ScalarLab@TRAC2024: Exploring Machine Learning Techniques for Identifying Potential Offline Harm in Multilingual Commentaries

Anagha H C, Saatvik Krishna M, Soumya Sangam Jha, Vartika T Rao, Anand Kumar M
Department of Information Technology
National Institute of Technology, Karnataka, Surathkal, India
{hcanagha.211it008, skm.211it056, soumyajha.211it068, vartikatrao.211it077,m_anandkumar}@nitk.edu.in

Abstract
The objective of the shared task, Offline Harm Potential Identification (HarmPot-ID), is to build models to predict the offline harm potential of social media texts. "Harm potential" is defined as the ability of an online post or comment to incite offline physical harm such as murder, arson, riot, rape, etc. The first subtask was to predict the level of harm potential, and the second was to identify the group to which this harm was directed towards. This paper details our submissions for the shared task that includes a cascaded SVM model, an XGBoost model, and a TF-IDF weighted Word2Vec embedding-supported SVM model. Our system ranked 4th in the first subtask and 3rd in the second. Several other models that were explored have also been detailed.

Keywords: Offline Harm, Harm Potential, HarmPot, Text classification, Offline harm, TF-IDF, weighted word embeddings

1. Introduction

There has been an increased use of social media in the current society. It is estimated that approximately 62.3 % of the population uses social media. This has led to a large section of society gaining access to airing their opinions on social media. While it might seem that this may give more people accountability, on the contrary, it has led to factions of people openly expressing their harmful discriminatory opinions online, by making use of pseudo-anonymity that many social media platforms allow, like Twitter and Reddit. While this is creating a harmful space for users online, it also exposes the mindset of people who have potentially dangerous views.

The shared task, Offline Harm Potential Identification (HarmPot-ID), aims to exploit the data online to predict the probability of a person committing a crime offline through their comments made on social media. Using the data, we were tasked to predict whether a specific social media post is likely to cause offline harm events like riots, arson, murder, rape, etc. With an increased rate of violent crimes across the world, early detection could potentially save many lives.

The shared task consisted of two subtasks. The first sub-task was a 4-class classification task to predict the level of harm potential. Class '0' refers to completely harmless content that poses no threat of causing any offline harm. Class '1' refers to the comment that could incite an offline harm event given specific conditions or context. Class '2' refers to the comments most likely to incite an offline harm event in most contexts. Class '3' refers to the comments that will incite or initiate an offline harm event in any context. The second sub-task required predicting five labels: Gender, Religion, Descent, Caste, and Political Ideology, each a binary classification task. This subtask could also be looked at as a multi-label classification task.

The dataset (Kumar et al., 2024b) provided consisted of multilingual, code-mixed (Hindi, English, and Meitei) comments collected from various social media platforms like YouTube, Twitter, and Telegram. A few records include text consisting of only emojis, numbers, or texts from other scripts. Details about the number of samples in the train, dev, and test sets are given in Table1, script distributions in Table 2 and class distributions in Table 3 and 4.

Firstly, the multi-lingual code-mixed data renders general pre-trained models ineffective. Moreover, the unbalanced nature of the dataset makes it hard for the models to accurately predict the categories of harm.

In this paper, we propose systems to overcome these challenges using methods such as balanced class weights, oversampling and even training word embedding models on our dataset.

The rest of this paper is organized as follows. Section II discusses the background and related works. Section III describes the methodology. Section IV contains the experimentation. Section V discusses the results, Section VI contains the conclusion, and Section VII concludes the paper with future directions.

File	Labelling	Number of Records
Train	Labelled	50,788
Dev	Labelled	6,349
Test	Unlabelled	6,349

Table 1: Number of Records

File	Hindi	Bengali	English	Others
Train	4,956	3,862	40,690	1,280
Dev	646	468	5,086	149
Test	644	449	5,093	163

Table 2: Script Distribution

Label	Class	Training	Validation
Gender	0	46,358	5,169
	1	10,779	1,180
Religion	0	51,616	5,704
	1	5,521	645
Descent	0	55,501	6,169
	1	1,636	180
Caste	0	56,518	6,291
	1	619	58
Political Ideology	0	56,682	6,301
	1	455	48

Table 4: Sub-Task 1b Class Distribution

2. Methodology

2.1. Data Preprocessing

Data preprocessing techniques are incorporated to make the data usable for training the models.

2.1.1. Lowering of Text and Removal of Punctuation

The initial preprocessing step involved the conversion of all Roman script text to lowercase and the subsequent removal of all punctuation marks.

2.1.2. Mapping Emojis

Emojis were systematically correlated with words by utilizing the Python library 'emoji.' This process entailed the conversion of emojis into their corresponding textual descriptions. For instance, the thumbs-up emoji was algorithmically assigned to the word 'thumbs_up.

2.2. Models Used

The textual data underwent vectorization utilizing the TF-IDF vectorizer. The resulting vector size was (50,788, 1,06,486). Subsequently, various models were implemented, each employing specific techniques as delineated below. Parameters other than the ones explicitly mentioned were set to default values.

2.2.1. Logistic Regression and XGBoost

The logistic regression (LR) model was trained using L2 regularization, and Stochastic Average Gradient descent was used as the optimization algorithm to solve the convex optimization problem during training. An LR model with the balanced class weights parameter was trained to assign higher significance to minority classes. To address the class imbalance, the data underwent oversampling via the Adaptive Synthetic Sampling (ADASYN) technique (He et al., 2008), and an LR model was trained on the augmented dataset. Furthermore, in sub-task 1a, the labels were subjected to one-hot encoding before model training. This encoding method was also applied to the oversampled data. The oversampled data was also trained on a model with the balanced class weights parameter. Sub-task 1b was treated as a separate multi-label binary classification task and an LR classifier was trained for each label. A similar training mechanism was used for XGBoost while giving equal importance to both positive and negative classes by adjusting the scale_pos_weight parameter.

Class	Training	Validation
0	16,135	2,017
1	21,554	2,695
2	12,211	1,526
3	888	111

Table 3: Sub-Task 1a Class Distribution

2.2.2. SVM

The SVM model was trained on the training dataset for sub-task 1a. Initially, a linear kernel was employed, with a regularization parameter (C) set to 1. Given the multi-label classification nature of the sub-task, distinct SVMs were trained for each label, maintaining the same parameter settings. Subsequently, an evaluation of model performance led to the adoption of the radial basis function (RBF) kernel for all SVM models, as it demonstrated superior performance compared to the linear kernel.

2.2.3. Cascaded SVM

A cascaded SVM was trained for sub-task 1b. A SVM was trained on the entire training data for sub-task 1a. The instances classified as 0 for sub-task 1a were directly classified as 0 for all the labels of sub-task 1b. This is inferred from the fact that if a comment does not pose any harm, it will not harm any of the sections mentioned as labels in sub-task 1b. Separate SVMs were then trained to classify the instances which were classified as 1,2, and 3 in sub-task 1a.

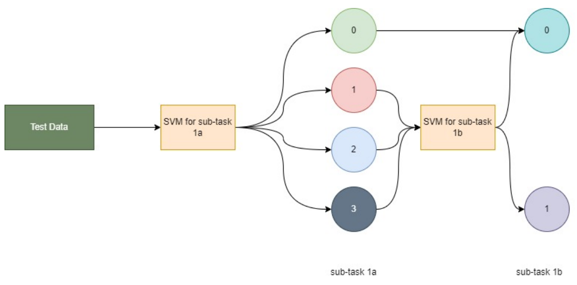

Figure 1: Cascaded SVM

2.2.4. Hierarchical SVM

From the above methods, it was noticed that the models were misclassifying classes 1, 2, and 3 in sub-task 1a. Hence, all the instances of classes 1, 2, and 3 were grouped together. A binary SVM classifier was trained to detect if there is no harm (class 0) or some form of harm (class 1, 2, and 3). Subsequently, another multi-class SVM classifier was trained to classify the level of harm to classes 1, 2, and 3.

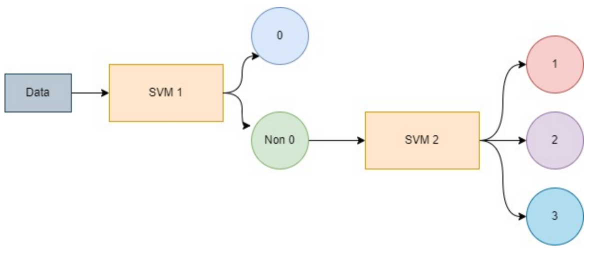

Figure 2: Hierarchical SVM

2.2.5. Using Word2Vec embeddings

To train the word2vec (Mikolov et al., 2013) model, we used the previous year's TRAC conference data (TRAC 2018, TRAC2020, TRAC 2022) along with this year's. We ensured that the distribution in scripts and languages was identical to that of the original train, dev, and test set and that none of the instances were repeated. We had 97,217 instances, of which 77,055, 12,257, 6,131, and 1,774 were in English, Hindi, Bengali, and undefined scripts, respectively.

We trained both a CBOW (Continuous Bag of Words) model and a skip-gram model. A simple DNN and an attention-based LSTM model were trained using the embeddings obtained. Both an embedding size of 100 and 300 were tried. Additionally, due to the code-mixed nature of the data, a tri-gram training method was used to accurately capture the language patterns, as Hindi and Bengali have some similarities in their word structure and improve the language model's overall performance.

Word Embeddings	Micro F1 Score
skipgram	0.5948
skipgram-tri	0.5248
cbow	0.6087
cbow-tri	0.5248
GloVe	0.42

Table 5: Word Embeddings and Micro F1 Score

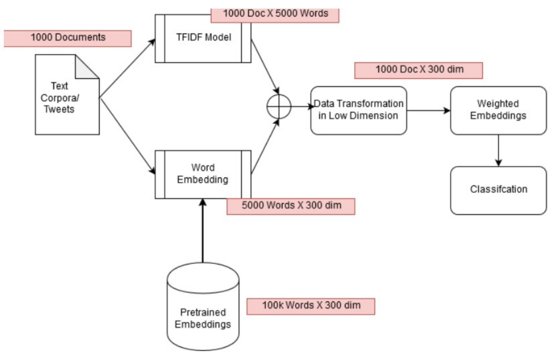

Figure 3: Weighted Document Embedding Framework adapted from (Sharmila et al., 2019)

2.2.6. Using TF-IDF weighted Word2Vec embeddings with SVM

Due to good results being shown by SVM, we used TF-IDF weighed Word2Vec embeddings (Sharmila et al., 2019) to obtain document embeddings and trained the SVM on that. The word embeddings were obtained as described previously. To obtain the TF-IDF weighted Word2Vec, the vocabulary of the TF-IDF and Word2Vec were matched, and the resultant embeddings were obtained by multiplying the TF-IDF embedding matrix and the word2vec embedding matrix.

2.2.7. Using GloVe embedings

GloVe (Global Vectors for Word Representation) (Pennington et al., 2014) is a technique to obtain word embeddings. The model was retrained on the same data used to train the Word2Vec (Mikolov et al., 2013) model. A simple DNN and an attention based model were trained on the word embeddings obtained. Additionally a TF-IDF weighed word embedding method was also used to train a SVM.

Method	Sub-task 1	Gender	Religion	Caste	Descent	Political Ideology
LR	0.632	0.851	0.929	0.990	0.975	0.992
LR with Balanced Class Weights	0.61	0.79	0.91	0.94	0.96	0.99
LR with Oversampling using AdaSYN	0.57	0.78	0.82	0.9	0.97	0.99
LR with One-Hot Encoded Data	0.62	-	-	-	-	-
LR with Oversampled One-Hot Encoded Data	0.56	-	-	-	-	-
LR with Oversampled Data and Balanced Class Weights	0.55	0.78	0.82	0.9	0.97	0.99
LR - Multi-Label Classifer	-	0.56	0.56	0.56	0.56	0.56
XGB	0.596	0.857	0.932	0.991	0.976	**0.995**
XGB with Balanced Class Weights	-	0.636	0.931	0.99	0.971	0.994
XGB with Oversampling using AdaSYN	0.548	0.699	0.922	0.972	0.964	0.994
XGB with One-Hot Encoded Data	0.492	-	-	-	-	-
XGB with Oversampled One-Hot Encoded Data	0.445	-	-	-	-	-
XGB with Oversampled Data and Balanced Class Weights	-	0.52	0.524	0.644	0.938	0.993
XGB - Multi-Label Classifer	-	0.495	0.495	0.495	0.495	0.495
SVM	**0.673**	0.869	0.932	**0.991**	0.975	0.992
SVM Cascade	0.673	**0.87**	**0.935**	0.98	**0.99**	0.994
word2vec dnn skipgram	0.594	-	-	-	-	-
word2vec dnn skipgram-tri	0.524	-	-	-	-	-
word2vec dnn cbow	0.608	-	-	-	-	-
word2vec dnn cbow-tri	0.524	-	-	-	-	-
Cascading SVM TF-IDF weighted Word2Vec	0.626	0.848	0.923	0.974	0.987	0.993
Hierarchical SVM	0.66	-	-	-	-	-

Table 6: Results (micro-F1 scores of each task)

3. Results

All the results (Micro-F1 scores) shown in table 6 are tested on the Dev set, whereas the final shared task results are evaluated on the test set.

3.1. Model Performance

Cascaded SVM gives the best results (Micro-F1 scores) on average for all tasks. The results are detailed in table 6.

3.2. Word Embeddings

The CBOW model worked the best among all the word embedding techniques. The results of a simple DNN trained on different word embeddings are detailed in the table 5. Due to this, for further analysis of combined models, we stick to CBOW Word2Vec.

3.3. Submission Details

Our team, ScalarLab, made 3 submissions - Cascading SVM, Cascading SVM with TF-IDF weighted word embeddings, and an XGBoost Model. Our standings at the end of the evaluation phase are shown in tables 7 and 8.

User	Team	Rank	Micro-F1
Yestin	CLTL	1.00	0.74
xsd		2.00	0.73
lazyboy.blk	1024m	3.00	0.71
ScalarLab		4.00	0.67

Table 7: Results of Sub-Task 1a

User	Team	Rank	Micro-F1
Yestin	CLTL	1.00	0.96
xsd		2.00	0.96
ScalarLab		3.00	0.95

Table 8: Results of Sub-Task 1b

4. Conclusion

From our extensive work with various models, we have concluded that the SVM model with cascading has performed the best with a 0.673 Micro F1 score on the first subtask and an average of 0.9455 micro F1 on the second subtask. The weighted document vectors attained less accuracy than the traditional TF-IDF-based SVM. For future work, BERT embeddings can be implemented. It would also be ideal to investigate the performance of this model on other code-mixed datasets. We believe this work can help further the understanding of code-mixed text classification and offline potential harm detection.

5. Bibliographical References

Haibo He, Yang Bai, Edwardo Garcia, and Shutao Li. 2008. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. pages 1322 – 1328.

Raj Kumar, Om Bhalla, Manohar Vanthi, S. M. Wani, and Shivam Singh. 2024a. Harmpot: An annotation framework for evaluating offline harm potential of social media text. *ArXiv*.

Ritesh Kumar, Ojaswee Bhalla, Shehlat Maknoon Vanthi, Madhu Wani, and Siddharth Singh. 2024b. Harmpot: An annotation framework for evaluating offline harm potential of social media text. In *Proceedings of the the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation*, Torino, Italy.

Kirti Kumari, Shaury Srivastav, and Rajiv Ranjan Suman. 2022. Bias, threat and aggression identification using machine learning techniques on multilingual comments. In *Proceedings of the Third Workshop on Threat, Aggression and Cyberbullying (TRAC 2022)*, pages 30–36. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space.

Jessica Pater and Elizabeth Mynatt. 2017. Defining digital self-harm. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing (CSCW '17)*, pages 1501–1513. Association for Computing Machinery.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

D. Sharmila, S. Kannimuthu, G. Ravikumar, and K. Anand. 2019. Kce dalab-apda@ fire2019: Author profiling and deception detection in arabic using weighted embedding. `https://ceur-ws.org/Vol-2517/T2-10.pdf`.