41043 42850
Natural Language Processing

Assignment 3 - Group 25

Multilingual Content Moderation

Megan Hastie	|	25775584
Dean Stokeld	|	24923029
Sophie Gaou	|	13914407
Alexander Easy	|	24506305
Tarlan Kameli	|	25092115

Table of Contents

Abstract

The rapid growth of social media content, coupled with its increasing linguistic diversity, has made the detection and management of harmful speech more complex than ever. In response to this challenge, this project examines the shortcomings of traditional moderation approaches in multilingual settings, especially in cases where code-mixing and transliteration obscure offensive language, limiting the effectiveness of conventional models. To address these issues, three transformer-based models were implemented and evaluated: XLM-RoBERTa, DistilBERT, and IndicBERT, which were trained on a cleaned and augmented version of the HASOC 2021 dataset, containing only English and Hindi tweets. The methodology included thorough data preprocessing, back-translation for data augmentation, and hyperparameter tuning using Optuna to improve model generalisation. Of the three models, XLM-RoBERTa consistently delivered the strongest results, achieving marginally higher F1-score and overall generalisation over the other models. However, real-world environments often include computational constraints, and although the XLM-RoBERTa model performed the best, its 1-2% increase in performance may not be the optimal choice for deployment as it's a much heavier model. The findings highlight the importance of multilingual pre-training and utilising context-aware models in achieving effective content moderation. Although the project primarily focused on technical performance, it also brought attention to the broader responsibility of designing moderation tools that not only detect harmful content but also safeguard freedom of expression. By addressing both linguistic complexity and ethical considerations, this project contributes to the advancement of AI moderation systems that are more inclusive and better suited to the needs of diverse online communities.

1. Introduction

In the age of social media, freedom of speech is now more accessible than ever. X (formally known as twitter) sees on average 500 million tweets per day (Twitter Usage Statistics - Internet Live Stats, 2009), and over 1 trillion tweets since its launch (Kalev Leetaru, 2019) – now anyone can talk to who they want, about what they want, when they want. Whilst this offers a plethora of benefits, it also poses a significant challenge: when does free speech go too far? This introduces the need for content moderation to detect and combat harmful speech and content on these platforms. Whilst it is easy enough to give a model a hard coded list of 'bad words' to look for and filter, this approach means models don't understand crucial context into exactly why something is 'bad' – whether it be tone, intent, cultural meaning, 'slang' etc – so may not flag harmful content correctly. This becomes even more challenging when multilingual environments are considered, which most social platforms are. With X, users may frequently mix languages within a single tweet. This code-mixing, coupled with transliteration (e.g. writing Hindi words using Latin characters), makes content moderation significantly harder as offensive language can be obscured or scattered across languages, meaning harmful content may bypass filters and fail to be detected by the model. This introduces the need for a model that is both multilingual and understands language in context, ensuring online spaces are kept safe without taking away users' ability to speak freely.

2. Background

2.1 Offensive Language and Its Impact

Platforms such as Discord and X (Twitter) incur high volumes of user-generated content daily, which, while encouraging online communication, also increases the risks of harmful or offensive speech. Offensive language is defined as derogatory or obscene language that can be presented in various forms (Mut Altın & Saggion, 2024). Hate speech is a form of abuse which targets individuals or groups based on identity traits such as ethnicity, religion, and sexual orientation, and is mainly harmful for silencing marginalised groups (Yin & Zubiaga, 2022). Offensive and abusive language on social media causes adverse impact on user psychological well-being, educational and social interactions, as well as creating societal polarisation. Exposure to digital abuse, such as cyberbullying, can potentially harm academic, social and emotional development. According to previous research, victims of cyberbullying experience significant emotional distress affecting their overall well-being. Additionally, the World Health

Organisation (WHO) reports that a rise in problematic social media use amongst young users can damage adolescent development (WHO, 2024). Moreover, emotional distress results in poorer concentration and lower academic achievement in cyberbullying victims (Faryadi, 2011).

2.2 Challenges in Current Moderation Systems

A study hiding toxic comments across Facebook, Twitter, and YouTube showed that reducing exposure to abusive or offensive language substantially reduces user engagement (Beknazar-Yuzbashev et al., 2025). These results indicate that offensive language, while socially harmful, acts as a potent engagement stimulus, underscoring the crucial need for more sophisticated text moderation approaches that can restrain offensive content without overly diminishing legitimate user interaction. Additionally, digital hate campaigns can lead to societal division, from public-health resistance to election misinformation, with cascading effects on democratic processes (Vasist et al., 2024).

2.3 Limitations of Existing Models

While improvements in automated abusive language detection models are observed, they underperform due to their reliance on lexical features, such as slur keywords and profanity (Yin & Zubiaga, 2022). Such models struggle to interpret the nuances, tones or context without any sentiment analysis. These Models' reliance on keywords results in struggles with implicit expressions without such features and non-abusive speech with such features. Reliance on keyword heuristics creates false negatives when abusive language is subtle and false positives when benign use overlaps with banned terms. For example, a relevant work in this context comes from AWS, where an attempt to fine-tune large models directly using label information was made. However, studies found that this approach lacks interpretable moderation process outputs and therefore results in poor performance (Ma et al., 2023).
These issues are multiplied in multilingual environments, where users often engage in code-mixing (switching languages within a single post), which can bypass primitive filtering systems.

2.4 Dataset Bias and Language Representation

Further challenges arise in machine learning model training, where imbalanced datasets can lead to biased moderation systems. Harmful content is often underrepresented in public datasets, resulting in models that perform poorly using real-world data. Current

text moderation models are predominantly trained and evaluated on an English dataset, leading to pronounced deficiencies when applied to non-English content. Models performing well in English could perform considerably worse when applied to underrepresented languages. Moreover, multilingual models must generalise across incredibly different grammatical structures and vocabulary conventions. The use of cross-lingual transfer learning methods can leverage high-resource data for better model generalisation on low-resource languages. Community-driven annotation, transparent reporting of performance, and fine-tuned moderation frameworks are suggested by recent studies to address these challenges (Mnassri et al., 2024).

2.5 Project Aim

Finally, this project aims to propose a moderation model that protects users from harmful and offensive content while preserving freedom of speech norms.

3. Methodology

3.1 Dataset

The dataset used was the Hate Speech and Offensive Content (HASOC) 2021 dataset. This dataset comprises short multilingual tweets that were designed to support research in detecting hate speech and offensive language on social media platforms.

This dataset was already annotated for the presence of offensive content, avoiding any need for self-supervised modelling. This data was comprised of English, Hindi (Devanagari script) and Marathi (also in Devanagari script) tweets. However, the Marathi data was limited and slightly differs from Hindi in terms of grammar, punctuation and other small details that would introduce unnecessary noise into the dataset; therefore, it was excluded, and only the English and Hindi data were used. The data was labelled either ‘HOT’ (Hateful or Offensive) or ‘NOT’ (Not offensive). It also included extra labels for the ‘HOT’ labelled data, such as ‘PRFN’ (Profane) and ‘OFFN” (Offensive). However, to simplify the classification task and align with the project’s goals, these subcategories were removed, and the task was framed as a binary classification problem.

The dataset consists of 8,437 rows, with 3,843 English tweets and 4,594 Hindi tweets. Of these, 3,934 were labelled as hateful or offensive, while 4,503 were labelled as non-offensive.

3.2 Data Preprocessing

Data preprocessing is a key component in NLP, in which data preparation is performed on raw data to prepare for the modelling procedure. This project applies various data preprocessing techniques to the HASOC dataset. Text preprocessing is particularly an important step in NLP as it aids machine learning algorithms to understand the data better by denoising, standardisation, tokenisation, label encoding, and augmentation. Each of these preprocessing techniques is further discussed in detail below.

Data cleaning and Denoising

Raw text often includes extra information such as URL tags, special characters, such as hashtags. In the case of tweets, a major issue with the raw input was the presence of hashtagged phrases, which are normally written in snake case (text is demonstrated as snake_case) or camel case (text is demonstrated as CamelCase) format. Cleaning the data confers several advantages, such as improved feature consistency, reducing vocabulary sparsity, and better model generalisation. For instance, applying hashtag expansion reduces variants of the same concept into single lexical items - #DataScience is converted to data science, lowering the effective vocabulary size. To remove such noises from the dataset, this report incorporated a split function where hashtags are split on underscore, digits, or camel cases are split based on where uppercase characters are present. Additionally, the data cleaning process ensures that HTML entities are decoded to English contractions for uniform tokenisation. For example, '&amp' is decoded to '&' and then further data preprocessing steps replace ampersands with 'and'. Repeated special characters are collapsed, and any non-Devanagari or non-ASCII characters, such as mojibake characters (corrupted characters during encoding/decoding), are removed. These preprocessing steps are applied to both the Hindi and English datasets to maintain consistency and reduce unnecessary vocabulary growth, as illustrated in Figure 2 below.

```
# Split hashtags into readable words (e.g. #NotCool → "Not Cool")
def split_hashtag(tag):
  """Split camelCase or snake_case hashtag into readable words."""
    tag = tag.replace("_", " ")
    tag = re.sub(r"([a-z])([A-Z])", r"\1 \2", tag)
    tag = re.sub(r"([A-Za-z])(\d)", r"\1 \2", tag)
    tag = re.sub(r"(\d)([A-Za-z])", r"\1 \2", tag)
    return tag
```

Figure 1. split_hashtag() function

```python
# Clean each tweet: expand contractins, decode html entities, remove non-roman characters, links, mentions, punctuations, and expand hashtags
def clean_text(text):
  """
    Clean tweet text: decode HTML, expand contractions, remove links/mentions,
    normalise punctuation/casing, and split hashtags.
    """
    text = html.unescape(text) # decodes HTML entities (i.e. '&amp;' -> '&')
    text = contractions.fix(text) # expands contractions (i.e. "don't" -> "do not")
    text = re.sub(r"([^\w\s])\1+", r"\1", text) # collapses repeated special characters (i.e. && -> &)
    text = re.sub(r"&", " and ", text) # replaces ampersands (&) with 'and'
    text = re.sub(r"[^\x20-\x7E\u0900-\u097F]", "", text) # removes any non-ASCII and non-Devanagari characters (mojibake characters were present)
    text = re.sub(r"http\S+|www\S+", "", text) # removes URLs
    text = re.sub(r"@\w+", "", text) # removes mentions (@'s)

    # expands hastags by splitting camelCase or snake_case instances (camelCase: "#CovidSucks" | snake_case: "#covid_sucks")
    hashtags = sorted(re.findall(r"#\w+", text), key=len, reverse=True)
    for tag in hashtags:
        tag_body = tag[1:]
        split = split_hashtag(tag_body)
        text = text.replace(tag, split)

    text = text.lower() # converts text to lowercase (e.g. "COVID" -> "covid")
    text = re.sub(r"[^\w\s]", "", text) # removes punctuation (e.g. "price: $1000" -> "price 1000")

    # Adds a trailing space after every instance of "covid" for hastags
    # that originally included "covid" or "COVID" before converting to lower case. Solving edge cases such as:
    # "#covidsucks" - hashtag splitting would result in "covidsucks", now --> "covid sucks"
    # "#COVIDSucks" - hashtag splitting would result in "COVIDSucks" ("covidsucks" after applying lowercase), now --> "covid sucks"
    text = re.sub(r"covid", "covid ", text)

    text = re.sub(r"_", " ", text) # normalises underscores (e.g. "covid_sucks" -> "covid sucks")
    text = re.sub(r"\s+", " ", text) # collapses multiple spaces (e.g. "covid   sucks" -> "covid sucks")
    text = text.strip() # strips edges (e.g. " covid " -> "covid")
    return text

# Apply cleaning to English and Hindi text
eng_df['text'] = eng_df['text'].astype(str).apply(clean_text)
hin_df['text'] = hin_df['text'].astype(str).apply(clean_text)
```

Figure 2. clean_text() function

Additionally, further data cleaning was performed to remove all links, mentions, and punctuations, and finally, hashtag separation was applied using the split_hashtag() function. These preprocessing steps resulted in much smoother input data for training the models. Figure 3 illustrates a small subset of the raw dataset text, which includes mentions and snake_case formatting, whereas Figure 4 shows the output after data cleaning, in which hashtags and mentions are removed and snake_case tokens are split into separate words, yielding smoother inputs better suited for modelling.

	id	lang	text	label
0	0	en	@wealth if you made it through this &...	HOF
1	1	en	Technically that's still turning back the cloc...	HOF
2	2	en	@VMBJP @BJP4Bengal @BJP4India @narendramodi @J...	NOT
3	3	en	@krtoprak_yigit Soldier of Japan Who has dick ...	HOF
4	4	en	@blueheartedly You'd be better off asking who ...	HOF

Figure 3. Five sample entries of the raw dataset

	id	lang	text	label
0	0	en	if you made it through this and were not only ...	HOF
1	1	en	technically that is still turning back the clo...	HOF
2	2	en	and you are the govt stop thinking about world...	NOT
3	3	en	soldier of japan who has dick head	HOF
4	4	en	you would be better off asking who does not th...	HOF

Figure 4. Sample entries from Figure 2 after the data cleaning process

3.3 Data Augmentation

Data augmentation is a preprocessing technique utilised to increase the volume, quality and diversity of training data (Mumuni & Mumuni, 2022). Back-translation is an augmentation technique in which the original texts are translated into a pivot language and then translated back into the source language. This process leads to producing paraphrased variants of the input data by introducing controlled semantic drift while preserving meaning. This project applies the Google Translate API to translate each tweet from English to Hindi as a pivot language and back to English, as well as Hindi to English and back to Hindi. In the process of back-translating Hindi to English to Hindi, a helper function was used to check for the script, to avoid corrupting Romanised Hindi. Each dataset is augmented by replacing the original text with its back-translated version. Finally, the augmented datasets are saved as new CSV files for future use.

3.4 Label Encoding

Machine learning models such as XLM-RoBERTa-base, DistilBERT-base-multilingual-cased, and IndicBERT used in this project only take numerical data. Label Encoding is a technique that is used to convert categorical columns into numerical values (aakarshachug, 2025). Label encoding assigns a unique integer to each category in the data. In this project, label encoding was applied to convert categorical labels ‘HOF’, ‘NOT’ into integers, 0 and 1, respectively.

3.5 Tokenisation

Tokenisation refers to splitting text into words or subwords, which are then converted to IDs (Hugging Face, n.d.). This process ensures that every input example is converted into fixed-size token ID sequences, which are compatible with the Transformer architecture. In this project, tokenisation was applied to both training and validation datasets, with automatic padding and truncation for variable-length input.

3.6 Dataset Splitting

Finally, the dataset is split into 60% train, 20% validation, and 20% for testing, preserving label distribution while using a fixed random seed to ensure reproducibility. When the random seed is fixed, it results in reproducible pseudo-random sequences, leading to the result being able to be reproduced (Kethinedi, 2023). Additionally, to maintain the balance between ‘HOF’ and ‘NOT’ classes, class weights are computed and converted into a tensor for later application in loss functions.

3.7 Modelling

This project conducts experiments with three pre-trained models for performing the offensive data classification task using XLM-RoBERTa-base, DistilBERT-base-multilingual-cased, and IndicBERT by AI4Bharat. This report further conducts a comprehensive analysis of the performance results of the mentioned model. Each model is further discussed below.

XLM-RoBERTa-base

XLM-RoBERTa base model, a multilingual version of RoBERTa, is pre-trained on 2.5TB of filtered Common Crawl data containing 100 languages. This model is a transformer model pre-trained on a large self-supervised corpus. This model was pre-trained with the Masked Language Modelling (MLM) which means the model randomly masks 15% of the words in the inp, ut then run the entire masked sentence through the model to predict the masked words allowing the model to learn a bidirectional representation of the sentences (Hugging Face, n.d.). This resulted in the model learning an inner representation of 100 languages that can be further used to extract features useful for tasks such as classification.

DistilBERT-base-multilingual-cased

This model is a distilled version of the BERT base multilingual model trained on the concatenation of Wikipedia in 104 different languages. The model contains six layers, 768 dimensions and 12 heads, totalling 134M parameters, which is twice as fast as mBERT-base model, making it more suitable in terms of computational resources (Hugging Face, n.d.). However, the reduced number of layers compared to the m-BERT model may degrade the performance on complex semantic meaning. Additionally, DistilBERT retains only 97% of BERT's original GLUE score, leading to an average drop in accuracy across tasks(Sanh et al., 2019).

IndicBERT

IndicBERT is a multilingual ALBERT model pre-trained exclusively on 12 major indian languages. It is pre-trained on a monolingual corpus of around 9B tokens. The 12 languages covered by this model include Hindi and English, which are concentrated in this project (Kakwani et al., 2020). Studies state that IndicBERT addresses the limitations often encountered by applying multilingual models like mBERT to the intricacies of Indian languages (Arora, 2025). The ALBERT architecture suggests a deliberate strategy to develop a high-performing model while minimising the number of parameters compared to standard BERT models, which makes it more suitable for broader utilizations such as this report's multilingual classification task (Arora, 2025). Despite various strengths, such as successful fine-tuning for sentiment classification tasks in code-mixed Dravidian languages, one notable limitation is its context window size, which is restricted to 128 tokens. This limitation can create challenges when analysing longer texts, which require splitting the longer dataset entries into smaller segments.

3.8 Hyperparameter Tuning

This report leverages Optuna, which is an automated hyperparameter optimisation framework commonly used in AI projects. Optuna was deployed to identify the most effective training configurations to maximise the model performance. This report configured Optuna to explore a variety of hyperparameters used in transformer fine-tuning, including learning rate, weight decay, batch size, and warm-up step. Continuous parameters were sampled uniformly while discrete choices were drawn from predefined lists as illustrated in Figure 4. In this instance, 40 trials per epoch. The best model configurations were selected based on the highest F1-score on the validation set, and further saved in dictionaries for further access without having to search for hyperparameters again, as illustrated in Figure 5.

```python
# Define hyperparameter search space
hp = {
    "learning_rate": trial.suggest_float("learning_rate", 1e-5, 5e-5, log=True),
    "weight_decay": trial.suggest_categorical("weight_decay", [0.0, 0.01, 0.1]),
    "per_device_train_batch_size": trial.suggest_categorical("per_device_train_batch_size", [8, 16, 32]),
    "warmup_steps": trial.suggest_categorical("warmup_steps", [0, 100, 250, 500]),
    "num_train_epochs": trial.suggest_int("num_train_epochs", 2, 5),
    "adam_beta1": trial.suggest_float("adam_beta1", 0.85, 0.95),
    "adam_beta2": trial.suggest_float("adam_beta2", 0.95, 0.999),
    "adam_epsilon": trial.suggest_float("adam_epsilon", 1e-8, 1e-6, log=True),
    "gradient_accumulation_steps": trial.suggest_categorical("gradient_accumulation_steps", [1, 2, 4]),
    "lr_scheduler_type": trial.suggest_categorical("lr_scheduler_type", ["linear", "cosine", "cosine_with_restarts", "polynomial"]),
    "label_smoothing_factor": trial.suggest_float("label_smoothing_factor", 0.0, 0.2),
}
```

Figure 4. Hyperparameter search space for optuna hyperparameter fine-tuning

```python
xlm_best_params = {
    "learning_rate": 3.5377e-5,
    "weight_decay": 0.01,
    "per_device_train_batch_size": 32,
    "warmup_steps": 100,
    "num_train_epochs": 3,
    "adam_beta1": 0.8814,
    "adam_beta2": 0.9763,
    "adam_epsilon": 1.279e-8,
    "gradient_accumulation_steps": 1,
    "lr_scheduler_type": "polynomial",
    "label_smoothing_factor": 0.1538
}

distil_best_params = {
    "learning_rate": 2.1513e-5,
    "weight_decay": 0.0,
    "per_device_train_batch_size": 8,
    "warmup_steps": 250,
    "num_train_epochs": 4,
    "adam_beta1": 0.9172,
    "adam_beta2": 0.9604,
    "adam_epsilon": 4.4486e-8,
    "gradient_accumulation_steps": 1,
    "lr_scheduler_type": "cosine",
    "label_smoothing_factor": 0.0297
}

indic_best_params = {
    "learning_rate": 4.1161e-5,
    "weight_decay": 0.0,
    "per_device_train_batch_size": 8,
    "warmup_steps": 0,
    "num_train_epochs": 4,
    "adam_beta1": 0.8600,
    "adam_beta2": 0.9667,
    "adam_epsilon": 1.5734e-8,
    "gradient_accumulation_steps": 2,
    "lr_scheduler_type": "cosine_with_restarts",
    "label_smoothing_factor": 0.1517
}
```

Figure 5. Best hyperparameters found after running optuna on the validation set

3.9 Original Plan and Deviations

Transliteration

The original plan included transliterating Hindi text from Devanagari script to Roman script in an attempt to normalise the input format and reduce variability in token representations. However, after investigation, it was revealed that the transformer models that were planned to be used already handle Devanagari script well due to their multilingual pretraining. Additionally, transliteration also introduces ambiguity in the data as multiple Devanagari script Hindi words map to the same Roman form, which introduces the risk of semantic loss and misinterpretation. After deliberation, transliteration was excluded from the planned data preprocessing pipeline.

Slang Normalisation

Slang normalisation was also originally planned, which includes converting slang back to their formal counterparts, to normalise the input data and reduce data variability. The motivation to include slang handling was due to the prevalence of slang used in social media posts, and since there are multiple forms of slang for specific words, it includes extra noise that would be fed into the models. However, upon further analysis of the models that were planned, it was found that these models already handle slang-containing text quite well. In addition to this, since this project aims to create a deployable model that can generalise well to real-world platforms, like Twitter (where slang is very common), it was decided to exclude any direct attempt at handling slang.

By excluding transliteration and slang normalisation from the data preprocessing pipeline, the natural linguistic characteristics of social media content were preserved, allowing the models to learn directly from authentic usage patterns and better generalise to real-world deployment scenarios.

4. Experimental Results

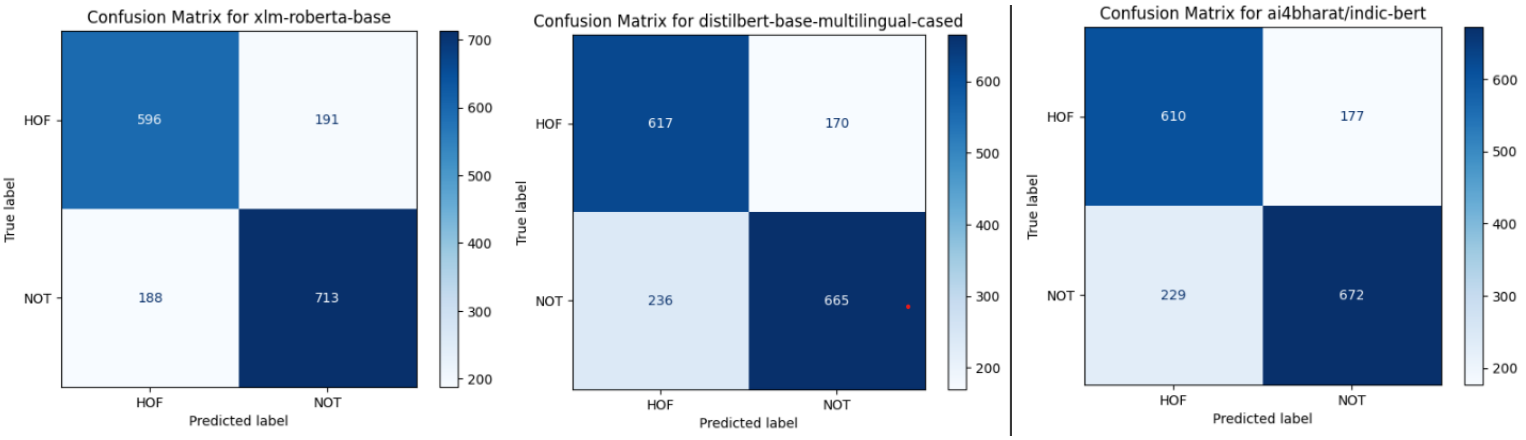
4.1 Results

All three models were evaluated on the HASOC 2021 dataset using precision, recall, F1-score and accuracy. The table below shows that all three models competed competitively.

Model	Accuracy	Precision	Recall	F1 (Macro)	F1 (Weighted)
XLM-RoBERTa	0.78	0.77	0.77	0.77	0.78
DistilBERT	0.76	0.76	0.76	0.76	0.76
IndicBERT	0.76	0.76	0.76	0.76	0.76

Table 1: Summary of evaluation metrics on models

As seen in the table above, XLM-RoBERTa achieved the highest percentages in all categories. The difference may seem small, but it is meaningful in large-scale systems where small gains reduce moderation errors significantly. The real differences between the models were shown when a confusion matrix was used on their test results.

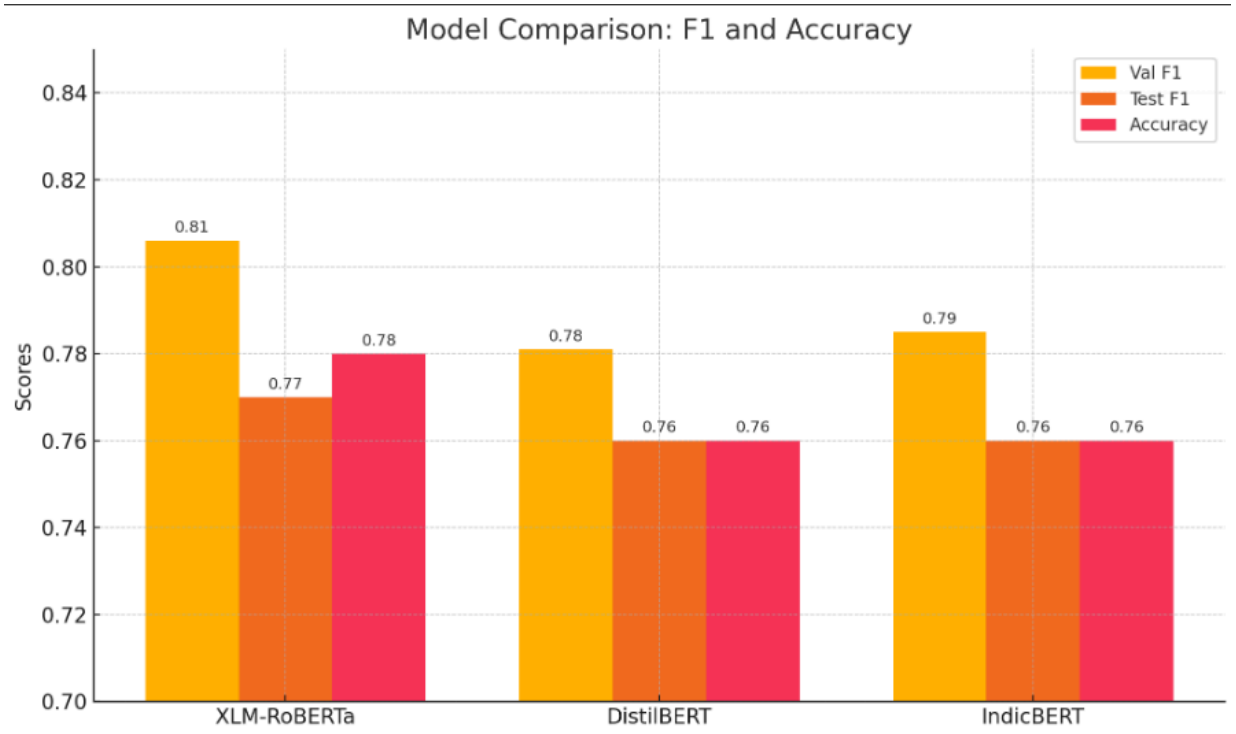


(a). XLM-RoBERTa *(b) DistilBert* *(c) IndicBert*

Figure 6. Confusion matrices for models

These confusion matrices show a key trade-off. DistilBERT and IndicBERT correctly flagged more hate/offensive tweets (higher true HOF) but also wrongly flagged many non-offensive tweets (higher false NOT). In contrast, XLM-RoBERTa has slightly fewer true positives but also significantly fewer false positives.

This is important when taking into consideration the user experience, as having non-offensive content flagged as being hateful or offensive may lead the user to feel wrongly punished or censored.



4.2 Discussion

Model Performance Overview

Looking at these results, the test F1-scores across the three models are modest, only ranging between 0.76 and 0.77. This shows that the nature of this dataset is challenging, as social media language is messy, informal and highly situational and context-dependent, which leads to models having a difficult time interpreting it accurately.

Comparative Model Analysis

XLM-RoBERTa not only had the best test results compared to the other models, but also showed the highest validation F1 Score, which suggests better generalisation and less overfitting. XLM-RoBERTa performed best likely because it's a larger multilingual model trained on over 100 languages. This gives it an upper hand in situations where code switching or non-standard text is used. DistilBERT is a more compressed version of BERT, which sacrifices some of its depth for speed, which explains its lower accuracy. IndicBERT is designed and trained on 12 major Indian languages, which may have given it an advantage on the Hindi data but possibly caused it to be less effective overall due to its limited pretraining coverage. Although XLM-RoBERTa produced the best results, it is also a larger model with a high number of parameters. This led to longer training times and increased system requirements. This could make it impractical for lower resource environments or when being implemented into real-time systems. On the other hand, DistilBERT is significantly smaller and faster; for this reason, it would be more suitable for real-time or constrained deployments. Therefore, despite being only slightly accurate, DistilBERT could be the preferred solution in certain situations.

Influence of Preprocessing and Optimisation

The preprocessing choices also made an impact on these results. The use of data augmentation through back translation to paraphrase the text while preserving its semantic meaning; however, it may have produced noise or unnatural phrasing in the returned text, slightly affecting the models' confidence. Tokenisation and padding were also applied to the dataset to ensure consistent input formats across the models, allowing them to handle the data consistently and efficiently. Additionally, the choice to use hyperparameter tuning was performed on each model with optuna, which optimised the parameters that the models use, such as learning rate, weight decay, batch size, etc. This choice was crucial in improving the models' performance as it was able to find the best configurations based on the highest validation F1-score, providing us with the most optimal solution.

5. Conclusion

Creating safe online spaces requires effective multilingual content moderation that can detect harmful content across different languages, respect cultural nuances, and provide equitable protection without compromising freedom of expression. Using the HASOC 2021 dataset, the challenges posed by multilingual content were investigated and discussed how traditional moderation systems often fall short in these contexts. Among the models tested, XLM-RoBERTa achieved the highest performance across key metrics such as accuracy, precision, recall, and F1-score. However, the marginal improvement it offered over DistilBERT came at the cost of significantly longer training times, raising questions about its practicality in resource-constrained settings. Despite this, its ability to generalise across languages and identify subtle forms of abuse makes it well-suited for real-world multilingual environments. The findings reinforce the idea that successful content moderation lies at the intersection of technical rigour and ethical responsibility, requiring a careful balance between user safety and freedom of expression.

To build upon the current findings, the next steps include incorporating larger and more diverse datasets to enhance the generalisability of the models, allowing them to provide better coverage of rare or nuanced linguistic patterns, particularly in multilingual text.

Secondly, a re-evaluation of the preprocessing and modelling pipeline may offer new insights. For example, exploring alternative text normalisation strategies, such as emoji interpretation and revisiting slang normalisation and transliteration. Additionally, experimenting with ensemble methods may yield an improved performance and reliability by allowing the models to leverage their complementary strengths, especially in ambiguous cases.

Thirdly, expanding the linguistic scope of the dataset by including additional languages. This would increase the model's inclusiveness and applicability to a wider range of users online.

Lastly, integrating real-time deployment scenarios enables practical evaluation of model performance under live conditions, such as reliability and scalability.

6. References

aakarshachug. (2025, February 12). *Label Encoding in Python*. GeeksforGeeks. Retrieved May 14, 2025, from http://geeksforgeeks.org/ml-label-encoding-of-datasets-in-python

Arora, P. (2025). *Indic BERT: A comprehensive analysis* [Blog post]. Research Kits. Retrieved May 14, 2025, from https://puneetarora.substack.com/p/indic-bert-a-comprehensive-analysis

Beknazar-Yuzbashev, G., Jiménez-Durán, R., McCrosky, J., & Stalinski, M. (2025). *Toxic content and user engagement on social media: Evidence from a field experiment* (CESifo Working Paper No. 11644). Munich Society for the Promotion of Economic Research. Retrieved from https://ssrn.com/abstract=5130929

Faryadi, Q. (2011). *Cyber bullying and academic performance*. International Journal of Computer Engineering Research, 1(1), 2250–3005.

HASOC 2021 Dataset. (2021). *Hate Speech and Offensive Content Identification in Indo-European Languages*. Shared Task at FIRE 2021. Retrieved from https://github.com/Kalit31/HASOC-2021

Hugging Face. (n.d.). *distilbert/distilbert-base-multilingual-cased* [Pre-trained language model]. Retrieved May 14, 2025, from https://huggingface.co/distilbert/distilbert-base-multilingual-cased

Hugging Face. (n.d.). *FacebookAI/xlm-roberta-base* [Pre-trained language model]. Retrieved May 14, 2025, from https://huggingface.co/FacebookAI/xlm-roberta-base

Hugging Face. (n.d.). *Summary of the tokenizers*. In *Transformers documentation*. Retrieved May 14, 2025, from https://huggingface.co/docs/transformers/en/tokenizer_summary

Kakwani, D., Kunchukuttan, A., Golla, S., N. C., G., Bhattacharyya, A., Khapra, M. M., & Kumar, P. (2020). *IndicNLPSuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages*. In *Findings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Kalev Leetaru. (2019, February 12). *Terabytes Of News Versus One Trillion Tweets: Is Social Media Really As Big As We Think?* Forbes. https://www.forbes.com/sites/kalevleetaru/2019/02/12/terabytes-of-news-versus-one-trillion-tweets-is-social-media-really-as-big-as-we-think/

Mumuni, A., & Mumuni, F. (2022). *Data augmentation: A comprehensive survey of modern approaches*. Array, 16, Article 100258. https://doi.org/10.1016/j.array.2022.100258

Mut Altın, L. S., & Saggion, H. (2024). *Review of offensive language detection on social media: Current trends and opportunities*. In F. P. García Márquez, A. Jamil, A. A. Hameed, & I. Segovia Ramírez (Eds.), *Emerging trends and applications in artificial intelligence: ICETAI 2023* (Lecture Notes in Networks and Systems, Vol. 960). Springer, Cham. https://doi.org/10.1007/978-3-031-56728-5_6

Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). *DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter* (arXiv preprint arXiv:1910.01108). https://arxiv.org/abs/1910.01108

Twitter Usage Statistics - Internet Live Stats. (2009). *Internetlivestats.com*. https://www.internetlivestats.com/twitter-statistics/

Vasist, P. N., Chatterjee, D., & Krishnan, S. (2024). *The polarizing impact of political disinformation and hate speech: A cross-country configural narrative*. Information Systems Frontiers, 26, 663–688. https://doi.org/10.1007/s10796-023-10390-w

Yin, W., & Zubiaga, A. (2022). *Hidden behind the obvious: Misleading keywords and implicitly abusive language on social media*. Online Social Networks and Media, 30, Article 100210. https://doi.org/10.1016/j.osnem.2022.100210

7. Appendix

7.1 Contributon Table

Team Member	Contribution
Dean Stokeld (Team lead)	➢ Coding ➢ Experimentation ➢ Edited/finalised slides ➢ Solo-presented ➢ Formatted the report ➢ Added 3.1 & 3.9 to methodology ➢ Appendix ➢ Edited/finalised abstract & conclusion ➢ Corrected spelling/grammatical errors in the report ➢ Delegated tasks & other organisational tasks
Tarlan Kameli	➢ Background ➢ Methodology
Megan Hastie	➢ Introduction ➢ Final editor
Alexander Easy	➢ Results interpretation ➢ Discussion
Sophie Gaou	➢ Abstract ➢ Conclusion ➢ Presentation slides draft

7.2 Generative AI Usage

ChatGPT prompt for Optuna search code assistance:

https://chatgpt.com/share/682437fb-7b0c-8006-99f9-bdf63ee902e7

7.3 GitHub Link

https://github.com/DeanStokeld/B.A.I/blob/main/41043%20%20Natural%20Language%20Processing%20-%20Autumn%202025/nlp_a3_24923029.ipynb