Gendered Abuse Detection in Indic Languages

Anonymous ACL submission

Abstract

The rise of online gender-based violence is a significant societal concern, exacerbating existing social and economic inequalities. This issue drives individuals away from digital spaces, limiting their access to critical opportunities in areas such as politics and the economy. In extreme cases, GBV can result in severe harm or even loss of life. This paper introduces a novel dataset designed to address this gap, focusing on gendered abuse in Hindi, Tamil, and Indian English. Each post is annotated to assess whether it constitutes gendered abuse, whether it is targeted at individuals from marginalized genders or sexualities, and whether it contains explicit or aggressive language. The evaluation of these models will focus on the F1 score, emphasizing the importance of balancing precision and recall in imbalanced classification tasks.

1 Introduction

The rise of online gender-based violence is a serious issue that makes existing social and economic problems even worse. It pushes people away from digital spaces, limiting their opportunities in important areas like politics and the economy. In extreme cases, this kind of abuse can even result in harm or loss of life. As the need for automated systems to detect gendered abuse grows, there is a significant gap in available datasets that can be used to tackle this problem, especially for content in Indian languages.

This project aims to fill that gap by using a new dataset focused on gendered abuse in Hindi, Tamil, and Indian English. The dataset, created by 18 activists and researchers who have experienced or studied gendered abuse, includes 7638 posts in English, 7714 posts in Hindi, and 7914 posts in Tamil. It has been annotated to answer three main questions about the content of each post:

Is the post gendered abuse when not directed at someone from a marginalized gender or sexuality? Is the post gendered abuse when directed at someone from a marginalized gender or sexuality? Is the post explicit or aggressive? The goal is to build a classifier that can accurately identify gendered abuse based on these annotations. The dataset includes labels that help determine whether a post meets these criteria. The labels are:

1: This means the annotator believes the post matches the given label (e.g., gendered abuse, explicit language). 0: This means the annotator doesn't believe the post matches the label. NL: This means the post was assigned to the annotator but not annotated. NaN: This means the post wasn't assigned to the annotator at all. The task consists of three subtasks:

Create a classifier using only the provided dataset to detect gendered abuse (label 1). Use transfer learning from other publicly available datasets for hate speech and toxic language detection in Indic languages to help build a classifier for gendered abuse. Build a multi-task classifier that predicts both gendered abuse (label 1) and explicit language (label 3) in a single model. The evaluation of these models will focus on the F1 score, a commonly used metric for multi-label classification tasks. The F1 score balances both precision (how many of the predicted labels were correct) and recall (how many of the true labels were found), making it a valuable tool, especially in situations where data is imbalanced.

2 Related Studies

2.1 Automated Hate Speech Detection and the Problem of Offensive Language

The study *Automated Hate Speech Detection and the Problem of Offensive Language* addresses the challenge of distinguishing hate speech from offensive language on social media. The authors define hate speech as language aimed at marginalized groups with intent to harm, while offensive language includes terms that may be inappropriate or insulting but are not necessarily hateful. The complexity of this distinction is emphasized, especially given the variety of language used online. To investigate this, the researchers collected a large dataset of tweets containing words from a hate speech lexicon and manually categorized them into three groups: hate speech, offensive language, or neither.

They trained machine learning classifiers, such as logistic regression, using a variety of linguistic and syntactic features to predict these categories. Although their best model showed high overall performance, it frequently misclassified hate speech as less offensive. This highlighted how often hate speech is either overlooked or wrongly labeled. The authors argue that lexical approaches, like using the Hatebase lexicon, were not sufficient for detecting hate speech accurately. Many tweets containing flagged keywords were not actually hate speech according to human annotators, and tweets without those keywords often went undetected. Terms like "fggot" and "ngger" were useful in identifying hate speech, but their absence posed a major challenge.

The study concludes that more nuanced models are needed to distinguish hate speech from merely offensive language, especially given the social and legal consequences of these labels. The authors suggest future research should explore social context, the intentions behind messages, and potential biases in annotation—particularly around issues like racism, sexism, and homophobia—to improve detection systems.

2.2 Detecting Hate Speech in Social Media

The paper *Detecting Hate Speech in Social Media* by Shervin Malmasi and Marcos Zampieri explores how to detect hate speech on social platforms and differentiate it from general profanity. The study introduces a new annotated dataset of English tweets categorized into three classes: hate speech (HATE), offensive but not hateful (OFFENSIVE), and clean content (OK). The authors highlight the need for better automated systems to monitor growing instances of online abuse and argue that existing binary classification approaches—hate vs. not hate—are too simplistic.

To build effective models, they apply a multi-class classification approach using a linear Support Vector Machine (SVM). They extract three types of features: character n-grams (from 2 to 8 characters), word n-grams (from 1 to 3 words), and word skip-grams (bigrams with skips of 1, 2, or 3 words). These features aim to capture stylistic patterns and word relationships that signal hateful or offensive intent. The dataset includes over 14,000 tweets, each labeled by at least three annotators. The authors use 10-fold stratified cross-validation to ensure the classifier's robustness across different data splits.

Their results show that character 4-grams provided the best performance, achieving 78% accuracy. Interestingly, combining all features—which increased the feature space to over 5 million—did not outperform the simpler character-based model. The study also notes that because of the class imbalance in the dataset, the majority class baseline was already high, and their oracle classifier (which combines outputs of all classifiers) reached up to 91.6% accuracy. This indicates that while the model performs reasonably well, there is still room for improvement.

In summary, the authors stress that distinguishing hate speech from offensive content is difficult due to overlapping language.

3 Methodology

3.1 Baseline 1

3.1.1 Task 1: Hate Speech Detection in English, Hindi, and Tamil

For Task 1, our goal was to build a reliable baseline model for detecting hate speech in three languages—English, Hindi, and Tamil. We approached this using a simple yet effective pipeline built around transformer-based models. We then designed a preprocessing function to clean the raw social media text data by removing URLs, user mentions, hashtags, and unnecessary whitespace. This helped reduce noise in the input and made the data more suitable for modeling. After that, we loaded the training and test datasets for each language, carefully handling missing files, incorrect labels, and malformed rows. For modeling, we selected language-appropriate pre-trained transformers: distilroberta-base for English, and bert-base-multilingual-cased for both Hindi and Tamil. These models were fine-tuned using HuggingFace's Trainer API. We tokenized the text inputs to a fixed maximum length and created custom PyTorch dataset objects. The models were trained for 3 epochs using the AdamW optimizer with a batch size of 64. Evaluation was done using accuracy and weighted F1-score, with detailed classification reports to ensure balanced performance.

3.1.2 Task 2: Transfer Learning

Our primary objective was to do transfer learning and fine-tune transformer-based models on curated datasets specific to each language. The English dataset was sourced from the widely-used tweet-eval benchmark, specifically its "hate" subset, while the Hindi and Tamil datasets were drawn from HASOC 2021 and the Tamil Offensive Language dataset respectively. The pipeline began with a preprocessing utility function (clean-text) that removed URLs, mentions, and hashtags, which often introduce noise into social media texts. For Hindi, we filtered the HASOC dataset to include only Task 1 labels (NOT and HOF), mapping them to binary values (0 and 1). Similarly, in the Tamil dataset, we removed non-Tamil entries and converted offensive labels into binary categories.

Next, we created a custom PyTorch dataset class (HateSpeechDataset) which handled tokenization and formatting using HuggingFace tokenizers. We tokenized all text to a maximum length of 128 tokens and returned attention masks and labels in the required format. For modeling, we selected language-appropriate pre-trained models—distilroberta-base for English and bert-base-multilingual-cased for both Hindi and Tamil. We used evaluation metrics like weighted F1-score and accuracy, which were computed using scikit-learn's f1-score and accuracy-score functions.

3.1.3 Task 3: Joint Classification Approach for Gendered Abuse and Explicit Language Detection

This Task 3 Baseline 1 implementation presents a joint classification approach to detect both gendered abuse and explicit language within a single model across three languages: English (en), Hindi (hi), and Tamil (ta). Instead of training separate models for each task, it uses one model (SafeClassifier) with a shared transformer encoder (like distilroberta-base for English and bert-base-multilingual-cased for Hindi/Tamil) and two separate classification heads: one for detecting gendered abuse (label1) and another for explicit language (label3).

The pipeline starts by loading the language-specific training and testing datasets using the load-joint-data() function, which merges two CSVs per language—one for each label—while filtering out malformed or missing entries. It also performs basic text cleaning to remove URLs, mentions, and hashtags. Once loaded, the texts are tokenized using a pre-trained transformer tokenizer suitable for that language. The custom RobustDataset class wraps these tokenized inputs and returns both labels as a single tensor for joint learning.

The SafeClassifier model takes the transformer's pooled output (typically the [CLS] token embedding) and passes it through two linear layers—gendered-head and explicit-head—to predict both labels. During training, it calculates the cross-entropy loss for each head and sums them to get the total loss. The compute-metrics() function evaluates model performance using weighted F1-scores and accuracy for both tasks. The results for each language are printed, showing how well the model performs in classifying gendered and explicit abuse.

3.2 Baseline 2

3.2.1 Task 1: Hate Speech Detection in English, Hindi, and Tamil

This Task 1 - Baseline 2 code implements a language-specific gendered abuse detection system using the multilingual transformer model xlm-roberta-base. We treat each language—English, Hindi, and Tamil—independently, training a separate binary classifier for each, which helps tailor the model to language-specific features and noise patterns.

The pipeline begins with robust preprocessing through preprocess-text(), which performs both universal cleaning (removing URLs, hashtags, and markup) and language-sensitive filtering to retain only characters relevant to each script (e.g., Devanagari for Hindi, Tamil Unicode for Tamil). Then, load-single-task-data() is used to gracefully handle corrupted files and label inconsistencies, converting various noisy representations (like 'NL', '1.0', '0.') into clean binary labels (0 or 1).

Once data is validated and cleaned, we tokenize it using AutoTokenizer for xlm-roberta-base, and wraps it into a PyTorch-compatible SilentDataset. The training logic dynamically adjusts batch size and number of epochs based on dataset size to prevent under- or overfitting.

After training, the model is evaluated on a test set and prints clean performance metrics.

3.2.2 Task 2: Transfer Learning

In this baseline approach for Task 2, we use transfer learning to train individual binary classification models for English, Hindi, and Tamil. Transfer learning allows us to start with powerful pre-trained language models—like DistilBERT for English and Multilingual BERT for Hindi and Tamil—that have already learned a deep understanding of language from large-scale text corpora. Instead of training a model from scratch, we fine-tune these pre-trained models on our specific abuse detection task, which makes the training faster and more effective, especially when working with limited data.

Each model is designed to detect whether a given social media post is abusive (label 1) or not (label 0). For English, we use the TweetEval dataset focused on hate speech, while for Hindi and Tamil, we use annotated datasets from HASOC 2021. Before training, we clean the text data by removing URLs, mentions, and hashtags to ensure the model focuses on the core linguistic content. The cleaned text is then tokenized and passed into the respective pre-trained models, which are fine-tuned over a few epochs using the AdamW optimizer.

During fine-tuning, the models learn task-specific patterns, adapting their general language understanding to recognize abusive language in each target language. After training, the models are evaluated using standard metrics like accuracy and F1 score. These evaluations help us measure how well the models can correctly classify posts while balancing precision and recall.

3.2.3 Task 3: Joint Multitask Classification for Gendered Abuse and Explicit Content

In Task 3 of Baseline 2, we are training a joint classification model that can detect both gendered abuse and explicit content in text across three languages: English, Hindi, and Tamil. The model uses a multitask learning approach, where a shared model (based on XLM-Roberta, a multilingual transformer) processes the input text and learns to classify both tasks simultaneously. This allows the model to benefit from shared knowledge across languages, improving performance on each task.

The model architecture consists of two parts: the shared encoder and two task-specific classification heads. The shared encoder is responsible for understanding the text, while the two heads independently classify the text for gendered abuse and explicit content. The predictions from both tasks are combined, and the model is trained to optimize performance for both tasks at the same time.

By training both tasks together, the model can learn to detect these types of harmful content more effectively, leveraging the shared knowledge of the language model while also focusing on the specific needs of each classification task. This approach helps the model handle both types of abuse detection without needing separate models for each task.

3.3 Final Advanced Model

3.3.1 Task 1: Hate Speech Detection in English, Hindi, and Tamil

The Finalised Advanced Model for Task 1 focuses on detecting gendered abuse and explicit content in social media posts in English, Hindi, and Tamil. It uses a series of advanced techniques to improve classification accuracy, particularly for imbalanced datasets.

First, the model cleans the input text by removing unnecessary elements like URLs, mentions, and hashtags. For Hindi and Tamil texts, it transliterates Romanized text into the native script, making it easier for the model to understand. This preprocessing helps standardize the text across languages.

To further enhance the training process, the model uses back translation, where English text is translated into Hindi and then back into English. This generates slightly different versions of the same text, which helps the model become more robust and generalize better on unseen data.

For training, the model employs a custom loss function called Focal Loss, which helps the model focus on harder-to-classify examples. This is especially useful in dealing with imbalanced datasets, where one class (e.g., non-abusive content) might dominate. The model also computes class weights to address this imbalance and uses a custom trainer to incorporate Focal Loss during training.

The model is fine-tuned using pre-trained transformer models, such as distilroberta-base for English and indic-bert for Hindi and Tamil. Evaluation metrics like F1-score and accuracy are used to assess performance, ensuring the model's effectiveness across multiple languages and diverse social media content.

3.3.2 Task 2: Transfer Learning

The finalized implementation for Task 2 involves training a multilingual model to detect offensive and abusive language in **English**, **Hindi**, and Tamil using the DeBERTa-v3 model for sequence classification. The first step in the process is data cleaning, where unnecessary elements such as mentions (@), hashtags (), and URLs are removed from the text. Additionally, language-specific characters are filtered out (e.g., Hindi characters are limited to Unicode characters in the Hindi script, and Tamil characters are similarly constrained). The data is then augmented using various techniques tailored to each language, such as synonym replacement and word swapping for English, and character-level swapping and word substitution for Hindi and Tamil, which introduces more variety into the training data and improves the model's robustness against different writing styles.

The external datasets for each language are loaded from CSV files containing text and labels (1 for offensive, 0 for non-offensive). These datasets are preprocessed and augmented before being merged with target datasets. In the case of the **English dataset**, the labels are derived from a combination of columns (e.g., toxic and severe_toxic). For **Hindi**, labels are mapped from categories such as "HOF" (Hate Offensive) to 1 and "NOT" (Non-offensive) to 0. **Tamil** data is cleaned and label-mapped to categories such as "not_offensive" (0) and various offensive types (1).

A custom **MultilingualModel** is constructed using the DeBERTa-v3 architecture. The model consists of a base DeBERTa-v3 sequence classification model, enhanced with **language-specific adapters** that modify the hidden states for each language. These adapters, implemented as linear layers, ensure the model is better at capturing language nuances. The model is designed to take in both the input text and the corresponding language identifier, allowing it to adjust its processing based on the language.

The training process uses the **Trainer** class from Hugging Face Transformers, with custom modifications for calculating loss. **Cross-entropy loss** is used for classification, and the **F1 score** and **accuracy** are computed as evaluation metrics. The model is trained for **3 epochs**, using a batch size of **16** for training and **32** for evaluation, with a learning rate of **2e-5** and gradient accumulation enabled to handle larger batch sizes. Additionally, **early stopping** is enabled to prevent overfitting by saving the best model based on validation performance.

After training, the model is evaluated for each language using separate validation sets. The evaluation results (accuracy and F1 score) are printed for each language, providing a detailed view of the model's performance across English, Hindi, and Tamil. The trained model and tokenizer are then saved for future use or deployment.

This implementation incorporates a combination of multilingual data augmentation, domain-specific model adaptation, and efficient training strategies to build a model capable of understanding and detecting offensive language across multiple languages. The final model is expected to generalize well to unseen data, especially in real-world multilingual settings where offensive language may vary widely across languages.

3.3.3 Task 3: Joint Multitask Classification for Gendered Abuse and Explicit Content

The goal of this task is to train a multilingual model to detect offensive and abusive language in texts written in English, Hindi, and Tamil. The model predicts two types of abuse: gendered abuse and explicit language. To achieve this, a BiLSTM-based architecture with multitask learning is used, allowing the model to handle both tasks simultaneously, which helps improve its performance by learning shared features.

Before training, the data is cleaned by removing unnecessary elements like mentions, hashtags, and URLs. This ensures that the model focuses on the actual content of the text. Additionally, the labels are defined for each text, indicating whether it contains gendered abuse or explicit language. The text is then tokenized using a language-specific tokenizer, making it ready for input to the model.

The model itself is a BiLSTM, which is effective for processing sequential data like text. It captures information from both the past and future context of each word, which helps improve the model's understanding of the text. The model is designed to predict both tasks using a shared architecture, making it more efficient by leveraging the similarities between the two tasks.

Finally, the model is trained using the Hugging Face Trainer, which handles the training process, including batching, loss calculation, and evaluation. The training involves a few epochs, and the model is evaluated based on accuracy and F1 score for both tasks. This multitask approach ensures that the model is well-equipped to detect different types of offensive language in multilingual settings.

4 Dataset and Experimental Setup

4.1 Overview

The dataset consists of annotated social media posts in three languages: English (EN), Hindi (HI), and Tamil (TA). The task involves detecting abusive language, gendered abuse, and explicit content, with annotations provided by multiple human annotators. Each language-specific file contains labeled data for training and testing a model. There are 9 total training and testing sets (3 languages × 3 labels), allowing for comprehensive evaluation across language and label types.

Each dataset for English , Hindi , and Tamil is structured with the following elements:

Text (post content): This is the social media post that has been labeled by annotators. Annotator labels: For each language file, there are multiple annotators' opinions on whether a post matches the given labels (abusive, gendered abuse, explicit language). English language dataset has columns for six annotators: en-a1, en-a2, ..., en-a6. Similarly, the Hindi and Tamil datasets have corresponding annotator columns (e.g., hi-a1, ..., ta-a1, ...).

4.2 Labels

The labels used by annotators are as follows:

1: The annotator believes the post matches the given label (e.g., abusive, explicit language). 0: The annotator does not believe the post matches the label. NL: The post was assigned to the annotator but not annotated. NaN: The post wasn't assigned to the annotator.

4.3 Data Distribution

Training and Testing Sets: The data is split into training and testing sets for each language and label combination, resulting in 9 training sets and 9 testing sets: Training: For each language, there are three training sets (one for each label: abusive, gendered abuse, and explicit language). Testing: Similarly, there are three testing sets for each language, corresponding to the same labels.

4.4 Annotators

The labeling is done by multiple annotators, which means the quality of labels can vary.

- Some posts may have disagreements between annotators, with some annotators marking a post as 1 (abusive), while others mark it as 0 (non-abusive).
- The use of "NL" and "NaN" values indicates that there may be posts where the annotator either skipped the post or wasn't assigned to annotate it. Handling such cases during training is important, as missing annotations can affect model performance.

4.5 Exploratory Data Analysis (EDA)

In this section, we perform an exploratory data analysis (EDA) on the dataset to understand its characteristics and identify any potential patterns or issues before training the models.

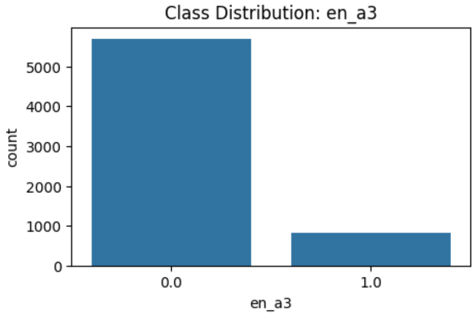

Figure 1: Distribution of labels in the dataset.

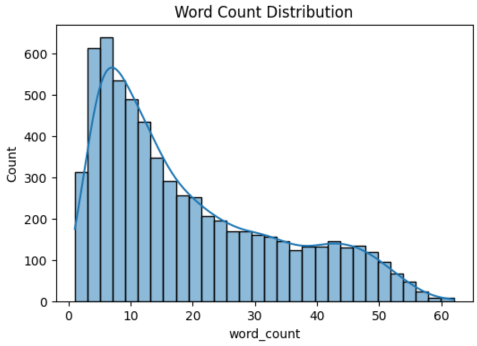

Figure 2: Word cloud representation of common terms in the dataset.

5 Results

Language	F1 Score / Accuracy
English (EN)	0.662 / 0.764
Hindi (HI)	0.708 / 0.798
Tamil (TA)	**0.813 / 0.815**

Table 1: Final Task 1 Evaluation Results (Baseline 1)

6 Observations

- **Task 1: Hate Speech Detection**

Language	**F1 Score / Accuracy**
English (EN)	0.665 / 0.766
Hindi (HI)	0.671 / 0.771
Tamil (TA)	**0.755 / 0.754**

Table 2: Final Task 1 Evaluation Results (Baseline 2)

Language	**F1 Score / Accuracy**
English (EN)	0.662 / 0.764
Hindi (HI)	0.708 / 0.798
Tamil (TA)	0.494/0.593

Table 3: Final Task 1 Evaluation Results (Final)

Language	**F1 Score / Accuracy**
English (EN)	0.48 / 0.53
Hindi (HI)	0.76 / 0.77
Tamil (TA)	**1/ 1**

Table 4: Final Task 2 Evaluation Results (Baseline 1)

Language	**F1 Score / Accuracy**
English (EN)	0.665 / 0.766
Hindi (HI)	0.56 / 0.69
Tamil (TA)	**1 / 1**

Table 5: Final Task 2 Evaluation Results (Baseline 2)

Language	**F1 Score / Accuracy**
English (EN)	**0.84 / 0.96**
Hindi (HI)	0.33 / 0.73
Tamil (TA)	0.41 / 0.83

Table 6: Final Task 2 Evaluation Results (Final)

Language	**Gendered Abuse (F1 / Accuracy)**
English (EN)	0.614 / 0.728
Hindi (HI)	0.70 / 0.79
Tamil (TA)	**0.64 / 0.66**
Language	**Explicit Language (F1 / Accuracy)**
English (EN)	0.199 / 0.370
Hindi (HI)	0.45 / 0.60
Tamil (TA)	**0.835 / 0.840**

Table 7: Baseline 1 Task 3 Evaluation Results across all three languages. Tamil performs best in both subtasks.

  - Baseline 1 achieved the highest performance in Tamil (F1: 0.813, Acc: 0.815), indicating strong lexical cues in Tamil data.
  - Baseline 2 showed reduced performance across all languages, although Tamil still led.
  - Final results showed a significant drop in Tamil performance (F1: 0.494), while Hindi maintained consistent accuracy (0.798), possibly due to data imbalance or preprocessing issues.

Language	**Gendered Abuse (F1 / Accuracy)**
English (EN)	0.536
Hindi (HI)	0.07
Tamil (TA)	0.56
Language	**Explicit Language (F1 / Accuracy)**
English (EN)	0.42
Hindi (HI)	0.21
Tamil (TA)	**0.61**

Table 8: Baseline 2 Task 3 Evaluation Results across all three languages. Tamil performs best in both subtasks.

Language	**Gendered Abuse (F1 / Accuracy)**
English (EN)	0.61 / 0.72
Hindi (HI)	**0.705 / 0.794**
Tamil (TA)	0.72 / 0.73
Language	**Explicit Language (F1 / Accuracy)**
English (EN)	0.42 / 0.57
Hindi (HI)	0.5 / 0.637
Tamil (TA)	**0.83 / 0.84**

Table 9: Final Task 3 Evaluation Results across all three languages. Tamil performs best in both subtasks.

- **Task 2: Multilingual Hate Speech Classification**
  - Baseline 1 and 2 showed perfect Tamil performance (F1/Acc = 1.0), suggesting either dataset simplicity, leakage, or overfitting.
  - In the final model, English outperformed both Hindi and Tamil (F1: 0.84, Acc: 0.96), highlighting the robustness of the English model (distilroberta-base).
  - Hindi and Tamil saw performance drops, suggesting potential issues in generalization or multilingual domain adaptation.

- **Task 3: Subtask Classification (Gendered Abuse and Explicit Language)**
  - For Gendered Abuse, final results showed Hindi performing best (F1: 0.705), followed by Tamil (F1: 0.72), with English slightly lower (F1: 0.61).

  - For Explicit Language, Tamil consistently outperformed the other languages across all baselines and final models, achieving F1: 0.83 in the final setup.
  - English showed the weakest performance in detecting explicit content (Final F1: 0.42), possibly due to higher linguistic ambiguity or less explicit labeling.

- **Overall Trends**
  - Tamil data appears to have clearer patterns for explicit language, contributing to consistently high scores in Task 3.
  - English benefits most from high-resource models like distilroberta-base, especially in Task 2.
  - Hindi performance varied widely across tasks and baselines, indicating possible inconsistencies in data quality or model generalization.
  - Performance fluctuations between baselines and final models highlight the importance of consistent preprocessing, training stability, and balanced evaluation sets.

7 Conclusion and Future Work

7.1 Conclusion

The rise of online gender-based violence poses a significant challenge, reinforcing social inequalities and driving marginalized groups away from digital spaces. This study addressed the critical need for automated detection systems by developing and evaluating models for identifying gendered abuse and explicit content in English, Hindi, and Tamil.

Our experiments demonstrated that:

- **Transformer-based models** (e.g., DistilRoBERTa, XLM-RoBERTa, DeBERTa-v3) effectively detect abusive content, with Tamil performing particularly well in explicit language classification (F1: 0.83).

- **Transfer learning** improved performance, especially for English, where high-resource models like DistilRoBERTa achieved strong results (F1: 0.84 in Task 2).

- **Multitask learning** proved viable for joint classification, though performance varied across languages, suggesting the need for better language-specific adaptation.

- **Data quality and annotation consistency** significantly impacted model reliability, highlighting the importance of balanced datasets and robust preprocessing.

7.2 Future Work

To further enhance detection systems, future research should explore:

1. **Improved Multilingual Adaptation** – Fine-tuning models with language-specific embeddings or adapter layers to better capture linguistic nuances in Hindi and Tamil.

2. **Context-Aware Models** – Incorporating user history, conversational context, and sociolinguistic cues to reduce false positives/negatives.

3. **Bias Mitigation** – Addressing annotation biases, particularly in gendered abuse labels, to ensure fairer model performance.

4. **Real-Time Deployment** – Optimizing models for low-latency applications, enabling faster moderation in social media platforms.

5. **Expanded Language Coverage** – Extending the framework to more Indian languages (e.g., Bengali, Marathi) to broaden impact.

By advancing these directions, we can build more equitable and effective tools to combat online abuse, fostering safer digital spaces for all users.

Check Point Link: https://drive.google.com/drive/u/0/folders/1FN5DKRgAqgtaRFbIfjdJPPFFd3EPESCY, https://drive.google.com/file/d/1DmYwExfmmeIxAeEm8h0CGBS5Ih2GTbJ-/view?usp=sharing.

References

1. Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, NC Gokul, Avik Bhattacharyya, Mitesh M Khapra, and Pratyush Kumar. 2020. *Indicnlpsuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages*. In Findings of the Association for Computational Linguistics: EMNLP 2020, pages 4948–4961. https://aclanthology.org/2020.findings-emnlp.445

2. Shana Poplack and James A. Walker. 2003. *Pieter Muysken, Bilingual Speech: A Typology of Code-Mixing*. Journal of Linguistics, 39(3):678–683. https://doi.org/10.1017/S0022226703252299

3. Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. O'Reilly Media, Inc. https://www.nltk.org/book/

4. Arnav Arora, Rashmi G, Devika R, Saket D, Apurv Verma, Raghav Juyal, and Anubha Agarwal. 2023. *The Uli Dataset: An Exercise in Experience Led Annotation of oGBV*. In Proceedings of the International AAAI Conference on Web and Social Media (ICWSM). https://ojs.aaai.org/index.php/ICWSM/article/view/14955

5. PyTorch Documentation. *Sequence Models and Long Short-Term Memory Networks*. https://pytorch.org/tutorials/beginner/nlp/sequence_models_tutorial.html

6. Advaita Vetagiri. 2023. *CNLP-NITS-PP GitHub Repository*. https://github.com/advaithavetagiri/CNLP-NITS-PP

7. Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. arXiv preprint arXiv:1712.06427. https://arxiv.org/abs/1712.06427

8. Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. *Automated Hate Speech Detection and the Problem of Offensive Language*. Proceedings of the International AAAI Conference on Web and Social Media, 11(1), 512–515. https://doi.org/10.1609/icwsm.v11i1.14955

9. Shervin Malmasi and Marcos Zampieri. 2017. *Detecting Hate Speech in Social Media*. arXiv preprint arXiv:1712.06427. https://arxiv.org/abs/1712.06427