracy of hate speech detection. The systematic exploration of varying threshold values is illustrated in Figures 3–5. This exploration allows for a nuanced understanding of how these values affect methods performance. The highest accuracy is consistently achieved at the most significant threshold value, particularly 0.9. It indicates the importance of optimizing similarity thresholds for accurate hate speech detection.

Optimization with Minimal Labeled Data: An intriguing observation is the improvement in accuracy when reducing labeled data from 20% to 5%. This aligns with the calculated weights of available data. It suggests that the smaller the data, the higher the accuracy of the annotation process. The self-learning performance of the meta-learner and meta-vectorizer relies on an ensemble approach to optimize calculated weights. It contributes to enhanced performance.

3.3.3. Discussion on Biases and Mitigation Strategies

Our hate speech detection model relies on Indonesian YouTube comments. It may be subject to linguistic and thematic biases inherent in the dataset. These biases are influenced by cultural nuances and the focus on specific themes, such as the 2019 Indonesian presidential debate and COVID-19. They could limit the model's applicability to diverse linguistic and thematic contexts. Additionally, biases are introduced through the model's design. It includes the choice of machine learning algorithms (SVM, DT, KNN, and Naive Bayes) and feature extraction methods (Word2Vec and TF-IDF). Annotation bias, stemming from subjective human judgment during annotation, further contributes to potential bias in the model's learning. Addressing these biases requires ongoing vigilance, continuous model evaluation, and a commitment to algorithmic fairness for equitable performance.

Future research should consider diversifying datasets to include various themes, languages, and cultural contexts to mitigate biases. Employing fairness-aware machine learning techniques and continuous model evaluation and updates can address algorithmic biases over time. Proactively managing biases is crucial for promoting fairness and reliability in the responsible deployment of hate speech detection models.

3.3.4. Limitations

While showcasing promising results, our proposed hate speech detection approach is subject to certain limitations that warrant in-depth analysis. By dissecting these limitations, we aim to provide a nuanced understanding of the constraints and challenges associated with our methods.

Linguistic Specificity and Stemming Challenges: A substantial limitation stems from the linguistic specificity of our dataset, which predominantly comprises comments in the Indonesian language. The choice of Sastrawi stemming, limited to the basic form of Indonesian words, raises concerns about the method's adaptability to diverse linguistic contexts. The intricacies of different languages, dialects, and linguistic nuances may impact the performance of our approach when applied to datasets outside the scope of the Indonesian language. Additionally, excluding online annotation tools like MonkeyLearn was strategic, considering potential mismatches with our linguistic context and stemming approach. It underscores the need for future research to explore the generalizability of our methodology to multilingual settings. Additionally, investigating alternative annotation tools is crucial.

Applicability to Hate Speech Variants and Contexts: While our approach demonstrates effectiveness in detecting hate speech related to the context of COVID-19, its generalizability to diverse hate speech variants remains a subject of scrutiny. The efficacy of our approach in identifying overt hate speech is well-established. However, we acknowledge the inherent challenges of recognizing sarcasm and subtle forms of expression. These may need to be more effectively captured by our current method.

The thematic specificity of our training data could pose challenges in adapting the model to recognize hate speech in different domains or contexts. Future research should prioritize expanding the dataset to encompass a broader spectrum of hate speech variants. It ensures the versatility and adaptability of our approach across a wide range of contexts. A robust hate speech detection model should transcend specific thematic constraints. It maintains relevance and effectiveness in real-world applications.

Dataset Limitation to COVID-19 Context: An inherent limitation of our study lies in the deliberate restriction of the dataset. It focuses specifically on hate speech related to the COVID-19 context. While this focus aligns with the contemporary relevance of the issue, it introduces a thematic bias. This bias may limit the model's ability to discern hate speech in unrelated contexts. Future research endeavors should prioritize the diversification of the dataset. This includes encompassing hate speech in various thematic domains, ensuring a more comprehensive evaluation of our method's robustness and adaptability.

4. Conclusions

This research introduces a novel approach to automated text annotation, leveraging semi-supervised and self-learning strategies. The study showcases the effectiveness of our method in enhancing machine learning by integrating a meta-learner for hate speech annotation in Indonesian text. The proposed concept combines a meta-learner and a meta-vectorizer to get an annotation process with a minimal dataset applied. Our approach utilizes a minimal labeled dataset of 20%, 10%, and 5%, incorporating a sophisticated weighting mechanism and a threshold system with four predefined values: 0.6, 0.7, 0.8, and 0.9. The optimization process for annotating small datasets yielded remarkable results, achieving automatic annotations with only 5% labeling and a threshold of 0.9. Notably, the DT-TF-IDF method achieved the highest accuracy, reaching an impressive 97.1%. In addition, the model that has the best accuracy is followed by KNN-Word2Vec (96.9%),

SVM-Word2Vec (96.8%), SVM-TF-IDF (93.4%), and DT-Word2Vec (93.4%) in the same scenario (5%, 80%, 0.9). The findings indicate a consistent trend of improved accuracy with increasing threshold values, with the highest threshold yielding superior accuracy across all scenarios. This result is inversely proportional to the percentage of labeled training data—lower percentages correlate with higher accuracy values. Based on Table 1, we observe that the accuracy value increases as the dataset size decreases from largest to smallest. These study findings can serve as a valuable reference for improving the accuracy of automatic annotation processes.

This approach could increase the number of datasets and their corpus in the future, optimizing annotations. In addition, there is a need to explore various methods, including the possibility of applying other machine learning techniques, to develop automated annotation processes further. In future projects, we can explore the development of multiple criteria for hate speech categorization, such as different classification categories like abusive, aggressive, offensive, and others.

Author Contributions: Conceptualization, S.S. and A.S.A.; Data curation, S.S., A.S.A., Y.F. and N.H.C.; Formal analysis, S.S., R.D. and A.S.A.; Funding acquisition, R.D.; Investigation, S.S. and A.S.A.; Methodology, S.S. and A.S.A.; Project administration, S.S.; Resources, S.S., A.S.A., Y.F. and N.H.C.; Software, S.S. and A.S.A.; Supervision, S.S. and R.D.; Validation, S.S., R.D. and A.S.A.; Visualization, S.S. and A.S.A.; Writing—original draft, S.S., R.D., F.A.D. and A.S.A.; Writing—review and editing, S.S., R.D. and F.A.D. All authors have read and agreed to the published version of the manuscript.

Funding: The research presented in this paper was partially supported by the funds of Polish Ministry of Education and Science assigned to AGH University of Krakow. Additionally, it was also partially supported by PLGrid Infrastructure with grant number PLG/2023/016757.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data available on request due to privacy restrictions.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A

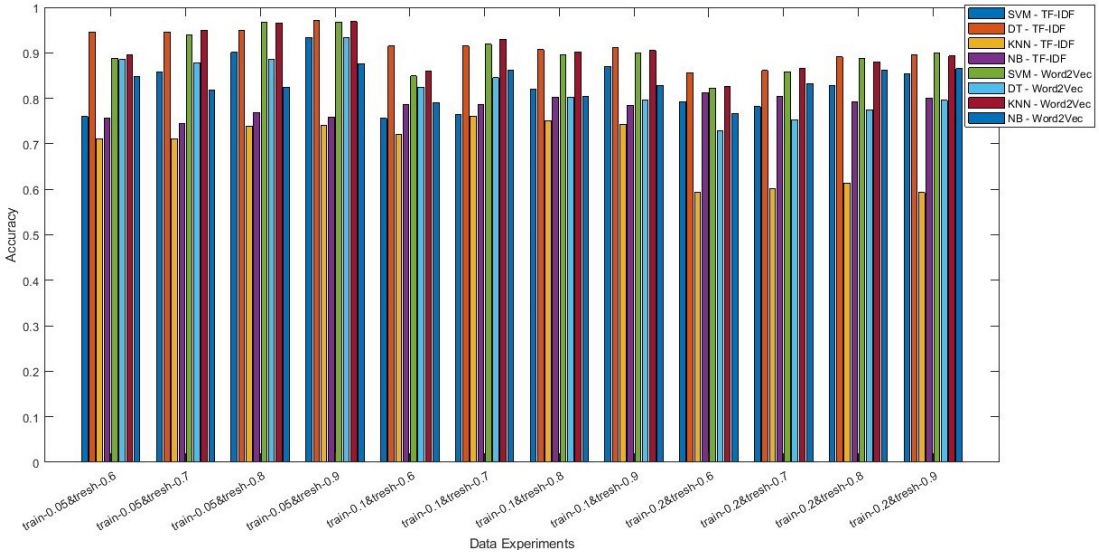


Figure A1. The accuracy results of the proposed models in text annotations for hate speech detection using a semi-supervised (self-learning) approach based on scenarios labeled data (5%, 10%, and 20%) and threshold (0.6, 0.7, 0.8, and 0.9).

Appendix B

Table A1. Accuracy results of the combination of meta-learning and meta vectorization modeling based on the scenario of Table 1.

No	Labeled Data	Unlabeled Data	Threshold	SVM TF-IDF	DT TF-IDF	KNN TF-IDF	NB TF-IDF	SVM Word2Vec	DT Word2Vec	KNN Word2Vec	NB Word2Vec
1	0.2	0.8	0.6	0.793	0.856	0.594	0.812	0.823	0.728	0.827	0.766
2	0.2	0.8	0.7	0.782	0.861	0.601	0.804	0.858	0.753	0.866	0.833
3	0.2	0.8	0.8	0.828	0.891	0.613	0.793	0.888	0.775	0.88	0.861
4	0.2	0.8	0.9	0.853	0.895	0.593	0.801	0.9	0.796	0.893	0.865
5	0.1	0.9	0.6	0.756	0.915	0.721	0.786	0.849	0.824	0.859	0.791
6	0.1	0.9	0.7	0.764	0.915	0.761	0.786	0.919	0.845	0.93	0.862
7	0.1	0.9	0.8	0.821	0.907	0.751	0.803	0.896	0.803	0.902	0.804
8	0.1	0.9	0.9	0.87	0.911	0.743	0.784	0.899	0.797	0.905	0.828
9	0.05	0.95	0.6	0.76	0.946	0.711	0.756	0.888	0.885	0.896	0.847
10	0.05	0.95	0.7	0.858	0.946	0.711	0.745	0.94	0.878	0.95	0.818
11	0.05	0.95	0.8	0.901	0.949	0.739	0.768	0.967	0.885	0.966	0.825
12	0.05	0.95	0.9	0.934	0.971	0.74	0.759	0.968	0.934	0.969	0.876

Appendix C

Table A2. Comparison of the increase and decrease in accuracy in each scenario.

Compared Scenarios	SVM TF-IDF (%)	DT TF-IDF (%)	KNN TF-IDF (%)	NB TF-IDF (%)	SVM Word2Vec (%)	DT Word2Vec (%)	KNN Word2Vec (%)	NB Word2Vec (%)
1–2	−1.10	0.50	0.70	−0.80	3.50	2.50	3.90	6.70
2–3	4.60	3.00	1.20	−1.10	3.00	2.20	1.40	2.80
3–4	2.50	0.40	−2.00	0.80	1.20	2.10	1.30	0.40
5–6	0.80	0.00	4.00	0.00	7.00	2.10	7.10	7.10
6–7	5.70	−0.80	−1.00	1.70	−2.30	−4.20	−2.80	−5.80
7–8	4.90	0.40	−0.80	−1.90	0.30	−0.60	0.30	2.40
9–10	9.80	0.00	0.00	−1.10	5.20	−0.70	5.40	−2.90
10–11	4.30	0.30	2.80	2.30	2.70	0.70	1.60	0.70
11–12	3.30	2.20	0.10	−0.90	0.10	4.90	0.30	5.10

References

1. Alrehili, A. Automatic Hate Speech Detection on Social Media: A Brief Survey. In Proceedings of the 2019 IEEE/ACS 16th International Conference on Computer Systems and Applications (AICCSA), Abu Dhabi, United Arab Emirates, 3–7 November 2019; IEEE: Piscataway, NJ, USA, 2019. [CrossRef]
2. Al-Makhadmeh, Z.; Tolba, A. Automatic hate speech detection using killer natural language processing optimizing ensemble deep learning approach. *Computing* **2019**, *102*, 501–522. [CrossRef]
3. Rajman, M.; Besançon, R. Text Mining: Natural Language techniques and Text Mining applications. In *Data Mining and Reverse Engineering*; Springer: New York, NY, USA, 1998; pp. 50–64. [CrossRef]
4. Fortuna, P.; Nunes, S. A Survey on Automatic Detection of Hate Speech in Text. *ACM Comput. Surv.* **2018**, *51*, 1–30. [CrossRef]
5. Cahyana, N.H.; Saifullah, S.; Fauziah, Y.; Aribowo, A.S.; Drezewski, R. Semi-supervised Text Annotation for Hate Speech Detection using K-Nearest Neighbors and Term Frequency-Inverse Document Frequency. *Int. J. Adv. Comput. Sci. Appl.* **2022**, *13*, 147–151. [CrossRef]
6. Aman, S.; Szpakowicz, S. Identifying Expressions of Emotion in Text. In *Text, Speech and Dialogue. TSD 2007*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 196–205. [CrossRef]
7. Krouska, A.; Troussas, C.; Virvou, M. The effect of preprocessing techniques on Twitter sentiment analysis. In Proceedings of the 2016 7th International Conference on Information, Intelligence, Systems & Applications (IISA), Chalkidiki, Greece, 13–15 July 2016; IEEE: Piscataway, NJ, USA, 2016. [CrossRef]
8. Savigny, J.; Purwarianti, A. Emotion classification on youtube comments using word embedding. In Proceedings of the 2017 International Conference on Advanced Informatics, Concepts, Theory, and Applications (ICAICTA), Denpasar, Indonesia, 16–18 August 2017; IEEE: Piscataway, NJ, USA, 2017. [CrossRef]
9. Ningtyas, A.M.; Herwanto, G.B. The Influence of Negation Handling on Sentiment Analysis in Bahasa Indonesia. In Proceedings of the 2018 5th International Conference on Data and Software Engineering (ICoDSE), Mataram, Indonesia, 7–8 November 2018; IEEE: Piscataway, NJ, USA, 2018. [CrossRef]
10. Mariel, W.C.F.; Mariyah, S.; Pramana, S. Sentiment analysis: A comparison of deep learning neural network algorithm with SVM and naïve Bayes for Indonesian text. *J. Phys. Conf. Ser.* **2018**, *971*, 012049. [CrossRef]
11. Mao, R.; Liu, Q.; He, K.; Li, W.; Cambria, E. The Biases of Pre-Trained Language Models: An Empirical Study on Prompt-Based Sentiment Analysis and Emotion Detection. *IEEE Trans. Affect. Comput.* **2022**, *14*, 1743–1753. [CrossRef]
12. Dashtipour, K.; Gogate, M.; Gelbukh, A.; Hussain, A. Extending persian sentiment lexicon with idiomatic expressions for sentiment analysis. *Soc. Netw. Anal. Min.* **2021**, *12*, 9. [CrossRef]

13. Imran, A.S.; Yang, R.; Kastrati, Z.; Daudpota, S.M.; Shaikh, S. The impact of synthetic text generation for sentiment analysis using GAN based models. *Egypt. Inform. J.* **2022**, *23*, 547–557. [CrossRef]
14. Balli, C.; Guzel, M.S.; Bostanci, E.; Mishra, A. Sentimental Analysis of Twitter Users from Turkish Content with Natural Language Processing. *Comput. Intell. Neurosci.* **2022**, *2022*, 2455160. [CrossRef]
15. Jain, D.K.; Boyapati, P.; Venkatesh, J.; Prakash, M. An Intelligent Cognitive-Inspired Computing with Big Data Analytics Framework for Sentiment Analysis and Classification. *Inf. Process. Manag.* **2022**, *59*, 102758. [CrossRef]
16. Kabakus, A.T. A novel COVID-19 sentiment analysis in Turkish based on the combination of convolutional neural network and bidirectional long-short term memory on Twitter. *Concurr. Comput. Pract. Exp.* **2022**, *34*, e6883. [CrossRef] [PubMed]
17. Al-Laith, A.; Shahbaz, M.; Alaskar, H.F.; Rehmat, A. AraSenCorpus: A Semi-Supervised Approach for Sentiment Annotation of a Large Arabic Text Corpus. *Appl. Sci.* **2021**, *11*, 2434. [CrossRef]
18. Saifullah, S.; Drężewski, R.; Dwiyanto, F.A.; Aribowo, A.S.; Fauziah, Y. Sentiment Analysis Using Machine Learning Approach Based on Feature Extraction for Anxiety Detection. In Proceedings of the Computational Science—ICCS 2023: 23rd International Conference, Prague, Czech Republic, 3–5 July 2023; Springer: Berlin/Heidelberg, Germany, 2023; pp. 365–372. [CrossRef]
19. Balakrishnan, V.; Lok, P.Y.; Rahim, H.A. A semi-supervised approach in detecting sentiment and emotion based on digital payment reviews. *J. Supercomput.* **2020**, *77*, 3795–3810. [CrossRef]
20. Ibrohim, M.O.; Budi, I. Multi-label Hate Speech and Abusive Language Detection in Indonesian Twitter. In Proceedings of the Third Workshop on Abusive Language Online, Florence, Italy, 1 August 2019; Association for Computational Linguistics: Stroudsburg, PA, USA, 2019. [CrossRef]
21. Khanday, A.M.U.D.; Rabani, S.T.; Khan, Q.R.; Malik, S.H. Detecting twitter hate speech in COVID-19 era using machine learning and ensemble learning techniques. *Int. J. Inf. Manag. Data Insights* **2022**, *2*, 100120. [CrossRef]
22. Zhang, Z.; Robinson, D.; Tepper, J. Detecting Hate Speech on Twitter Using a Convolution-GRU Based Deep Neural Network. In *The Semantic Web*; Springer International Publishing: New York, NY, USA, 2018; pp. 745–760. [CrossRef]
23. Davidson, T.; Warmsley, D.; Macy, M.; Weber, I. Automated Hate Speech Detection and the Problem of Offensive Language. In Proceedings of the International AAAI Conference on Web and Social Media, Montreal, QC, Canada, 15–18 May 2017; Volume 11, pp. 512–515. [CrossRef]
24. Cahyani, D.E.; Patasik, I. Performance comparison of TF-IDF and Word2Vec models for emotion text classification. *Bull. Electr. Eng. Inform.* **2021**, *10*, 2780–2788. [CrossRef]
25. Abduljabbar, D.A.; Omar, N. Exam questions classification based on Bloom's taxonomy cognitive level using classifiers combination. *J. Theor. Appl. Inf. Technol.* **2015**, *78*, 447–455.
26. Soliman, A.B.; Eissa, K.; El-Beltagy, S.R. AraVec: A set of Arabic Word Embedding Models for use in Arabic NLP. *Procedia Comput. Sci.* **2017**, *117*, 256–265. [CrossRef]
27. Kumar, C.S.P.; Babu, L.D.D. Novel Text Preprocessing Framework for Sentiment Analysis. In *Smart Intelligent Computing and Applications*; Springer: Singapore, 2018; pp. 309–317. [CrossRef]
28. Ramachandran, D.; Parvathi, R. Analysis of Twitter Specific Preprocessing Technique for Tweets. *Procedia Comput. Sci.* **2019**, *165*, 245–251. [CrossRef]
29. Mohammed, M.; Omar, N. Question classification based on Bloom's taxonomy cognitive domain using modified TF-IDF and word2vec. *PLoS ONE* **2020**, *15*, e0230442. [CrossRef] [PubMed]
30. Babanejad, N.; Agrawal, A.; An, A.; Papagelis, M. A Comprehensive Analysis of Preprocessing for Word Representation Learning in Affective Tasks. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020; Association for Computational Linguistics: Stroudsburg, PA, USA, 2020. [CrossRef]
31. Albalawi, R.; Yeap, T.H.; Benyoucef, M. Using Topic Modeling Methods for Short-Text Data: A Comparative Analysis. *Front. Artif. Intell.* **2020**, *3*, 42. [CrossRef] [PubMed]
32. Arora, M.; Kansal, V. Character level embedding with deep convolutional neural network for text normalization of unstructured data for Twitter sentiment analysis. *Soc. Netw. Anal. Min.* **2019**, *9*, 12. [CrossRef]
33. Elgibreen, H.; Faisal, M.; Sulaiman, M.A.; Abdou, S.; Mekhtiche, M.A.; Moussa, A.M.; Alohali, Y.A.; Abdul, W.; Muhammad, G.; Rashwan, M.; et al. An Incremental Approach to Corpus Design and Construction: Application to a Large Contemporary Saudi Corpus. *IEEE Access* **2021**, *9*, 88405–88428. [CrossRef]
34. Rai, A.; Borah, S. Study of Various Methods for Tokenization. In *Applications of Internet of Things*; Springer: Singapore, 2020; pp. 193–200. [CrossRef]
35. Manalu, S.R. Stop words in review summarization using TextRank. In Proceedings of the 2017 14th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), Phuket, Thailand, 27–30 June 2017; IEEE: Piscataway, NJ, USA, 2017. [CrossRef]
36. Zeroual, I.; Lakhouaja, A. Arabic information retrieval: Stemming or lemmatization? In Proceedings of the 2017 Intelligent Systems and Computer Vision (ISCV), Fez, Morocco, 17–19 April 2017; IEEE: Piscataway, NJ, USA, 2017. [CrossRef]
37. AlKhwiter, W.; Al-Twairesh, N. Part-of-speech tagging for Arabic tweets using CRF and Bi-LSTM. *Comput. Speech Lang.* **2021**, *65*, 101138. [CrossRef]
38. Sharma, A.; Kumar, S. Ontology-based semantic retrieval of documents using Word2vec model. *Data Knowl. Eng.* **2023**, *144*, 102110. [CrossRef]

39. Liang, H.; Sun, X.; Sun, Y.; Gao, Y. Text feature extraction based on deep learning: A review. *EURASIP J. Wirel. Commun. Netw.* **2017**, *2017*, 211. [CrossRef] [PubMed]
40. Garouani, M.; Ahmad, A.; Bouneffa, M.; Hamlich, M.; Bourguin, G.; Lewandowski, A. Using meta-learning for automated algorithms selection and configuration: An experimental framework for industrial big data. *J. Big Data* **2022**, *9*, 57. [CrossRef]
41. Kamyab, M.; Liu, G.; Adjeisah, M. Attention-Based CNN and Bi-LSTM Model Based on TF-IDF and GloVe Word Embedding for Sentiment Analysis. *Appl. Sci.* **2021**, *11*, 11255. [CrossRef]
42. Saifullah, S.; Fauziyah, Y.; Aribowo, A.S. Comparison of machine learning for sentiment analysis in detecting anxiety based on social media data. *J. Inform.* **2021**, *15*, 45. [CrossRef]
43. Fauziah, Y.; Saifullah, S.; Aribowo, A.S. Design Text Mining for Anxiety Detection using Machine Learning based-on Social Media Data during COVID-19 pandemic. In Proceedings of the LPPM UPN "Veteran" Yogyakarta Conference Series 2020—Engineering and Science Series, Yogyakarta, Indonesia, 27 October 2020; pp. 253–261. [CrossRef]
44. Capelle, M.; Hogenboom, F.; Hogenboom, A.; Frasincar, F. Semantic news recommendation using wordnet and bing similarities. In Proceedings of the 28th Annual ACM Symposium on Applied Computing—SAC'13, Coimbra, Portugal, 18–22 March 2013; ACM Press: New York, NY, USA 2013. [CrossRef]
45. Sivakumar, S.; Videla, L.S.; Kumar, T.R.; Nagaraj, J.; Itnal, S.; Haritha, D. Review on Word2Vec Word Embedding Neural Net. In Proceedings of the 2020 International Conference on Smart Electronics and Communication (ICOSEC), Trichy, India, 10–12 September 2020; IEEE: Piscataway, NJ, USA, 2020. [CrossRef]
46. Landgraf, A.J.; Bellay, J. word2vec Skip-Gram with Negative Sampling is a Weighted Logistic PCA. *arXiv* **2017**, arXiv:1705.09755. [CrossRef]
47. Alkomah, F.; Ma, X. A Literature Review of Textual Hate Speech Detection Methods and Datasets. *Information* **2022**, *13*, 273. [CrossRef]
48. Saifullah, S.; Drezewski, R. Non-Destructive Egg Fertility Detection in Incubation Using SVM Classifier Based on GLCM Parameters. *Procedia Comput. Sci.* **2022**, *207*, 3248–3257. [CrossRef]
49. Bansal, M.; Goyal, A.; Choudhary, A. A comparative analysis of K-Nearest Neighbor, Genetic, Support Vector Machine, Decision Tree, and Long Short Term Memory algorithms in machine learning. *Decis. Anal. J.* **2022**, *3*, 100071. [CrossRef]
50. Kuchipudi, R.; Uddin, M.; Murthy, T.; Mirrudoddi, T.K.; Ahmed, M.; P, R.B. Android Malware Detection using Ensemble Learning. In Proceedings of the 2023 International Conference on Sustainable Computing and Smart Systems (ICSCSS), Coimbatore, India, 14–16 June 2023; IEEE: Piscataway, NJ, USA, 2023; pp. 297–302. [CrossRef]
51. Degirmenci, A.; Karal, O. Efficient density and cluster based incremental outlier detection in data streams. *Inf. Sci.* **2022**, *607*, 901–920. [CrossRef]
52. Kesarwani, A.; Chauhan, S.S.; Nair, A.R. Fake News Detection on Social Media using K-Nearest Neighbor Classifier. In Proceedings of the 2020 International Conference on Advances in Computing and Communication Engineering (ICACCE), Las Vegas, NV, USA, 22–24 June 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–4. [CrossRef]
53. Xu, S. Bayesian Naïve Bayes classifiers to text classification. *J. Inf. Sci.* **2018**, *44*, 48–59. [CrossRef]
54. Mwaro, P.N.; Ogada, D.K.; Cheruiyot, P.W. Applicability of Naïve Bayes Model for Automatic Resume Classification. *Int. J. Comput. Appl. Technol. Res.* **2020**, *9*, 257–264. [CrossRef]
55. Zhang, F.; Fleyeh, H.; Wang, X.; Lu, M. Construction site accident analysis using text mining and natural language processing techniques. *Autom. Constr.* **2019**, *99*, 238–248. [CrossRef]
56. Saifullah, S.; Cahyana, N.H.; Fauziah, Y.; Aribowo, A.S.; Dwiyanto, F.A.; Drezewski, R. Text Annotation Automation for Hate Speech Detection using SVM-classifier based on Feature Extraction. In Proceedings of the International Conference on Advanced Research in Engineering and Technology, Thai Nguyen, Vietnam, 1–2 December 2022.
57. Kocoń, J.; Figas, A.; Gruza, M.; Puchalska, D.; Kajdanowicz, T.; Kazienko, P. Offensive, aggressive, and hate speech analysis: From data-centric to human-centered approach. *Inf. Process. Manag.* **2021**, *58*, 102643. [CrossRef]
58. Maniruzzaman, M.; Rahman, M.J.; Ahammed, B.; Abedin, M.M. Classification and prediction of diabetes disease using machine learning paradigm. *Health Inf. Sci. Syst.* **2020**, *8*, 7. [CrossRef]
59. Machova, K.; Mach, M.; Vasilko, M. Comparison of Machine Learning and Sentiment Analysis in Detection of Suspicious Online Reviewers on Different Type of Data. *Sensors* **2021**, *22*, 155. [CrossRef]