wangkongqiang@CASE 2025: Detection and Classifying Language and Targets of Hate Speech using Auxiliary Text Supervised Learning

Kongqiang Wang
School of Information Science,
Yunnan University,
Yunnan Baiyao Street, Kunming,
650500, Yunnan, China.
wangkongqiang@stu.ynu.edu.cn

Peng Zhang
School of Information Science,
Yunnan University,
Yunnan Baiyao Street, Kunming,
650500, Yunnan, China.
zpp1219@gmail.com

Abstract

Our team was interested in content classification and labeling from multimodal detection of Hate speech, Humor, and Stance in marginalized socio-political movement discourse. We joined the task: Subtask A-Detection of Hate Speech and Subtask B-Classifying the Targets of Hate Speech. In this two task, our goal is to assign a content classification label to multimodal Hate Speech. Detection of Hate Speech: The aim is to detect the presence of hate speech in the images. The dataset for this task will have binary labels: No Hate and Hate. Classifying the Targets of Hate Speech: Given that an image is hateful, the goal here is to identify the targets of hate speech. The dataset here will have four labels: Undirected, Individual, Community, and Organization. Our group used a supervised learning method and a text prediction model. The best result on the test set for Subtask-A and Subtask-B were F1 score of 0.6209 and 0.3453, ranking twentieth and thirteenth among all teams.

1 Introduction

First of all, let's introduce the Overview of Shared Task on Multimodal Hate, Humor, and Stance Detection in Marginalized Movement@CASE2025 (Hürriyetoğlu et al., 2025). The complexity of text-embedded images presents a formidable challenge in ML given the need for multimodal understanding of multiple aspects of expression conveyed by them. Particularly, the marginalized movement stands as a prominent subject of online discourse, where text-embedded images like memes serve as vehicles of both solidarity and resistance, reflecting the multifaceted dynamics of attitudes and perceptions within the community and beyond. In this context, the distinction between humor and harm becomes blurred, as memes straddle the line between satire and offense, challenging researchers and platforms alike to navigate the complexities of online content moderation. As one label generally fails to encompass multiple aspects of linguistics, this shared task classifies images on four aspects: hate, targets of hate, stance, and humor as subtasks.

Our group mainly participated in the following two sub-tasks. Subtask A-Detection of Hate Speech: The aim is to detect the presence of hate speech in the images. The dataset for this task will have binary labels: No Hate and Hate. Subtask B-Classifying the Targets of Hate Speech: Given that an image is hateful, the goal here is to identify the targets of hate speech. The dataset here will have four labels: Undirected, Individual, Community, and Organization. We have made tremendous progress in these two sub-tasks.

2 Dataset

In this section, we describe various aspects of task dataset including data collection, annotation guidelines, and dataset statistics. Task dataset comprises 5,063 text-embedded images that encompass memes, posters, and infographics relevant to the LGBTQ+ movement. Official dataset only include images from 2020-2024 as this period saw an upsurge of social media content in this domain (Oz et al., 2023). This also allows task dataset to represent contemporary social media interactions through memes. Note that by the term LGBTQ+, official dataset refer to all gender identities and sexual orientations inclusively.

2.1 Data Collection

To maintain diversity in the dataset, organizer collected data from three popular social media platforms: Facebook, Twitter, and Reddit, through manual search and extraction. For Twitter, organizer used hash tags such as #lgbt, #pride, #trans, #transrights, #nonbinary, and #genderidentity to filter images related to LGBTQ+ discussions. For Facebook, organizer targeted groups that frequently

Proceedings of the 8th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Texts associated with RANLP 2025, pages 62–70, Varna, Bulgaria, Sep 13, 2025.
https://doi.org/10.26615/978-954-452-099-1-007

discussed LGBTQ+ content. Similarly, for Reddit, organizer identified subreddits where discussion related to LGBTQ+ was more prominent. Further, to ensure the relevance and quality of the dataset, the data collection process was subject to filtering criteria. Detailed filtering criteria for our dataset can be found in (Shah et al., 2024). As different annotators may encounter and collect the same image, organizer sequentially employed two image deduplication tools: dupeGuru[1] and difPy[2], to search for duplicates and retain the highest quality image out of each batch of duplicates. Organizer used the OCR application provided by Google Cloud Vision API[3] to extract textual data from the images. Organizer removed non-alphanumeric elements such as special characters, hyperlinks, symbols, and non-English characters to reduce noisy text data and ensure data quality. Note that the text may occasionally contain unintentional noisy artifacts.

3 Data Annotation

Organizer engaged five experienced annotators, well-versed in NLP and computational linguistics, to annotate data samples for PrideMM. The annotators had a prior understanding of the LGBTQ+ movement and meme archetypes on social media. Organizer presented them with comprehensive annotation guidelines to ensure uniform and unbiased annotations, and asked them to annotate each image separately for all four tasks. A 3-phase annotation schema was used to ensure accurate and consistent annotations. First, a dry run was conducted to evaluate the understanding of the annotation guidelines among the annotators where every annotator was given an identical batch of 50 images for annotation. Second, a revision phase was conducted where every annotator was given another identical batch of 200 images and received a revised set of instructions based on the results of the first phase. Finally, in the consolidation phase, the annotators annotated a final batch of 50 images while discussing and revising the annotation guidelines until a consensus was reached. These steps were taken to minimize misannotations and noisy labels in the PrideMM dataset. The meticulously devised annotation guidelines were followed to ensure consistency in the annotations. Each image in their dataset was independently annotated for the three aspects and one sub-class, apart from the connection between 'Hate' and 'Hate Targets'.

[1]https://github.com/arsenetar/dupeguru

[2]https://github.com/elisemercury/Duplicate-Image -Finder

[3]https://cloud.google.com/vision/docs

4 Annotation Guidelines

In this section, organizer describe the annotation guidelines used to annotate the dataset. They devise separate guidelines for each of the four tasks.

Hate Speech. This task aimed to identify instances of hate speech in the images. The primary focus was on identifying images that intentionally conveyed hateful sentiments. Annotators needed to distinguish between images expressing strong disagreement without resorting to offensive language and those containing genuine elements of hate speech. This differentiation aimed to guarantee accurate labeling, ensuring that images conveying genuinely hateful sentiment through visual content, language, or a combination of both were appropriately identified.

Hate Targets. This task required annotators to identify the targets of hate in hateful images by classifying the images into one of the four classes: Undirected, Individual, Community, and Organization. Images were labeled as Undirected when they targeted abstract topics, societal themes, or ambiguous targets like 'you' that were not directed toward any specific individuals, entities, or groups. Hateful images targeting specific people including political leaders, celebrities, or activists like 'Joe Biden' and 'J.K. Rowling' were annotated as Individual. Likewise, the label Community was used for instances of images targeting broader social, ethnic, or cultural groups like 'LGBT' or 'trans'. Lastly, images targeting corporate entities, institutions, or similar organizations like 'Chick-fil-A' and 'government" were annotated as Organization.

Stance. This task involved annotating the images into either of three distinct categories: Support, Oppose, and Neutral, determined by their stance within the context of the LGBTQ+ movement. The Support label was given to images that expressed support towards the goals of the movement, agreed with efforts in fostering equal rights for LGBTQ+ individuals, and promoted awareness for the movement's goals. The Oppose label was given to images that conveyed disagreement with the goals of the movement, denied the problems faced by individuals who identified as LGBTQ+, and dismissed the need for equal rights and acceptance. The Neutral label was given to images that were contextually relevant to the movement but

did not exhibit support or opposition towards the movement.

Humor. In this task, annotators were asked to identify images showcasing humor, sarcasm, or satire related to the LGBTQ+ Pride movement. Annotators were instructed to discern the presence of humor in the images regardless of whether they presented a lighthearted or insensitive perspective on serious subjects. Note that annotators were asked to annotate images based on whether the creator of the image intended for it to be humorous, and not based on whether the annotator personally found it humorous. This task aimed to capture the nuanced use of text-embedded images for comedic or satirical purposes, thereby helping disentangle hate and humor in the images related to this movement.

5 Statistics and Inter-Annotator Agreement

Table 1: Dataset Statistics for PrideMM. The data consists of 5,063 samples for Hate, Stance, and Humor tasks, and 4,482 samples for the Target classification task.

Task	Label	#Samples	%
Hate	No Hate	2,581	50.97%
	Hate	2,482	49.03%
Target	Undirected	771	31.07%
	Individual	249	10.03%
	Community	1,164	46.90%
	Organization	298	12.00%
Stance	Neutral	1,458	28.80%
	Support	1,909	37.70%
	Oppose	1,696	33.50%
Humor	No Humor	1,642	32.43%
	Humor	3,421	67.57%

Table 1 shows the distribution of images in PrideMM across all class labels. For the hate detection task, the dataset has a balanced distribution of binary labels. The target classification task exhibits a heavily imbalanced distribution. Given the context of this study, most hateful images convey undirected hate or are targeted toward communities, with a low frequency of hate against individuals and organizations. For the stance classification task, the number of images is well-balanced across three labels. On the other hand, as memes are often meant to be humorous, the majority of the images in the dataset are annotated to humor.

Organizer used the Fleiss' Kappa (κ) (Falotico and Quatto, 2015) as a statistical measure to assess the inter-annotator agreement across all four tasks. For Task A (Hate Speech detection), κ was 0.66/0.74 in the dry run and final phase respectively, for Task B(Target detection), κ was 0.68/0.81, for Task C (Stance detection), κ was 0.62/0.75, and for Task D (Humor detection), κ was 0.60/0.74. The increase in κ from the dry run phase to the final phase across all tasks reflects the effectiveness of the 3-phase annotation schema.

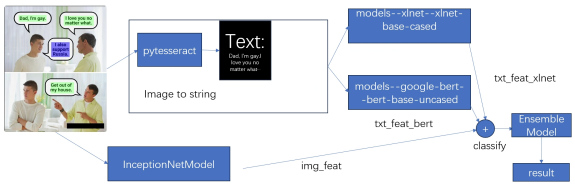

Figure 1: The framework diagram of the Ensemble model.

6 Methodology

6.1 Ensemble Model

MultiModal Hate Speech Detection In our task, there are two modalities of image and text. We use the pytesseract tool to extract the text in the image and obtain the features of the text through the xlnet and bert models. The corresponding image features of the pictures are obtained through the inception net model, and then the classification results are finally obtained through the classifier of the ensemble model. The framework diagram of the model is shown in the Figure 1. The classifier in the Ensemble model uses a neural network classifier. The specific parameters of the classifier are shown in the Table 2.

Table 2: Neural Network classifier structure for Subtask A.

layer number	layer type	Input dimension	Output dimension / parameter
1	Linear	image_features + xlnet_hidden + bert_hidden	512
2	ReLU	512	512
3	BatchNorm1d	512	512
4	Dropout	512	512 (p=0.3)
5	Linear	512	256
6	ReLU	256	256
7	BatchNorm1d	256	256
8	Dropout	256	256 (p=0.3)
9	Linear	256	num_classes = 2

MultiModal Hate Targets Classification The method of MultiModal Hate Targets Classification is the same as that of MultiModal Hate Speech Detection, except that we use two different classifiers for experiments. It is found that using the Linear classifier alone is slightly better than using

the Neural Network classifier. Their structures are respectively shown in Table 3 and Table 4.

Table 3: Neural Network classifier structure for Subtask B.

layer number	layer type	Input dimension	Output dimension/parameter
1	Linear	image_features + xlnet_hidden + bert_hidden	512
2	ReLU	512	512
3	BatchNorm1d	512	512
4	Dropout	512	512 (p=0.3)
5	Linear	512	256
6	ReLU	256	256
7	BatchNorm1d	256	256
8	Dropout	256	256 (p=0.3)
9	Linear	256	num_classes = 4

Table 4: Linear classifier structure.

layer number	layer type	Input dimension	Output dimension/parameter
1	Linear	image_features + xlnet_hidden + bert_hidden	num_classes = 4

6.2 K-max pooling neural network with recurrent learning rate (CLR) scheduling

In the Ensemble Model, the addition of image features leads to poor classification effect of the model. Relying solely on the text extracted from the images may improve the performance of the model. So in this model, we first use the Google Vision API to extract text from images and classify the text content. Facts have also proved the usefulness of our conjecture.

6.2.1 Problem Setup

We address the binary and multi-task classification problem on textual content, where each sample may be annotated with hierarchical labels: a binary label for Task 1 (e.g., Hate vs. Not-Hate) and a fine-grained four-class label for Task 2 (e.g., Undirected , Individual , Community , Organization). The dataset is preprocessed and split into training and test sets accordingly, using a k-fold cross-validation scheme to improve generalizability and model robustness.

6.2.2 Preprocessing and Tokenization

All text inputs are tokenized using nltk.RegexpTokenizer, preserving only word characters. A Keras Tokenizer is then applied to convert the text into sequences of word indices. These sequences are padded to a maximum sequence length based on the longest sample in the training set.

6.2.3 Embedding Layer Construction

We utilize pre-trained GloVe embeddings (840B.300d) to initialize the word embedding matrix. Each word in the vocabulary is mapped to a 300-dimensional dense vector. If a word is missing from the GloVe vocabulary, it is assigned a vector initialized from a normal distribution with the same mean and standard deviation as the pre-trained embeddings.

6.2.4 Model Architecture

The core of the proposed model is based on a K-max pooling neural network architecture enhanced by embedding initialization and dense transformations:

- Input Layer: Tokenized and padded sequences.
- Embedding Layer: Initialized with pre-trained GloVe vectors, this layer is frozen (non-trainable) during training.
- K-Max Pooling Layer: Extracts the top-k activations across the sequence dimension, effectively capturing the most informative word features regardless of position.
- Dense Transformation: The pooled features are passed through a fully connected layer with tanh activation, followed by a softmax classification head.

This simple yet effective architecture is chosen to reduce overfitting and training time while maintaining competitive performance.

6.2.5 Learning Rate Scheduling

To improve convergence, we apply Cyclic Learning Rate (CLR) scheduling, as proposed by (Smith, 2017). Specifically, we use the exp_range policy to periodically vary the learning rate between 0.001 and 0.006 using a base-2 exponential decay factor (gamma=0.99994). This prevents premature convergence and encourages the model to escape local minima during training.

6.2.6 Training Strategy

The model is trained using the Adam optimizer with default hyperparameters ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 1e-8$). The loss function used is categorical_crossentropy, and additional evaluation metrics include accuracy and a custom-defined F1-score metric implemented using Keras backend operations.

To ensure reliable evaluation, we perform 4-fold cross-validation. In each fold:

- The model is trained on k-1 folds and validated on the remaining fold.

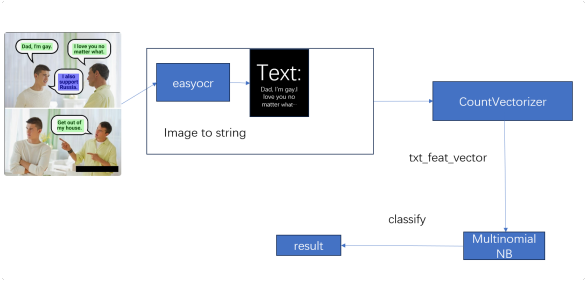

Figure 2: The framework diagram of the Multinomial Naive Bayes classification model.

- Predictions on the test set are collected and averaged across folds for final ensemble predictions.

6.2.7 Prediction and Submission

After training, the ensemble of models predicts the labels for both tasks. The final predictions are computed by averaging softmax outputs across all folds and selecting the label with the highest averaged probability. Results are saved into submission files corresponding to each task.

6.2.8 Evaluation

The model performance is evaluated using macro-averaged precision, recall, and F1-score for both tasks. This metric choice reflects the need to account for class imbalance in the multi-class setting.

6.2.9 Summary

Overall, our model integrates pre-trained embeddings, efficient pooling mechanisms, cyclic learning rates, and ensemble training via cross-validation to deliver a robust solution for binary and multi-task text classification.

6.3 Multinomial Naive Bayes classification model applying easyocr text extraction

We use the English (en) version in the easyocr application for text extraction, and then perform text vectorization through the CountVectorizer in the sklearn.feature_extraction.text library. Finally, the Multinomial Naive Bayes classification model is used for classification. The overall architecture diagram of the model is shown in the Figure 2.

6.3.1 Introduction to EasyOCR

EasyOCR is an open-source, deep learning-based Optical Character Recognition (OCR) tool developed by Jaided AI, supporting text recognition in over 80 languages. This library is implemented based on PyTorch, adopts multi-layer convolutional neural network (CNN) and sequence modeling (such as LSTM) structures, and combines CTC (Connectionist Temporal Classification) loss for the training and recognition of unaligned text sequences.

The main features of EasyOCR are as follows: Multilingual support. Built-in support for multiple languages including Chinese, English, Japanese, Korean, etc., suitable for multilingual text scenarios; Strong end-to-end recognition capability. It can automatically detect text areas from the original image and recognize their contents; No complex preprocessing required. Supports complex backgrounds, slanted text and multi-font recognition; Simple and easy to use. The API interface is friendly and suitable for quick integration into applications; Strong scalability. Users can customize training data and fine-tune the model to adapt to character sets or styles in specific domains.

Brief description of the workflow:

- Text Detection: The CRAFT (Character Region Awareness for Text Detection) model is adopted to locate the text regions in the image;
- Text Recognition: Use deep neural networks (CNN + LSTM + CTC) to conduct sequence modeling and character recognition for the extracted text regions;
- Language post-processing (optional) : Language-level correction in combination with dictionaries or rules.

6.3.2 Principle Explanation of CountVectorizer

CountVectorizer is one of the most fundamental and commonly used methods in text feature extraction, used to convert text data into vector format, facilitating subsequent processing by machine learning models. Its basic idea is: to count the number of occurrences of each word (or n-gram) in the text and take these count values as the elements of the feature vector.

Fundamental: The main processing flow of CountVectorizer is as follows:

- Tokenization. Segment the text to divide sentences or documents into individual words (tokens).
- Vocabulary Building. Based on all the input text, count all the words that have appeared

(or the specified n-gram), and assign a unique index to each word.

- Word Frequency Statistics (Vectorization). For each input text, count the number of times each word appears in the vocabulary to form a sparse vector.

Suppose the vocabulary is ['apple ', 'banana', 'orange'] and the text is 'apple orange orange', then the conversion result is [1, 0, 2].

Mathematical Treatment: Given a example: Vocabulary size V; Input text dataset $D = \{d_1, d_2, \ldots, d_n\}$; Each text d_i is converted to a vector $\vec{x_i} \in R^V$ Then:

$$x_{ij} = \text{count}(\text{term}_j \text{ in } d_i) \tag{1}$$

where x_{ij} represents the occurrence frequency of the j-th term in the vocabulary within the i-th document.

Configurable Parameters: CountVectorizer provides several key parameters that affect the vectorization:

- `ngram_range=(1,1)`: Specifies the use of 1-gram (single words), 2-gram (word pairs), etc.
- `stop_words='english'`: Removes English stop words
- `max_features=1000`: Retains only the top 1000 most frequent terms
- `min_df` / `max_df`: Filters terms that appear too rarely or too frequently
- `binary=True`: Does not count frequencies, only marks whether terms appear or not

Summary: CountVectorizer is a simple and efficient feature extraction method for text frequency; It produces a one-hot sparse matrix, suitable for text classification and retrieval modeling; It does not consider word order or context meaning, but performs well in many tasks that depend on word frequency; For more complex language modeling, it is often combined with TF-IDF, word vectors (Word2Vec), or Transformer models.

6.3.3 Explanation of the principle of Multinomial Naive Bayes

Basic Conception: Multinomial Naive Bayes (Polynomial Naive Bayes) is a type of Naive Bayes classifier, mainly used for classification problems of discrete features, especially suitable for text classification, such as spam recognition, sentiment analysis, news classification, etc. It is based on Bayes' theorem and assumes that features are conditionally independent of each other, that is, the source of "naive".

Core Formula: Given a document d, we want to find the most likely class $y \in \{c_1, c_2, \ldots, c_k\}$. Using Bayes' theorem:

$$P(y|d) = \frac{P(d|y) \cdot P(y)}{P(d)} \tag{2}$$

Since $P(d)$ is the same for all classes, we only need to maximize the numerator:

$$\hat{y} = \arg\max_y P(y) \cdot P(d|y) \tag{3}$$

In the multinomial model, the document is represented as a word frequency vector $\vec{x} = (x_1, x_2, \ldots, x_n)$, where x_i is the frequency of word w_i in the document. Assuming word independence, the likelihood of document under class y is:

$$P(d|y) = \prod_{i=1}^{n} P(w_i|y)^{x_i} \tag{4}$$

The final classification formula:

$$\hat{y} = \arg\max_y \log P(y) + \sum_{i=1}^{n} x_i \cdot \log P(w_i|y) \tag{5}$$

Parameter Estimation: Prior probability $P(y)$: The occurrence ratio of class in training data.

$$P(y = c) = \frac{\text{count}(y = c)}{\text{total samples}} \tag{6}$$

Conditional probability $P(w_i|y)$: The relative frequency of word w_i in class y. To avoid zero probabilities, Laplace smoothing is applied.

$$P(w_i|y) = \frac{\text{count}(w_i \text{ in } y) + \alpha}{\sum_j \text{count}(w_j \text{ in } y) + \alpha \cdot V} \tag{7}$$

α: Smoothing parameter, usually 1 (Laplace) or a small value (Lidstone). V: Vocabulary size.

Advantages and Applications: Advantages: Simple model with high computational efficiency; Performs well on high-dimensional sparse data (e.g., text); Fast training speed, no need for gradient descent; Not prone to overfitting, good generalization. Application scenarios: Text classification (news, reviews, spam filtering); Discrete counting data (click-through rates, purchase data). Compared with other Naive Bayes variants, see Table 5.

Table 5: Naive Bayes Model Variants and Applications

Model	Feature Type	Application Domain
BernoulliNB	Binary features	Text word presence / absence
MultinomialNB	Count features	Text word frequency
GaussianNB	Continuous features	Images, sensor data, etc.

Conclusion: MultinomialNB is one of the most classic models in text classification; It classifies by counting word frequencies in text; Simple and efficient, making it a good baseline model for text classification; Performs well when combined with `CountVectorizer` or `TfidfVectorizer`.

7 Experimental Results

According to the official instructions (Thapa et al., 2025a) for OCR extraction: If participants want to extract OCR, they can use Google Vision API, tesseract, EasyOCR, etc. In the paper that benchmarks this dataset, organizer have used Google Vision API to extract OCR for training the models. Since a lot of participants may not have access to the vision API, they can use the extracted text from organizer's benchmark paper (Bhandari et al., 2023). So when we used the Google vision API to extract the text version database, we directly used the dataset indicated by the official. For the other two methods, namely the pytesseract and EasyOCR methods, we manually extracted them through our own python script code. The complete code of this entire project can be found at our GitHub address[4].

For Subtask A-Detection of Hate Speech and Subtask B-Classifying the Targets of Hate Speech, the results obtained by our three and four methods on the test set are shown in Table 6 and Table 7.

Table 6: The results obtained by our three methods for Subtask A-Detection of Hate Speech on the test set.

Model	Recall	Precision	F1	Accuracy
Ensemble Model (Neural Network classifier)	0.4928	0.4733	0.3761	0.4852
K-max Pooling Neural Network	0.5925	0.6389	0.5585	0.5976
Multinomial Naive Bayes Classification Model	0.6365	0.6591	0.6209	0.6331

[4]https://github.com/WangKongQiang/Case2025

Table 7: The results obtained by our four methods for Subtask B-Classifying the Targets of Hate Speech on the test set.

Model	Recall	Precision	F1	Accuracy
Ensemble Model (Neural Network classifier)	0.2500	0.1175	0.1598	0.4699
Ensemble Model (Linear classifier)	0.2525	0.2503	0.2477	0.3695
K-max Pooling Neural Network	0.2544	0.2846	0.1723	0.4739
Multinomial Naive Bayes Classification Model	0.3322	0.5552	0.3453	0.4779

8 Discussion

For Multimodal Hate, Humor and Stance Detection in Marginalized Movement@CASE2025 sharing task, we referred to the relevant tasks of CASE 2024 (Thapa et al., 2024) and CASE 2023 (Thapa et al., 2023) shared tasks on multimodal hate speech detection and derived our own method. Although the effect of the experiment needs to be strengthened. However, these contents and ideas have given us a lot of inspiration. Multimodal content analysis is a longstanding tradition of the CASE workshop series. We believe that with our further research and more detailed optimization and training of the model, we will achieve even greater success in future competitions.

9 Conclusion

We employed multiple methods in Subtask A-Detection of Hate Speech and Subtask B-Classifying the Targets of Hate Speech, which respectively involved the transformer model, deep learning models and machine learning models in these two tasks. Our final leaderboards are respectively shown in the Table 8 and in the Table 9.

Table 8: The Final Leaderboard of Subtask A: Detection of Hate Speech.

#	User	Team Name	Recall	Precision	F1	Accuracy
1	wangxiuxian	TUJ-MI	0.8422 (1)	0.8422 (1)	0.8422 (1)	0.8422 (1)
2	Ryuan		0.8288 (2)	0.8291 (2)	0.8284 (2)	0.8284 (2)
3	jiarranDiana	IMU-L	0.8211 (3)	0.8217 (3)	0.8205 (3)	0.8205 (3)
4	ray-sushant	Phantom Troupe	0.8189 (4)	0.8193 (4)	0.8191 (4)	0.8190 (4)
5	Neuron-Force	MemeMasters	0.8086 (5)	0.8086 (5)	0.8086 (5)	0.8086 (5)
6	shrutigurung	Multimodal Kathmandu	0.8004 (6)	0.8028 (6)	0.8005 (6)	0.8008 (6)
7	Sujal_Maharjan		0.7932 (7)	0.7928 (7)	0.7932 (7)	0.7929 (7)
8	NextTry		0.7927 (8)	0.7928 (8)	0.7927 (8)	0.7929 (8)
9	ankitbk07		0.7927 (9)	0.7930 (9)	0.7927 (9)	0.7929 (9)
10	Rashfi		0.7868 (10)	0.7870 (10)	0.7868 (10)	0.7869 (10)
11	rohanmainali	Silver	0.7847 (11)	0.7833 (11)	0.7847 (11)	0.7830 (11)
12	prerana3		0.7799 (12)	0.7812 (12)	0.7789 (12)	0.7810 (12)
13	TomalJoy		0.7417 (13)	0.7416 (13)	0.7417 (13)	0.7416 (13)
14	Tanvir_77		0.7405 (14)	0.7414 (14)	0.7406 (14)	0.7411 (15)
15	bidhan_*b*		0.7380 (15)	0.7382 (15)	0.7377 (15)	0.7377 (16)
16	AkshYat		0.7360 (16)	0.7360 (16)	0.7360 (16)	0.7360 (14)
17	akshayyy22		0.7225 (17)	0.7234 (17)	0.7217 (17)	0.7219 (17)
18	ysb	YS	0.6926 (18)	0.6927 (18)	0.6923 (18)	0.6923 (18)
19	Durgeshverma24iitrm	MLP	0.6636 (19)	0.6644 (19)	0.6632 (19)	0.6636 (19)
20	**wangkongqiang**	**wang**	**0.6365 (20)**	**0.6359 (20)**	**0.6360 (20)**	**0.6331 (17)**
21	MDSagorChowdhury	Musafir	0.6179 (21)	0.6862 (19)	0.5828 (21)	0.6233 (18)

10 Limitations of the Work

we are interested in learning about LLMs in computational social science (Thapa et al., 2025b), our paper mainly focuses on making discussions on

Table 9: The Final Leaderboard of Subtask B: Classifying the Targets of Hate Speech.

#	User	Team Name	Recall	Precision	F1	Accuracy
1	wangxiuxian	TUJ-MI	0.6383 (1)	0.6759 (1)	0.6530 (1)	0.6426 (1)
2	Ryuan		0.6204 (2)	0.6556 (2)	0.6335 (2)	0.6426 (1)
3	ray-sushant		0.6021 (5)	0.6169 (4)	0.6057 (3)	0.6305 (2)
4	jiarranDiana	IMU-L	0.6038 (3)	0.6230 (3)	0.6015 (4)	0.6305 (3)
5	Sujal_Maharjan	Multimodal Kathmandu	0.5922 (6)	0.5666 (5)	0.5777 (5)	0.5823 (4)
6	bidhan_cb		0.6032 (4)	0.5407 (10)	0.5628 (6)	0.5703 (5)
7	prerana3		0.5504 (7)	0.5653 (7)	0.5539 (7)	0.5904 (3)
8	ankitbk07		0.5249 (9)	0.6044 (6)	0.5486 (8)	0.5823 (6)
9	shrutigurung	MemeMasters	0.5059 (10)	0.5427 (9)	0.5150 (9)	0.5382 (7)
10	rohanmainali	Silver	0.5422 (8)	0.5092 (12)	0.5018 (10)	0.5181 (8)
11	akshayyy22		0.4869 (11)	0.5289 (11)	0.4984 (11)	0.5151 (10)
12	MDSagorChowdhury	Musafir	0.4143 (12)	0.4008 (13)	0.3739 (12)	0.4418 (10)
13	**wangkongqiang**	**wang**	**0.3322 (13)**	**0.5552 (8)**	**0.3405 (13)**	**0.4779 (9)**
14	Durgeshverma24iitrm	MLP	0.2757 (14)	0.3158 (14)	0.2739 (14)	0.4096 (11)

hate speech for this task. This is because we are quite interested in and good at identifying hate and offense categories in the text (Parihar et al., 2021). Due to our lack of utilization of image features, we are unable to make good use of the image content in the dataset of this sharing task. Also, the model used for extracting text features is similar to that in the sklearn package of CountVectorizer, but it cannot extract the content from the text very well. We believe that by combining a better image feature model, more refined text feature extraction, and conducting appropriate text preprocessing, our model can achieve better results. These are all our future tasks.

11 Ethical Considerations

Our work focuses on hate speech detection and target classification within LGBTQ+ related multimodal content, a domain that is inherently sensitive and requires heightened ethical awareness throughout all research stages. We address the following key ethical concerns:

We used public OCR tools (Google Vision API, pytesseract, EasyOCR) and open-source libraries (e.g., scikit-learn, TensorFlow) to extract and analyze text. We disclose our models, source code, and hyperparameters openly at our GitHub repository, promoting transparency and reproducibility. However, we caution that open-source release of models detecting sensitive content must be accompanied by ethical usage disclaimers and limitations.

We aim to improve our methods by integrating more robust and interpretable models, minimizing biases, and involving domain experts—especially from affected communities—in future annotation and evaluation processes. Ethical AI practices will remain a guiding principle in our ongoing research.

Acknowledgments

We are very grateful to the organizers of the Shared Task on Multimodal Hate, Humor,and Stance Detection in Marginalized Movement@CASE2025 and the School of Information of Yunnan University for providing the environment and equipment.

References

Aashish Bhandari, Siddhant B Shah, Surendrabikram Thapa, Usman Naseem, and Mehwish Nasim. 2023. Crisishatemm: Multimodal analysis of directed and undirected hate speech in text-embedded images from russia-ukraine conflict. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1994–2003.

Rosa Falotico and Piero Quatto. 2015. Fleiss' kappa statistic without paradoxes. *Quality & Quantity*, 49:463–470.

Ali Hürriyetoğlu, Surendrabikram Thapa, Hristo Tanev, and Surabhi Adhikari. 2025. Findings and insights from the 8th workshop on challenges and applications of automated extraction of socio-political events from text. In *Proceedings of the 8th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE 2025)*.

Mustafa Oz, Akan Yanik, and Mikail Batu. 2023. Under the shadow of culture and politics: Understanding lgbtq social media activists' perceptions, concerns, and strategies. *Social Media+ Society*, 9(3):20563051231196554.

Anil Singh Parihar, Surendrabikram Thapa, and Sushruti Mishra. 2021. Hate speech detection using natural language processing: Applications and challenges. In *2021 5th International Conference on Trends in Electronics and Informatics (ICOEI)*, pages 1302–1308. IEEE.

Siddhant Bikram Shah, Shuvam Shiwakoti, Maheep Chaudhary, and Haohan Wang. 2024. Memeclip: Leveraging clip representations for multimodal meme classification. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17320–17332, Miami, Florida, USA. Association for Computational Linguistics.

Leslie N. Smith. 2017. Cyclical learning rates for training neural networks. In *Computer Vision and Pattern Recognition*. arxiv.org.

Surendrabikram Thapa, Farhan Ahmad Jafri, Ali Hürriyetoğlu, Francielle Vargas, Roy Ka Wei Lee, and Usman Naseem. 2023. Multimodal hate speech event detection-shared task 4. In *CASE 2023-Proceedings of the 6th Workshop on Challenges and Applications of Automated Extraction of Socio-Political Events from Text, associated with 14th International Conference on Recent Advances in Natural*

Language Processing, RANLP 2023, pages 151–159. Association for Computational Linguistics.

Surendrabikram Thapa, Kritesh Rauniyar, Farhan Ahmad Jafri, Hariram Veeramani, Raghav Jain, Sandesh Jain, Francielle Vargas, Ali Hürriyetoğlu, and Usman Naseem. 2024. Extended multimodal hate speech event detection during russia-ukraine crisis-shared task at case 2024. In *7th Workshop on Challenges and Applications of Automated Extraction of Socio-Political Events from Text, CASE 2024*, pages 221–228. Association for Computational Linguistics.

Surendrabikram Thapa, Siddhant Bikram Shah, Kritesh Rauniyar, Shuvam Shiwakoti, Surabhi Adhikari, Hariram Veeramani, Kristina T. Johnson, Ali Hürriyetoğlu, Hristo Tanev, and Usman Naseem. 2025a. Multimodal hate, humor, and stance event detection in marginalized sociopolitical movements. In *Proceedings of the 8th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE 2025)*.

Surendrabikram Thapa, Shuvam Shiwakoti, Siddhant Bikram Shah, Surabhi Adhikari, Hariram Veeramani, Mehwish Nasim, and Usman Naseem. 2025b. Large language models (llm) in computational social science: prospects, current state, and challenges. *Social Network Analysis and Mining*, 15(1):1–30.