applied sciences

Article

A Novel Stacked Ensemble for Hate Speech Recognition

Mona Khalifa A. Aljero [1] and Nazife Dimililer [2,*]

[1] Department of Applied Mathematics & Computer Sciences, Faculty of Arts and Sciences, Eastern Mediterranean University, Via Mersin 10, Famagusta 99628, North Cyprus, Turkey; mnny1985@yahoo.com
[2] Department of Information Technology, School of Computing and Technology, Eastern Mediterranean University, Via Mersin 10, Famagusta 99628, North Cyprus, Turkey
* Correspondence: nazife.dimililer@emu.edu.tr

Abstract: Detecting harmful content or hate speech on social media is a significant challenge due to the high throughput and large volume of content production on these platforms. Identifying hate speech in a timely manner is crucial in preventing its dissemination. We propose a novel stacked ensemble approach for detecting hate speech in English tweets. The proposed architecture employs an ensemble of three classifiers, namely support vector machine (SVM), logistic regression (LR), and XGBoost classifier (XGB), trained using word2vec and universal encoding features. The meta classifier, LR, combines the outputs of the three base classifiers and the features employed by the base classifiers to produce the final output. It is shown that the proposed architecture improves the performance of the widely used single classifiers as well as the standard stacking and classifier ensemble using majority voting. We also present results on the use of various combinations of machine learning classifiers as base classifiers. The experimental results from the proposed architecture indicated an improvement in the performance on all four datasets compared with the standard stacking, base classifiers, and majority voting. Furthermore, on three of these datasets, the proposed architecture outperformed all state-of-the-art systems.

Keywords: hate speech recognition; stacking ensemble; text classification; binary classification; stacked ensemble

Citation: Aljero, M.K.A.; Dimililer, N. A Novel Stacked Ensemble for Hate Speech Recognition. *Appl. Sci.* **2021**, *11*, 11684. https://doi.org/10.3390/app112411684

Academic Editor: Valentino Santucci

Received: 5 August 2021
Accepted: 23 November 2021
Published: 9 December 2021

1. Introduction

An undesirable side effect of the increase in social media usage has been the rapid growth of hate speech on these platforms. Hate speech can be defined as an attack on a specific person or group based on race, ethnicity, religion, gender, age, disability, or sexual orientation. Each platform of social media has its definition of hate speech. However, all agree that hate speech attacks specific target groups based on some discriminating characteristic. Nevertheless, if automatic hate speech detection is not performed, the platforms may become hate speech platforms. Even though recent research on automatic detection of hate speech is well-presented in [1–5], to the best of our knowledge, no such system has been fully implemented yet. Facebook implemented a model in 2019 called RoBERTa to detect toxic posts, dependence on user reports for the detection of hate speech has not been eliminated yet.

Waseem and Hovy [6] studied the detection of hate speech in the form of racism and sexism on social media. They created their dataset from English Twitter which was the first public dataset for hate speech. The availability of this dataset allowed other researchers to use it to evaluate their model and compare the results with other approaches. Their study addressed hate speech detection as a multiclassification problem. They labeled the tweets as 'Sexism', 'Racism' and 'Neither'. Character N-gram and linguistic features performed the best as a feature, with their dataset scoring 73.93% F1-score.

Davidson et al. [7] also collected a corpus of tweets, which became the most extensive publicly available corpus of more than 24 k tweets in 2017. They labeled tweets into three categories: 'Hate', 'Offensive', and 'Neither'. The authors presented a study on "Automated Hate Speech Detection and the Problem of Offensive Language" as they developed a multiclass model. As features, they used N-gram with Term Frequency-Inverse Document Frequency (TF-IDF), Part of Speech (POS), a sentiment lexicon, and several characters, words, and syllables in each tweet. In their study, they applied SVM and LR as classifiers. The most outstanding result they obtained was a 90% F1-score utilizing the LR as a classifier. The authors of this study observed that incorporating information about the user (the person who tweeted) can provide interesting features that improve the classifier's performance.

Different approaches have been employed to detect hate speech in social media. These approaches varied from individual models to ensemble models. Authors in [8] employed several machine learning approaches, namely; SVM, Random Forest (RF), and Naïve Bayes (NB) on multiple datasets. The sentiment lexicon and N-gram were combined as features. Using RF, they were able to achieve the best result for binary classification, with an accuracy of 77.36%. Furthermore, they built a new dataset from Indonesian Twitter. On the other hand, in [9], Indurthi et al. employed one classifier, namely SVM with universal sentence encoder (USE) as a feature. The authors evaluated their model on one publicly available HatEval dataset from SemEval-2019 Task 5 achieving an F1-score of 65.1%.

Stacking is one of the most frequently used algorithms first presented by Wolpert in 1992 [10]. One of the advantages of this approach is that each base classifier's misclassification is minimized [10]. It has been used on multiple real-world datasets, and it achieved higher results than the stand-alone classifiers [11]. Furthermore, in some competitions, such as Kaggle [12], this approach has been employed very frequently. The popularity of the stacking approach among the users of Kaggle is due to the improved performance on real-world datasets and the fact that high-ranking participants have adopted it.

Badjatiya et al. [13] applied three deep learning methods to develop a hate speech classifier namely convolutional neural networks (CNNs), FastText, and long short-term memory (LSTM) networks with word embedding. The authors employed different combinations of neural networks, Gradient Boosted Decision Trees (GBDTs), and word embedding utilized by random embeddings or GloVe. As baselines, the authors employed machine learning algorithms, namely; SVM, LR, GBDT, and RF. For the evaluation of the classifier's performance, they used the Waseem and Hovy [6] dataset. Their approach outperformed the baselines methods and achieved the best result with the ensemble approach of LSTM + GBDT with Random Embedding achieving a 93% F1-score.

Aria et al. [14] employed an ensemble approach of three classifiers to detect hate speech on the English HatEval dataset from SemEval-2019 Task 5. They used SVM, RF, and Bidirectional Long Short-Term Memory (BiLSTM). As features, with the SVM and RF, they used bi/tri-grams, and word embedding with BiLSTM. The majority voting applied on the predictions of the three classifiers, which obtained a low score of 39.2% F1-score on the test set while achieving a 75.2% F1-score on the development set. The authors refer to the fact that there is a big difference between the training and testing sets, and the dataset did not shuffle before the split. In [15], Kokatnoor and Krishnan proposed a stacked weighted ensemble model for the binary classification of hate speech. They ensembled five classifiers, namely LR, NB, RF, soft voting, and hard voting. The authors observed an improvement in the results obtained by the proposed approach compared with the stand-alone classifiers, as they achieved an accuracy of 95.54%.

An ensemble approach was proposed by Gao and Huang [16] to detect hate speech. The model used LR and neural network models, LSTM. The authors assessed each model individually as a stand-alone model, then ensembled them and made a comparison. The ensemble model outperformed both of the stand-alone models by 7% in F1-score on the Fox News corpus. In [17], MacAvaney et al. proposed another stacked ensemble of multiple SVMs. The authors used one different feature with each SVM. For the evaluation of their model, they tested their model on four hate speech datasets in terms of accuracy and F1-score. Their model performed less than the neural ensemble model on two out of four datasets. Furthermore, the authors discussed the challenges of automatic detection for hate speech in social media.

Another study that provided evidence that the ensemble approach can achieve significant performance over an individual classifier is Zimmerman et al. [18], who proposed an ensemble method in which ten CNNs with various weight parameters/initializations are combined. They performed 10-fold cross-validation on their method using the best settings with 10 epochs and batch size. Two datasets were used for the evaluation of the proposed method, were split into 85% for training and 15% for testing. The authors used word embedding as a feature. They achieved a higher performance compared with the individual classifiers with an average improvement of 1.97% F1-score.

Zhang et al. [19], employed a combination of CNN and gated recurrent unit network (GRU) trained using the word2vec feature for hate speech detection. They evaluated their approach on different datasets. The authors achieved higher performance compared with the state-of-the-art. They conducted a comparison between their results and the baselines and the state-of-the-art.

All datasets used in this study are retrieved messages from Twitter in the English language. Twitter is the third most popular social media platform in the world [20]. A large number of users use this platform to express their hate and anger toward another person or group; therefore, most of the papers on hate speech collect data from Twitter.

In this work, hate speech recognition is treated as a binary decision task that is expected to benefit the efforts in the domain. We aim to develop a model for binary text classification using a novel stacked ensemble to improve the performance. The stand-alone classifiers have various limitations, such as bias increment and variance. Therefore, this work is motivated by the fact that using the stacked ensemble can overcome these concerns and improve the performance of stand-alone classifiers. The main contributions of this work include:

- Develop a novel stacked ensemble model for binary detection of hate speech on social media platforms;
- Use of the four publicly available hate speech datasets with varying sizes to allow comparison of future work;
- Compare the proposed model's result against the standard stacking, single classifiers, majority voting, and state-of-the-art results;
- Improve the performance of the standard stacking approach.

This work is organized as follows. The proposed stacked ensemble architecture is presented in Section 2. The main phases of the proposed architecture, namely the preprocessing steps for cleaning the data, features employed training of base classifiers, and training of the meta classifier, which are discussed in this section. In Section 3, we present the publicly available datasets used in the experiments, and the results obtained from the stacked ensemble and non-stacked ensemble approaches. Section 4 presents and provides a discussion on the experimental results of the proposed approach, Base-Level classifiers using the test set, majority voting, and standard stacking. Furthermore, we compare our proposed architecture with the state-of-the-art for each dataset. Finally, Section 5 contains the conclusion and future work.

2. Materials and Methods

2.1. Proposed Approach

In this section, we present the novel proposed approach for detecting hate speech. The general idea of the stacked ensemble is using predictions of classifiers from the first level as input of the classifier of the next level. However, in our approach, we added the extracted features from the development dataset as input to the next level beside the predictions of the first level.

The general framework of the proposed stacking ensemble consists of two levels of classifiers: Base-Level classifiers at the first level and a Meta-Level classifier at the second level. As shown in Figure 1, the workflow of our proposed approach is composed of a preprocessing stage, feature extraction stage, a two-level stacked ensemble, and the final output. At the first step, the raw input data are preprocessed and prepared for feature extraction. Then, two features, namely word embedding, and USE are extracted at the feature extraction step. The data from the train set thus prepared is fed into the Base-Level classifiers (C1, C2, C3, . . . , Cn) at the first level of the stacked ensemble. Subsequently, the outputs of the Base-Level classifiers (the predictions provided by three Base-Level classifiers on the development set) together with the extracted features from the development set are fed into the Meta-Level classifier as inputs. Finally, the final prediction is obtained from the Meta-Level classifier. In the proposed stacking ensemble SVM, LR, and XGB are employed as Base-Level classifiers and LR is used as the Meta-Level classifier.

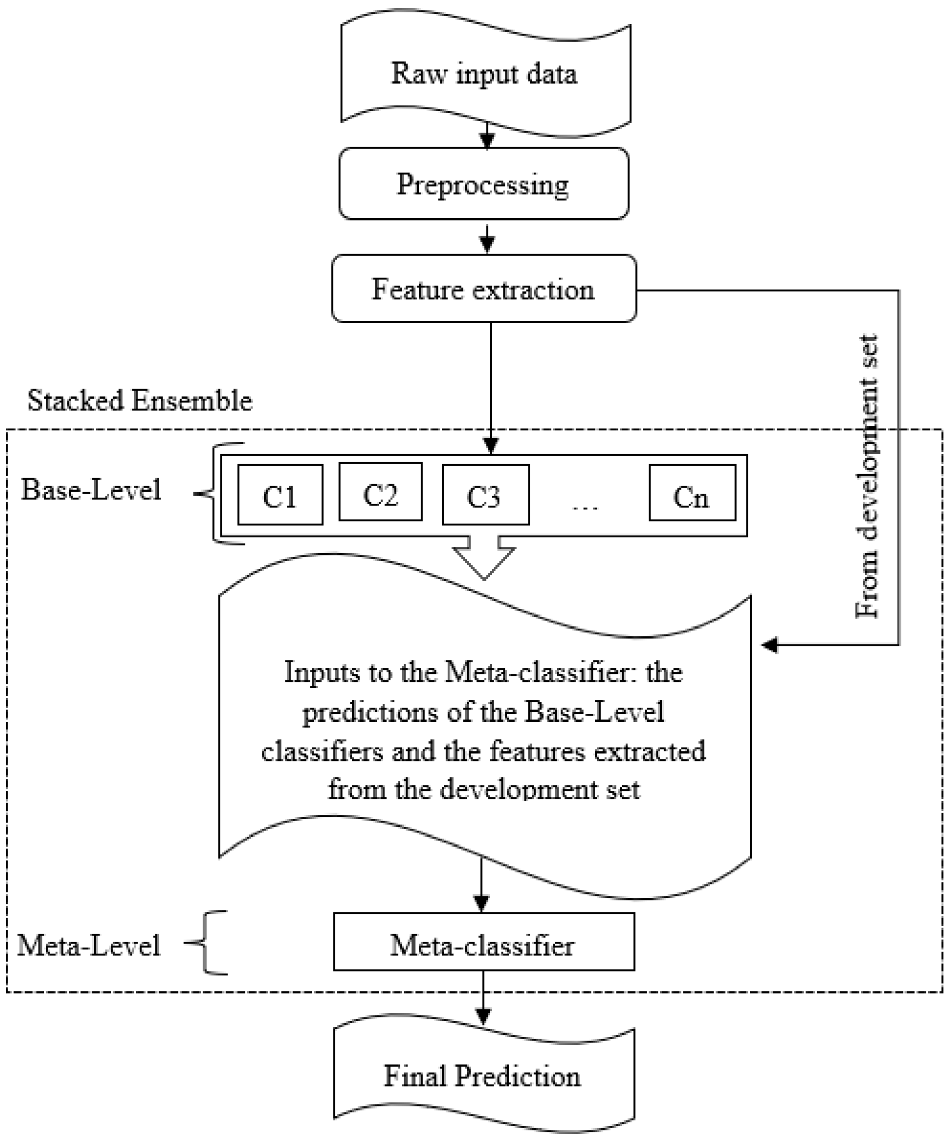

Figure 1. Proposed stacking ensemble architecture with two levels of classifiers.

The proposed architecture encompasses two training phases. The Base-Level classifiers are trained on the training set using the extracted features in the first training phase. In the second training phase, the Meta-Level classifier is trained using the development set and the outputs of the Base-Level classifiers trained in the first phase. The same preprocessing and feature extraction stages applied to the training data are also applied to the development data. The pre-trained Base-Level classifiers are used to make their individual predictions. The Meta-Level classifier is trained with the Base-Level predictions and the extracted features from the development set. Once the training of the Base-Level and Meta-Level classifiers are completed, the architecture is constructed. The preprocessing and feature extraction processes are repeated for the test set. The Meta-Level classifier generates the final result, which is the predictions on the test set.

2.1.1. Preprocessing

This step is applied to filter and prepare the dataset for the model. The tweets of the dataset are preprocessed prior to feature extraction as listed below:

- Hashtag symbols '#' are removed;
- Mentions '@' are removed;
- URLs are removed;
- All characters are changed to lowercase;
- All words are stemmed;
- Non-words or single characters are removed.

2.1.2. Feature Extraction

Due to their effectiveness in text classification, semantic relations between words or sentences have been used as features by numerous researchers [21]. In more recent studies, Cer et al. [22] and Kim [23] have demonstrated that embedding features significantly improves the classification performance in hate speech.

Therefore, we used both 'word to vector' and 'sentence to vector' embedding features to ensure that semantic information at both sentence and document levels are represented.

All the Base-Level classifiers used in this study employed concatenation of the following features:

- Word embeddings: we employed word2vec to represent each word by a vector of size 200, with the skip-gram technique, which predicts the context from the given word;
- Sentence embeddings: we employed USE to encode tweets into vectors [22]. The encoder takes preprocessed text as input and outputs the sentence embeddings as vectors with 512 dimensions.

Therefore, diversity among the base classifiers is ensured through the use of different machine learning algorithms.

2.1.3. Base-Level Classifiers

We trained seven of the most frequently used machine learning algorithms, namely K-Nearest Neighbor (KNN), LR, SVM, NB, RF, Extra trees (E-trees), and XGB, using 3-fold cross-validation.

KNN is a nonparametric supervised machine learning method known as lazy learning as it has no training stage. The assumption in KNN is that similar data can be found in the same neighborhood. By the very nature of its decision rule, the performance of KNN classification depends crucially on the way that distances or similarities are computed between different examples [24].

LR is a supervised machine learning algorithm that has been successfully utilized to solve various classification problems. It is a classification approach with two types: binary (two classes) and multi classes (more than two classes). A prediction is made by feeding a set of features to LR; it is referred to as a probabilistic classifier since it predicts the probability of an output.

SVM is a very frequently used non-probabilistic binary classifier. It is used to classify linearly separable data into two classes, such as hate and no-hate, in two-dimensional space. One-vs-all or one-vs-one approaches may be used to adapt SVM to multiclass classification. The goal of SVM classification is to maximize the margin separating the target classes.

NB is a simple but efficient probabilistic classifier based on the Bayes theorem with the assumption that all features are independent of each other. An NB classifier uses the Bayes theorem to compute each data item's probability of belonging to each class. The data item is tagged with the class that has the highest probability.

RF is a large group of decision trees developed by Breiman in 2001, it is an ensemble model that employs several classification trees [25]. For each data item to be classified, each tree predicts a class; then, the class with the majority votes wins. The individual trees in the RF are trained on random subsets of the training data using different features for splitting the data. These two properties of the training phase ensure that the individual decision trees are uncorrelated, and thus, the ensemble classifier RF produces a better prediction.

E-tree is a form of ensemble classification algorithm that outputs a classification result by combining the results of several less correlated decision trees gathered in a forest. E-tree is a variation of the RF model where the entire training set is used for training, whereas the RF trains the individual classifiers on subsets of the training data.

XGB is an ensemble method based on Gradient Boosting, where the gradient descent algorithm is used to minimize loss. An iterative training process where a new tree trained using the previously incorrectly classified data items added to the ensemble is employed.

2.1.4. Meta-Level Classifier

The meta classifier used in the proposed architecture is LR which has recently been the most frequently used Meta-Level classifier in the stacking approach, according to Agarwal and Chowdary [26]. We chose the LR as the Meta-Classifier because it requires fewer parameters and it is faster to train [27,28].

In the proposed stacked ensemble, the pre-trained Base-Level classifiers are employed to predict the category of the tweets in the development set. During the training of the meta classifier LR, the predictions of the Base-Level classifiers are used, along with the extracted features from the development set. After this training phase, the architecture utilizes Base-Level classifiers trained on the training set and the Meta-Level classifier trained on the development set for the predictions on unseen data.

2.1.5. Evaluation Metrics

We present the results of the proposed approach using the F1-score, which is the harmonic mean of recall and precision, as an evaluation metric for comparison with other work. The F1-score is a frequently used metric for the classification model effectiveness [28]. Additionally, in all experiments, accuracy was calculated and presented. The formulas of the metrics presented in this study are listed below.

$$\text{Accuracy} = (\text{TP} + \text{TN})/(\text{TP} + \text{TN} + \text{FP} + \text{FN}) \tag{1}$$

$$\text{Precision} = (\text{TP})/(\text{TP} + \text{FP}) \tag{2}$$

$$\text{Recall} = (\text{TP})/(\text{TP} + \text{FN}) \tag{3}$$

$$\text{Specificity} = (\text{TN})/(\text{TN} + \text{FP}) \tag{4}$$

$$\text{F1-score} = 2(\text{Precision} \times \text{Recall})/(\text{Precision} + \text{Recall}) \tag{5}$$

where TP, TN, FN, and FP represent the: True Positive, True Negative, False Negative, and False Positive, respectively.

3. Experimental Setup and Results

In order to evaluate the performance of the proposed architecture, we tested the proposed stacked ensemble using different machine learning classifiers as Base-Level classifiers. Additionally, we compared the performance of the proposed system to that of the Base-Level classifiers individually and the full ensemble of Base-Level classifiers using majority voting and the standard stacking approach. In each setting of the stacking ensembles, the Meta-Level classifier used was LR. The F1-scores of these different combinations of Base-Level classifiers are presented in the following discussion.

This section introduces details about the used datasets, preprocessing, and feature extractions applied for the hate speech detection model. All the experiments utilized Python 3.7.

3.1. Datasets

Table 1 presents the four publicly available datasets from Twitter used in this work. For training the stacking model, three sets, namely train, development, and test set, are needed. The training set is used to train the Base-Level classifiers, and the development set is used to train the Meta-Level classifier. The HatEval dataset was already available in three sets. The datasets were first randomly split into training and testing sets. Furthermore, we divided the training set to obtain 10% for the development set, similar to Abuzayed et al. [29]. This splitting aims to verify the performance of the approach, as it is tested on unknown data. We applied different splits on the dataset to perform a comparison with the state-of-the-art as they used different split percentages on each one.

Table 1. Datasets used in this study.

Dataset Name	Link
HatEval	http://hatespeech.di.unito.it/hateval.html (accessed on 30 June 2021)
Davidson	https://data.world/thomasrdavidson/hate-speech-and-offensive-language
COVID-HATE	http://claws.cc.gatech.edu/covid/#dataset
ZeerakW	https://github.com/ZeerakW/hatespeech/blob/master/NAACL_SRW_2016.csv

3.1.1. HatEval Dataset

The HatEval dataset is considered a challenging dataset since the labeling of the tweets is not unarguably correct. The tweets in the development and test sets target different minority groups, and the two sets have different word distributions. The HatEval dataset distribution of binary classification is presented in Table 2. According to the definition of hate speech, this dataset contains hateful tweets that attack a group based on gender or ethnicity (immigrants or women).

Table 2. HatEval dataset distribution of Non-hateful and Hateful tweets.

	Non-Hateful Tweet (0)	**Hateful Tweet (1)**	**Total**
Train	5217 (58%)	3783 (42%)	9000
Development	573 (57%)	427 (43%)	1000
Test	1740 (58%)	1260 (42%)	3000
Total	7530 (58%)	5470 (42%)	13,000

3.1.2. Davidson Dataset

Davidson dataset, which contains 24,783 tweets, is the largest dataset for hate speech collected from Twitter [8]. The tweets in the dataset are labeled as ‘Hate’, ‘Offensive’, and ‘Neither’. We combined the tweets with labels ‘Hateful’ and ‘Offensive’ to be ‘Hateful Tweet’ and the tweets with ‘Neither’ label to become ‘Non-hateful Tweet’ according to the label combinations used by Salminen et al. [30]. The ‘Hateful Tweet’ in this dataset are the tweets that attack a woman, Table 3 presents the distribution of the Hateful and Non-hateful Tweets across the training, development, and test sets of the Davidson dataset.

Table 3. Davidson dataset distribution of Non-hateful and Hateful tweets.

	Non-Hateful Tweet (0)	**Hateful Tweet (1)**	**Total**
Train	3010 (17%)	14,833 (83%)	17,843
Development	327 (17%)	1656 (83%)	1983
Test	826 (17%)	4131 (83%)	4957
Total	4163 (17%)	20,620 (83%)	24,783

3.1.3. COVID-HATE Dataset

The COVID-HATE dataset contains tweets containing hate speech sparked by the global spread of the COVID-19 pandemic. The hate speech in this dataset is targeted in general at the Asian race and in particular at Chinese communities. The COVID-HATE dataset was manually annotated to separate the 2319 tweets into four classes which are 'Hate', 'Non-Asian Aggression', 'Counter Hate', and 'Neutral'. We combined the 'Hate' and 'Non-Asian Aggression' categories into the 'Hateful Tweet' category, and 'Counter Hate' and 'Neutral' categories into the 'Non-hateful Tweet' category. Table 4 shows the distribution of the Hateful and Non-hateful tweets used to train the ML classifiers presented in this work.

Table 4. COVID-HATE dataset distribution of Non-hateful and Hateful tweets.

	Non-Hateful Tweet (0)	**Hateful Tweet (1)**	**Total**
Train	965 (58%)	704 (42)	1669
Development	106 (57%)	80 (43%)	186
Test	249 (54%)	215 (46%)	464
Total	1320 (57%)	999 (43%)	2319

3.1.4. ZeerakW Dataset

This dataset contains 16,135 tweets labeled as 'Sexism', 'Racism', and 'Neither'. In this study, we combined the tweets labeled as 'Sexism' and 'Racism' to be labeled as 'Hateful Tweet', which we refer to as 1, and the tweets labeled as 'Neither' to be under the 'Non-hateful Tweet' label which is 0 as shown in Table 5. The 'Hateful Tweet' here is the tweet that attacks a person or group based on gender or race.

Table 5. ZeerakW dataset distribution of Non-hateful and Hateful tweets.

	Non-Hateful Tweet (0)	**Hateful Tweet (1)**	**Total**
Train	7935 (68%)	3682 (32%)	11,617
Development	877 (68%)	414 (32%)	1291
Test	2221 (69%)	1006 (31%)	3227
Total	11,033 (68%)	5102 (32%)	16,135

It is observed that the datasets Davidson (17% Non-hateful—83% Hateful tweets) and ZeerakW (68% Non-hateful—32% Hateful tweets) are imbalanced. Therefore, to obtain balanced data, we oversampled the minority class using SMOTE [31]. Thus, for the Davidson training set, we oversampled Non-hateful tweets, and for the ZeerakW training set, we oversampled the Hateful tweets. In [32], the SMOTE method was used to balance the data, improving the classification performance.

Moreover, as we mentioned previously, we applied different percentages to split the datasets except for HatEval. In line with the approach used by Nobata et al. [33] and Gibert et al. [34], we split the other datasets into 80% for the training and 20% for the testing set. The COVID-HATE dataset is divided into 80% for the training set and 20% for the testing set. The other two datasets have another splitting. These datasets are ZeerakW with 85% for the training set and 15% for the testing set, and the Davidson dataset with 75% and 25% for training and testing sets, respectively.

3.2. Results from Proposed Stacking Experiment

After the Base-Level classifiers are trained using training data, we tested their performance on the development data in each dataset. Table 6 presents the F1-score of each Base-Level classifier used in the proposed stacking architecture on the development set. These Base-Level classifiers were employed in the stacking architecture with five different combinations. Table 7 shows the performance of the proposed stacking ensembles on the test set. The Meta-Level classifier in each proposed stacking ensemble is trained using the development set.

Table 6. F1-score of the Base-Level classifiers on the development set. Bold entries show the highest performance for each classifier on each dataset.

Base-Level Classifier	HatEval	Davidson	COVID-HATE	ZeerakW
KNN	0.6061	0.9296	0.6982	0.7480
LR	0.6434	0.9343	0.7761	**0.8406**
SVM	0.6180	**0.9517**	0.7632	0.8398
NB	**0.6845**	0.8669	0.7201	0.7402
RF	0.6443	0.8671	**0.7963**	0.6081
E-tree	0.5808	0.9358	0.7745	0.7180
XGB	0.6745	0.9500	0.7960	0.8253

Table 7. F1-score of the stacking ensembles using different base classifiers on test sets. Bold entries show the highest performance for each classifier on each dataset.

Base-Classifiers Combination	HatEval	Davidson	COVID-HATE	ZeerakW
SVM, LR, XGB	**0.6551**	**0.9713**	**0.7301**	**0.7392**
KNN, SVM, NB	0.6405	0.9613	0.6920	0.7049
LR, NB, RF	0.6407	0.9601	0.7136	0.7226
KNN, LR, NB	0.6410	0.9710	0.7261	0.7150
SVM, LR, E-tree	0.6428	0.9710	0.7255	0.7253

In our experiment, we tried all different combinations of the seven individual classifiers. Among all combinations of these single classifiers, we present the top five combinations in terms of the F1-score. Examining the five ensembles using the four test sets, we found that the ensemble of SVM, LR, and XGB is the best combination among other ensembles. As Table 7 shows, the best results for all datasets are achieved by the proposed architecture containing SVM, LR, XGB as Base-Level classifiers.

As Table 6 showed, NB performed worst on the Davidson dataset, while KNN, RF, and E-tree performed worst on COVID-HATE, ZeerakW, and HatEval datasets, respectively. On the other hand, NB and RF performed the best on the HatEval and COVID-HATE datasets, respectively, while SVM achieved the best performance with the Davidson dataset, and LR was the best on ZeerakW datasets. We observed that the stacking ensemble combination of (KNN, SVM, NB) performed the worst on three out of four datasets (HatEval, COVID-HATE, and ZeerakW).

3.3. Results from Single Classifier Experiment

We trained seven single classifiers with the train set of the four datasets, using the same stages of preprocessing and feature extractions. Table 8 presents the performance of the single classifiers on test sets in terms of the F1-score.

Table 8. F1-score of the Base-Level classifiers on the test set. Bold entries show the highest performance for each classifier on each dataset.

Base-Level Classifier	HatEval	Davidson	COVID-HATE	ZeerakW
KNN	0.5885	0.9281	0.6580	0.6037
LR	**0.6407**	0.9365	0.7146	**0.7179**
SVM	0.6394	**0.9530**	0.6843	0.7030
NB	0.6024	0.8745	0.6539	0.5508
RF	0.6016	0.8797	0.6710	0.6274
E-tree	0.6031	0.9397	0.6542	0.6747
XGB	0.6353	0.9498	**0.7246**	0.7058

It can be seen that the SVM classifier produced the best performance on two of the datasets, Davidson and ZeerakW. For the other two datasets HatEval and COVID-HATE, the single classifiers LR and XGB, respectively, achieved the best F1-score.

We tested the Base-Level classifiers on each dataset's development and test sets and presented the results in Tables 6 and 8, respectively. The challenge here is that the most commonly used words in the test and development sets are different, which may adversely affect the performance of our Meta-Level classifier leading to misclassification. Fortuna et al. [35] also attributed the low performance of their system to the inconsistency between train and test sets, which was also observed by [14]. It can be observed in Tables 6 and 8 that, as mentioned before, the data in the test, development, and training portion of the dataset vary, and that is reflected in the performance of the Base-Level classifiers. Moreover, it can be argued that training the Meta-Level classifier on development data might be harmful.

3.4. Results from Majority Voting Experiment

In this experiment, we used the same combinations of the Base-Level classifiers in the proposed stacking approach to compare the results in the two approaches. As shown in Table 9, the results achieved by this approach are not suitable as the ones achieved by the proposed stacking approach. The improvements of the proposed stacking approach over the majority voting are 1.44%, 1.61%, 0.55%, and 1.71% on HatEval, Davidson, COVID-HATE, and ZeerakW datasets, respectively.

Table 9. F1-score of the majority voting ensembles using different base classifiers on test sets. Bold entries show the highest performance for each classifier on each dataset.

Base-Classifiers Combination	HatEval	Davidson	COVID-HATE	ZeerakW
SVM, LR, XGB	0.6296	0.9521	0.7050	**0.7221**
KNN, SVM, NB	0.6170	0.9509	0.7188	0.7005
LR, NB, RF	0.6135	0.9309	0.6950	0.6280
KNN, LR, NB	0.6167	0.9547	**0.7208**	0.6289
SVM, LR, E-tree	**0.6307**	**0.9552**	0.7081	0.7210

It can be seen in Table 9 that among the majority voting ensembles (LR, NB, RF) consistently has the lowest performance for all datasets. RF is an ensemble method consisting of tree predictions; these predictions are provided by majority voting of each tree [36]. In the (LR, NB, RF) combination, in most cases, RF agrees with LR or NB or both in wrong class predictions leading to the misclassification in the majority voting approach. Moreover, RF performs better with multiclassification tasks.

The majority voting ensemble containing SVM, LR, and E-tree provide the highest F1-score on both HatEval and Davidson datasets. The best F1-score on the COVID-HATE dataset is achieved with the (KNN, LR, NB) combination. Finally, for the ZeerakW dataset, the majority voting ensemble using the (SVM, LR, XGB) combination achieved the highest F1-score.

3.5. Results from Standard Stacking Experiment

This experiment is similar to the proposed approach except that we use only the predictions from the Base-Level classifiers as an input to the Meta-Level classifier (in the novel proposed approach we used the Base-Level predictions and the extracted features from the development set). F1-scores of the standard stacking are given in Table 10 on each dataset. The results achieved by this approach are very low on the HatEval dataset with 59.43% as the highest F1-score for the (SVM, LR, E-tree) combination. On the Davidson, COVID-HATE, and ZeerakW datasets, the best results were obtained by the (SVM, LR, XGB), (LR, NB, RF), and (KNN, LR, NB) combinations, respectively.

Table 10. F1-score of the standard stacking on test sets. Bold entries show the highest performance for each classifier on each dataset.

Base-Classifiers Combination	HatEval	Davidson	COVID-HATE	ZeerakW
SVM, LR, XGB	0.5910	**0.9517**	0.7036	0.6657
KNN, SVM, NB	0.5746	0.9434	0.6623	0.6806
LR, NB, RF	0.5804	0.9412	**0.7047**	0.6754
KNN, LR, NB	0.5873	0.9437	0.7046	**0.6815**
SVM, LR, E-tree	**0.5943**	0.9491	0.6859	0.6472

Table 11 demonstrates the best results from the four experiments in this work on all test sets. The experimental results on the test sets show that our proposed model achieved the best performance in terms of the F1-score over the other three approaches. Specifically, it can be seen in Table 11 that the proposed novel ensemble scheme outperforms the standard stacking approach for all datasets. These results provide evidence of the effectiveness of our system.

Table 11. Performance of the four hate speech recognition systems in terms of F1-score on tests sets of the four datasets. Bold entries show the highest performance for each classifier on each dataset.

Approaches	HatEval	Davidson	COVID-HATE	ZeerakW
Proposed Stacking	**0.6551**	**0.9713**	**0.7301**	**0.7392**
Standard Stacking	0.5943	0.9517	0.7047	0.6815
Single classifiers	0.6307	0.9530	0.7208	0.7179
Majority voting	0.6407	0.9552	0.7246	0.7221

4. Discussion

Table 12 illustrates the summary of the state-of-the-art on used datasets. Furthermore, Table 13 compares the proposed approach with the state-of-the-art for each of the four datasets. Zhang et al. [19], outperformed the state-of-the-art result on Davidson scoring a 94% F1-score. Our proposed approach that combined the USE and word2vec features and two-level stacking ensemble surpassed that result by 3.1%, indicating that USE contributed to the classification performance. For the same dataset, it can be seen from Table 9 that unlike the proposed stacking ensembles listed in Table 7, majority voting improves the performance of the base classifiers in only one combination. The results of the SemEval-2019 competition for Task 5 HatEval can be found in [37]. Even though the proposed system outperformed the highest-ranked system, which employed SVM with sentence embeddings features, the performance improvement is not as pronounced as the one on the Davidson dataset. For the ZeerakW dataset, the best result for binary classification was achieved by Zimmerman et al. [18] using ensemble neural networks with various weight initialization along with word embedding features. To the best of our knowledge, there are no published results on binary hate speech classification using the COVID-HATE dataset.

Table 12. Summary of the state-of-the-art approaches on used datasets.

State-of-the-Art	Dataset	F1-Score	Model	Features
Indurthi et al. [9]	HatEval	65.1%	SVM	USE
Zhang et al. [19]	Davidson	94.0%	CNN, GRU (Ensemble)	word2vec
Zimmerman et al. [18]	ZeerakW	77.8%	Neural networks (Ensemble)	word2vec

We observed that LR and SVM were the most successful models among all the used models on the test set in terms of the F1-score. As shown in Table 7 in the proposed stacking ensemble approach, the two best combinations have the LR and SVM in them (SVM, LR, XGB and SVM, LR, E-tree). One can argue that these combinations were the best because they used LR as a Meta-Level classifier; however, LR was also the best in the other two approaches that did not utilize a Meta-Level classifier. In Table 8, LR achieved the best F1-score on two datasets: HatEval and ZeerakW, among Base-Level classifiers. Furthermore, in the majority voting, LR was in the combinations that achieved the highest performed F1-score on each dataset, (SVM, LR, XGB), (KNN, LR, NB), and (SVM, LR, E-tree) as shown in Table 9.

Table 13. F1-score (%) of the state-of-the-art and the proposed stacking ensemble on all datasets. Bold entries show the highest performance for each dataset.

Dataset	State of the Art	Proposed Stacking Ensemble
HatEval	65.1 [9]	**65.5**
Davidson	94.0 [19]	**97.1**
COVID-HATE	-	**73.0**
ZeerakW	**77.8** [18]	73.9

When the results of the standard stacking approach from Table 10 are compared with the results of the proposed stacking approach, it is seen that the proposed approach outperformed the standard stacking approach on all datasets in terms of the F1-score. Table 11 shows that among all approaches, the standard stacking approach had the worst performances on all datasets. According to these results, our novel stacking method achieved the highest F1-scores across all datasets in all four experiments. Among all approaches, standard stacking performed the worst, whereas novel stacking performed the best. This improvement confirmed the effectiveness of our novel approach.

As shown in Table 13, the proposed approach outperformed the state-of-the-art in both HatEval and Davidson datasets. Our approach has a lower performance than the state-of-the-art only on the ZeerakW dataset. This may be attributed to the combination of the two features word2vec and USE employed here. Indeed, Waseem and Hovy [6] observed that using character N-grams on this dataset outperformed other features. They also noted that the other features had an adverse effect on the performance. Goldberg [38] stated that in comparison with N-gram features, which eliminate the concept of location in a text (apart from immediate surrounding terms for bi/tri-grams), CNN and word embeddings can utilize a sequence of tokens by concatenating token embeddings into a matrix. In fact, Zimmerman et al. [18], achieved a higher F1-score than our approach by using this combination. In addition, combining these features with the LR (which performed the best, as explained previously) improves the performance. This observation is consistent with our findings on the ZeerakW dataset, where the proposed approach's result was the only one that was lower than the state-of-the-art.

It should be noted that no specific combination of classifiers or a specific single classifier can deliver the highest performance for all types of datasets. Each dataset has different characteristics such as the context of the tweet [14], the targets of the hate speech, the classes used for categorizing the tweets, the distribution of the classes, and the overall dataset size. All these unique characteristics of the datasets indicate that specially tailored systems with architecture and features best suited for a particular dataset may not generalize well and adapt to the dynamic hate speech domain. Overall, the proposed approach can improve performance by overcoming the inadequacies of the base classifiers while considering the dataset's appropriate features. The results presented here show that the proposed architecture with the (SVM, LR, XGB) combination as Base-Level classifiers improves the performance of the base classifiers, and in general, outperformed most of the state-of-the-art.

5. Conclusions

In this work, we proposed a novel stacking ensemble for distinguishing between hateful and non-hateful tweets from English Twitter datasets of various sizes.

We applied the proposed stacking classifiers approach at two levels: Base and Meta levels. The Base-Level classifiers are trained using the features extracted from the training set. Then, the Meta-Level classifier is trained on the predictions of the Base-Level classifiers using the development set and the features extracted from the development set. The preprocessed unseen test set was then classified using our model. On all of the datasets studied, we compared the results of the proposed approach to those of the single classifiers, majority voting ensembles, standard stacking, and the state-of-the-art. The results indicated that distinguishing Hateful and Non-hateful tweets is a challenging task. Compared with single classifiers, standard stacking, and majority voting, the proposed technique achieved the highest F1-score on all datasets. Furthermore, the proposed approach outperformed the state-of-the-art in three out of the four datasets using the same combinations of features (word2vec and USE), and the same combination of Base-Level classifiers (SVM, LR, XGB), with LR as Meta-Level classifier. Even though there are two phases for training the classifiers, this is justifiable as it is not repetitive. Nonetheless, the features that may be useful for correct classification are related to the nature and composition of the data to be classified; thus, fine-tuning the feature engineering stage may improve the performance even further. More comprehensive datasets that include a more extended timeframe would exemplify the time-varying nature of the tweets and allow the classifiers to capture the dynamic nature of hate speech.

Future work will address the imbalance problem. The data imbalance where the distribution of samples per class is not equal influences machine learning classifiers in favor of the majority class. This is not desirable especially in cases such as hate speech recognition where identifying the minority class is more important. Therefore, balancing the dataset used for training the base classifiers in our approach is expected to improve the performance of the proposed system. Resampling data for balancing the dataset can be achieved by oversampling the minority class and if needed, undersampling the majority class. During this process it is important to ensure useful information is not lost due to undersampling and the oversampled data do not cause overfitting in machine learning classifiers. Our proposed architecture will be extended by employing alternative resampling methods at the base-classifier training phase of the proposed architecture. Specifically, the effect of the minority oversampling techniques on the final performance will be studied. Furthermore, the proposed stacking approach will be applied to different domains of text classification with binary as well as multiple classes.

Author Contributions: Conceptualization, N.D. and M.K.A.A.; methodology, N.D., and M.K.A.A.; software, M.K.A.A.; validation, M.K.A.A. and N.D.; visualization, N.D., and M.K.A.A.; data curation, M.K.A.A.; formal analysis, N.D. and M.K.A.A.; writing—original draft preparation, N.D. and M.K.A.A.; writing—review and editing, N.D., and M.K.A.A.; supervision, N.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are openly available in repositories listed in Table 1.

Acknowledgments: This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

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

References

1. Sachdeva, J.; Chaudhary, K.K.; Madaan, H.; Meel, P. Text based hatespeech analysis. In Proceedings of the International Conference Artifitial Intellegent Smart System (ICAIS), Coimbatore, India, 25–27 March 2021; pp. 661–668.
2. Ibrohim, M.O.; Budi, I. Multi-label hate speech and abusive language detection in Indonesian Twitter. In Proceedings of the 3rd Workshop Abusive Language Online, Florence, Italy, 3 August 2019; pp. 46–57.
3. Graff, M.; Miranda-Jiménez, S.; Tellez, E.; Ochoa, D.A. INGEOTEC at SemEval-2019 task 5 and task 6: A genetic programming approach for text classification. In Proceedings of the 13th International Workshop Semantic Evaluation, Minneapolis, MN, USA, 6–7 June 2019; pp. 639–644.

4. Roy, P.K.; Tripathy, A.K.; Das, T.K.; Gao, X.Z. A Framework for Hate Speech Detection Using Deep Convolutional Neural Network. *IEEE Access* **2020**, *8*, 204951–204962.
5. Mohapatra, S.K.; Prasad, S.; Bebarta, D.K.; Das, T.K.; Srinivasan, K.; Hu, Y.-C. Automatic Hate Speech Detection in English-Odia Code Mixed Social Media Data Using Machine Learning Techniques. *Appl. Sci.* **2021**, *11*, 8575. [CrossRef]
6. Waseem, Z.; Hovy, D. Hateful symbols or hateful people? Predictive features for hate speech detection on twitter. In Proceedings of the NAACL Student Research Workshop, San Diego, CA, USA, 13–15 June 2016; pp. 88–93. [CrossRef]
7. 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. Available online: https://ojs.aaai.org/index.php/ICWSM/article/view/14955 (accessed on 29 June 2021).
8. Ibrohim, M.O.; Budi, I. A dataset and preliminaries study for abusive language detection in Indonesian social media. *Procedia Comput. Sci.* **2018**, *135*, 222–229. [CrossRef]
9. Indurthi, V.; Syed, B.; Shrivastava, M.; Chakravartula, N.; Gupta, M.; Varma, V. Fermi at semeval-2019 task 5: Using sentence embeddings to identify hate speech against immigrants and women in twitter. In Proceedings of the 13th International Workshop on Semantic Evaluation, Minneapolis, MN, USA, 6–7 June 2019; pp. 70–74.
10. Wolpert, D.H. Stacked generalization. *Neural Netw.* **1992**, *5*, 241–259. [CrossRef]
11. Oza, N.C.; Tumer, K. Classifier ensembles: Select real-world applications. *Inform. Fusion* **2008**, *9*, 4–20. [CrossRef]
12. Kaggle. Available online: http://kaggle.com (accessed on 15 July 2021).
13. Badjatiya, P.; Gupta, S.; Gupta, M.; Varma, V. Deep learning for hate speech detection in tweets. In Proceedings of the 26th International Conference on World Wide Web Companion, Perth, Australia, 3–7 April 2017; pp. 759–760. [CrossRef]
14. Aria, N.; Vermeer, F.; Wiltvank, G.; Goot, R. Sthruggle at SemEval-2019 Task 5: An ensemble approach to hate speech detection. In Proceedings of the 13th International Workshop on Semantic Evaluation, Minneapolis, MN, USA, 6–7 June 2019; pp. 484–488.
15. Kokatnoor, S.A.; Krishnan, B. Twitter hate speech detection using stacked weighted ensemble (SWE) model. In Proceedings of the 2020 Fifth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN) 2020, Bangalore, India, 26 November 2020; pp. 87–92. [CrossRef]
16. Gao, L.; Huang, R. Detecting online hate speech using context aware models. *arXiv* **2017**, arXiv:1710.07395. Available online: https://aclanthology.org/2020.lrec-1.758 (accessed on 28 June 2021).
17. MacAvaney, S.; Yao, H.R.; Yang, E.; Russell, K.; Goharian, N.; Frieder, O. Hate speech detection: Challenges and solutions. *PLoS ONE* **2019**, *14*, e0221152.
18. Zimmerman, S.; Kruschwitz, U.; Fox, C. Improving hate speech detection with deep learning ensembles. In Proceedings of the Eleventh International Conference on Language Resources and Evaluation, Miyazaki, Japan, 7–12 May 2018; pp. 2546–2553.
19. Zhang, Z.; Robinson, D.; Tepper, J. Detecting hate speech on twitter using a convolution-gru based deep neural network. In Proceedings of the European Semantic Web Conference, Anissaras, Crete, Greece, 3–7 June 2018; Springer: Cham, Switzerland, 2018. [CrossRef]
20. Antonakaki, D.; Fragopoulou, P.; Ioannidis, S. A survey of Twitter research: Data model, graph structure, sentiment analysis and attacks. *Expert Syst. Appl.* **2021**, *164*, 114006. [CrossRef]
21. Camacho-Collados, J.; Pilehvar, M.T. From word to sense embeddings: A survey on vector representations of meaning. *J. Artif. Intell. Res.* **2018**, *63*, 743–788. [CrossRef]
22. Cer, D.; Yang, Y.; Kong, S.Y.; Hua, N.; Limtiaco, N.; John, R.S.; Constant, N.; Guajardo-Céspedes, M.; Yuan, S.; Tar, C.; et al. Universal sentence encoder. *arXiv* **2018**, arXiv:1803.11175.
23. Kim, Y. Convolutional neural networks for sentence classification. In Proceedings of the EMNLP, Doha, Qatar, 25–29 October 2014; pp. 1746–1751. [CrossRef]
24. Weinberger, K.Q.; Saul, L.K. Distance metric learning for large margin nearest neighbor classification. *J. Mach. Learn. Res.* **2009**, *10*, 207–244.
25. Breiman, L. Bagging predictors. *Mach. Learn.* **2001**, *45*, 123–140. [CrossRef]
26. Agarwal, S.; Chowdary, C.R. A-stacking and A-bagging: Adaptive versions of ensemble learning algorithms for spoof fingerprint detection. *Expert Syst. Appl.* **2020**, *146*, 113160. [CrossRef]
27. Van Thin, D.; Le, L.S.; Nguyen, N.L.T. Nlp@ uit: Exploring feature engineer and ensemble model for hate speech detection at vlsp 2019. *Training* **1991**, *5*, 3–51.
28. Verma, G.; Chhaya, N.; Vinay, V. To target or not to target: Identification and analysis of abusive text using ensemble of classifiers. *arXiv* **2020**, arXiv:2006.03256.
29. Abuzayed, A.; Elsayed, T. Quick and simple approach for detecting hate speech in Arabic tweets. In Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection, Marseille, France, 11–16 May 2020.
30. Salminen, J.; Hopf, M.; Chowdhury, S.A.; Jung, S.G.; Almerekhi, H.; Jansen, B.J. Developing an online hate classifier for multiple social media platforms. *Human-Cent. Comput. Inform. Sci.* **2020**, *10*, 1–34. [CrossRef]
31. Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: Synthetic minority over-sampling technique. *J. Artif. Intell. Res.* **2002**, *16*, 321–357. [CrossRef]
32. Li, Y.; Nie, X.; Huang, R. Web spam classification method based on deep belief networks. *Expert Syst. Appl.* **2018**, *96*, 261–270. [CrossRef]

33. Nobata, C.; Tetreault, J.; Thomas, A.; Mehdad, Y.; Chang, Y. Abusive language detection in online user content. In Proceedings of the 25th International Conference on World Wide Web, Montréal, QC, Canada, 11–15 April 2016; International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva: Geneva, Switzerland, 2016; pp. 145–153. [CrossRef]
34. de Gibert, O.; Perez, N.; García-Pablos, A.; Cuadros, M. Hate speech dataset from a white supremacy forum. In Proceedings of the 2nd Workshop on Abusive Language Online, Brussels, Belgium, 31 October 2018.
35. Fortuna, P.; Nunes, S. Stop PropagHate at SemEval-2019 Tasks 5 and 6: Are abusive language classification results reproducible? In Proceedings of the 13th International Workshop on Semantic Evaluation, Minneapolis, MN, USA, 6–7 June 2019; pp. 745–752.
36. Lorena, A.C.; Jacintho, L.F.; Siqueira, M.F.; De Giovanni, R.; Lohmann, L.G.; De Carvalho, A.C.; Yamamoto, M. Comparing machine learning classifiers in potential distribution modelling. *Expert Syst. Appl.* **2011**, *38*, 5268–5275. [CrossRef]
37. Basile, V.; Bosco, C.; Fersini, E.; Debora, N.; Patti, V.; Pardo, F.M.R.; Rosso, P.; Sanguinetti, M. Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter. In Proceedings of the 13th International Workshop on Semantic Evaluation, Minneapolis, MN, USA, 6 June 2019; pp. 54–63.
38. Goldberg, Y. A primer on neural network models for natural language processing. *J. Artif. Intell. Res.* **2016**, *57*, 345–420.