Hate Detection in MMH150K Dataset

Suryaansh Rathinam
School of Computing
National University of Singapore
A0307215N
suryaansh.rathinam@u.nus.edu

Sneha Sarkar
School of Computing
National University of Singapore
A0304787U
e1373875@u.nus.edu

Bhushan Mohol
School of Computing
National University of Singapore
A0304884X
E1373972@u.nus.edu

Tan Zhiqiang
School of Computing
National University of Singapore
A0290862X
E1327893@u.nus.edu

Ni Shinan
School of Computing
National University of Singapore
A0304558B
E1373646@u.nus.edu

ABSTRACT

Hate speech in multimodal formats, such as image-text pairs, poses growing challenges to online safety. We evaluated numerous multimodal classifiers on the `MMHS150K` dataset, a large-scale resource containing 150,000 tweet-image pairs labeled with multiple hate categories. This report highlights the findings of our two best-performing models. We fuse visual and textual features using two architectures: `EfficientNet-B0 + BiLSTM` and `MobileNetV2 + RoBERTa`. Both outperform text-only baselines, with the best model (`EfficientNet + BiLSTM`) achieving an F1-score of **72.17%**. Our results show that dynamic fusion and sequence-aware text encoding significantly improve hate speech detection and offer a scalable path for robust multimodal moderation systems.

KEYWORDS

Hate Detection, MMH150K, EfficientNet, BiLSTM, MobileNetV2, RoBERTa

1 BACKGROUND

With the rise of user-generated content, detecting hate speech has become essential for keeping online communities safe. Major social platforms like Facebook and Twitter (now X) have introduced policies to ban content targeting protected characteristics such as race, gender, religion, and disability [1, 2]. These platforms face the challenge of balancing free expression with the need to protect users, which requires automated systems that can process large volumes of content, intervene in real time, and apply rules consistently.

As online hate evolves—often using subtle language or coded expressions—detection systems must also adapt. Building robust detection models not only helps platforms keep harmful content in check, but also reduces the burden on human moderators. Ultimately, it's crucial to maintain this balance between freedom of speech and protection from targeted harm.

CS5344 Team 17, April 2025, National University of Singapore

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2 INTRODUCTION

This project focuses on building multimodal classifiers that combine both text and image features for binary hate speech detection using the `MMHS150K` dataset [3].

Our main contributions are:

(1) A full preprocessing pipeline for class balancing and multimodal data augmentation.
(2) Two fusion-based architectures: `EfficientNet-B0 + BiLSTM` and `MobileNetV2 + RoBERTa`.
(3) A comparative evaluation showing a 6.27% F1-score improvement over text-only baselines.

3 CHALLENGES

This project presented several significant challenges that shaped our methodology and approach:

3.1 Limited PySpark MLlib Capabilities

Initially our pipeline was PySpark MLlib based, however due to its very limited support for complex deep learning architectures like ResNet50 and LSTMs. This limitation prevented direct learning on image data within the PySpark framework, necessitating alternative approaches.

3.2 Large Dataset Processing

The substantial size of the MMHS150K dataset (150,000 tweet-image pairs) presented significant processing challenges, especially when working with multimodal data that requires more complex preprocessing than traditional text-only datasets.

3.3 Batch Training Requirements

Due to the dataset's size and the memory requirements of deep learning models, training had to be implemented in batches rather than loading the entire dataset at once. This required careful optimization of batch sizes to balance between memory constraints and training efficiency.

3.4 Subjectivity in Manual Annotations

The dataset relies on human annotators who may introduce subjective interpretations, particularly for borderline cases. The co-occurrence patterns shown in Figure 2 demonstrate this subjectivity,

with many samples receiving mixed annotations across different categories.

3.5 Model Architecture Constraints

While state-of-the-art models like ResNet would typically be preferred for image processing tasks, we had to optimize for both performance and efficiency. This led us to explore smaller, more efficient architectures like EfficientNet-B0 and MobileNetV2 that could deliver good accuracy while remaining computationally feasible.

4 DATASET

The `MMHS150K` dataset [3] is a multimodal hate speech dataset containing around 150,000 tweet–image pairs collected from Twitter. Each sample includes both textual and visual content, labeled by three annotators across multiple hate categories.

4.1 Dataset Structure

The dataset is provided in JSON format, where each entry represents one tweet–image pair with metadata, labels, and content. Key fields include:

- **`tweet_url`**: Direct link to the tweet.
- **`labels`**: Array of 3 labels (0–5), annotated by different annotators.
- **`img_url`**: Link to the associated image.
- **`tweet_text`**: Raw tweet text, may contain profanity or symbols.
- **`labels_str`**: Human-readable version of `labels`.

Listing 1: Example entry from MMHS150K dataset

```
"1063020048816660480": {
  "tweet_url": "https://twitter.com/user/status
      /1063020048816660480",
  "labels": [5, 5, 5],
  "img_url": "http://pbs.twimg.com/...jpg",
  "tweet_text": "My horses are r****** https://t
      .co/HYhqc6d5WN",
  "labels_str": ["OtherHate", "OtherHate", "
      OtherHate"]
}
```

Each tweet is assigned a numeric label (0–5), corresponding to:

- `0`: NotHate
- 1: Racist
- 2: Sexist
- 3: Homophobe
- 4: Religion
- 5: OtherHate

Note: We apply majority voting to the 3 annotations to get a final label (see Section 5.1). This helps resolve disagreements and ensures consistent training targets.

4.2 Class Distribution and Imbalance

As shown in Figure 1, `NotHate` makes up nearly 70% of the dataset, with the rest unevenly split across hate categories. `Racist` is the most common hate label (14.1%), followed by `OtherHate` (7.0%), and `Sexist` (5.1%). The `Religion` class is rare (0.6%), highlighting the need for class balancing (see Section 5.1).

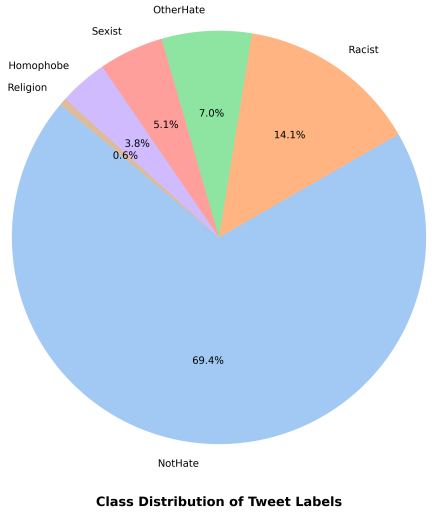

Figure 1: Class distribution in the dataset

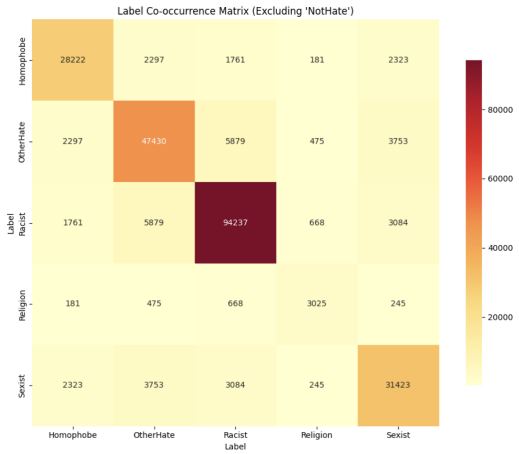

Figure 2: Label co-occurrence heatmap

4.3 Label Co-occurrence

Some tweets have mixed labels, as visualized in Figure 2. For example, `Racist` and `NotHate` often co-occur, which may reflect subtle hate or annotator disagreement. Other strong overlaps include `Homophobe` and `Sexist`, as well as `Racist` and `OtherHate`.

4.4 Common Label Combinations

Figure 3 shows the most common multi-label patterns. Many of them involve `NotHate` along with one or more hate labels, further confirming annotation ambiguity and overlapping interpretations.

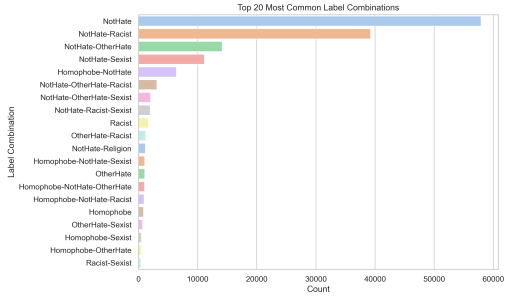

Figure 3: Most frequent label combinations

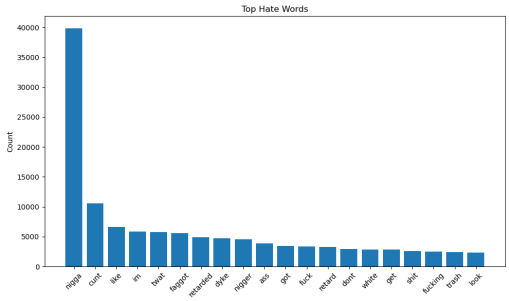

Figure 4: Most frequent words in hate samples

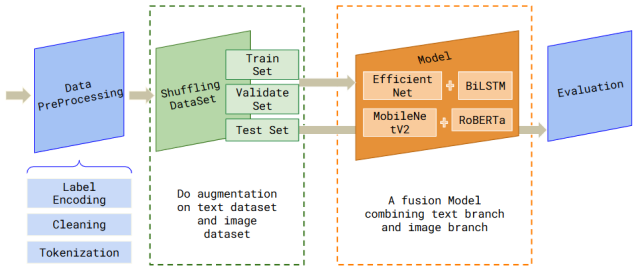

Figure 5: Workflow

4.5 Lexical Patterns in Hate Speech

Figure 4 highlights frequently used offensive terms in hate-labeled tweets. This analysis helped guide the text augmentation process (see Section 5.1) and demonstrates why simple keyword detection is insufficient.

5 WORKFLOW AND METHODOLOGY

Figure 5 shows our overall pipeline. We first clean and preprocess the raw data, then apply augmentation to handle class imbalance. The dataset is shuffled and split into 80% for training/validation and 20% for testing. Multimodal models are then trained and evaluated.

5.1 Data Preprocessing

- **Label Transformation**: Each sample has three labels from different annotators. We use majority voting and convert all hate-related classes (Racist, Sexist, etc.) into a single *Hate* label (1). Non-hate samples are labeled as 0.
- **Text Cleaning**: Tweets are lowercased, and we remove URLs, mentions, hashtags, numbers, and punctuation.
- **Tokenization**: We use `bert-base-uncased` from HuggingFace to tokenize text. All sequences are padded/truncated to 100 tokens for consistency.

5.1.1 ***Text Tokenization.*** We tokenize each tweet using a BERT-based tokenizer. It converts text into token IDs with special tokens and attention masks. This ensures all inputs have the same shape (100 tokens), ready for training.

5.1.2 ***Image Transformation.*** Images go through a simple transform pipeline:

- **Resize**: 224×224 for EfficientNet, 96×96 for MobileNet.
- **To Tensor**: Convert to PyTorch tensor, with inbuilt normalization.

Listing 2: Image transformation script

```
image_transform = transforms.Compose([
    transforms.Resize((IMAGE_SIZE,
        IMAGE_SIZE)),
    transforms.ToTensor()
])
```

5.1.3 ***Data Augmentation.*** To balance the dataset and improve generalization:

- **Text (Hate only)**: Each sample gets two random augmentations: synonym replacement, random deletion, or word swap.
- **Image (Hate only)**: One random augmentation: rotation, brightness change, or a vertical/horizontal flip.
- **Balancing**:
  - NotHate: downsampled to 75,000
  - Hate: oversampled to 75,000 using augmented samples

Figure 6 confirms that both classes are evenly balanced after augmentation, which helps reduce bias during training.

5.2 Model Architectures

We build two multimodal classifiers by modifying pretrained backbones and adding lightweight fusion layers. Changes are mainly in the final projection and fusion parts, while encoders are frozen. BiLSTM uses trainable fusion for deeper alignment, while MobileNet-RoBERTa applies fixed α-weighted fusion for speed. Table 1 summarizes the main design choices.

5.2.1 EfficientNet-B0 + BiLSTM.

- Remove the original classifier head → add `FC(1280→256)` + ReLU + dropout 0.5 as new image projection.
- For text: use bert-base-uncased tokenizer, then:
  - custom embedding layer (dim=128)
  - BiLSTM (hidden size = 128, bidirectional)
  - `FC(256→128)` + ReLU + dropout 0.5 to project text features
- Fusion: concatenate image (256 d) and text (128 d) → pass through `FC(384→128→1)` to produce final logit.

Table 1: Key settings for each model

Component	EffNet+Bilstm	MobileNet+RoBERTa
Image projection dim	256	256
Text projection dim	128	256
Image dropout	0.5	0.2
Text dropout	0.5	—
Fusion method	Learned concat	Static α-weight
α values	—	{0.0, 0.25, ..., 1.0}
LSTM hidden size	128 (bidirectional)	—

Table 2: Training hyperparameters

Parameter	Value
Optimizer	Adam
Learning rate	1×10^{-3}
Loss function	BCEWithLogitsLoss
Batch size	128
Epochs	30
Early stopping patience	3
Device	GPU / MPS
Image size (EffNet-B0)	224×224
Image size (MobileNetV2)	96×96

5.2.2 MobileNetV2 + RoBERTa (Alpha Fusion).

- Replace original classifier → add `FC(1280→256)` + dropout 0.2
- For text: use a specialized RoBERTa based tokenizer, then extract [CLS] token (768 d) and project with `FC(768→256)`
- Fusion: compute weighted sum:

$$F = \alpha \cdot \text{ImgFeat} + (1 - \alpha) \cdot \text{TextFeat}$$

where α is tuned on validation set. Final output is given by `FC(256→1)`.

5.3 Training Details

We trained both models using the Adam optimizer and early stopping based on validation F1. The loss function is `BCEWithLogitsLoss`, suitable for binary classification. The dataset was pre-balanced, so no class weights were applied.

For MobileNetV2+RoBERTa, we performed a grid search over $\alpha \in \{0.0, 0.25, 0.5, 0.75, 1.0\}$ and retrained the model with the best value. We logged per-epoch accuracy, precision, recall, and F1, and used custom plots to visualize training progress.

These hyperparameter values (learning rate = 1×10^{-3}, batch size = 128, patience = 3) were chosen based on small-scale debugging and were found to balance convergence speed with training stability.

6 EVALUATION RESULTS

6.1 Text-only Model Performance

The text-only baseline achieved an F1 score of 0.6590 and accuracy of 0.6791. As shown in Figure 7, its learning plateaued early and

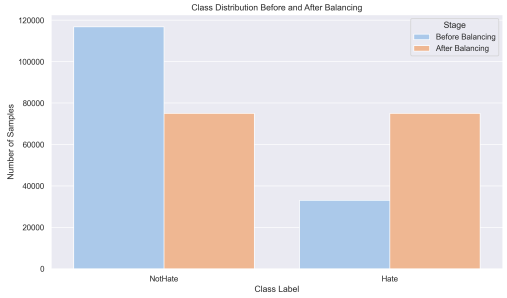

Figure 6: Class distribution before and after balancing

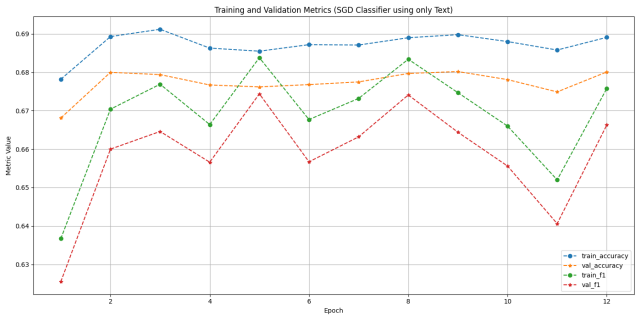

Figure 7: Training and validation metrics – Text-only model (SGD)

Figure 8: Training and validation metrics – BiLSTM + EfficientNet

struggled with generalization. The model often missed visually implied hate, confirming the importance of multimodal fusion.

6.2 EfficientNet-B0 + BiLSTM Performance

The BiLSTM + EfficientNet model shows steadily increasing validation F1 and accuracy, with training and validation curves closely aligned, suggesting stable learning and low overfitting.

This model achieved consistent training and validation trends (Figure 8). The F1 score peaked at **0.7217**, and the model showed strong generalization across both hate and non-hate classes.

Table 3: Classification Report – EfficientNet-B0 + BiLSTM

Class	Precision	Recall	F1-Score	Support
NotHate	0.72	0.71	0.72	22559
Hate	0.72	0.73	0.72	22441
Accuracy	–	–	0.7206	45000
Macro Avg	0.72	0.72	0.72	45000
Weighted Avg	0.72	0.72	0.72	45000

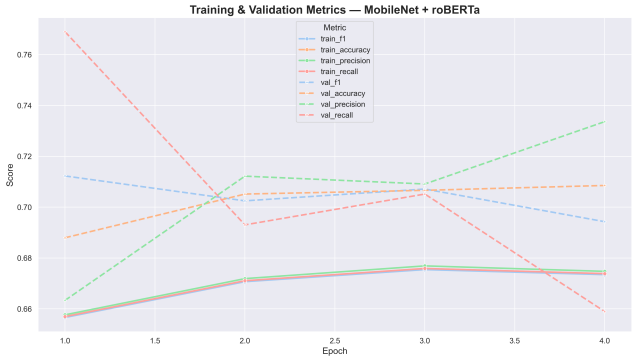

Figure 9: Training and validation metrics – MobileNet + RoBERTa

Table 4: Classification Report – MobileNetV2 + RoBERTa

Class	Precision	Recall	F1-Score	Support
NotHate	0.68	0.76	0.72	22559
Hate	0.73	0.65	0.69	22441
Accuracy	–	–	0.7050	45000
Macro Avg	0.71	0.70	0.70	45000
Weighted Avg	0.71	0.70	0.70	45000

6.3 MobileNetV2 + RoBERTa Performance

The MobileNet + RoBERTa model converges early but exhibits a gap between training and validation F1, indicating limited generalization capacity and signs of mild overfitting.

Although training loss decreased steadily, validation F1 slightly fluctuated (Figure 9). The model achieved the highest precision (**73.07%**) on the Hate class but had relatively lower recall.

6.4 Alpha Tuning for Fusion Weight

We experimented with $\alpha \in \{0.0, 0.25, 0.5, 0.75, 1.0\}$ to tune the relative weight of image features. As shown in Figure 10, $\alpha = 0.75$ achieved the best F1 score. Performance drops sharply when $\alpha = 1.0$, indicating that text features are more informative, while image features provide useful but secondary cues.

6.5 Overall Comparison

EfficientNet + BiLSTM achieves lower loss and higher F1 over time, while MobileNet + RoBERTa flattens early in most metrics.

EfficientNet + BiLSTM outperforms across all metrics except precision, highlighting better recall and overall balance.

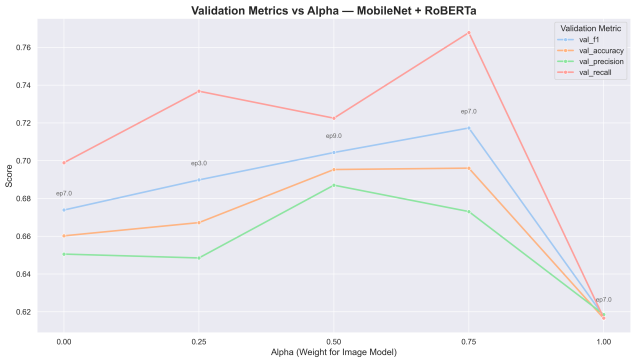

Figure 10: Validation metrics vs. alpha – MobileNet + RoBERTa

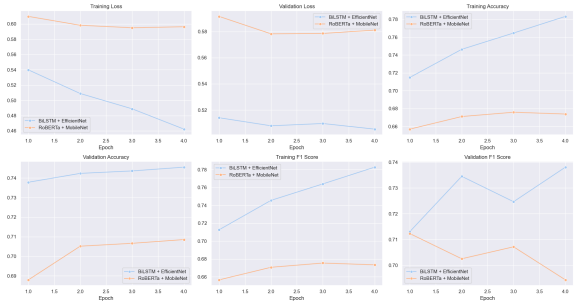

Figure 11: Grid comparison – loss, accuracy, F1 (both models)

Table 5: Overall test performance comparison

Model	Accuracy	F1	Precision	Recall
EffNet + BiLSTM	0.7206	0.7217	0.7171	0.7263
MobileNet + RoBERTa	0.7050	0.6862	0.7307	0.6469

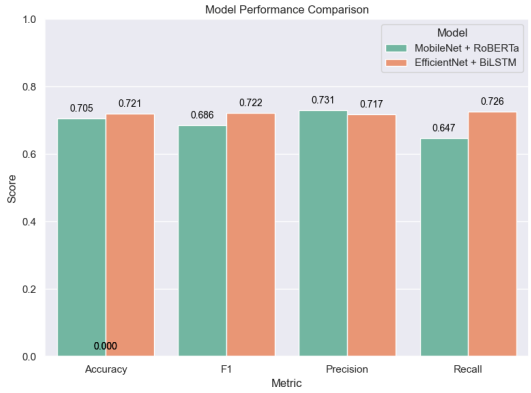

Figure 12: Final model test performance

The performance gap also reflects the benefits of sequence-aware text encoders and dynamic feature fusion. The **BiLSTM** model adapts better to ambiguous or image-supported hate, while **RoBERTa** relies more on textual clues alone.

Model Strength Summary:

EfficientNet-B0 + BiLSTM

- Balanced precision and recall (F1 = 0.7217)
- Strong recall for hate class (72.63%) → better safety coverage
- Stable learning curve and generalization

MobileNetV2 + RoBERTa

- Higher precision (73.07%) → fewer false positives
- Lightweight and fast to train
- Better recall on non-hate (76%)

Interpretation: EfficientNet-BiLSTM is more balanced and robust across classes and training stages. It is suitable for general moderation tasks. MobileNet-RoBERTa is more conservative and efficient, making it ideal for real-time systems where precision matters most.

7 DISCUSSION AND ANALYSIS

7.1 Model Design Differences

The performance difference is explained by:

- **Text modeling**: BiLSTM reads full token sequences, while RoBERTa uses [CLS] token only.
- **Fusion strategy**: BiLSTM uses dynamic, learned fusion; RoBERTa uses fixed alpha weighting.
- **Projection depth**: BiLSTM includes 2 FC layers after fusion; RoBERTa only 1.

7.2 Limitations

While our models perform well on the `MMHS150K` dataset, there are several limitations:

- The dataset mainly consists of English tweets from Twitter, which may limit generalization to other languages or platforms.
- Rare categories like `Religion` remain underrepresented, leading to lower F1 scores (up to 5% drop) even after augmentation.
- Our binary setup simplifies a task that is often multi-label or severity-aware in real-world moderation systems.

7.3 Use Case Recommendations

- **Precision-critical scenarios**: For applications like auto-flagging or content removal where false positives must be minimized, **MobileNet + RoBERTa** is preferred due to its higher precision on the hate class.
- **Balanced moderation tasks**: For general-purpose moderation, semi-automated review, or when recall is equally important, **EfficientNet + BiLSTM** is a better choice due to its more balanced precision and recall.
- **Low-resource deployment**: When speed and efficiency matter more than peak accuracy, such as in mobile or edge settings, **MobileNet + RoBERTa** offers faster inference with fewer parameters.
- **Subtle or ambiguous content**: If detecting sarcasm, memes, or visually implied hate is a key requirement, **EfficientNet + BiLSTM** performs better due to its dynamic fusion and sequence-aware text encoding.

7.4 Fusion Strategy Insights

- **Fixed-weight fusion (MobileNet + RoBERTa)** uses a static α-weighted sum of image and text features across all samples:

$$\text{FusedFeat} = \alpha \cdot \text{ImgFeat} + (1 - \alpha) \cdot \text{TextFeat}$$

  This method is simple and efficient, but does not adapt to content-specific modality importance.
- **Learned fusion (EfficientNet + BiLSTM)** uses a concatenated feature vector followed by multiple fully connected layers. This allows the model to dynamically learn which modality matters more for each sample.
- **Fusion layer depth matters**: The RoBERTa model uses only a single projection layer after fusion, while BiLSTM-based fusion includes two layers with ReLU and dropout. This makes it more expressive and better suited for capturing complex cross-modal relationships.
- While image features provide marginal improvement overall, they are critical for detecting visually implied hate (e.g., memes), where text alone may appear benign.

8 CONCLUSION AND FUTURE WORK

This study explored multimodal hate speech detection using the `MMHS150K` dataset, which contains 150,000 tweet–image pairs labeled across multiple hate categories. We built and compared two multimodal classifiers: `EfficientNet + BiLSTM` and `MobileNetV2 + RoBERTa`. Among them, `EfficientNet + BiLSTM` achieved the best performance with 72.06% accuracy and an F1 score of 72.17%.

Our results challenge earlier claims [4] that text-only models outperform multimodal ones. We show that with dynamic fusion and sequence-aware encoding, multimodal architectures can achieve better generalization and higher F1-scores, especially on subtle or visually implied hate speech.

In future work, we plan to experiment with deeper fusion strategies, larger pre-trained models, and more robust augmentation techniques. We also aim to extend the model beyond binary classification by incorporating multi-label predictions and investigating severity-level tagging.

REFERENCES

[1] Facebook. (2013). Controversial, Harmful and Hateful Speech on Facebook. https://www.facebook.com/notes/facebook-safety/controversial-harmful-and-hateful-speech-on-facebook/574430655911054/
[2] Twitter. (2023). Hateful conduct policy. https://help.twitter.com/en/rules-and-policies/hateful-conduct-policy
[3] MMH150K Dataset (2019) https://gombru.github.io/2019/10/09/MMHS/
[4] Gomez, R., Gibert, J., Gomez, L., & Karatzas, D. (2019). *Exploring Hate Speech Detection in Multimodal Publications.* arXiv preprint arXiv:1910.03814.

APPENDIX

The source code and implementation details for this project are available in the public GitHub repository below:

`https://github.com/suryaansh2002/CS5344-Project`