Available online at website: https://jurnal.iaii.or.id/index.php/RESTI

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 10 No. 2 (2026) 355 - 365 e-ISSN: 2580-0760

Leveraging Sarcasm Text Detection in Meme Polarity Classification

Zeva Patu Assyadid[1], Ema Rachmawati[2]*
[1,2]School of Computing, Universitas Telkom, Bandung, Indonesia

*emarachmawati@telkomuniversity.ac.id

Abstract

Memes are a popular internet format and spread quickly across many social media platforms. People can express their ideas, criticisms, interests, or aversions through memes. But in some cases, other people may interpret memes differently and feel bad about it. This variation in meme interpretation is a challenge in sentiment analysis, as a meme can be judged negative or positive by different individuals. Therefore, there is a need for an automated system that can consistently predict the sentiment polarity of memes. A meme is multimodal content that could consist of visual and textual components, which is suitable for a sentiment polarity analysis study. To model a system that effectively leverages multimodal features, the model needs to understand the meme features. This study proposes a joint deep learning model—BERT and Densenet121—that concatenates text, image, and cluster features based on the extracted face encodings. To assess the context of the texts better, BERT was trained with a sarcasm dataset. The widely used 'SemEval 2020 Task 8: Memotion Analysis dataset' was used in this study due to its comprehensive annotation of meme-based sentiment and sarcasm, which aligns with this study's approach. The result demonstrates that the model achieved 0.3738 Accuracy (+2.52%) and 0.3735 Weighted F1 (+1.04%), while maintaining competitive Macro F1 (0.3047). This result shows effective sarcasm adaptation on imbalanced datasets, with improved ability to detect positive and neutral sentiments and reduce sarcastic false negatives compared to the base model. This highlights the effectiveness of integrating sarcasm detection into the model framework for robust sentiment classification in memes.

Keywords: deep learning; meme; memotion; sarcasm detection; sentiment classification

How to Cite: Z. P. Assyadid and E. Rachmawati, "Leveraging Sarcasm Text Detection in Meme Polarity Classification", *J. RESTI (Rekayasa Sist. Teknol. Inf.)* , vol. 10, no. 2, pp. 355 - 365, Apr. 2026.
Permalink/DOI: https://doi.org/10.29207/resti.v10i2.7516

Received: January 22, 2026
Accepted: April 5, 2026
Available Online: April 22, 2026

This is an open-access article under the CC BY 4.0 License
Published by Ikatan Ahli Informatika Indonesia

1. Introduction

In the last few years, memes have been popular on many social media platforms to express feelings or just share some jokes with friends[1-3] . Studies show that meme-based posts have approximately 30% higher share rates than standard promotional content, reflecting their dominant role in digital communication. But in some cases, other people may interpret memes differently and feel bad about it. This variation in meme interpretation poses a significant challenge in sentiment analysis, as the same meme can be perceived as negative or positive by different viewers [4-11]. Therefore, there is a need for an automated system that can consistently predict the sentiment polarity of memes [12-15].

Sentiment polarity is a common field of sentiment analysis in natural language processing. With the multimodal form, memes are a perfect subject for sentiment polarity analysis. Multimodal analysis utilizes multiple inputs in data processing, such as image, video, text, or audio. With multiple inputs, multimodal systems can capture more accurate context and often provide better results than unimodal [2], [16-18]. One of the studies with a multimodal approach that is in the top rank in the 'SemEval 2020: Memotion Analysis competition' [19] is the model from Guoym [20] which employs both data-based and feature-based ensemble strategies combining ResNet50 visual features with BERT and ELMo text embeddings. However, most existing approaches rely on general-purpose models and fail to exploit meme-specific multimodal features, such as facial expressions and spatial object positions, resulting in suboptimal performance [21-25].

Cluster-based Deep Ensemble Learning (CDEL) model by Guo et al. [26] addresses this limitation by exploiting facial features in memes and achieves state-of-the-art performance, outperforming all previous models on benchmark emotion classification datasets. The implementation of the CDEL model begins with the

selection stage of the features to be extracted, which are images and texts in the meme. The next stage is to select clustering and classification algorithms that produce optimal joint performance. Then, in the final stage, the algorithm will be combined with a deep learning model. Although the CDEL model is superior in utilizing multimodal features, it is still lacking in sarcasm text detection in memes. Guo et al. stated that the analysis error of sarcasm text detection reached 38%. This happens when the text implies an emotion that is opposite to the actual one.

As can be seen in Figure 1, there is one example of error sarcasm text detection on the CDEL model. There is a text "launch the nukes" which explicitly contains a negative meaning, but the meme has a positive label. Sarcasm text detection is a challenging sub-task of sentiment analysis, as it requires understanding the implied meaning, which is often the opposite of the literal meaning [27]. Sarcasm is influenced by context, tone, or cultural knowledge that is usually not obvious in plain text [28]. Traditional models could struggle with this, as they may misclassify sarcastic comments due to their surface-level semantics. BERT, in the other case, has shown robust performance in sarcasm detection tasks by leveraging its bidirectional attention mechanisms to capture subtle contrasts in sentiment. Recent studies show that BERT-based models achieve state-of-the-art performance in sarcasm detection, with precision, recall, and F1-score reaching 0.917 on benchmark datasets [27].



Figure 1. Example of Sarcasm Text Detection Error

Studies from Parameswaran et al. [29] have demonstrated that BERT on sarcasm-specific datasets leads to improved classification accuracy and generalization. They proposed BERT-based models, which are TD-BERT and BERT-AEN, for detecting the target of sarcastic comments posted on social media. The proposed models were evaluated on various social media datasets like Reddit, book snippets, and tweets. And the results showed that their model outperformed up to 15.2% of the previous state-of-the-art system on both the Reddit and tweets datasets.

Recent advances in sentiment analysis have extended beyond traditional text-based approaches to incorporate domain-specific contexts. As shown in [30], which demonstrates that IndoBERT effectively captures emotional nuances in customer reviews for drinking water services, outperforming traditional methods. Their approach enables accurate emotion classification—such as satisfaction, frustration, or concern—which helps service providers identify critical issues and improve responsiveness.

Despite significant progress in multimodal sentiment analysis and meme classification, there is still limited integration of sarcasm detection in sentiment models. While existing studies treat sarcasm detection and sentiment classification as separate tasks [21], [22], [27], there is insufficient research on explicitly integrating sarcasm understanding into sentiment prediction models. Current approaches fail to model sarcasm as a sentiment modifier, leading to misclassification when sarcastic content contradicts surface-level sentiment indicators [28].

In this study, we develop an automatic system for meme sentiment polarity analysis that leverages facial features as cluster-based features inspired by the CDEL model, but with sarcasm text detection applied in the text model. The text model is trained with the same Memotion sarcasm dataset to improve the understanding of sarcasm in memes. This study also tests several hyperparameters to get a combination of parameters that produce the best results. Our study highlights the importance of leveraging sarcasm text detection in the joint deep learning model to process multimodal subjects like memes.

The remaining part of this paper is organized as follows: In Section 2, we discuss the Memotion dataset and the joint deep learning model architecture. Then, in Section 3, we present experimental results on evaluating the model performance. Finally, we conclude the paper in Section 4.

2. Methods

The model developed in this study is illustrated in Figure 2, with inputs at the top and outputs at the bottom. Briefly, the model integrates a pretrained BERT for processing text in images, DenseNet121 for image feature extraction, and additional cluster features based on face encoding.

2.1 Memotion Dataset

The dataset used in this study is the well-known Memotion dataset from the SemEval 2020 Task 8 competition [19]. The competition organizers had previously divided the data into a training set and a test set, consisting of 6,992 and 1,878 samples, respectively. The details of the sentiment distribution for both training and test datasets are presented in Table 1.

Table 1. Sentiment Distribution on Training and Test Data

Sentiment	Training Data	Test Data
Positive	4,160	1,111
Neutral	2,201	594
Negative	631	173

The Memotion training data contain several columns: *image_name, text_ocr, text_corrected, humour, sarcasm, offensive, and motivational.* In this study, only the *image name, text,* and the corresponding *sarcasm* and *sentiment* labels were used. The dataset includes

image files with seven unique extensions: jpg, jpeg, JPG, png, PNG, jpe, and bmp.

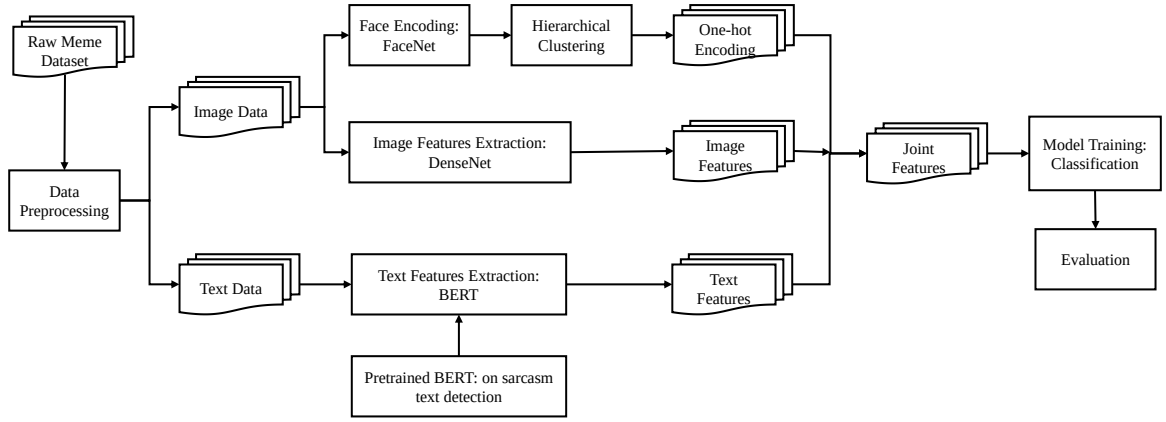


Figure 2. Meme Polarity Classification using Sarcasm Detection

2.2 Preprocessing

The preprocessing stage consisted of three main steps. First, meme data containing null values in the training and test sets were handled by copying the text from the *text_ocr* column to the *text_corrected* column whenever the latter contained a null value and the former did not. Any remaining null values in the *text_ocr* column were then replaced with an empty string (""), and the column data type was converted to string to ensure proper tokenization.

Following this, sentiment labels in the training data were standardized by converting *very_positive* to *positive* and *very_negative* to *negative*, aligning them with the available ground truth labels. Finally, image processing was carried out through a custom function, *load_and_preprocess_image*, which resized each image to 224 × 224, converted it into a NumPy array, and applied the *preprocess_input* function from the keras.applications.resnet50 library to prepare the data for feature extraction. During this stage, an IOError was raised indicating that image_5119.png was corrupted, which was resolved by setting LOAD_TRUNCATED_IMAGES = True.

2.3. Face Encoding Extraction

To extract face encodings from images, the related function from the DeepFace library was utilized, employing FaceNet as the backbone model within a custom function named *extract_face features()* (as illustrated in Figure 3). This function generates a vector representation of detected faces within the images. Specifically, it extracts embeddings containing the face encoding (facial vector representation) along with *face_confidence*, which indicates the detection confidence level.

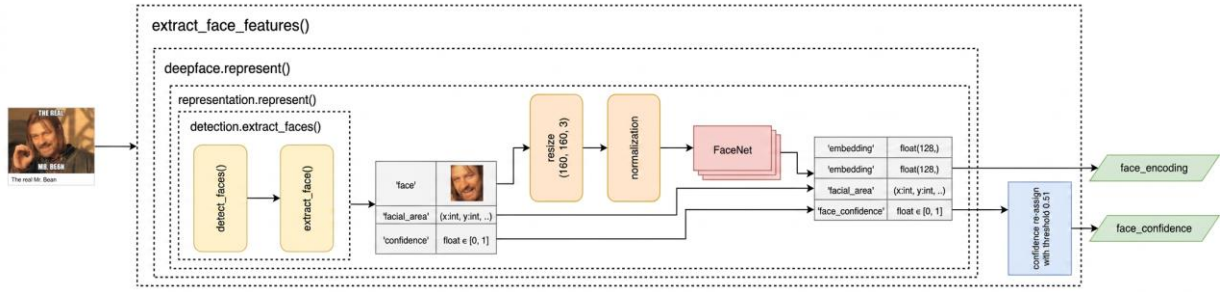


Figure 3. Face Encoding Extraction Scheme

FaceNet operates by mapping facial images into a compact 128-dimensional embedding space, where distances between embeddings correspond to degrees of facial similarity. This embedding-based representation enables robust face recognition even under variations in pose, illumination, and occlusion. The DeepFace library provides a high-level interface to FaceNet, handling internal preprocessing steps such as face alignment and normalization prior to feature extraction.

When no face is detected in an image, the function returns a null or empty embedding, and such images are subsequently assigned to a default cluster (cluster ID – 1) in the clustering stage. The *face_confidence* score returned by FaceNet reflects the model's certainty in detecting a valid face within the image region. Only embeddings with a confidence value exceeding a predefined threshold (0.51 in this study) are retained for subsequent clustering; images falling below this threshold are likewise assigned to the default cluster. This filtering step is critical to prevent noisy or low-quality face representations from distorting the cluster structure and, consequently, the classification signal derived from facial features.

2.4 Face Clustering

The hierarchical clustering was applied post-extraction to group memes based on similar facial features. Agglomerative hierarchical clustering was employed using Euclidean distance and Ward linkage, starting with each face encoding as its own cluster and iteratively merging the closest clusters to minimize within-cluster variance.

This method was selected because it does not require pre-specifying the number of clusters, making it suitable when the exact number of facial identities in memes is unknown. Additionally, it produces a hierarchical structure in the form of a dendrogram, enabling similarity analysis at various granularity levels—from nearly identical faces to those with general resemblance. Although this study does not explicitly analyze the dendrogram, this hierarchical representation offers potential for further exploration of facial similarity structures in memes.

To determine the optimal number of clusters, various clustering thresholds (t) were evaluated using metrics Comprehensive Indicator (CI), which combines Silhouette Score (SC), Calinski-Harabasz Index (CHI), and Davies-Bouldin Index (DBI) as defined by [26], which is shown in Equation 1.

$$CI = SC_N + CHS_N - DBI_N \quad (1)$$

$SC_N, CHS_N, and\ DBI_N$ are normalized SC, CHS, and DBI, which are obtained as shown in Equations 2-4.

$$SC_N = \frac{SC - SC_{min}}{SC_{max} - SC_{min}} \quad (2)$$

$$CHS_N = \frac{CHS - CHS_{min}}{CHS_{max} - CHS_{min}} \quad (3)$$

$$DBI_N = 1 - \frac{DBI - DBI_{min}}{DBI_{max} - DBI_{min}} \quad (4)$$

SC assesses how well each sample is placed in the correct cluster by comparing its proximity to its own cluster and the nearest other cluster. CHS measures the ratio of between-cluster to within-cluster variance, making it effective for statistically evaluating cluster separation. Meanwhile, DBI penalizes clusters that are not compact enough or insufficiently separated. Combining all three provides a more balanced evaluation between internal consistency and inter-cluster separation.

In Equation 1, CI is used to select the optimal clustering threshold. In Equations 2–4, SC, CHS, and DBI were computed for a given threshold t. The subscripts “min” and “max” denote the minimum and maximum values of each metric observed across all evaluated thresholds, used for min-max normalization into the range [0, 1]. Specifically, SC ranges from −1 to 1, where values closer to 1 indicate well-separated and cohesive clusters; CHS is a non-negative value where higher values reflect better-defined cluster boundaries; and DBI is a non-negative value where lower values indicate more compact and well-separated clusters. The DBI normalization in Equation 4 applies a subtraction from 1 so that a higher DBI_n corresponds to a lower (better) raw DBI score, making all three normalized components contribute positively to CI.

The evaluation identified an optimal threshold of 14.9, yielding 614 clusters. Images without detected faces or with *face_confidence* below the threshold (0.51) were assigned to a default cluster (-1). Finally, centroids for each cluster were computed from the constituent face encodings, excluding the default cluster. These centroids served as representative points and were used to assign clusters to face encodings in the test data. For each detected face in test images, the face encoding was compared against all pre-computed centroids, assigning the test data to the cluster with the minimum Euclidean distance. An example of the meme clustering results based on face encodings is shown in Figure 4.



(a) (b)

Figure 4. Example of Clustering Result (ID cluster: 377) in Training Data (a) and Testing Data (b)

2.5 Sarcasm Text Detection on Pretrained BERT

This sarcasm text detection model is built upon the ‘*bert-base-cased*’ architecture, augmented with additional layers including multiple Dense layers with ReLU activation and Dropout layers to prevent overfitting. The model is designed to classify sarcasm into four categories: ‘*not_sarcastic*’, ‘*general’*, ‘*twisted_meaning*’, and ‘*very_twisted*’, utilizing the ‘*sarcasm’* column from the training data.

After training, the model weights are saved and subsequently loaded into a replica reconstruction of the same architecture. From this replica model, the fine-tuned BERT component (‘*bert_model*’) is extracted and integrated as the BERT layer within the primary multimodal deep learning model. This approach enables the combined deep learning architecture to directly leverage fine-tuned BERT representations optimized for sarcasm text detection.

2.6 Text Encoding

This study utilizes the pretrained bert-base-cased BERT model, which leverages transfer learning from a massive English text corpus to generate rich linguistic representations. The encoding process (shown in Figure 5) begins with WordPiece tokenization. Following encoding, the input text is converted into input_ids (numeric token IDs) and token_type_ids (segment identifiers) for input to the BERT model. BERT processes these inputs to produce contextual embeddings for each token.

To obtain a fixed-size representation for the entire text, features are extracted from the BERT output: this study employs the [CLS] token representation, commonly used as a summary of the entire word sequence, along with mean and max pooling across all token embeddings to capture diverse aspects of textual meaning. These text features are then concatenated and fed into multiple dense layers, batch normalization, and dropout for further processing and dimensionality reduction.

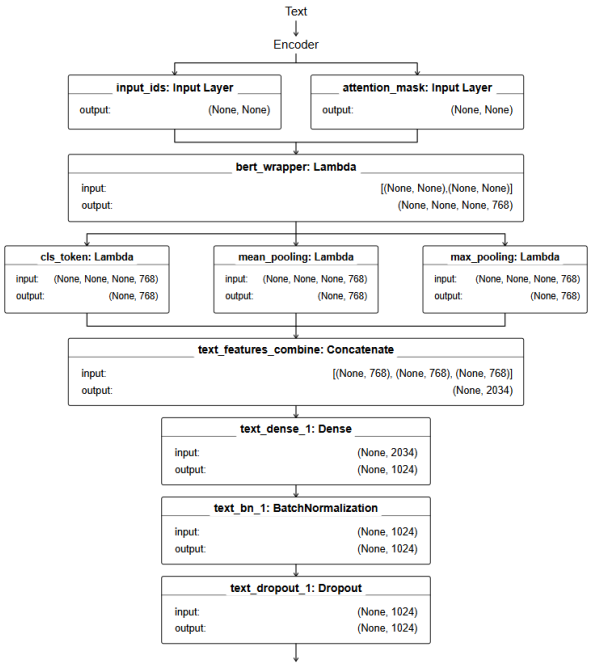


Figure 1. Text Encoding Process

2.7 Image Encoding

Figure 6 illustrates the workflow for image encoding, where image arrays are initially extracted using the pretrained DenseNet121 model with ImageNet weights to leverage learned visual features. To adapt the model to the specific task and Memotion dataset, the final layers of DenseNet121 were set to *trainable = True*, enabling fine-tuning on the Memotion images.

The next step applies global average pooling and global max pooling. These pooling operations reduce the spatial dimensions of feature maps, producing fixed-size vectors that capture the overall image content. Global average pooling aggregates the mean activations across each feature map, while global max pooling extracts the most prominent features.

The aggregated features are then concatenated and processed through dense layers, batch normalization, and dropout to refine representations, prevent overfitting, and enhance stable model learning. Each vector is encoded as a one-hot representation with a value of 1 at the position corresponding to the cluster label and 0 elsewhere. This categorical encoding ensures effective processing of cluster information by subsequent dense layers.

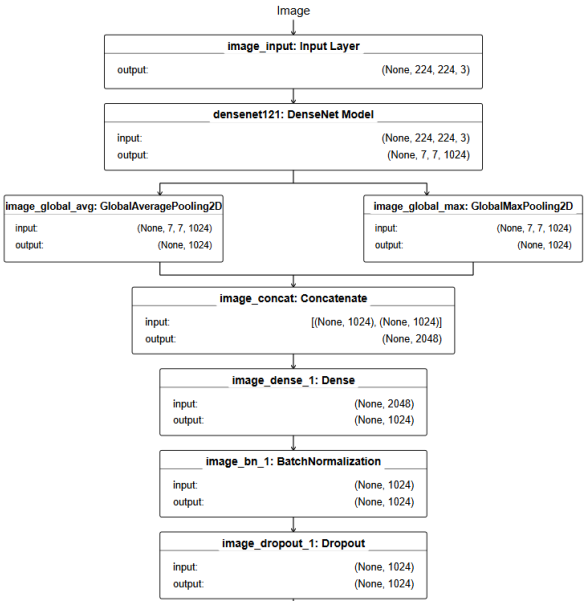


Figure 2. Image Encoding Process

2.8 Clustering Features: One-hot encoding

After each meme image is assigned to a cluster label, the label is converted to a one-hot encoding format to serve as input features for the combined deep learning model. One-hot encoding represents each cluster as a binary vector of dimension k, where k denotes the total number of clusters formed.

In this study, k corresponds to the 614 clusters identified during the hierarchical clustering stage (Section 2.4), plus one additional dimension for the default cluster (cluster ID –1), yielding a vector of length 615. For a meme assigned to cluster j, the one-hot vector contains a value of 1 at position j and 0 at all other positions. This sparse, fixed-length representation provides a structured and computationally efficient means of encoding cluster membership, enabling the subsequent dense layers to learn associations between facial identity groups and sentiment polarity classes.

2.9 Classification

The final step involves concatenating the three feature sets—text, image, and cluster—into a single comprehensive feature vector using a concatenation layer. This combined feature vector serves as input to the model's classification head, enabling predictions based on integrated insights from text, images, and clustering data.

The final dense layer produces a three-dimensional output corresponding to the three target sentiment classes: positive, neutral, and negative. A softmax activation function is applied to this output to convert raw logits into class probability distributions, where the predicted class is determined by the index with the highest probability. The model is trained end-to-end using the Focal Loss function (described in Section 2.11) to address class imbalance, with model weights optimized via the Adam optimizer using the learning

rate schedule defined by the ReduceLROnPlateau callback.

2.10 Evaluation Metrics

As proposed by the official evaluation of the Memotion challenge, we used the Macro-F1 score to measure the performance of sentiment polarity classification. To obtain a more comprehensive evaluation, the weighted average of Precision, Recall, and F1-score is also calculated. Weighted average accounts for the proportion of each class in the dataset, avoiding potential overestimation of model performance when it excels on the majority classes but performs poorly on minority classes. Additionally, this study evaluates per-class metrics by assessing Precision, Recall, and F1-score individually for each sentiment class.

Finally, the confusion matrix is visualized to better understand model performance across all sentiment classes. Model evaluation uses the confusion matrix—a table recording correct and incorrect predictions for each sentiment class, such as positive, negative, and neutral. For each class, metrics are computed using a one-vs-all approach, treating one class as positive. In these calculations, true positive (TP) represents data truly from that class correctly predicted by the model; false positive (FP) is data from other classes incorrectly predicted as that class; and false negative (FN) is data from that class predicted as another class. Precision as in Equation 5 is calculated as the ratio between TP and the total predictions directed to that class (TP + FP). Meanwhile, recall as shown in Equation 6 measures how many of the total actual labels were successfully identified correctly. F1 score as shown in Equation 7 is the harmonic mean between precision and recall, providing a balance between the two.

$$Precision = \frac{TP}{TP+FP} \quad (5)$$

$$Recall = \frac{TP}{TP+FN} \quad (6)$$

$$F1\ score = \frac{2\times Precision\times Recall}{Precision+Recall} \quad (7)$$

2.11 Experimental Setup

We only experimented with classifier models that combine text, image, and cluster features. The models used the following default hyperparameters: learning rate of 5e-6, default class weight setting, batch size of 32, and a maximum number of 100 training epochs.

The experiments were conducted using the Google Colab Notebook platform with the following specifications: 12-core processor, 83 GB RAM, and an NVIDIA A100-SXM4 GPU with 40 GB of GPU memory.

Focal Loss for Class Imbalance: To address the issue of class imbalance in the Memotion dataset, we employed Focal Loss as shown in Equation 8, which is designed to reduce the impact of well-classified samples and focus learning on difficult or minority class examples. Unlike standard categorical cross-entropy, Focal Loss modifies the loss by a modulating factor $(1-p_t)^{\gamma}$, where p_t is the predicted probability for the true class, and γ is a focusing parameter.

$$FL(p_t) = -\alpha_t(1-p_t)^{\gamma}\log(p_t) \quad (8)$$

In our implementation, we set $\alpha_t = 1.0$ and $\gamma = 2.0$, which are commonly adopted values in prior works. This setup allowed the model to emphasize learning on misclassified or minority class samples without requiring manual resampling or complex class weighting.

Training Callbacks: To optimize the training process and ensure robust model convergence, we employed several training callbacks. The ModelCheckpoint callback was used to save the model's weights whenever an improvement in the validation Macro-F1 score was observed, ensuring that the best-performing model on the validation set is retained. To prevent overfitting and reduce training time, EarlyStopping was applied by monitoring the same validation Macro-F1 score, stopping training if no improvement was observed for 10 consecutive epochs and restoring the best weights thereafter. Additionally, the ReduceLROnPlateau callback was employed to dynamically adjust the learning rate, reducing it by a factor of 0.75 whenever the validation Macro-F1 score plateaued for 5 consecutive epochs, with a minimum learning rate of 1e-6 to avoid stalling the training process.

3. Results and Discussions

The experiments aim to highlight the performance comparison between the baseline model using the standard BERT and the model enhanced with sarcasm-trained BERT.

3.1 Testing: Validation Split Ratio

For the experiments, we tried various validation splits to assess the model's robustness and performance consistency. The validation split ratios used were 0.10, 0.15, 0.20, and 0.25. These variations were chosen to observe how the different proportions of training and validation data affect the model's ability to generalize the classification.

The validation split experiment results in Table 2 demonstrate that the validation split ratio significantly impacts model performance. The best Macro-F1 scores for the sarcasm model and baseline model were achieved with a 0.25 validation split, yielding 0.3047 and 0.3095, respectively.

Table 2. Macro-F1 Scores for Validation Split Testing

Validation Split Ratio	BERT+sarcasm	Base BERT
0.1	0.2786	0.2460
0.15	0.2198	0.2226
0.2	0.2821	0.2479
0.25	0.3047	0.3095

These findings indicate that allocating a larger proportion of data to validation (25%) enhances the

model's generalization ability, likely due to more reliable validation during training. In contrast, smaller splits like 0.15 resulted in poorer performance, attributable to insufficient validation or instability in the learning dynamics.

3.2 Testing: Learning Rate

To optimize model performance, experiments were conducted on learning rate settings using values of 5e-6, 1e-5, and 2e-5 to assess their impact on overall performance. The results in Table 3 confirm the critical role of learning rate in model optimization. Among the tested values, a learning rate of 5e-6 yielded the highest Macro-F1 scores for both models: 0.3047 for the sarcasm model and 0.3095 for the baseline model. A smaller learning rate of 1e-6 resulted in substantially lower performance, particularly for the baseline model (Macro-F1 of 0.1289), indicating overly slow learning and potential entrapment in suboptimal loss landscape regions. Conversely, higher rates like 1e-5 and 2e-5 led to performance degradation, suggesting unstable convergence or overshooting during training. These outcomes underscore the importance of fine-tuning the learning rate, especially in multimodal architectures integrating complex features from text, images, and clusters.

Table 3. Macro-F1 Scores for Learning Rate Testing

Learning Rate	BERT+sarcasm	Base BERT
1E-6	0.2082	0.1289
5E-6	0.3047	0.3095
1E-5	0.2697	0.2750
2E-5	0.1978	0.2107

3.3 Testing: Class Weight Strategy

There are 3 options of class weight strategies used, including *default* class weight, computationally '*balanced'* class weight, and class weight based on '*class distribution' ratio*. Class weights are set to balanced compute weights inversely proportional to class frequencies, assigning higher weights to minority classes. In contrast, class weights based on class distribution ratios calculate weights using the relative frequency or proportion of each class within the entire dataset.

The results in Table 4 indicate that the choice of class weighting strategy substantially affects model performance. For both the sarcasm and baseline models, the highest Macro-F1 scores—0.3047 and 0.3095, respectively—were achieved using the default class weight setting.

Applying a balanced strategy led to significant performance degradation in both models. Similarly, the class distribution approach, where weights reflect the ratio of majority to minority class proportions, produced mixed outcomes: the baseline model showed a slight improvement over balanced (0.2778 vs. 0.2454), while the sarcasm model experienced further decline.

These findings suggest that in this multimodal sentiment classification context, manual reweighting strategies may introduce instability or overcompensation for minority classes. The default uniform weighting across samples yields the most stable results and effective learning dynamics for both models

Table 4. Macro-F1 Scores for Class Weight Testing

Class Weight	BERT+sarcasm	Base BERT
Default	0.3047	0.3095
Balanced	0.2331	0.2454
Class Distribution	0.2237	0.2778

3.4 Confusion Matrix

We use the optimal parameters determined through the experiments described above. As shown in the results, the base model achieved a slightly higher Macro-F1 score of 0.3095 compared to the sarcasm model's 0.3047, with overall accuracy scores of 0.3488 and 0.3738, respectively. Despite the minor gap in Macro-F1, a deeper look at the confusion matrices reveals important behavioral differences.

As can be noticed in Figures 7 and 8, the sarcasm model considerably lowers false negatives, particularly for positive samples. It misclassifies only 107 positive instances as negative, compared to 284 in the base model. This shows improved sensitivity to positive sentiment when sarcasm is involved. This is further supported by the sarcasm model's higher per-class F1 score for the positive class (0.3966 vs. 0.3766), despite a slightly lower overall Macro-F1. Additionally, the sarcasm model produces a better balance in class performance. It achieves the highest F1 score for the neutral class (0.4067) and displays a fairly even macro recall of 0.3348. Although achieving a slightly better Macro-F1, the base model shows greater imbalance in the negative class with a recall of 0.2717 but low precision (0.1061), indicating many false positives. A summary of per-class performance metrics is presented in Table 5.

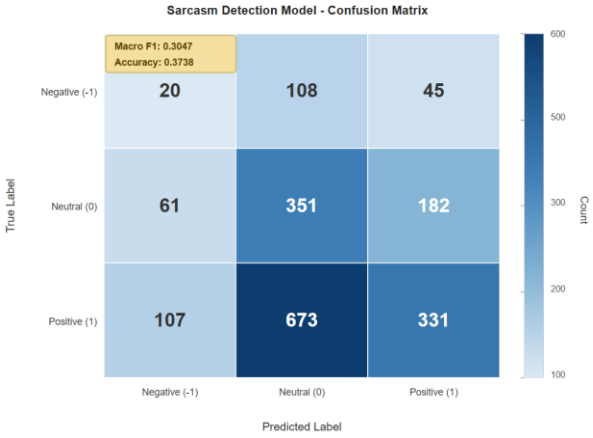


Figure 3. Confusion matrix for classification model with BERT + sarcasm

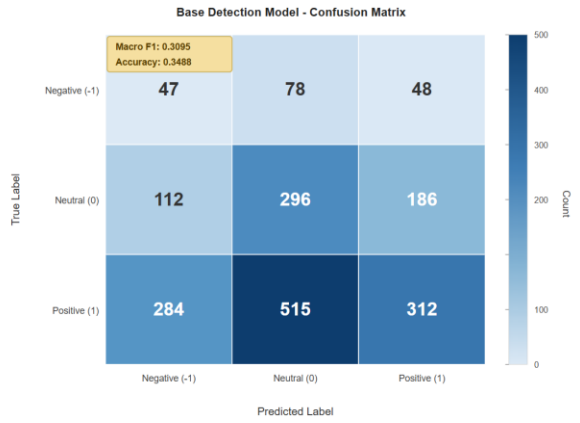


Figure 4. Confusion matrix for a classification model with base BERT

Table 5. Summary of per-class performance metrics

Sentiment	BERT+Sarcasm			Base BERT		
	P	R	F1	P	R	F1
Negative	**0.1064**	0.1156	0.1108	0.1061	**0.2717**	**0.1526**
Neutral	0.3101	**0.5909**	**0.4067**	**0.3330**	0.4983	0.3992
Positive	**0.5932**	**0.2979**	**0.3966**	0.5714	0.2808	0.3766

In terms of weighted metrics (Table 6), the sarcasm model also yields slightly better results: Weighted F1 = 0.3735 (vs. 0.3631), and Weighted Recall = 0.3738 (vs. 0.3488). These findings suggest that while the base model performs marginally better in terms of overall Macro-F1, the sarcasm model demonstrates improved robustness, especially in correctly detecting positive sentiment and reducing false negatives in sarcastic contexts. This trade-off highlights the potential benefit of integrating sarcasm-specific understanding in sentiment classification tasks involving nuanced or ambiguous language, such as internet memes.

Table 6. Comparison of Overall Evaluation Metrics

Metric	BERT+Sarcasm	Base BERT
Macro F1	0.3047	**0.3095**
Accuracy	**0.3738**	0.3488
Weighted F1	**0.3735**	0.3631
Weighted Precision	**0.4588**	0.4531
Weighted Recall	**0.3738**	0.3488

3.5 Discussion: Error Analysis

To gain deeper insights into the limitations of the sarcasm model, an analysis was conducted on samples that were misclassified across each class.

In the positive class, the sarcasm model exhibited notable misclassification errors, with 673 samples incorrectly classified as neutral and 107 as negative. Analysis of the model's confidence scores revealed that errors toward the neutral class often occurred with relatively small confidence differences (delta) between positive and neutral predictions. The average delta confidence for these errors was 0.3002, with a wide distribution (standard deviation of 0.1942), indicating frequent model ambiguity in distinguishing between the two classes.

Figure 9 presents a sample meme misclassified as neutral sentiment. The image depicts a puppy wrapped in a blanket in a bathroom, accompanied by the text *"I found a seal in the bathroom. Doesn't like fish but loves pepperoni pizza"*. The combination of an adorable visual and absurdly humorous text creates a lighthearted, enjoyable effect that intuitively suggests positive sentiment. However, the model classified this meme as neutral, despite the positive class confidence being only slightly lower than neutral by a margin of 0.00495. This razor-thin confidence difference highlights the model's high ambiguity and failure to convincingly capture the positive context, likely due to the absence of explicit textual indicators typically associated with positive emotions and limitations in understanding the visual contribution to sentiment formation.



Figure 9. Example of Positive to Neutral Sentiment Misclassification

Figure 10 shows a sample meme featuring a child with prominent ears and a handsome adult man wearing headphones, which carries positive sentiment but was predicted as negative by the model. This error stems from the model's failure to grasp the visual humor context. Instead of interpreting it as a positive narrative of self-acceptance and transformation, the model fixated on potentially negative elements. It placed heavy emphasis on the text "Let me see your ears," which could be construed as mockery or an offensive request. Additionally, the image of the child with protruding ears was treated as a negative cue. This is reflected in the confidence scores: the highest for negative (0.4066), followed closely by the true positive class (0.4037), and neutral (0.1897). The minuscule gap between negative and positive classes (delta ≈ 0.0029) indicates the model was on the verge of selecting the correct class but ultimately tilted negative due to a literal interpretation of visual and textual elements.



Figure 10. Example of Positive to Negative Sentiment Misclassification

Based on the delta confidence data, neutral-to-positive misclassifications occurred more frequently (182 cases) than neutral-to-negative ones (61 cases). This indicates a model bias toward interpreting neutral content as positive. The average delta confidence for positive

errors (0.212) was slightly higher than for negative errors (0.197), suggesting marginally stronger model conviction in positive sentiment over negative in these instances.



Figure 11. Example of Neutral to Positive Sentiment Misclassification

Figure 11 depicts a neutral meme sample predicted as positive by the model. The image shows Spider-Man hanging upside down with the text "*Mommy come quick! I think I saw a spider*." The text conveys ironic humor, as Spider-Man—whose powers derive from a spider—is afraid of spiders. However, from a more objective viewpoint, the model's positive classification could be deemed reasonable. The presence of the superhero Spider-Man inherently carries strong positive connotations, irrespective of the textual narrative.

Moreover, the irony arising from the contradiction between Spider-Man and his spider phobia can be seen as an element rendering the meme amusing and entertaining, a form of positive sentiment. Thus, the model likely did not misinterpret the sentiment but instead prioritized the positive weight associated with the popular character and the resulting humorous effect, overriding the narrative's neutrality.



Figure 12. Example of Neutral to Negative Sentiment Misclassification

Figure 12 presents another meme sample classified as negative by the model: Spider-Man emerging from a wardrobe with the text "*Fuck You Narnia!*" This error arises from the model's failure to comprehend the broader visual context, fixating instead on the explicitly negative wording. The word "*Fuck*" carries a strongly negative connotation, prompting the model to assign it overwhelming weight. Consequently, it overlooked the visual cue of Spider-Man exiting the wardrobe. The humor stems from the unexpected fusion of the popular superhero with a reference to *The Chronicles of Narnia*, where the wardrobe serves as a portal to another world. Lacking cultural contextual knowledge, the model could not grasp that Spider-Man is attempting to enter Narnia, with the joke arising from the clash of fictional universes. Thus, it fixated solely on textual negativity, leading to an erroneous classification despite the meme's overall neutral humor.

Negative-to-neutral prediction errors occurred more frequently than negative-to-positive ones. This suggests the model tends to overlook latent negative sentiments in memes. The average delta for negative-to-neutral errors (0.358) was substantially higher than for negative-to-positive errors (0.283), indicating stronger model conviction in classifying negative memes as neutral.



Figure 13. Example of Negative to Positive Sentiment Misclassification

Figure 13 shows a sample negative-class meme misclassified by the model as positive. The meme features the text "*ENEMY IS SLEEPING. TIME TO HANGOUT.*" alongside the cartoon characters from *Tom and Jerry*. Objectively, the model's positive classification can be viewed as accurate due to the humor that reframes negativity into positivity. The phrase "*ENEMY IS SLEEPING*" carries a negative connotation referencing an "enemy," but it is immediately countered by the upbeat "*TIME TO HANGOUT.*" This contrast generates humor that subverts the original negative meaning. Furthermore, the image of peacefully sleeping Tom and mischievously smiling Jerry adds a layer of playful cheerfulness, reinforcing positive sentiment. The model likely succeeded in capturing this sentiment shift from a sleeping foe's threat to an opportunity for fun, interpreting the meme holistically as positive.



Figure 14. Example of Negative to Neutral Sentiment Misclassification

Figure 14 features a man holding chalk in front of a whiteboard with the text "*IF YOUR FRIENDS DIDN'T WISH YOU HAPPY FRIENDSHIP DAY, YOU HAVE AMAZING FRIENDS.*" The ground truth negative label stems from the joke's core, conveying social rejection and loneliness: the punchline "*you have amazing friends*" is sarcastic, implying the opposite—that if friends skip Friendship Day wishes, you likely lack true friends. The model likely predicted neutral due to conflicting signals in the meme. It contains strong positive keywords like "*friends*," "*friendship*," "*happy*," and "*amazing*," which bias scores toward positivity.

However, the model failed to detect sarcasm in "you have amazing friends," preventing reconciliation of positive words with the underlying negative themes.

When faced with ambiguous or contradictory signals—such as positive keywords paired with neutral subtext or visuals lacking explicit sentiment support—the model defaults to neutral with high confidence. This occurs not from subtle balance but from interpreting ambiguity as the absence of strong sentiment, opting for the "*safe*" neutral choice. This is evidenced by the model's predictions: neutral confidence of 0.762441, far exceeding negative (0.114548) and positive (0.123011).

3.6 Comparison with Previous Research

To contextualize these results within the broader landscape of meme sentiment classification research, we present a comprehensive comparison with recent methods. Table 7 presents a comprehensive comparison of our proposed approach with recent state-of-the-art methods for multimodal meme sentiment classification. The comparison focuses on studies that used the SemEval 2020 Task 8 (Memotion) dataset or similar meme sentiment benchmarks, highlighting differences in methodology and main contribution.

Table 7. Comparison with Recent Methods for Multimodal Meme Sentiment Classification

Study	Dataset	Method/Model	Main Contribution
[19]	Memotion	Traditional ML+CNN	Benchmark dataset creation
[20]	Memotion	BERT+DenseNet	Multimodal concatenation
[31]	Memotion Task A	VGG-16/Inception-V3 + LSTM/BERT	Multimodal fusion comparison
[14]	Memotion	CNN + LSTM	Combined CV and NLP features
[13]	Memotion	Multimodal + spatial encoding + face embeddings	Spatial and facial feature integration
[32]	Memotion 2.0	BERT + EfficientNetV4 + CLIP	Multitask multimodal transformer
Ours	Memotion	BERT (sarcasm-trained) + DenseNet121 + Face Clusters	explicit sarcasm integration, improved positive/neutral detection

3.7 Limitations and Future Research Directions

This study effectively integrates sarcasm detection into multimodal meme sentiment classification but has several key limitations suggesting future directions. This study relies on the ~7,000 English memes from SemEval 2020 Task 8, limiting generalizability across languages, cultures, and formats. Class imbalance impacts macro-F1 despite sarcasm-aware training, where balanced datasets or oversampling could help. Performance hinges on sarcasm dataset quality for BERT pre-training, potentially missing context-dependent or cultural sarcasm variations, with optimal data size untested. Face encoding applies only to memes with faces, reducing impact, and lacks facial expression analysis for richer context. Modest macro-F1 (0.3047) highlights ongoing minority class challenges requiring further imbalance strategies.

4. Conclusions

Overall, this study successfully developed an automatic sentiment polarity classification system tailored to the Memotion dataset using multimodal architecture. By integrating textual information (with sarcasm text detection applied to BERT), visual content (via DenseNet121), and facial clustering features, the model was able to capture the nuances and implied emotions embedded within the memes. Experimental results indicate that the BERT+Sarcasm outperforms baseline BERT across weighted metrics—Accuracy (0.3738 vs. 0.3485), Weighted F1 (0.3735 vs. 0.3631), Precision (0.4588 vs. 0.4531), and Recall (0.3738 vs. 0.3488)—with only a marginal Macro F1 trade-off (0.3047 vs. 0.3095), validating its efficacy for sarcasm detection. Despite this, the sarcasm model showed stronger performance in classifying both positive and neutral sentiments, and in minimizing false negatives. Nevertheless, the overall performance indicates that there is still potential for improvement. Challenges persist in accurately identifying negative sentiment, and the impact of dataset imbalance remains noticeable. Future work could explore more diverse datasets, contrastive training techniques, or context-aware modeling frameworks that better capture humor, irony, and cultural references prevalent in memes.

References

[1] S. Tabatabaei and Elena Anatolievna Ivanova, "The Role of Memes on Emotional Contagion," *Elementary Education Online*, vol. 20, no. 5, pp. 6028–6036, 2021, doi: 10.17051/ilkonline.2021.05.676.

[2] M. Velmala, S. Rajiakodi, K. Pannerselvam, and B. Sivagnanam, "Multimodal Sentiment Analysis of Online Memes: Integrating Text and Image Features for Enhanced Classification," in *Procedia Computer Science*, Elsevier, Jan. 2025, pp. 355–364. doi: 10.1016/J.PROCS.2025.04.272.

[3] M. Hazman, S. McKeever, and J. Griffith, "What Makes a Meme a Meme? Identifying Memes for Memetics-Aware Dataset Creation," *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 19, pp. 745–759, Jul. 2024, doi: 10.1609/icwsm.v19i1.35843.

[4] R. K. Routhu and U. Baruah, "Sentiment Analysis on Memes: A Review," *Expert Syst.*, vol. 42, no. 11, p. e70133, Nov. 2025, doi: https://doi.org/10.1111/exsy.70133.

[5] C. Ling, I. Abuhilal, J. Blackburn, E. De Cristofaro, S. Zannettou, and G. Stringhini, "Dissecting the Meme Magic: Understanding Indicators of Virality in Image Memes," in *Proceedings of the ACM on Human-Computer Interaction*, Association for Computing Machinery, Apr. 2021. doi: https://doi.org/10.1145/3449155.

[6] M. S. Hee, R. Cao, T. Chakraborty, and R. K. W. Lee, "Understanding (Dark) Humour with Internet Meme Analysis," in *WWW 2024 Companion - Companion Proceedings of the ACM Web Conference*, Association for Computing Machinery, Inc, May 2024, pp. 1276–1279. doi: https://doi.org/10.1145/3589335.3641249.

[7] L. Bonheme and M. Grzes, "SESAM at SemEval-2020 Task 8: Investigating the Relationship between Image and Text in Sentiment Analysis of Memes," *14th International Workshops on Semantic Evaluation, SemEval 2020 - co-located 28th International Conference on Computational Linguistics, COLING 2020, Proceedings*, pp. 804–816, 2020, doi: 10.18653/V1/2020.SEMEVAL-1.102.

[8] Z. Zhao *et al.*, "MemeReaCon: Probing Contextual Meme Understanding in Large Vision-Language Models," in *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, Suzhou, China: Association for

Computational Linguistics (ACL), Nov. 2025, pp. 3559–3582. doi: 10.18653/V1/2025.EMNLP-MAIN.176.

[9] S. Sharma *et al.*, "Detecting and Understanding Harmful Memes: A Survey," *IJCAI International Joint Conference on Artificial Intelligence*, vol. 6, pp. 5597–5606, Jul. 2022, doi: 10.24963/IJCAI.2022/781.

[10] K. P. N. Nguyen and V. Ng, "Computational Meme Understanding: A Survey," *EMNLP 2024 - 2024 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, pp. 21251–21267, 2024, doi: 10.18653/V1/2024.EMNLP-MAIN.1184.

[11] L. Bojić *et al.*, "Comparing large Language models and human annotators in latent content analysis of sentiment, political leaning, emotional intensity and sarcasm," *Scientific Reports 2025 15:1*, vol. 15, no. 1, pp. 11477-, Apr. 2025, doi: 10.1038/s41598-025-96508-3.

[12] A. Konyspay, P. Shamoi, M. Ziyada, and Z. Smambayev, "Meme Similarity and Emotion Detection using Multimodal Analysis," in *2025 International Conference on Activity and Behavior Computing (ABC)*, Al Ain, United Arab Emirates: IEEE, Mar. 2025. doi: https://doi.org/10.1109/ABC64332.2025.11118558.

[13] M. Hazman, S. McKeever, and J. Griffith, "Meme Sentiment Analysis Enhanced with Multimodal Spatial Encoding and Face Embedding," *Communications in Computer and Information Science*, vol. 1662 CCIS, pp. 318–331, 2023, doi: 10.1007/978-3-031-26438-2_25/TABLES/6.

[14] S. Gundapu and R. Mamidi, "Gundapusunil at SemEval-2020 Task 8: Multimodal Memotion Analysis," *14th International Workshops on Semantic Evaluation, SemEval 2020 - co-located 28th International Conference on Computational Linguistics, COLING 2020, Proceedings*, pp. 1112–1119, 2020, doi: 10.18653/V1/2020.SEMEVAL-1.147.

[15] S. Chi, G. Sang, and X. Shi, "Research on Multimodal Sentiment Classification of Internet Memes Based on Transformer," in *ACM International Conference Proceeding Series*, Association for Computing Machinery, Jan. 2024, pp. 445–450. doi: https://doi.org/10.1145/3673277.3673354.

[16] E. Hossain, O. Sharif, and M. Moshiul Hoque, "MemoSen: A Multimodal Dataset for Sentiment Analysis of Memes," in *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, Marseille, France: European Language Resources Association, Jun. 2022, pp. 1542–1554. Accessed: Jan. 21, 2026. [Online]. Available: https://aclanthology.org/2022.lrec-1.165/

[17] Y. Duan, X. Zhao, and H. Guo, "Sentiment-aware cross-modal semantic interaction model for harmful meme detection," *Decis. Support Syst.*, vol. 196, p. 114509, Sep. 2025, doi: 10.1016/J.DSS.2025.114509.

[18] T. H. Afridi, A. Alam, M. N. Khan, J. Khan, and Y. K. Lee, "A Multimodal Memes Classification: A Survey and Open Research Issues," *Lecture Notes in Networks and Systems*, vol. 183, pp. 1451–1466, 2021, doi: 10.1007/978-3-030-66840-2_109.

[19] C. Sharma *et al.*, "SemEval-2020 Task 8: Memotion Analysis-the Visuo-Lingual Metaphor!," *14th International Workshops on Semantic Evaluation, SemEval 2020 - co-located 28th International Conference on Computational Linguistics, COLING 2020, Proceedings*, pp. 759–773, 2020, doi: 10.18653/V1/2020.SEMEVAL-1.99.

[20] Y. Guo, J. Huang, Y. Dong, and M. Xu, "Guoym at SemEval-2020 Task 8: Ensemble-based Classification of Visuo-Lingual Metaphor in Memes," *14th International Workshops on Semantic Evaluation, SemEval 2020 - co-located 28th International Conference on Computational Linguistics, COLING 2020, Proceedings*, pp. 1120–1125, 2020, doi: 10.18653/V1/2020.SEMEVAL-1.148.

[21] A. Maladry, E. Lefever, C. Van Hee, and V. Hoste, "The limitations of irony detection in Dutch social media," *Language Resources and Evaluation 2023 58:4*, vol. 58, no. 4, pp. 1355–1386, Jul. 2023, doi: 10.1007/S10579-023-09656-1.

[22] R. A. Potamias, G. Siolas, and A. G. Stafylopatis, "A transformer-based approach to irony and sarcasm detection," *Neural Computing and Applications 2020 32:23*, vol. 32, no. 23, pp. 17309–17320, Jun. 2020, doi: 10.1007/S00521-020-05102-3.

[23] C. Mello, · Gullal, S. Cheema, · Gaurish Thakkar, G. S. Cheema, and G. Thakkar, "Combining sentiment analysis classifiers to explore multilingual news articles covering London 2012 and Rio 2016 Olympics," *International Journal of Digital Humanities 2022 5:2*, vol. 5, no. 2, pp. 131–157, Nov. 2022, doi: 10.1007/S42803-022-00052-9.

[24] N. Novielli *et al.*, "Assessment of off-the-shelf SE-specific sentiment analysis tools: An extended replication study," *Empirical Software Engineering 2021 26:4*, vol. 26, no. 4, pp. 77-, Jun. 2021, doi: 10.1007/S10664-021-09960-W.

[25] U. Naseem, I. Razzak, P. Eklund, and K. Musial, "Towards Improved Deep Contextual Embedding for the identification of Irony and Sarcasm," *Proceedings of the International Joint Conference on Neural Networks*, Jul. 2020, doi: 10.1109/IJCNN48605.2020.9207237.

[26] X. Guo, J. Ma, and A. Zubiaga, "Cluster-based deep ensemble learning for emotion classification in Internet memes," *J. Inf. Sci.*, vol. 51, no. 1, pp. 265–283, Feb. 2025, doi: https://doi.org/10.1177/01655515221136241.

[27] W. Chen, F. Lin, G. Li, and B. Liu, "A survey of automatic sarcasm detection: Fundamental theories, formulation, datasets, detection methods, and opportunities," *Neurocomputing*, vol. 578, p. 127428, Apr. 2024, doi: 10.1016/J.NEUCOM.2024.127428.

[28] S. Farabi, T. Ranasinghe, D. Kanojia, Y. Kong, and M. Zampieri, "A Survey of Multimodal Sarcasm Detection," *IJCAI International Joint Conference on Artificial Intelligence*, pp. 8020–8028, 2024, doi: 10.24963/IJCAI.2024/887.

[29] P. Parameswaran, A. Trotman, V. Liesaputra, and D. Eyers, "BERT's The Word : Sarcasm Target Detection using BERT," in *Proceedings of the 19th Annual Workshop of the Australasian Language Technology Association*, Australasian Language Technology Association, Dec. 2021, pp. 185–191. Accessed: Jan. 21, 2026. [Online]. Available: https://aclanthology.org/2021.alta-1.21/

[30] I. N. Y. Sumadewa, A. Anggrawan, H. Hairani, I. P. Hariyadi, C. Satria, and D. S. C. Saputri, "Emotion Classification on Customer Reviews of Drinking Water Services Using IndoBERT and Machine Learning Algorithms," in *2025 7th International Conference on Cybernetics and Intelligent System (ICORIS)*, Mataram, Indonesia: Institute of Electrical and Electronics Engineers (IEEE), Dec. 2025, pp. 1–5. doi: 10.1109/icoris67789.2025.11296077.

[31] A. Baruah, K. A. Das, F. A. Barbhuiya, and K. Dey, "IIITG-ADBU at SemEval-2020 Task 8: A Multimodal Approach to Detect Offensive, Sarcastic and Humorous Memes," *14th International Workshops on Semantic Evaluation, SemEval 2020 - co-located 28th International Conference on Computational Linguistics, COLING 2020, Proceedings*, pp. 885–890, 2020, doi: 10.18653/v1/2020.semeval-1.112.

[32] A.-M. Bucur, A. Cosma, and I.-B. Iordache, "BLUE at Memotion 2.0 2022: You have my Image, my Text and my Transformer," *CEUR Workshop Proc.*, vol. 3168, Apr. 2022, Accessed: Feb. 26, 2026. [Online]. Available: http://arxiv.org/abs/2202.07543