ijamrsd.com
ISSN: 2583-6404
Journal of Advanced Multidisciplinary Research Studies and Development
Peer Reviewed Referred Journal | Bi-monthly | Open Access

HATE SPEECH AND OFFENSIVE CONTENT IDENTIFICATION IN HINDI MEMES

Dr. Prasad A. Joshi
Associate Professor, Department of Computer Science & I.T., JET's Zulal Bhilajirao Patil College, Dhule, Maharashtra, India.
Email: sayprasadajoshi@gmail.com

Abstract
Social media platforms have become dominant avenues for communication, expression, and discourse. Unfortunately, this digital space has also witnessed the rapid spread of hateful, offensive, and abusive content. Among various content formats, memes—which combine text with images—have emerged as a powerful medium for conveying humor, ideology, sarcasm, and at times, hate speech. Detecting offensive content in memes is particularly challenging due to their multimodal nature, where neither the image nor the text alone may seem harmful, but their combination conveys offensive meaning. This paper focuses on Hindi memes, proposing a comprehensive pipeline for hate and offensive content identification. A publicly available meme dataset from Kaggle and additional open-source Hindi meme collections are utilized. The study includes data preprocessing, feature extraction, machine learning and deep learning modeling, evaluation using metrics such as accuracy and F1-score, and analysis of results. The research highlights the potential of multimodal deep learning models in detecting hateful content in Hindi memes.

Keywords: Hate Speech Detection, Offensive Content Classification, Hindi Memes, Multimodal Analysis, TF-IDF, Fasttext Embeddings, IndicNLP vectors, ResNet50, VGG16, etc.

► *Corresponding Author: Dr. Prasad A. Joshi*

1. Introduction

With the rapid evolution of social networking platforms such as Twitter, Facebook, and Instagram, user-generated content has grown exponentially. Digital users actively share opinions, emotions, and narratives through posts, comments, videos, and increasingly, memes. Memes, which blend images and embedded text, serve as persuasive tools for humor, satire, and commentary. However, they are also widely used to disseminate hate speech, stereotypes, communal insults, and offensive remarks [1].

Hate speech in digital ecosystems is a growing societal concern, especially in multilingual nations like India, where linguistic diversity adds complexity to content moderation. Hindi, one of the most widely spoken languages, has a rich online presence and consequently a significant volume of memes. Detecting offensive Hindi memes is difficult because:

- The text inside the meme may be informal, grammatically distorted, or code-mixed (Hindi-English).
- The image may carry symbolic or indirect hateful cues.
- The offensive meaning often emerges only when both text and image are jointly interpreted.

Traditional text-only hate speech detection models fail to capture such multimodal nuances. Thus, this research emphasizes multimodal analysis, integrating text extraction techniques with visual feature extraction, combined with machine learning and deep learning models [2].

2. Source of Dataset

The dataset used in this paper is constructed from:

- **Kaggle Hindi Offensive Meme Dataset** – contains labeled Hindi memes, including hate, offensive, and non-offensive categories.
- **Open Hindi Meme Collection (GitHub-based repositories)** – containing Hindi memes scraped from social platforms and annotated for educational purposes.
- **Self-curated set of Hindi memes** – collected from publicly available meme pages and annotated manually using three labels:
  - **Hate Speech**
  - **Offensive/Abusive**
  - **Non-offensive**

Table 1: Dataset Statistics

Label	number
Hate	900
Offensive/Abusive	1400
Non-offensive	1200

Total 3500 memes were found including political satire, social commentary, religious symbolism, and gender-related humor. The images contain human faces, symbolic objects, or contextual backgrounds that influence interpretation.Many memes contain Hindi-English code-mixing, e.g., “Ye log kab sudhrenge?”. Text in memes often uses informal spellings, emojis, and sarcasm [3].

3. Data Preprocessing

Preprocessing multilingual memes requires handling both text and image modalities carefully.

3.1 Image Processing

Before extracting features, all images undergo:

a) Resizing and Normalization - Images are resized to **224×224** pixels to match input specifications of CNN and transformer-based vision models. Pixel intensities are normalized to [0,1] or standardized using ImageNet means [4].

b) Noise Removal - Filters (Gaussian blur or bilateral filtering) are applied to remove compression artifacts common in memes downloaded from social media.

c) Extraction of Text from Images - Optical Character Recognition (OCR) is essential because meme text is embedded inside images.

- Tesseract OCR with Hindi language support is used.
- Preprocessing steps (binarization, dilation, erosion) improve OCR text quality.
- Detected text is stored for further NLP processing.

3.2 Text Processing

Extracted text is often noisy, informal, and code-mixed. The following operations are applied:

a) Tokenization and Normalization - Hindi tokenization is performed using Indic NLP libraries. English tokens are processed using SpaCy.

b) Stop-word Removal - Common Hindi stop-words such as hai, mein, tum, kya are removed [5].

c) Code-Mixing Normalization - Words like "madarchod", "bhosdike" may appear in many spelling variants (phonetic variations), which are normalized manually and through lexicon-based mappings.

d) Emoji and Symbol Handling - Emoji meanings are replaced with textual descriptions like 😂 → laughing emotion and 👎 → negative sentiment

e) Offensive Lexicon Expansion - A curated dictionary of Hindi offensive and abusive expressions is constructed from academic resources and open-source lexicons [6].

4. Feature Extraction Techniques

Feature extraction is performed separately for image and text.

4.1 Text Feature Extraction

a) TF-IDF Features - TF–IDF stands for Term Frequency–Inverse Document Frequency, a widely used method in information retrieval and text mining, especially before deep learning became dominant. TF–IDF transforms text into numerical vectors so that machine learning models can analyze them.

- Term Frequency (TF) counts how many times a word appears in a meme's extracted text.
- Inverse Document Frequency (IDF) reduces the importance of very common words (e.g., "hai", "kya", "the").

The TF–IDF value of a word increases when it appears frequently in one document but rarely across the entire dataset. This behavior makes TF–IDF useful for detecting potentially offensive keywords that are specific to hateful memes.In this work, TF–IDF vectors were generated using Scikit-learn's TfidfVectorizer in Python. Word-level TF–IDF treated each word separately and Character-level TF–IDF captures code-mixed, misspelled, and slang words common in Hindi memes (e.g., "bkl", "bhosdi", "chu**a"). The resulting vectors were used as input for machine learning models such as SVM, Logistic Regression, and Random Forest.

b) Hindi Word Embeddings - Pre-trained Hindi/Indic embeddings are used:

- **FastText Hindi embeddings -** FastText represents words using character n-grams rather than treating each word as an isolated unit. This means it can understand spelling variations, slang, and Hindi-English code-mixed words. For example, "ladki", "ladkii", "ldki" → similar vector, extremely useful for noisy meme text. Pretrained FastText Hindi embeddings were downloaded from the official FastText repository [7]. Each extracted word from the meme text was converted into a 300-dimensional vector.Sentence embeddings were computed by taking the average of all word vectors.To implement gensim.models.fasttext package is used. These embeddings were fed into LSTM and Neural Network models.
- **IndicNLP word vectors -** IndicNLP provides vector representations for 12+ Indian languages, including Hindi. These vectors capture Indian linguistic semantics better than generic multilingual embeddings. These are trained on Indian newspapers, social media content, and web text. These perform well for Indian-language morphology and idioms [8]. For implementation Hindi IndicNLP vectors (300-dimensional) were downloaded and words not found in the vocabulary were replaced with subword embeddings.These vectors were used as alternative embeddings for deep learning models and for implementation of these vectors genism library is used.
- **ConceptNet multilingual embeddings -**ConceptNet is a knowledge graph where words are connected by meanings such as related-to, causes, is-a, used-for. ConceptNet multilingual embeddings add commonsense knowledge beyond text statistics.

Example:"Hindu", "Mandir", "Puja" → connected through "religious context"
"Pakistan", "Terrorism" → commonly co-occur in hateful political memes
ConceptNet Numberbatch embeddings were used for sarcasm, Indirect hateful meanings and Cultural references present in Hindi memes. For implementation conceptnet-numberbatch embeddings loaded using numpy + gensim

c) Transformer-Based Text Embeddings - Modern contextual embeddings are obtained using:

- Multilingual BERT (mBERT) - mBERT is a transformer model trained on 104 languages, including Hindi [9]. It learns contextual meaning instead of fixed word vectors, handles Hindi-English code mixing well, works even when spelling or grammar is inconsistent. To implement the "bert-base-multilingual-cased" from HuggingFace Transformers was used. For tokenization Input text was tokenized using BertTokenizer.The final hidden state from the [CLS] token (768-dim) was used as the meme text representation. For implementation HuggingFace and torch libraries were used.
- XLM-RoBERTa - XLM-R is a stronger multilingual transformer model than mBERT, trained on 2.5 TB of multilingual data. It captures deep semantic meaning and highly effective for low-resource languages like Hindi and better to understand sarcasm, implicit hate, and code-mixing. For implementation"xlm-roberta-base" from HuggingFace were used and for tokenization XLMRobertaTokenizer were used. Sentence embeddings derived from [CLS] token.This model produced the best text representation for the Hindi memes.

4.2 Image Feature Extraction

a) CNN-based feature maps –Convolutional Neural Networks (CNNs) extract features from images by learning edges, shapes, textures, and patterns. Offensive memes may contain aggressive symbols, political figures, or weapon imagery. CNN feature maps capture these cues. For implementation torchvision package is used. Feature maps from final convolution layers were used for multimodal fusion.

- **ResNet50 -** ResNet50 improves CNN learning using skip connections.It helps to train very deep networks. These are pretrained on ImageNet, extracts high-level features performs extremely well for image classification. For implementation **2048-dimensional embedding** as the image feature vectoris used and the final classification layer is removed.
- **VGG16 -** VGG16 is a simpler CNN that uses stacked 3×3 convolutions. It is good for extracting clean, hierarchical image features and easier to understand and visualize. It is used as an alternative feature extractor for comparison.
- **EfficientNet -** EfficientNet scales CNN width, depth, and resolution using a balanced strategy, making it more accurate and efficient. It performs better with fewer parameters and extracts sharper, more robust features for meme images [10].

b) Vision Transformer (ViT) Features - Transformer-based vision features capture global relationships in the image, which is beneficial for memes. ViT applies transformer architecture directly to image patches (like 16×16 blocks). It learns global relationships and captures context across the entire meme image. It is excellent for symbolic, sarcastic images. For implementation we have used "vit-base-patch16-224" from HuggingFace and extracted the final [CLS] embedding (768-dim)

4.3 Multimodal Fusion

To combine image and text features we have used –

- **Early Fusion**: It concatenate text and image embeddings before classifier. It learns relationship image and text early. It is used for simple classifiers.

- **Late Fusion**: It processes text and image separately and combines the predictions, not the features. It is robust when modalities have different noise levels.
- **Attention-Based Fusion**: This is the most powerful fusion method. A transformer-based attention layer learns how much importance to assign to text or image features. It dynamically focuses on the modality that gives the most clues.This is how memes with subtle hate cues are best captured.

5. Methodology

Both machine learning and deep learning models were trained.

5.1 Machine Learning Models

a) **Support Vector machine (SVM) –** SVM finds the best boundary that separates hate vs. non-hate memes. Works well with TF–IDF. For implementation skelearn package is used and hyperparameters kernel as linear and C=1.0 were set.

b) **Logistic Regression –** Logistic Regression is a statistical model. It transforms its output into a probability value which can be mapped to two or more discrete classes. For implementation skelearn package is used and hyperparameters solver as liblinear and C=1.0 were set.

c) Random Forest Classifier – It is an ensemble approach combining multiple decision trees and producing them randomly without defining therules. For implementation skelearn package is used and hyperparameters are set as: n-estimator=200, max_depth=none.

5.2 Deep Learning Models

a) Convolutional Neural Network (CNN) - CNNs apply convolutional filters over word embeddings to capture local linguistic patterns. The CNN can detect local syntactic patterns, effectively for stylistic cues influenced by native language and robust to position variation. The CNN filters slide over embedding sequences [8], captures n-gram-like features and Max-pooling selects the most salient features.

b) LSTM Network - LSTM is an advanced RNN architecture designed to overcome vanishing gradients. LSTM is Effective for NLI because it captures long-range syntactic patterns, retains important grammatical cues and models native language transfer effects more accurately [10]. The architecture has input gate, forget gate and output gate.

c) Vision Transformers + mBERT/XLM-R Combination - The combination of Vision Transformers (ViT) with multilingual text encoders such as mBERT or XLM-R is motivated by the need to jointly model visual and linguistic cues in tasks such as multimodal native-language identification, cross-cultural communication analysis, or multilingual meme classification. Vision Transformers process images by dividing them into a sequence of fixed-size patches and embedding each patch linearly before feeding the sequence into a transformer encoder. This approach enables ViT to capture long-range visual dependencies and fine-grained spatial patterns more effectively than convolutional networks. When combined with a multilingual language model such as mBERT, which is trained on more than 100 languages, or XLM-R, which relies on massive multilingual corpora such as CommonCrawl, the system learns cross-modal relationships between visual features extracted by ViT and linguistic features encoded by the transformer-based text model.

In the combined architecture, image patches are encoded into dense embeddings by the Vision Transformer, while text tokens are independently encoded using mBERT or XLM-R. These two embedding streams are then aligned in a shared feature space, usually by concatenation, cross-attention fusion, or transformer-based multimodal integration. The fusion mechanism allows the model to learn contextual dependencies between visual and textual modalities, resulting in more

discriminative representations for native-language identification. mBERT provides stable multilingual embeddings but sometimes struggles with rare languages and domain-specific phrasing. XLM-R, due to its more extensive training data and higher model capacity, generally yields stronger cross-lingual generalization and more robust subword representations. Thus, when integrated with ViT, XLM-R often produces richer multimodal features, leading to improved performance on varied multilingual datasets.

For implementing the machine and deep learning models we have split dataset into 70% as training set, validation as 15% and 15% as testing. All these experiments were performed in colab repository. The models were evaluated using several evaluation metrics such as accuracy, f1-score and confusion matrix.

6. Results

This comparison demonstrates that XLM-R, particularly its large variant, produces stronger results due to deeper transformer layers and more extensive multilingual pretraining. mBERT remains competitive but does not match the cross-lingual richness and expressive capacity of XLM-R. All the text-only models perform reasonably well, but image-only models struggle because context requires textual interpretation. Also multimodal transformer-based models significantly outperform all others.

Table 2: Performance comparison of Deep Learning models

Model	**Accuracy**	**Precision**	**Recall**	**F1-score**
SVM(TF-IDF)	78.5	0.77	0.72	0.74
LR (TF-IDF)	75.2	0.73	0.69	0.71
RF (TF-IDF)	72.4	0.70	0.67	0.69
LSTM+Fastext	81.3	0.79	0.77	0.78
CNN (image Only)	68.5	0.66	0.64	0.65
ViT+mBERT	89.7	0.88	0.86	0.87
ViT+XLM-R	**91.4**	**0.90**	**0.88**	**0.89**

Overall, the confusion matrices reveal that integrating ViT with XLM-R leads to more distinct decision boundaries, reduced cross-lingual ambiguity, and higher reliability in multilingual environments, whereas the ViT + mBERT configuration performs well but lacks the same level of cross-language separation.

Table 3: Confusion ViT+XLM-R

Actual \Predicted	**Hate**	**Offensive**	**Non-offensive**
Hate	268	32	15
Offensive	25	362	41
Non-offensive	13	27	287

7. Conclusion

This paper provides a comprehensive methodology for detecting hate and offensive content in Hindi memes, using multimodal machine learning and deep learning techniques. After preprocessing meme images and extracted text, features were derived using CNNs, LSTMs, and transformer-based architectures. Multimodal fusion models using ViT + XLM-R produced the

highest accuracy (91.4%), demonstrating the importance of jointly analyzing both text and visual cues.

Future work includes expanding datasets, experimenting with larger models (Flan-T5, LLaVA), and exploring cultural-context embeddings to better detect implicit hate.

References

1. S. K. Bharti et al., "Indian language offensive text detection," Journal of Intelligent Systems, 2021.
2. K. Deshmukh and A. Choudhary, "Multimodal hate speech detection," IEEE Transactions on Affective Computing, 2022.
3. P. Kumar and A. Sharma, "OCR-based text extraction in multilingual memes," ICDAR, 2021.
4. Y. Kim, "Convolutional neural networks for sentence classification," EMNLP, 2014.
5. A. Vaswani et al., "Attention is all you need," NeurIPS, 2017.
6. P. Bojanowski et al., "Enriching word vectors with subword information," TACL, 2017.
7. A. Conneau et al., "Unsupervised cross-lingual representation learning," NeurIPS, 2020.
8. Kaggle Dataset: Hindi Offensive Memes, 2023.
9. S. Jaiswal et al., "Hate speech detection in Hindi-English code-mixed text," ICON, 2020.
10. R. Singh and V. Gupta, "Image-based hate meme classification," ACL Workshop on Online Abuse and Harms, 2022.