electronics

MDPI

Article

MTDOT: A Multilingual Translation-Based Data Augmentation Technique for Offensive Content Identification in Tamil Text Data

Vaishali Ganganwar and Ratnavel Rajalakshmi *

School of Computer Science and Engineering, Vellore Institute of Technology, Chennai 600127, Tamilnadu, India
* Correspondence: rajalakshmi.r@vit.ac.in

Abstract: The posting of offensive content in regional languages has increased as a result of the accessibility of low-cost internet and the widespread use of online social media. Despite the large number of comments available online, only a small percentage of them are offensive, resulting in an unequal distribution of offensive and non-offensive comments. Due to this class imbalance, classifiers may be biased toward the class with the most samples, i.e., the non-offensive class. To address class imbalance, a Multilingual Translation-based Data augmentation technique for Offensive content identification in Tamil text data (MTDOT) is proposed in this work. The proposed MTDOT method is applied to HASOC'21, which is the Tamil offensive content dataset. To obtain a balanced dataset, each offensive comment is augmented using multi-level back translation with English and Malayalam as intermediate languages. Another balanced dataset is generated by employing single-level back translation with Malayalam, Kannada, and Telugu as intermediate languages. While both approaches are equally effective, the proposed multi-level back-translation data augmentation approach produces more diverse data, which is evident from the BLEU score. The MTDOT technique proposed in this work achieved a promising improvement in F_1-score over the widely used SMOTE class balancing method by 65%.

Keywords: offensive content identification; imbalanced data; data augmentation; MuRIL; back translation

check for updates

Citation: Ganganwar, V.; Rajalakshmi, R. MTDOT: A Multilingual Translation-Based Data Augmentation Technique for Offensive Content Identification in Tamil Text Data. *Electronics* **2022**, *11*, 3574. https://doi.org/10.3390/electronics11213574

Academic Editor: Tae-Sun Chung

Received: 14 September 2022
Accepted: 21 October 2022
Published: 1 November 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The term 'offensive' refers to behavior that irritates, angers, or upsets a person or group. The rise of social media has led to a surge in offensive and hateful content on the internet. Because of the user's ability to remain anonymous, they believe they may express themselves freely and without constraints. All of these unpleasant comments deteriorate the mental health of the targeted individual or group. One way to avoid this from happening is to remove offensive comments. A manual approach to finding and removing the offending comment can take considerable effort. Therefore, automated systems that use machine learning and natural language processing tools are being applied in research and technology around the world to detect and reduce the use of offensive content on social media.

Several researchers have studied the issue of identifying offensive content, but as the amount of multilingual content has increased, it has become more challenging. The majority of research has focused on high-resource languages such as English, and only recently have low-resource languages such as different Indic languages been given more attention [1–3]. Dravidian languages, which are mainly spoken in south India and northeast Sri Lanka, include Tamil, Malayalam, Telugu, and Kannada. These languages also have a large number of offensive comments on social media, and there is a huge demand for automated systems that can classify into offensive and non-offensive regional YouTube comments. The limitations of this low-resource language include a small corpus and the lack of standard/benchmarked annotated data.

In this paper, the focus is on identifying YouTube comments written in Tamil script as either offensive or non-offensive. The dataset from the HASOC'21 shared task [4] that included YouTube comments in Tamil is used in this work. Every comment is classified as either offensive or non-offensive in this annotated corpus. The dataset included a total of 5877 YouTube comments, each labeled as offensive or non-offensive. The distribution of data among classes is substantially skewed, particularly for the offensive class. The number of offensive comments is considerably lower than that of non-offensive comments. Out of the 5877 comments, 4724 are not offensive and 1153 are offensive. The ratio of offensive comments to non-offensive comments is 1:4. As a result, the text classifiers become biased toward the non-offensive class which is having a large number of samples, and offensive class samples are frequently misclassified.

To address this class imbalance, a Multilingual Translation based Data augmentation technique for Offensive content identification in Tamil text data (MTDOT) is proposed in this work. In data augmentation, new data points are generated artificially from existing data points. Data augmentation is useful for reducing the cost of collecting and labeling data as well as improving model prediction accuracy. The class imbalance problem is addressed through data augmentation by generating artificial samples of a rare class in the dataset. Data augmentation approaches are divided into two categories: linguistic and non-linguistic [5]. The meaning is preserved after data augmentation in the linguistic category. In the linguistic category, the word or sentence is replaced or an entirely new statement is generated. Back translation is one such linguistic data augmentation techniques in which text is translated into another language and then back into the original language. In this work, offensive class comments are generated using the back-translation data augmentation technique.

Extensive experiments are conducted by applying the widely used SMOTE data level method [6], single-level back-translation augmentation method and our proposed MTDOT method for balancing the HASOC'21 Tamil dataset. Then, the text embedding vectors are generated for the balanced dataset using the MuRIL pre-trained model embedding layer. Six different classifiers namely Support Vector Machine, Naive Bayes, K-Nearest Neighbor (K-NN), Decision Tree, Random Forest and Majority Voting are trained using the text embedding vectors. The experimental findings demonstrate that the balanced dataset achieved precision, recall, and F_1-score of 0.82, 0.80, and 0.81, respectively, using the MTDOT class balancing approach.

The key contributions of this paper are:

- We proposed the data-level class balancing technique for addressing class imbalance in offensive content datasets. To the best of our knowledge, we are the first to use data augmentation for handling unequal class distribution in a non-English offensive content dataset i.e., Tamil dataset.
- In order to achieve a completely balanced dataset, new offensive comments are generated through single-level back translation and multi-level back translation using Malayalam and English as an intermediate language.
- Extensive experiments are conducted to demonstrate the effectiveness of proposed method using various classifiers and existing oversampling and undersampling methods.

The rest of this paper is structured as follows. Section 2 discusses existing work in offensive content detection and class balancing methods. Section 3 contains a description of the dataset used in our study. Section 4 describes the methodology, and in Section 5, details of the experiments conducted are provided. The results and discussion are in Section 6. Finally Section 7 concludes the paper.

2. Related Works

The existing works in offensive content identification are described in Section 2.1. In Section 2.2, the different existing techniques for handling class imbalance in text data are discussed.

2.1. Offensive Content Identification

Many researchers studied the problem of offensive content detection in social media comments in widely used English language, and some work is also existing for languages such as Hindi and German [1,2]. However, few efforts are made to identify offending content in low-resource Dravidian languages such as Tamil, Malayalam, and Kannada. The offensive content detection method for code-mixed language pairs such as Kannada–English, Malayalam–English, and Tamil–English is proposed by Garain et al. [7]. It is a multi-label classification model based on the ensemble of IndicBERT and generic BERT (Bidirectional Encoder Representations from Transformers) models.

Kedia and Nandy [8] proposed a multi-label classifier to classify code-mixed offensive language comments on social media platforms. They used multi-label classifiers including Multinomial NB, Linear SVM, and Random Forest to conduct the classification. Results significantly improved when a trained-from-scratch vanilla LSTM model and a transfer learning framework called ULMFiT were used. The proposed system is an ensemble of an AWD-LSTM-based model and two distinct transformer model architectures based on BERT and RoBERTa.

Jayanthi and Gupta [9] proposed a transformer model using multilingual BERT and XLM-RoBERTa. The model is an ensemble of six different mBERT and XLM-RoBERTa models. They also proposed a fusion architecture to boost performance by combining character-level, subword-level, and word-level embeddings. They applied their model to datasets in the languages Kannada, Malayalam and Tamil.

Ghanghor et al. [10] also proposed a model for the same dataset used by Jayanthi and Gupta [9]. They proposed a multilingual BERT cased model for offensive language detection and troll meme classification in Dravidian languages such as Tamil, Malayalam, and Kannada. To fix the class imbalance, loss functions such as NLL with class weights and Sadice were used. mBERT-cased was better than the other multilingual models used such as XLM-RoBERTa and IndicBERT. This model was used for text modality in the meme classification task.

Hate speech detection in a code-mixed dataset was proposed by Dave et al. [11] using Google's MuRIL pre-trained Transformer model. TF-IDF character n-grams and the MuRIL model are used for feature extraction from the text. Logistic regression and linear SVM models were used as classifiers to perform predictions.

Chinnappa and Dhivya [12] proposed a framework for hope speech prediction in the English, Tamil, and Malayalam datasets. The system uses a two-phase approach. In the first phase, a classifier is built to identify the text's language. In the second phase, the classifier is trained to distinguish between hope speech, non-hope speech, and not-lang labels. Ref. [13] proposed a transformer-based approach for identifying hate speech comments in code-mixed Tamil-English tweets. The dataset contains Tamil and English words and phrases. They first converted Tamil words to English using a Tamil to English mapping corpus. On the validation and test sets, the F_1-scores were 65% and 64%, respectively. Ref. [14] presented machine learning-based methods for the HASOC'21 dataset to identify offensive content. Additionally, they tried pre-trained multilingual transformer models including mBERT, MuRIL, and XLM-RoBERTa, of which XLM-RoBERTa achieved the maximum accuracy.

The studies on Tamil offensive content identification mostly used the HASOC-Dravidian-CodeMix shared task dataset [4]. This dataset contains YouTube comments collected from movie trailers using the YouTube comment scraper. Some comments in the dataset contain more than one sentence, but the average sentence length is one. Each comment is labeled as 'OFF' (offensive comment) or 'NOT' (non-offensive comment).

2.2. Handling Class Imbalance in Text Data

Imbalanced datasets are those in which the distribution of class labels is unequal. The number of samples from one class greatly outnumber those from the other. For example, in HASOC'21, the dataset used in this work, the number of non-offensive comments is four

times higher than offensive comments. Here, the class 'Not Offensive' is called the majority class, and the class 'Offensive' having fewer samples is called the minority class. In the presence of imbalanced data, it is crucial to identify minority classes accurately. Therefore, the model should not be biased to identify only the majority class but should also provide equal weight to the minority class.

There are three types of methodologies for addressing class imbalance in machine learning: data-level techniques, algorithm-level methods, and hybrid approaches [15]. Data-level strategies use various data sampling methods to try to reduce the degree of imbalance. The learning process is adjusted to increase the importance of minority class during training in algorithm-level methods. Typically, this is achieved by incorporating class weights into the loss function. In hybrid methods, both data-level and algorithmic-level methods are combined.

Data-level methods such as SMOTE and its variants [6,16,17] have been shown to work well in the past. Synthetic Minority Oversampling Technique or SMOTE is a technique to oversample the minority class. In SMOTE, new instances are created from the existing data. SMOTE examines minority class examples and uses K-NN to find a random nearest neighbor; then, a synthetic instance is generated at random in feature space. In Borderline-SMOTE [16], the samples near class borders are oversampled, while in Safe-Level-SMOTE [17] safe regions are defined to prevent oversampling in overlapping or noise regions.

In the undersampling data level method, the samples from the majority class are removed to balance the class distribution of underlying data. RUS (random undersampling) is the simplest undersampling method, where instances from the majority class are randomly chosen and removed from the training dataset. The disadvantage of random undersampling is that examples are eliminated without regard for their potential importance or value in determining the decision boundary between the classes [18]. Different Near-Miss algorithms have been proposed by Zhang and Mani [19], in which the majority samples are removed based on their distance from minority samples using the K-nearest neighbors (K-NN) classifier. Kubat and Matwin [20] proposed a one-sided selection method in which noisy and repeated majority class instances discovered through the 1-NN rule and Tomek Links are removed. Undersampling has the disadvantage of removing potentially important information, and oversampling may result in overfitting due to producing duplicate copies of existing samples.

Data augmentation refers to a set of algorithms that creates synthetic data from an existing dataset. In comparison to computer vision, NLP is still in its early stages of applying data augmentation. The most common application of data augmentation is to prevent overfitting. Data augmentation is accomplished through the use of symbolic or neural approaches. Neural approaches [21] include back-translation, style augmentation, and generative data augmentation. According to Shorten et al. [21], data augmentation may be effective for addressing the issue of class imbalance. Data augmentation tasks are classified as conditional or unconditional augmentation tasks. Unconditional data augmentation models, such as generative adversarial networks and variational autoencoders, generate random texts regardless of context. Unconditional data augmentation is used for this task because the context of the information as defined by the label must be preserved.

Another augmentation technique that is used as a pipeline for text generation is back translation (BT) or round-trip translation [22]. The BT method changes the text from language A to language B, then back to language A for the same text. By preserving the context of the original data, this back translation [5] aids in data diversification.

In this paper, we proposed a back-translation data augmentation method MTDOT using Malayalam and English as an intermediate language for handling class imbalance.

3. Dataset

The dataset is collected from FIRE 2021's Dravidian CodeMix under the sub-division of HASOC'21 [4] competition. The data set contains Tamil YouTube comments along with

the labels and text IDs. The labels indicate whether the comment is offensive (OFF), non-offensive (NOT), or not-Tamil. Since there were only three not-Tamil comment entries and the testing data did not include any other language comments, they were discarded. The examples of offensive comment and non-offensive comment are shown in Figure 1.

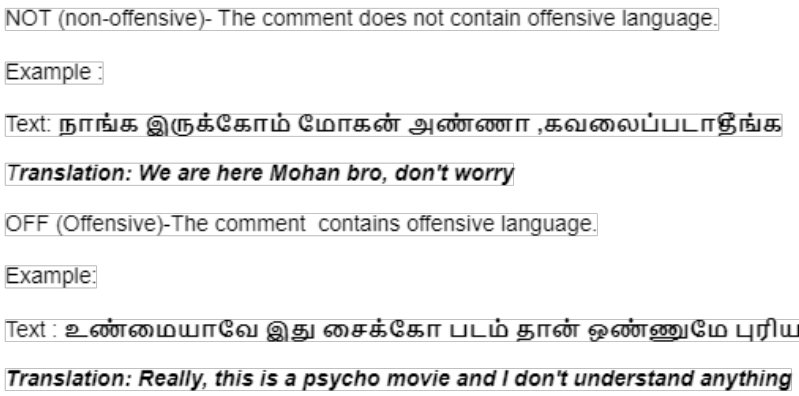

Figure 1. YouTube Comments in HASOC'21 dataset

There are 6531 YouTube comments without the not-Tamil comments. The class distribution in the dataset is shown in Table 1. The labels are thus: OFF indicating offensive and NOT indicating non-offensive. A total of 5877 comments were used as training data, where the data were further split into 80–20 percent, with 4701 serving as training data and 1176 serving as validation data. The remaining 654 comments from the overall data set are used for testing purposes. The testing set contains 536 non-offensive comments and 118 offensive entries.

Table 1. HASOC'21 Dataset class distribution.

Label	Category	Count
NOT	Non-offensive	4724
OFF	Offensive	1153
not-Tamil	Language other than Tamil/code-mixed Tamil and English	3

4. Methodology

In this work, a class balancing method is proposed for balancing the HASOC'21 [4] dataset in Tamil for offensive content identification. The architecture of proposed system is shown in Figure 2.

The proposed system (Figure 2) for the detection of offensive content from YouTube comments in Tamil contains the following steps:

- The proposed Multilingual Translation based Data augmentation technique (MTDOT) is applied on the imbalanced Tamil text data, and the balanced data are obtained in the first step.
- In the second step, MuRIL (Multilingual Representation for Indian Languages) is used as an embedding layer to obtain the representation of YouTube comments written in Tamil script.
- The embedding vectors generated from MuRIL layers are then provided to different classifiers.

The detailed description of each step in the proposed system is provided in the following subsections.

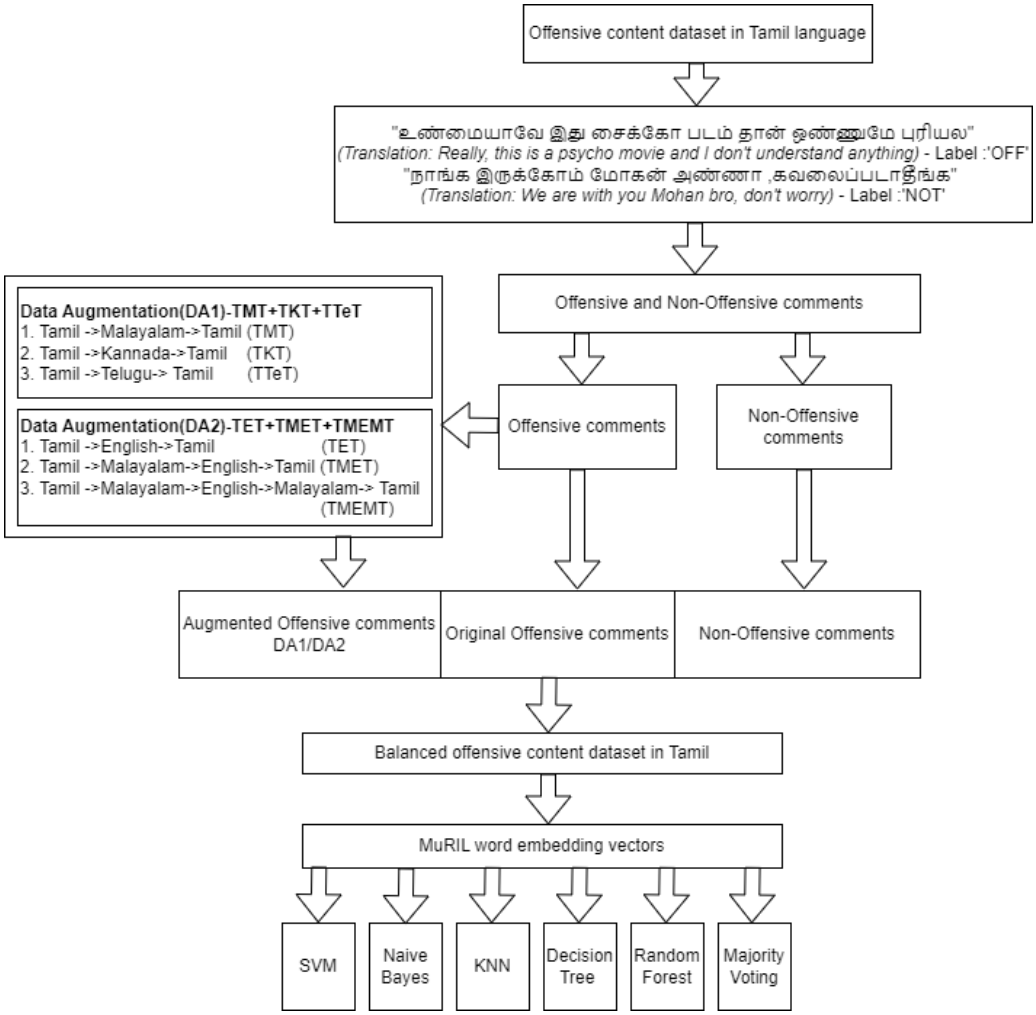

Figure 2. Architecture of proposed MTDOT method.

4.1. Balancing the Dataset

In Data Augmentation (DA) algorithms, synthetic data are constructed from the samples of the dataset. One application of data augmentation is fixing imbalance by augmenting minority class samples. In the back-translation method, text is translated from one language to another and then back to the original language. The process of data augmentation using back translation is shown in Figure 3.

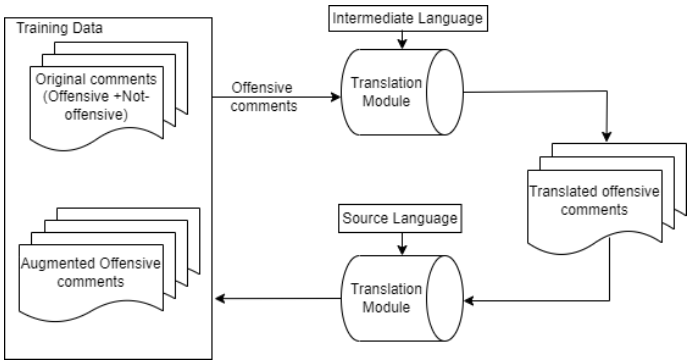

Figure 3. Balancing the dataset using the back-translation method for data augmentation.

In this method, three different ways of back translation have been used. The comments in the dataset were in Tamil language and written in Tamil script. Because the imbalance ratio is 4, the offensive comments must be augmented three times to balance the dataset

completely. For each offensive comment, we generated three augmented comments using the following methods:

- Tamil comment is translated to English and then back to Tamil (TET).
- Tamil comment is translated to Malayalam; then, the Malayalam comment is translated to English, and it is then back-translated to Tamil (TMET). Here, Malayalam and English are used as intermediate languages.
- Tamil comment is translated to Malayalam; then, the Malayalam comment is translated to English, and finally to Tamil (TMEMT). Here, Malayalam is used as an intermediate language twice.

The first balanced dataset (TET+TMET+TMEMT) contains all comments from the original dataset and augmented comments generated by the above methods.

The second balanced dataset (TMT+TKT+TTeT) contains the original dataset and the augmented offensive comments. Each offensive comment is augmented three times using the following methods:

- Tamil comment is translated to Malayalam and then back to Tamil (TMT).
- Tamil comment is translated to Kannada and then back to Tamil (TKT).
- Tamil comment is translated to Telugu and then back to Tamil (TTeT).

4.2. Word Embeddings

Word embedding is a term used for the representation of words for text analysis, which is typically in the form of a real-valued vector that encodes the meaning of the word such that the words that are closer in the vector space are expected to be similar in meaning.

In text embedding, tokens are represented as vectors which are a numerical representation of the word´s semantic meaning. These vectors can be utilised to train machine learning models for NLP-related tasks. TF-IDF is one of these text-embedding techniques. Classic embedding models have one disadvantage, though. They undergo polysemy disambiguation, in which a token or word with multiple meanings is represented by the same vector.

MuRIL [23], or Multilingual Representations for Indian Languages, is a BERT model that has been pre-trained by Google's Indian Research Unit. It is a multilingual language model trained exclusively on Indian text corpora. The corpora used by the authors is also subject to augmentation by translation and transliteration. The model has been trained on 17 languages, including English and 16 different Indian languages. It is trained in two phases: masked language modeling and translation language modeling. By evaluating the model on Indian language tasks and comparing it to the mBERT model, the authors conclude that MuRIL outperforms mBERT for all objectives. In this work, the MuRIL model is used as an embedding layer. After tokenization, the tokens are provided to the MuRIL pre-trained model to generate embedding vectors.

The first dimension of output represents the number of layers (12 layers + embedding layer), the second represents the number of tokens, and the third is the hidden size. The sentence embedding can be extracted by averaging over the layers and tokens (usually, the last four layers are considered, but one can take the average over all the layers as well). The contextual word embeddings can be extracted by the sum over the corresponding token outputs (as input, tokens here are subword units and not the words) with an average over the layers.

Although some comments in the offensive content dataset contain multiple sentences, the average sentence length is one. Each comment is labeled as 'OFF' (Offensive) or 'NOT' (Non-offensive). The text in the comment (single or multi-sentence) is tokenized and provided to the MuRIL embedding layer, which generates embedding vectors for the input tokens. These embedding vectors and their labels are fed into classifiers as training data.

4.3. Classifiers

Various classifiers that were trained on the dataset are used to study the effect of balancing the dataset. The embedding vectors generated from the pre-trained model are provided as input to these classifiers. Classifiers such as Support Vector Machine, Naive Bayes, K-NN, Decision Tree, Random Forest and Majority Voting are used for the experiments.

Support Vector Machine (SVM) is a supervised learning model used for classification, regression, and outlier detection. The objective of the SVM algorithm is to generate the optimal line or decision boundary that divides n-dimensional space into classes, so that subsequent data points can be classified easily. This optimal decision boundary is referred to as a hyperplane.

Naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' theorem. It is a highly scalable classifier used in various classification application domains. The fundamental idea behind Naive Bayes is that each feature makes an equal and independent contribution to the outcome.

K-Nearest Neighbor (K-NN) is a supervised machine learning algorithm. The K-NN algorithm assumes similarity between the new case/data and existing cases and places the new case in the category that is most similar to the existing categories.

A Decision Tree is a simple yet powerful and useful tool for data prediction and classification. The primary purpose of this model is to forecast an instance's values by learning decision rules based on data properties. An instance begins at the root node and descends to the next equivalent node based on the categorization produced by the test property. This instance advances down the tree branch and repeats the operation with the next sub-tree.

In ensemble learning, several models are combined to achieve better predictive performance compared to a single model. Ensemble learning is frequently used to enhance a model's performance (classification, prediction, function approximation, etc.) Bagging, stacking, and boosting are the three main categories of ensemble learning methods. In bagging, many Decision Trees are trained on different samples of the same dataset, and the predictions are averaged. In stacking, different models are trained on the same data, and another model is used to determine the best way to combine the predictions. In boosting, models are added sequentially to correct the prior model predictions, and the weighted average of the predictions is produced.

One such ensemble model is the Random Forest, which uses the bagging ensemble approach and decision trees as individual models. When working collectively, a large number of highly non-correlated models will outperform each of the component models separately. Here, the main element is the low correlation between models. Despite the fact that individual trees may make false predictions, the majority of them will be right; thus, the tree moves in the right direction as a group.

The Majority Voting Ensemble is yet another approach to ensemble learning in which the predictions from multiple other models are combined. When it comes to classification, the predictions for each label are added together, and the label with the most votes is predicted. It is a meta model i.e., a model of models with a collection of existing machine learning models. The models used are: Naive Bayes, K-NN, Decision Tree and Random Forest.

The above-mentioned models have been abbreviated for ease of representation, as shown in Table 2.

Table 2. Classifiers and their abbreviations.

Classifier	Abbreviation
Support Vector Machine	SVM
Naive Bayes	NB
K-Nearest Neighbor	KNN
Decision Tree	DT
Random Forest	RF
Majority Voting Ensemble	MV

4.4. Evaluation Metrics

To assess the effectiveness of the downstream model, performance metrics such as accuracy, precision, recall, and F_1-score were used. The ratio of the number of correct predictions over the total number of predictions is referred to as accuracy and is given in Equation (1). Precision (Equation (2)), recall (Equation (3)), and F_1-score were used as performance metrics because accuracy alone is insufficient to measure true performance. F_1-score is a harmonic mean of precision and recall. The formula for F_1-score is given in Equation (4).

$$Accuracy = \frac{True\ Positive + True\ Negative}{True\ Positive + False\ Positive + False\ Negative + True\ Positive} \quad (1)$$

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \quad (2)$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \quad (3)$$

$$F_1 = \frac{2 * Precision * Recall}{Precision + Recall} \quad (4)$$

Since the dataset is imbalanced with more non-offensive comments than offensive ones, accuracy is not a suitable evaluation metric. The offensive class, which has fewer samples in the dataset we used for this study, is more important, but its misclassification has little effect on accuracy. So, the weighted average for precision, recall and F_1-score has been used. Here, the precision, recall, and F_1-scores are calculated for each label, and then, the average is weighted by support.

In addition to the above listed metrics, the Bilingual Evaluation Understudy Score (BLEU score) proposed by Kishore Papineni [24] has also been used. It is a metric used to compare a generated sentence to a reference sentence. A score of 1.0 indicates a perfect match, whereas a score of 0.0 indicates a perfect mismatch. Although it was designed for translation, it can also be used to assess text output for a variety of natural language processing tasks. The BLEU score can also be used to various language generating issues using deep learning techniques, such as text summarization, language generation, image caption generation, and speech recognition. The BLEU score is used to evaluate how the generated augmented comments varies from the original one. An implementation of the BLEU score is provided by the Python Natural Language Toolkit library or NLTK.

5. Experiments

In this study, we have used the HASOC'21 dataset [4] for offensive content identification. Since the data are written in native/Tamil script, the task of identifying offensive content is challenging. As a result, techniques applicable to commonly used English-language NLP models will not be applicable to this type of data.

Various experiments are conducted with the following objectives:

1. To study the classifiers' performance for the original imbalanced Tamil text data.
2. To study the effect of balancing the data using the NearMiss undersampling technique.

3. To study the effect of balancing the data using the SMOTE oversampling technique.
4. To study the effect of balancing the data using the single-level back-translation augmentation method for balancing.
5. To study the impact of the proposed MTDOT class balancing method.

5.1. Experimental Setup

The dataset is preprocessed by removing special characters such as [, +, /, #, @, &, etc. After preprocessing, the dataset is split into a training and validation set with a validation set size of 20% and a random state of 42. The tokenizer provided for the MuRIL model is used to tokenize the dataset's texts. The model is imported using the Python package for the Hugging Face Transformer. The maximum token size is 512 characters. The pre-trained MuRIL model embedding layer receives the tokenized text as input and returns embedding vectors. After max pooling, the model generates a final vector with a length of 768. The different downstream classifiers, such as Support Vector Machine, Naive Bayes, K-Nearest Neighbor, Decision Tree, Random Forest, and Majority Voting, are then trained using the vectors. The Translators python library is used for translating the Tamil text into different languages.

5.2. Baseline Experiment with Original Dataset

In the first experiment, the classifiers were trained on the original offensive content identification dataset. The steps followed in training are described in the experimental setup section. The precision, recall and F_1-score for all the classifiers for the non-offensive class and offensive class are shown in Figures 4 and 5, respectively. For the non-offensive class, the highest precision of 0.93 (Naive Bayes), recall of 1.00 (Support Vector Machine) and F_1-score of 0.90 (Random Forest and Majority Voting) are achieved. Except for Naive Bayes (NB), all classifiers achieved almost the same precision for the non-offensive class.

For the offensive class, which is of more interest, the highest precision of 0.66 (Random Forest), recall of 0.75 (Naive Bayes) and F_1-score of 0.53 (Naive Bayes) is achieved. SVM does not perform well in the classification of offensive class comments. Due to the presence of imbalanced data, separating the hyperplane produced by an SVM is biased toward the minority class. The ratio of positive (offensive) to negative (non-offensive) support vectors becomes more imbalanced as the data imbalance increases; thus, samples at the boundary of the hyperplane are therefore more likely to be labeled as non-offensive. Naive Bayes performs comparatively well for both the offensive and non-offensive class as compared to other classifiers.

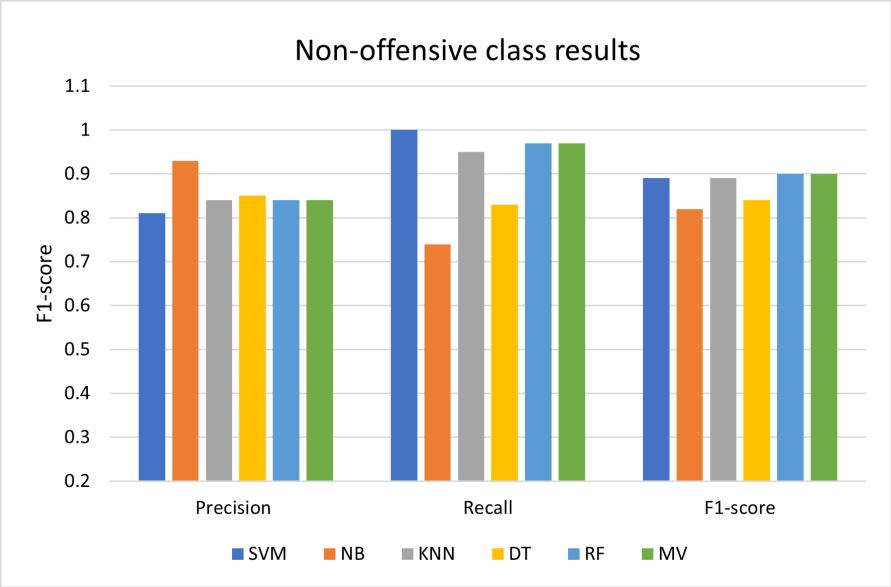

Figure 4. Non-offensive class results for original imbalanced dataset.

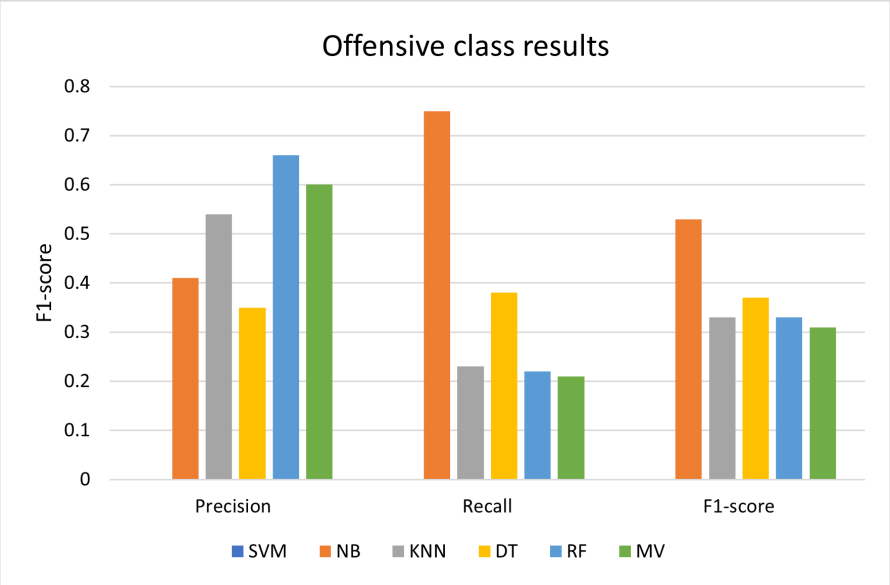

Figure 5. Offensive class results for original imbalanced dataset.

5.3. *Near Miss Undersampling Method for Addressing Class Imbalance*

The classifiers perform poorly for the offensive class in comparison with the non-offensive class, which is evident from Figures 4 and 5. This degradation in performance is due to the class imbalance in the dataset. In this experiment, the Near Miss undersampling method [19] is used to obtain a balanced dataset. Near Miss is a group of undersampling techniques that pick examples based on the distance between the majority and minority class examples. The technique comes in three versions: NearMiss-1, NearMiss-2, and NearMiss-3. In this experiment, NearMiss-3 is used, where the majority of the class examples that are closest to each minority class example is selected.

The classifier results for the offensive class are shown in Figure 6. Except for Naive Bayes, the offensive class F_1-score increased after applying the Near Miss undersampling algorithm. For Random Forest and Majority Voting, the highest F_1-score of 0.48 is achieved. For the balanced dataset using Near Miss, the classifiers' F_1-scores range from 0.36 to 0.48 points.

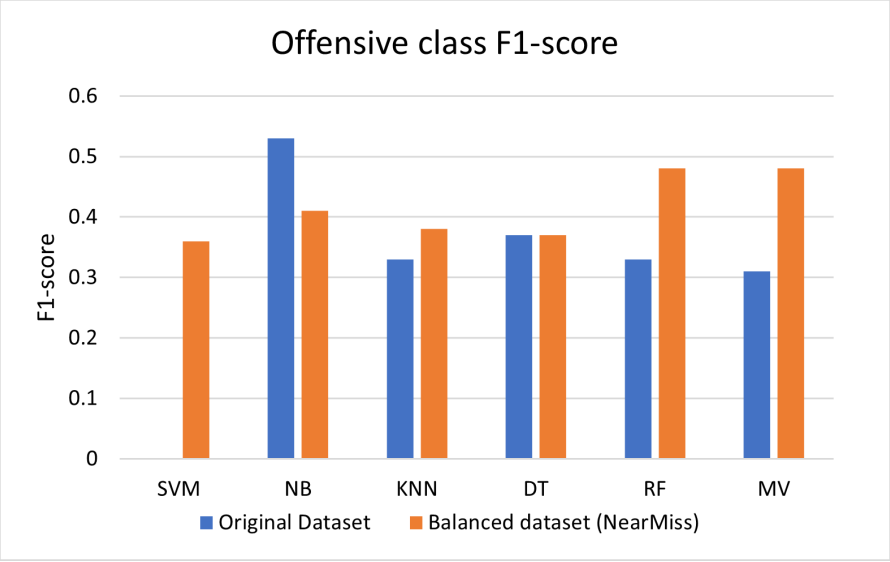

Figure 6. Offensive class F_1-scores for original and balanced dataset (Near Miss).

5.4. SMOTE Based Method for Addressing Class Imbalance

In this experiment, the dataset is balanced using the SMOTE [6] oversampling method. Synthetic Minority Oversampling TEchnique (SMOTE) [6] is a widely used oversampling approach in which new synthetic examples are generated for the minority class (offensive class). In this technique, first, a random minority class example is selected. Next, K-Nearest Neighbors for that example are located (k is normally equal to 5). A synthetic example is created at a randomly chosen point in feature space between two examples and its randomly chosen neighbor. As many synthetic examples of the minority class as needed can be created using this approach.

The classifier results for the offensive class are shown below in Figure 7. The F_1-score for the offensive class has increased after balancing, with the exception of Naive Bayes. Because some minority samples are repeated in SMOTE, so no new information is gained, Naive Bayes does not perform well in this situation. Support Vector Machines achieve the highest F_1-score of 0.53. For the Random Forest ensemble classifier, the maximum precision of 0.52 is achieved. For the balanced dataset using SMOTE, the classifiers' F_1-scores range from 0.39 to 0.53 points.

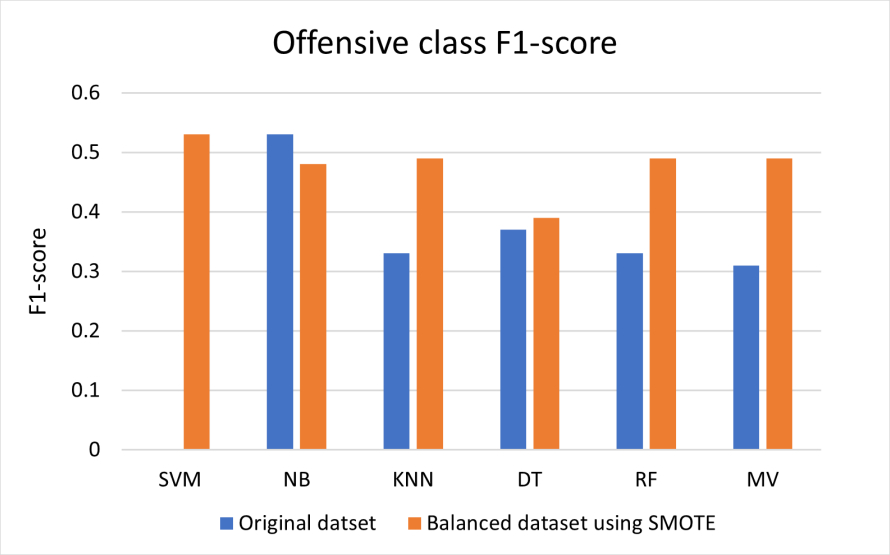

Figure 7. Offensive class F_1-scores for original and balanced dataset (SMOTE).

5.5. Single-Level Back-Translation Augmentation Method for Addressing Class Imbalance

The ratio of offensive comments to non-offensive comments in the offensive content identification dataset is 1:4. In this experiment, data augmentation is used to generate new offensive comments to obtain a balanced dataset. The offensive class samples are generated using the back-translation data augmentation method. Each offensive comment is augmented three times using TMT, TKT, and TTeT back-translation, with Malayalam, Kannada, and Telugu as an intermediate language, respectively. Figure 3 illustrates the back-translation process. The number of comments per class before and after balancing using data augmentation is shown in Figure 8.

The offensive class F_1-score achieved by all the classifiers is shown in Figure 9. There is a significant improvement in performance compared to SMOTE. The F_1-score improvement is highest for ensemble classifiers, Random Forest and Majority Voting, rising from 0.49 to 0.82.

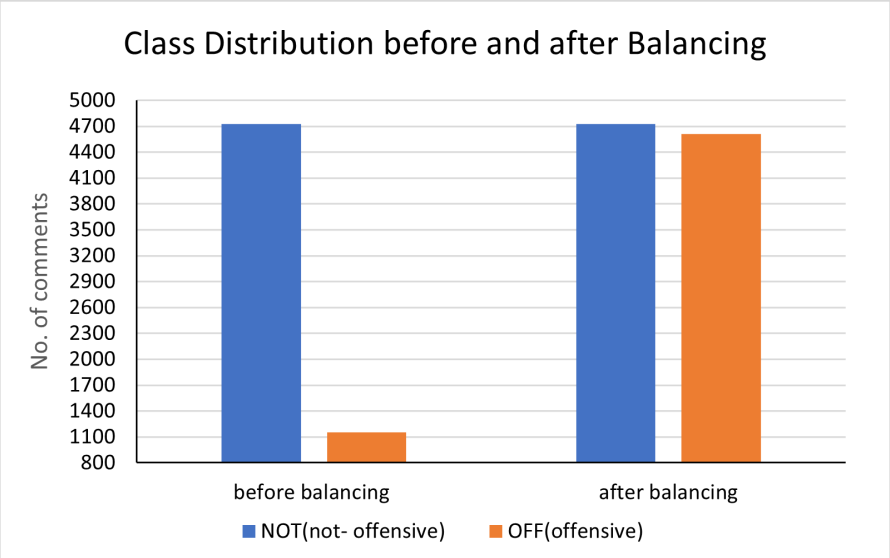

Figure 8. Class distribution before and after balancing the dataset.

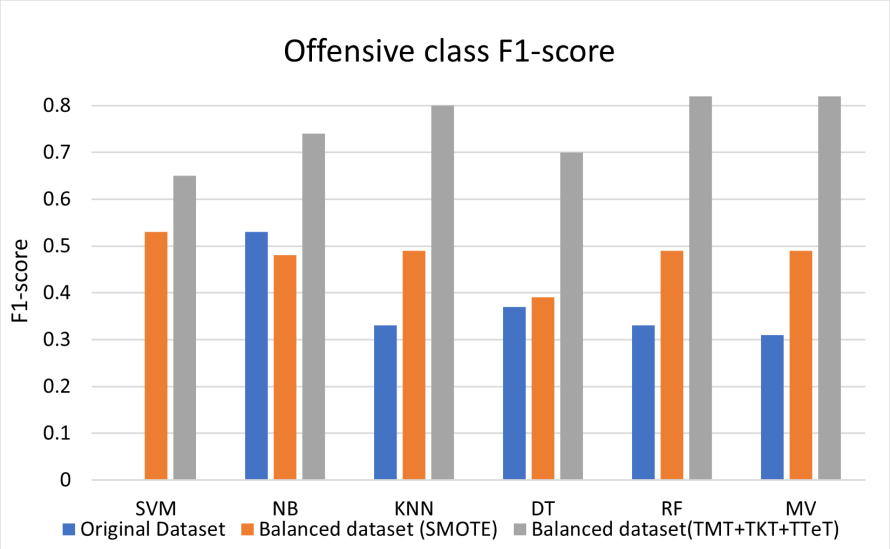

Figure 9. Offensive class F_1-scores for original and balanced dataset (TMT + TKT + TTeT).

5.6. Multi-Level Back-Translation Augmentation Method for Addressing Class Imbalance

In this experiment, we studied how the quality of generated samples and the performance of classifiers varied when multiple languages were used as intermediate languages instead of just one. Based on the back-translation augmentation method, we have proposed the class balancing method 'MTDOT', where more than one intermediate language is used. Malayalam and the English language are used for augmenting the offensive comments in three ways. The first method involves translating Tamil comments into English and then back into Tamil. The Tamil comment is translated to Malayalam, then to English, and then back to Tamil in the second method. Finally, the Tamil comment is translated to Malayalam, then to English, then back to Malayalam, and finally to Tamil. Figure 10 depicts the offensive class F_1-score obtained by all classifiers.

Compared to SMOTE, this method also achieved a significant improvement in offensive class F_1-score. The highest improvement in F_1-score is achieved for ensemble classifiers, Random Forest and Majority Voting, rising from 0.49 to 0.81.

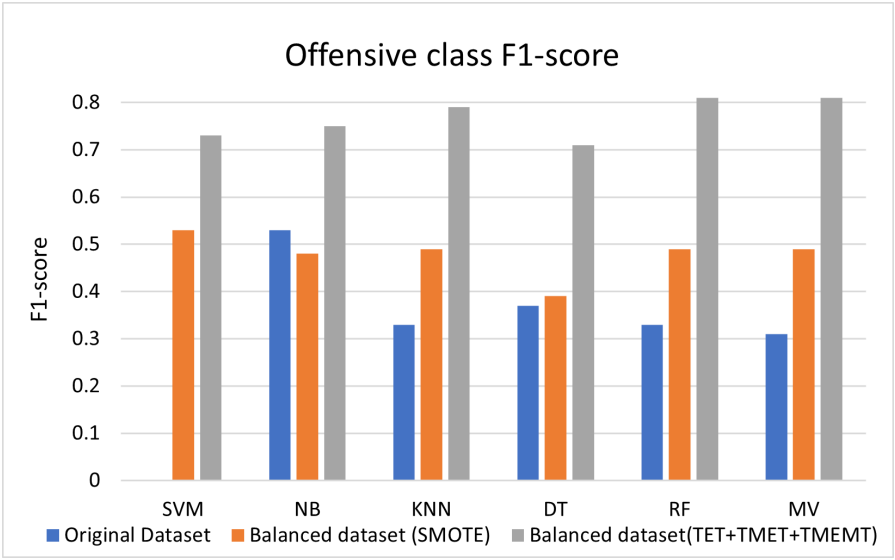

Figure 10. Offensive class F_1-scores for original and balanced dataset (TET + TMET + TMEMT).

6. Results and Discussion

To study the effect of balancing the offensive content data, experiments are conducted using four methods, namely: Near Miss, SMOTE, the single-level back-translation augmentation method and our proposed MTDOT class balancing method. Table 3 displays the precision, recall, and F_1-score for each class as well as the weighted average for all classifiers for the original dataset and the class-balanced dataset. The precision (P), recall (R), and F_1-score (F1) for the HASOC'21 dataset are listed in the column named Original Dataset.

Table 3. Performance analysis of existing undersampling, oversampling and proposed method.

Classifier	Class	Original Dataset			Balanced Dataset											
					Near-Miss			SMOTE			TMT + TKT + TTeT			TET + TMET + TMEMT		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
SVM	NOT	0.81	1.00	0.89	0.84	0.95	0.90	0.93	0.74	0.82	0.65	0.87	0.74	0.73	0.80	0.76
	OFF	0.00	0.00	**0.00**	0.57	0.26	**0.36**	0.41	0.75	**0.53**	0.81	0.54	**0.65**	0.77	0.69	**0.73**
	Wt.avg	0.65	0.81	0.72	0.79	0.82	0.79	0.83	0.74	0.77	0.73	0.70	0.69	0.75	0.75	0.75
NB	NOT	0.93	0.74	0.82	0.88	0.68	0.77	0.90	0.75	0.82	0.74	0.71	0.72	0.76	0.75	0.76
	OFF	0.41	0.75	**0.53**	0.31	0.61	**0.41**	0.38	0.66	**0.48**	0.72	0.75	**0.74**	0.74	0.76	**0.75**
	Wt.avg	0.83	0.74	0.77	0.77	0.67	0.77	0.80	0.73	0.75	0.73	0.73	0.73	0.75	0.75	0.75
KNN	NOT	0.84	0.95	0.89	0.85	0.90	0.87	0.97	0.56	0.71	0.78	0.85	0.81	0.78	0.85	0.81
	OFF	0.54	0.23	**0.33**	0.44	0.33	**0.38**	0.33	0.92	**0.49**	0.84	0.76	**0.80**	0.83	0.76	**0.79**
	Wt.avg	0.78	0.81	0.78	0.77	0.79	0.78	0.84	0.63	0.67	0.81	0.81	0.81	0.81	0.80	0.80
DT	NOT	0.85	0.83	0.84	0.87	0.63	0.73	0.86	0.80	0.83	0.70	0.72	0.71	0.72	0.70	0.71
	OFF	0.35	0.38	**0.37**	0.27	0.59	**0.37**	0.35	0.44	**0.39**	0.71	0.70	**0.70**	0.70	0.72	**0.71**
	Wt.avg	0.75	0.75	0.75	0.75	0.62	0.66	0.76	0.73	0.75	0.71	0.71	0.71	0.71	0.71	0.71
RF	NOT	0.84	0.97	0.90	0.34	0.77	0.48	0.88	0.90	0.89	0.82	0.82	0.82	0.81	0.82	0.82
	OFF	0.66	0.22	**0.33**	0.34	0.77	**0.48**	0.52	0.46	**0.49**	0.82	0.82	**0.82**	0.81	0.80	**0.81**
	Wt.avg	0.81	0.83	0.79	0.81	0.67	0.71	0.81	0.81	0.81	0.82	0.82	0.82	0.81	0.81	0.81
MV	NOT	0.84	0.97	0.90	0.90	0.77	0.83	0.89	0.80	0.85	0.81	0.84	0.83	0.81	0.83	0.82
	OFF	0.60	0.21	**0.31**	0.39	0.63	**0.48**	0.42	0.59	**0.49**	0.84	0.80	**0.82**	0.82	0.80	**0.81**
	Wt.avg	0.79	0.82	0.78	0.80	0.74	0.76	0.80	0.76	0.78	0.83	0.82	0.82	0.82	0.82	0.82

The results of the Near Miss undersampling algorithm are displayed in the Near Miss column in Table 3. The performance of the majority of classifiers is improved after balancing the dataset using the Near Miss algorithm. In the third column, named SMOTE, results from the widely used SMOTE oversampling method are presented to demonstrate the efficacy of our balancing technique. SMOTE performed better than Near Miss. The dataset

used in this work is of small size, containing 5880 samples. In undersampling, the majority of the class samples are removed, making the dataset size smaller. The less training data is the reason for Near Miss performing worse than SMOTE.

The results for the HASOC'21 dataset balanced using the back-translation data augmentation method with Malayalam (TMT), Kannada (TKT), and Telugu (TTeT) as intermediate languages are displayed in the column named TMT + TKT + TTeT. In the last column named TET + TMET + TMEMT, the results for the HASOC'21 dataset balanced using the multi-level back-translation data augmentation method (MTDOT) with Malayalam and English as intermediate languages are listed. Because accurately identifying offensive comments is of more importance, the F_1-scores for the offensive class are shown in bold. The highest F_1-score for the offensive class is displayed in blue. The graph of the comparative performance of all classifiers for the offensive class is shown in Figure 11.

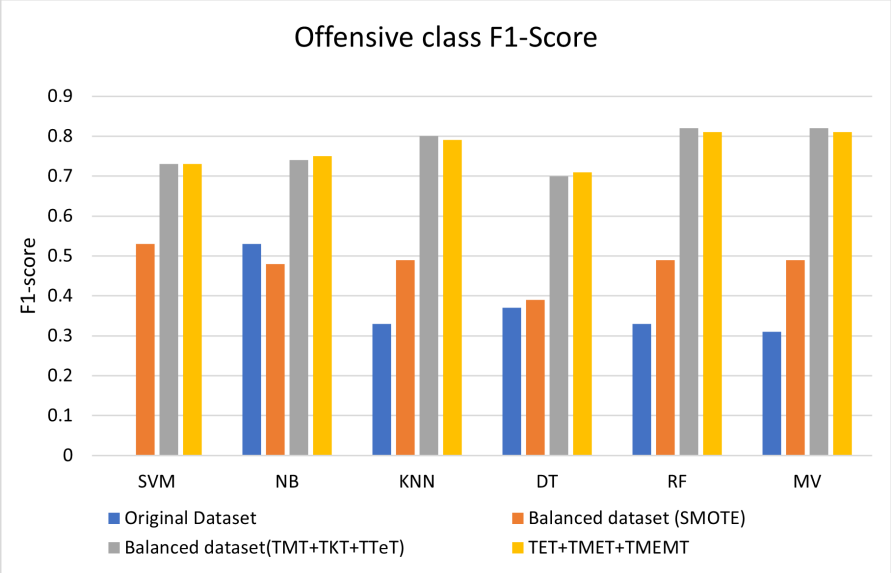

Figure 11. Offensive class F_1-score for original dataset and balanced dataset.

The F_1-score for the 'OFF' (offensive) class in the original dataset is extremely low, ranging from 0 to 0.31, which is a serious issue. The F_1-score of the offensive class for the balanced dataset has improved significantly for all classifiers, as seen in Table 3. The highest F_1-score of 0.81 is achieved for the 'OFF'(offensive) class for Random Forest and majority voting ensemble models. The proposed data-augmentation-based class balancing technique (MTDOT) outperforms the SMOTE oversampling method. For Support Vector Machines, the performance improvement is greatest when compared to SMOTE.

Data augmentation is implemented in two ways. For the first, three intermediate languages are used for back-translation: Tamil–Malayalam–Tamil (TMT), Tamil–Kannada–Tamil (TKT), and Tamil–Telugu–Tamil (TTeT). In multi-level back translation, English and Malayalam are used as intermediate languages at three levels: Tamil–English–Tamil (TET), Tamil–Malayalam–English–Tamil (TMET), and Tamil–Malayalam–English–Malayalam–Tamil (TMEMT). The performance improvement over the original unbalanced dataset is nearly identical for both methods. As a result, any method can be adopted for data augmentation. The BLEU score is computed to assess the quality of the generated statements as discussed in Section 4.4. The BLEU scores for the balanced dataset are shown in Table 4.

Table 4. BLEU Score.

Dataset	BLEU Score
TMT + TKT + TTeT	0.258
TET + TMET + TMEMT	0.205

For the TMT + TKT + TTeT dataset, the BLEU score is 0.258, and for TET + TMET + TMEMT, it is 0.205. This signifies that multi-layer back translation augmentation generates more diversified data than single-level back translation augmentation.

The F_1-score for the offensive class, as well as the weighted average of the F_1-scores for the offensive class and the non-offensive class, increased significantly for the balanced dataset, as shown in Table 3. Support Vector Machine shows the highest improvement in F_1-score of offensive class from 0.0 to 0.73, Naive Bayes from 0.53 to 0.75, K-NN from 0.33 to 0.80, Decision Tree from 0.37 to 0.71, Random Forest from 0.33 to 0.81, and Majority Voting from 0.31 to 0.81.

7. Conclusions

In this research work, we proposed a Multilingual Translation-based Data augmentation technique for Offensive content identification in Tamil text data (MTDOT) that can boost the performance of low-resource language and imbalanced data classification tasks. The back-translation augmentation method is used in three levels for generating synthetic offensive comments. The efficacy of the proposed method on the Tamil offensive content identification dataset (HASOC'21) is demonstrated by comparing it with the popular SMOTE class balancing method. The experimental results showed the significant improvement in the performance of various classifiers, after applying the class balancing MTDOT method. The advantage of the proposed method is that diversified data samples can be obtained without changing the contextual information. By using single-level back translation with Malayalam, Kannada, and Telugu as intermediate languages, we constructed another balanced dataset. Despite the fact that both methods are equally effective, our proposed MTDOT method yields more diverse data, which is evident with the improved BLEU score. The proposed method can be applied to any language, and it can be used for balancing extremely imbalanced datasets without the need for a large number of intermediate languages. In the future, we aim to apply this method to other similar tasks.

Author Contributions: Conceptualization, V.G. and R.R.; methodology, V.G. and R.R.; software, V.G.; validation, R.R.; formal analysis, V.G.; investigation, V.G.; resources, V.G.; data curation, V.G.; writing—original draft preparation, V.G. and R.R.; writing—review and editing, V.G. and R.R.; visualization, V.G.; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The dataset can be obtained from the organizers on request. https://dravidian-codemix.github.io/HASOC-2021/datasets.html.

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

References

1. Rajalakshmi, R.; Reddy, B.Y. DLRG@HASOC 2019: An Enhanced Ensemble Classifier for Hate and Offensive Content Identification. In Proceedings of the Working Notes of FIRE 2019—Forum for Information Retrieval Evaluation, Kolkata, India, 12–15 December 2019; Volume 2517, pp. 370–379.
2. B, Y.R.; Rajalakshmi, R. DLRG@HASOC 2020: A Hybrid Approach for Hate and Offensive Content Identification in Multilingual Tweets. In Proceedings of the Working Notes of FIRE 2020—Forum for Information Retrieval Evaluation, CEUR Workshop Proceedings, Hyderabad, India, 16–20 December 2020; Volume 2826, pp. 304–310.
3. Rajalakshmi, R.; Reddy, P.; Khare, S.; Ganganwar, V. Sentimental Analysis of Code-Mixed Hindi Language. In *Proceedings of the Congress on Intelligent Systems*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 739–751.
4. Chakravarthi, B.R.; Kumaresan, P.K.; Sakuntharaj, R.; Madasamy, A.K.; Thavareesan, S.; B, P.; Chinnaudayar Navaneethakrishnan, S.; McCrae, J.P.; Mandl, T. Overview of the HASOC-DravidianCodeMix Shared Task on Offensive Language Detection in Tamil and Malayalam. In Proceedings of the Working Notes of FIRE 2021—Forum for Information Retrieval Evaluation, CEUR, Gandhinagar, India, 13–17 December 2021.
5. Corbeil, J.P.; Ghadivel, H.A. Bet: A backtranslation approach for easy data augmentation in transformer-based paraphrase identification context. *arXiv* **2020**, arXiv:2009.12452.
6. 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]

7. Garain, A.; Mandal, A.; Naskar, S.K. JUNLP@DravidianLangTech-EACL2021: Offensive Language Identification in Dravidian Langauges. In *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*; Association for Computational Linguistics: Kyiv, Ukraine, 2021; pp. 319–322.
8. Kedia, K.; Nandy, A. indicnlp@kgp at DravidianLangTech-EACL2021: Offensive Language Identification in Dravidian Languages. *CoRR* **2021**, *abs/2102.07150*. Available online: http://xxx.lanl.gov/abs/2102.07150 (accessed on 12 September 2022).
9. Jayanthi, S.M.; Gupta, A. SJ_AJ@DravidianLangTech-EACL2021: Task-Adaptive Pre-Training of Multilingual BERT models for Offensive Language Identification. *CoRR* **2021**, *abs/2102.01051*. Available online: http://xxx.lanl.gov/abs/2102.01051 (accessed on 12 September 2022)).
10. Ghanghor, N.; Krishnamurthy, P.; Thavareesan, S.; Priyadharshini, R.; Chakravarthi, B.R. IIITK@DravidianLangTech-EACL2021: Offensive Language Identification and Meme Classification in Tamil, Malayalam and Kannada. In Proceedings of the DRAVIDIANLANGTECH, Kyiv, Ukraine, 20 April 2021.
11. Dave, B.; Bhat, S.; Majumder, P. IRNLP_DAIICT@ LT-EDI-EACL2021: Hope Speech detection in Code Mixed text using TF-IDF Char N-grams and MuRIL. In Proceedings of the First Workshop on Language Technology for Equality, Diversity and Inclusion, Kyiv, Ukraine, 19 April 2021; pp. 114–117.
12. Chinnappa, D. dhivya-hope-detection@ LT-EDI-EACL2021: Multilingual hope speech detection for code-mixed and transliterated texts. In Proceedings of the First Workshop on Language Technology for Equality, Diversity and Inclusion, Kyiv, Ukraine, 19 April 2021; pp. 73–78.
13. Rajalakshmi, R.; Reddy, Y.; Kumar, L. DLRG@DravidianLangTech-EACL2021: Transformer based approachfor Offensive Language Identification on Code-Mixed Tamil. In *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*; Association for Computational Linguistics: Kyiv, Ukraine, 2021; pp. 357–362.
14. Subramanian, M.; Ponnusamy, R.; Benhur, S.; Shanmugavadivel, K.; Ganesan, A.; Ravi, D.; Shanmugasundaram, G.K.; Priyadharshini, R.; Chakravarthi, B.R. Offensive language detection in Tamil YouTube comments by adapters and cross-domain knowledge transfer. *Comput. Speech Lang.* **2022**, *76*, 101404. [CrossRef]
15. Krawczyk, B. Learning from imbalanced data: Open challenges and future directions. *Prog. Artif. Intell.* **2016**, *5*, 221–232. [CrossRef]
16. Han, H.; Wang, W.Y.; Mao, B.H. Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. In *Proceedings of the International Conference on Intelligent Computing*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 878–887.
17. Bunkhumpornpat, C.; Sinapiromsaran, K.; Lursinsap, C. Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 475–482.
18. Sun, Z.; Song, Q.; Zhu, X.; Sun, H.; Xu, B.; Zhou, Y. A novel ensemble method for classifying imbalanced data. *Pattern Recognit.* **2015**, *48*, 1623–1637. [CrossRef]
19. Mani, I.; Zhang, I. kNN approach to unbalanced data distributions: A case study involving information extraction. In Proceedings of the Workshop on Learning from Imbalanced Datasets, Washington DC, USA, 21 August 2003; Volume 126, pp. 1–7.
20. Kubat, M.; Matwin, S. Addressing the curse of imbalanced training sets: One-sided selection. In Proceedings of the ICML, Citeseer, Nashville, TN, USA, 8–12 July 1997; Volume 97, p. 179.
21. Shorten, C.; Khoshgoftaar, T.M.; Furht, B. Text data augmentation for deep learning. *J. Big Data* **2021**, *8*, 1–34. [CrossRef] [PubMed]
22. Sennrich, R.; Haddow, B.; Birch, A. Improving neural machine translation models with monolingual data. *arXiv* **2015**, arXiv:1511.06709.
23. Khanuja, S.; Bansal, D.; Mehtani, S.; Khosla, S.; Dey, A.; Gopalan, B.; Margam, D.K.; Aggarwal, P.; Nagipogu, R.T.; Dave, S.; et al. Muril: Multilingual representations for indian languages. *arXiv* **2021**, arXiv:2103.10730.
24. Papineni, K.; Roukos, S.; Ward, T.; Zhu, W.J. Bleu: A method for automatic evaluation of machine translation. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, PA, USA, 7–12 July 2002; pp. 311–318.