SENTIMEMENET: A DEEP LEARNING APPROACH FOR MEME/NON-MEME CLASSIFICATION AND SENTIMENT ANALYSIS IN INTERNET MEMES

An Artificial Neural Networks Final Project

Presented to Mr. John Cristopher Mateo

Faculty, College of Information and Communications Technology

West Visayas State University

La Paz, Iloilo City

In Partial Fulfillment

of the Requirements for the Course

CCS 248: Artificial Neural Networks

By

Lean Vince A. Cabales

Zuriel Eliazar D. Calix

Jessie Loraine P. Porras

BSCS 3-B AI

December 2025

Table of Contents

Section 1. Introduction and Problem Definition

This section details the problem that the project is aiming to solve, justifications for its relevance to real-world applications, current existing solutions, and impact.

1.1 Project Overview

The goal of this project is to train two complementary deep learning models for comprehensive meme analysis. One model will be trained on a chosen dataset for a deep learning binary classification. This first model will be used to distinguish between memes and non-memes using a Mini-ResNet architecture. Following the initial classification, the identified memes will be fed into another model for sentiment extraction. This multimodal deep learning model performs fine-grained context and sentiment analysis and classifies memes based on sentiment contexts: humor, motivational, offensive, and sarcasm.

1.2 Problem Selection and Justification

In this modern war for online attention, memes have become the cultural implication for viralness. From personal blogs and pages to business channels and even school organizations, every social media user aims to ride this bandwagon to gain fame and not feel left out (FOMO). Studies show that 35% of young people use memes as a tool to shed light on complicated emotions (Thinkhouse, n.d.). Meanwhile, brands harness this new marketing trend to make their digital presence relatable, attractive, shareable, and culturally relevant.

One widely circulated use of memes on social media platforms today is in political media. In the Philippines, memes have been used by citizens as a grassroots medium for sharing political ideas (Halversen & Weeks, 2023). Mostly, the intention is to make fun of or expose politicians, using humor to disparage them or vent out people's frustrations and criticisms towards their failings as political leaders. Meme sharers support these meme trends often with the intention to raise awareness about political issues, for instance, and persuade others of their own views and perspectives.

While memes are used as a form of freedom of expression and are commonly for fun and entertainment, the use of memes, such as these political memes, makes way for several potential

negative consequences. Halversen and Weeks (2023) stated that exposure to political memes can promote anger and polarization. Humor is a more powerful tool for persuasion, and it can fuse laughter with politically oriented anger. This kind of media is also seen as frivolous and lacking in the capacity to communicate sophisticated ideas, making it a spreader of political misinformation. Scholars are even concerned that the relevance and simplicity of meme humor might restrict meaningful discourses.

The ambiguity of memes presents significant challenges for content moderation in social media platforms, especially since the Philippines does not have a strong media regulation protocol for monitoring social media content for minors and kids. This underscores the need for correctly analyzing the sentiment embedded in memes to identify nuances like sarcasm, hate speech, and other content, and potentially prevent false positives to create a fairer and more accurate platform governance. Furthermore, developing a model capable of fine-grained analysis can help brands and researchers understand meme trends, which they can use to improve their audience targeting and digital strategy.

Current solutions related to sentiment analysis on memes involve the fusion of image and text, treating them as multimodal classification problems. Sentiment analysis is mostly used to detect hate posts within social media platforms. Meta AI, for example, has released the Hateful Memes dataset as part of its initiative in creating an effective tool for detecting multimodal hate speech. Their research community focuses on developing tools that can extract context (image and text) within a meme and fuse them during classification to analyze the sentiment (Meta, n.d.). Similarly, Gundapu & Mamidi (2020) have developed a SemEval-2020 Task 8, a multimodal memotion analysis which classifies internet memes as positive, negative, and neutral. Their model further analyzed the type of humor expressed within the meme. Its system integrates a Convolutional Neural Network (CNN) for image understanding and a Long Short-Term Memory (LSTM) network for text sequence analysis. Other studies focus on subtle visual cues (faces, object context, layout) and expand to non-English or under-resourced languages, including those within the Philippine context. Amper & Baradillo (2025) conducted a multimodal sentiment analysis of memes in the course of the Philippine elections. Their study utilized a VADER-based NLP tool to interpret meme text alongside the visual context.

The multimodal nature of internet memes, where both the image and text must be processed and fused together to correctly classify them, makes this problem fundamentally suited to Deep Neural Networks. With these, SentimemeNet will use a combination of specialized DNN models working in sequence to address the central challenge of ambiguity in meme meaning. Our goal is to develop a comprehensive, multi-layered computational tool for memes capable of performing fine-grained, context-specific sentiment analysis on memes. This project hopes to contribute to the ongoing worldwide effort of addressing meme ambiguity in social media platforms.

Section 2. Data Collection and Preparation

This section details the selection, sources, and validation process for the datasets used to train SentimemeNet.

2.1 Dataset Choice and Source

SentimemeNet is trained using a composite strategy that combines several meme and non-meme sources for its two core classification tasks: Binary Detection (Stage 1) and Fine-Grained Sentiment Analysis (Stage 2).

2.1.1 Stage 1: Meme Detection Dataset (Meme vs Non-Meme)

For the binary meme versus not meme classification of SentimemeNet, three (3) different meme datasets were used: the Memotion Dataset 7k, the unlabeled test images from the Gyanendra Memotion dataset, and a large Reddit Memes collection to provide diverse and real-world examples of meme visual and textual patterns. This is combined with a non-meme dataset (Caltech-256) to supply visually diverse negative samples.

2.1.2 Stage 2: Sentiment Analysis Dataset

For the sentiment-related components, two comprehensive multimodal meme datasets related to the Memotion Analysis tasks from Kaggle, the world's largest and most active data science community, were used. Both of the datasets provide the necessary images and corresponding labels required for multimodal training.

The first dataset, officially known as "Memotion Dataset 7k", is a carefully curated collection of memes for sentiment and emotion. It has been systematically annotated for sentiment and emotion analysis. For this reason, it is a lot more suitable for research that investigates how memes express sentiments. The dataset is part of the shared task SemEval-2020 Task 8: Memotion Analysis, that is focused on computational analysis of internet memes (Sharma et al., 2020). The collection contains approximately 8,000 memes, all

annotated across three different analytical dimensions: classification (Task A), Humor Type Classification (Task B), and Intensity/Scale of Each Humor Type (Task C). The second dataset, “Memotion”, released by Gyanendra Das, includes additional meme images and their corresponding annotations. This serves as an extension of the original SemEval dataset and provides more labeled examples for the sentiment-based tasks.

2.2 Data Validation, Balancing, and Pre-processing

For the first model of SentimemeNet, the dataset was loaded from two folders — meme/ and not_meme/ —- and reads all the filenames with typical image extensions (jpg, png, jpeg). There are no other labels stored elsewhere because the folder names already act as the labels: meme is label 1 and not_meme is label 0. Then, the dataset is validated using several validation checks:

1. Checks if folders exist - verifies that meme and not_meme directories exist.
2. Check the number of images in each class - validates whether the dataset is usable and whether class imbalance exists.
3. Checks for imbalance - it compares the class sizes and checks for imbalance.

If one class has more images than the other, that class is downsampled using random sampling. A fixed seed (42) is then used to ensure that both classes have equal size before training, which is important for binary classification.

During the dataset preparation for model one, the maximum images per class were initialized to only 10000. Special configurations were made to accurately scan and collect the images in nested folders and filter out all invalid or corrupted images that cannot be used for training. The results of the dataset preparation (Table 1) show that the meme folder initially has 11780 valid image files and 37 invalid ones. On the other hand, the not_meme folder contains 61214 image files with no invalid images detected. Since both folders exceed the maximum images per class limit (10000) and do not have an equal number of images, both dataset was downsampled using random sampling.

Folder	No. of Potential Image Files	No. of Valid Images	No. of Invalid Images	Seed (for reproducibility)	Maximum images per class
meme	11817	11780	37	42	10000
not_meme	61214	61214	0	42	10000

Table 1. Dataset Class Imbalance/ Downsampling Results (Stage 1)

A temporary balanced dataset was then created to copy each image with new filenames (meme_00000.jpg, etc.), identical file extensions, and preserved metadata for a cleaner and uniform dataset to be used for training without modifying the original dataset.

2.3 Data Splitting

In total, there are already 20,000 images divided into two classes, prepared after balancing for the first model. Then, during the hyperparameter configuration, meme vs. non-meme classification, the data has been split into three ratios: 70% for training, 15% for validation, and 15% for testing. Specifically, the training set contains approximately 14,000 images, while the validation and test sets each contain around 3,000 images. This split ensures that the model has sufficient data to learn from during training, while also providing representative subsets for validation and final evaluation.

Similarly, in stage 2, the dataset contains a total of 11,352 samples split into similar ratios of 70/15/15. The training set includes 7,946 samples, the validation set 1702 samples, and the test set 1704 samples. Specific split indices determine how many samples go into each split, as shown in Table 2.

Split	Total Samples	Positive Samples	Non-Positive Samples
Training	7946	3,922 (49.4%)	4,024 (50.6%)
Validation	1702	906 (53.2%)	796 (46.8%)
Testing	1704	848 (49.8%)	856 (50.2%)

Table 2. Label distribution in Splits for the Sentiment Classification

The nearly 50/50 label distribution in each split ensures that the train, validation, and test sets are representative of the overall dataset, preventing potential class imbalance in splits. Additionally, it ensures reliable validation and testing, which are both necessary to tune hyperparameters and evaluate model performance. If these sets are skewed toward one class, the accuracy, precision, and recall results will become misleading.

Section 3. Model Architecture and Methodology

In this section, SentimemeNet's chosen models and methodology for each stage will be explained. This includes the different input pipelines, implementation, and optimization techniques applied.

3.1 SentimemeNet Model Overview

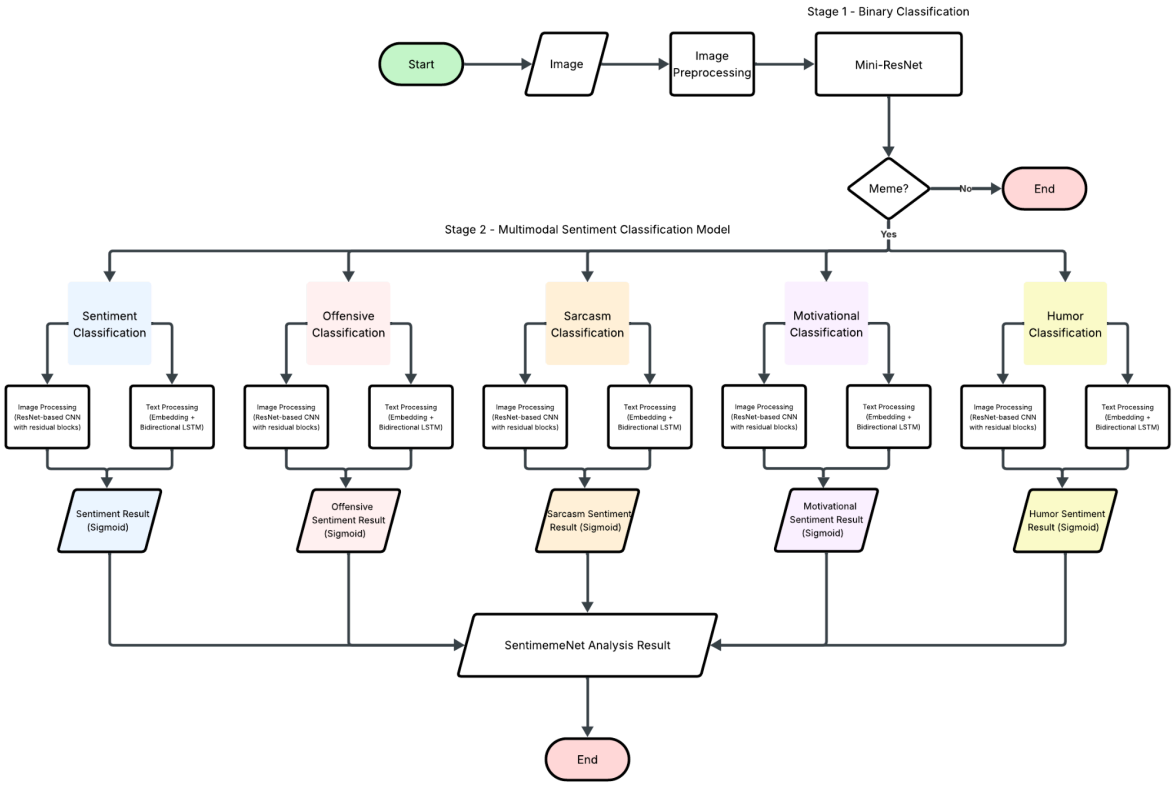

Figure 1. SentimemeNet Architecture Flowchart

The overall system of SentimemeNet follows a two-stage setup. The first stage involves initial binary classification of meme and non-meme images. Here, a mini version of ResNet is used with skip connections. The second stage is responsible for the sentiment classification and uses five versions of the multimodal model developed to suit each sentiment classification, specifically sentiment, offensive, sarcasm, motivational, and humor. The image branch employs a ResNet-based CNN with residual blocks, while the text branch, responsible for text processing, uses embedding with bidirectional Long Short-Term Memory (LSTM). These separate branches in the second stage are then

fused for the final classification, which outputs a sigmoid (binary) result. All results from each sentiment are then shown together in SentimemeNet's Analytics.

3.2 Input Pre-processing Pipeline

This pipeline serves as a foundational step for all subsequent architecture discussions. It applies to both Stage 1 and Stage 2 of SentimemeNet.

3.2.1 Image Pre-processing

The raw images in Stage 1 and Stage 2 models undergo an identical and optimized image preprocessing pipeline (Figure 2) when loaded into a TensorFlow Dataset. Here, the images (e.g., .png, .jpg) are first read before decoding them into a raw image tensor. Next, they are resized into a 224 x 224 Tensor for uniformity during training and are normalized, in which the output becomes the final input of the CNN model.

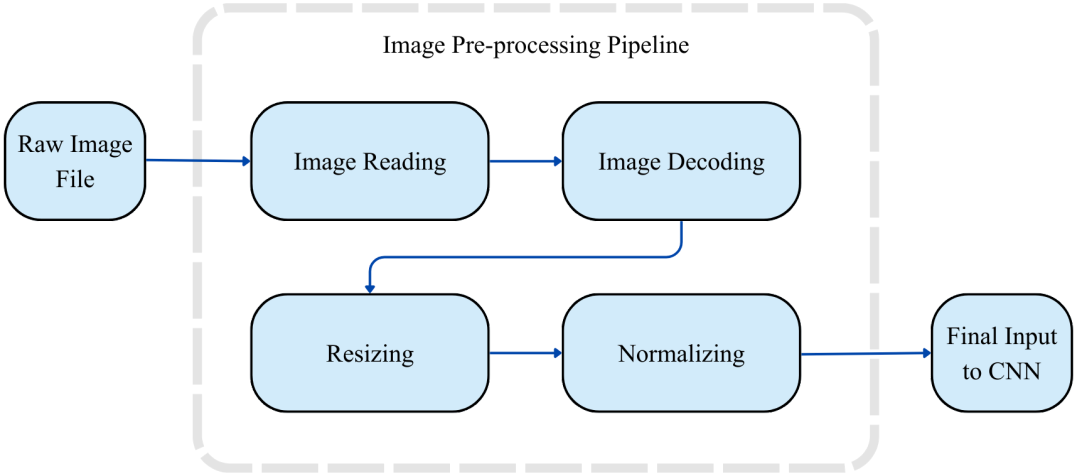

Figure 2. Image Pre-processing Pipeline

For the sentiment classification tasks, the same high-level pipeline was followed, but each sentiment operates on different labels from the Memotion dataset:

- Sentiment Classification → predicts positive/negative/neutral
- Sarcasm Classification → predicts sarcastic vs. not sarcastic

- Offensive Classification → predicts offensive vs. not offensive content
- Motivational Classification → predicts motivational vs. non-motivational memes
- Humor Classification → predicts funny vs. not funny memes

The first process is loading two separate Kaggle Memotion datasets and then merging them to produce a single unified dataset for training. This combined dataset is then validated and filtered, weeding out invalid images to clean the dataset. Balancing is also performed in this stage to ensure equal samples per class to prevent model bias. All sentiment tasks apply the same image pre-processing as the first model.

3.2.2 Text Pre-Processing

For multimodal sentiment tasks in stage 2, an additional Text Processing pipeline is needed to extract the sentiment from the memes. But before the actual model training/inference (processing), the raw texts need to be prepared first.

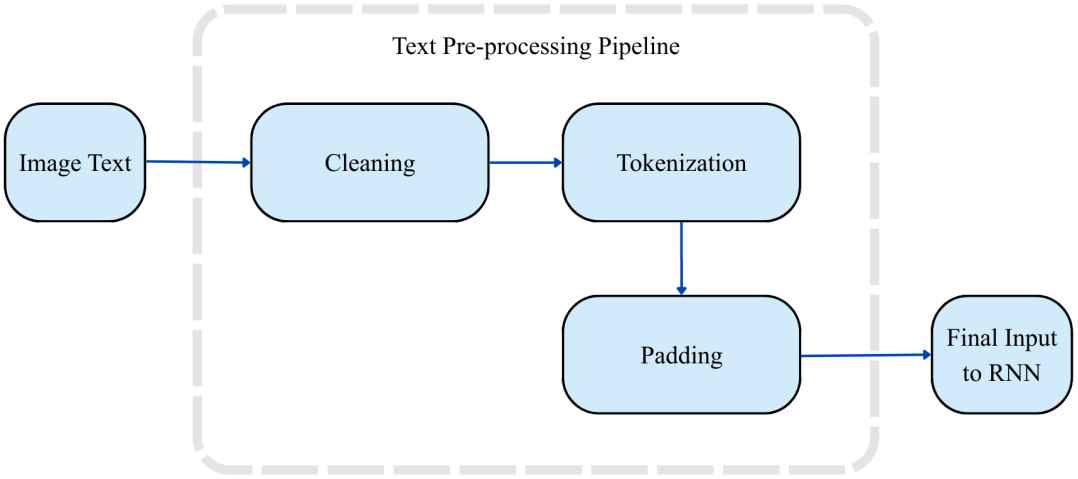

Figure 3. Text Pre-processing Pipeline

The first step in preprocessing texts is cleaning them. This is because the Optical Character Recognition (OCR) outputs can be messy, often with irregular spacing, URLs, or inconsistent casing. By cleaning them, the model can understand the texts better. Then, tokenization converts the cleaned texts into tokens or integer sequences since NLP models

cannot process raw strings and require numerical inputs. Lastly, padding ensures that all sequences have equal length before feeding them into the RNN.

3.3 Stage 1: Meme Detection Architecture

Stage 1 of SentimemeNet utilizes a lightweight version of ResNet, which is designed for the binary classification of memes and non-memes. This serves as a filter for the next stage that only requires memes for sentiment analysis.

The model works by first reducing the image size and extracting the low-level features, consisting 7 x 7 convolution with 32 filters, batch normalization, a ReLU activation, and max pooling. After this stage, the initial image with size 224 x 224 x 3 becomes 56 x 56 x 32. Next, in the core of the model, six residual blocks are arranged in three stages with increasing complexities: 64, 128, and 256 filters. Each of these blocks contains two convolution-batch normalization layers in its main path, along with a skip connection that either passes the input directly or uses a 1 x 1 convolution to match dimensions when downsampling occurs.

The network becomes easier to train with the use of skip connections because they help preserve important information and stabilize the gradient flow. As the model progresses, the spatial dimensions shrink from 56 x 56 to 28 x 28 and finally 14 x 14, while the number of feature channels increases. After this residual stage, the network uses Global Average Pooling to collapse each feature map into a single value, drastically reducing parameters compared to flattening. The model's classifier head consists of dense layers with 256 and 128 units, where each is followed by a dropout for regularization, and ends with a single sigmoid-activated neuron for binary classification.

Overall, the model contains roughly 2.8 million parameters and is significantly smaller but more efficient than the full ResNet architecture. It still retains the benefits of residual learning, structure feature extraction, and strong generalization capability.

3.4 Stage 2: MultiModal Sentiment Analysis Architecture

All types of classification in stage 2 use identical model architectures for image and text processing and only differ in class labeling. These two processes are done simultaneously after the first stage confirms that the image is a meme.

3.4.1 Image Branch

The image branch of the multimodal sentiment classification model uses a ResNet-based CNN with residual blocks. It is designed to extract meaningful visual features from meme images by gradually transforming raw pixels into a compact vector that captures the emotional cues, objects, text style, layout, and other visual patterns in memes. Then, it builds deeper features through residual learning without losing information. So, with each stage, the images become smaller while the feature depth becomes higher and more semantic and conceptual. This branch also utilizes Global Average Pooling (vector compression) that forces the model to extract global meaning and not pixel-level details. This pooling also reduces parameter count to prevent overfitting. A compact 256-dimensional representation of the image sentiment cues is then produced as the final image embedding, which is ready to be merged with the text features in the fusion layer.

3.4.2 Text Branch

While the image branch captures visual patterns and cues, the text branch processes the caption or text part of a meme. It converts this into a numerical vector that captures sarcasm, tone, sentiment, context, and linguistic cues. This branch is based on the embedding layer that converts the words into vectors, the Bidirectional Long Short-Term Memory (BiLSTM) to understand context from both directions, and the dense layer that produces the final 128-dimensional text representation.

This process receives the tokenized raw text that was prepared before and transforms each token ID into a vector in the embedding layer. For context understanding, BiLSTM reads the sentence forward and backward, which helps detect sarcasm, emotional tone, causal

relationships, etc. For higher-level meaning, a second BiLSTM takes the context-rich word representations and summarizes them into one vector. This is the exact moment when the model actually forms a global understanding of the text. Finally, the dense layer compresses the LSTM output and produces the 128-dimensional text embedding, and is ready to be fused with the image branch.

3.4.1 Feature Fusion and Classification Heads

The fusion layer combines the complementary information — embedded image and embedded text — from the two separate processing branches into a single representation for the classifier to use. Concatenation allows the network to use both modalities together through fully connected layers that learn cross-modal interactions.

Meanwhile, to map the fused features to the final output class (e.g., sentiment: positive or non-positive), the final 128-dimensional fused vector is reduced to one output unit within the dense layer. Using Sigmoid, the value is converted to a probability between 0 and 1, which is suitable for binary classification. The classification head, therefore, refers to the combination of the last few dense layers after fusion and the final output layer.

The following table shows the classification mapping of each sentiment type:

Classification	0	1
Sentiment	Non-Positive	Positive
Sarcasm	Not Sarcastic	Sarcastic
Offensive	Not Offensive	Offensive
Motivational	Not Motivational	Motivational
Humor	Not Funny	Funny

Table 3. Binary Class Mapping of the Multimodal Sentiment Classification Model

3.5 TensorFlow Dataset Optimizations

To improve speed, several optimizations were used, including shuffle() for randomizing the order of samples, batch() for grouping the samples, cache() to avoid re-reading images per epoch, and prefetch() for overlapping preprocessing and GPU execution.

Section 4. Model Training and Hyperparameter Tuning

4.1 Training Setup

4.1.1 Loss Function

Since SentimemeNet uses Sigmoid activation for the binary classification task in stage 1, the Binary Cross-Entropy is its loss function, which works perfectly with a sigmoid output. Similarly, in the multimodal stage, since the tasks are also binary, given by the classification mapping shown in Table 3, the loss function is also Binary Cross-Entropy.

4.1.2 Optimizer choice

The Adam or the Adaptive Moment Estimation Optimizer with a learning rate of 0.0001 was used within this project as the optimizer choice. This is because it is used to update the network weights during training using adaptive learning rates for each parameter.

4.1.3 Epochs, Batch Size, and Training Control

For stage 1, the maximum training length was set to 15 epochs. Training was then controlled using early stopping (patience = 5) and model checkpoint, where both monitor the validation loss. Its goal was to stop training when the loss ceased to decrease to ensure that the model minimized the classification error.

For stage 2 of SentimemeNet, training was configured with 25 epochs to allow sufficient time for convergence. However, this training length was dynamically controlled using advanced callbacks to ensure optimal performance and prevent overfitting. The primary training control mechanism employed in this stage was Early Stopping, which monitors the validation accuracy with a patience of five epochs. Using this, the weights from the epoch that yielded the highest validation accuracy are restored. There is also a model checkpoint callback that is used to save the best-performing model weights based on the same metric and the validation accuracy, which ensures that the final output model represents the peak performance achieved during training.

4.2 Hyperparameter Configurations

4.2.1 Learning Rate and Regularization

The overall hyperparameter strategy included dynamic adjustments to the learning rate to aid convergence. The base learning rate, initialized to be 0.0001, was used for the Adam optimizer. On the other hand, A ReduceLROnPlateau callback was implemented to manage the learning rate during plateaus in the loss landscape. It monitors the validation loss, and if no improvement is visible after a patience of 3 epochs, it reduces the learning rate by a factor of 0.5, down to a minimum rate of 0.00000001. Dropout layers are then used in the final dense layers of the model to further serve as a regularization technique in enhancing generalization.

Parameter	Stage 1: Meme Detection	Stage 2: Multimodal Sentiment Analysis
Input Image Size	224 x 224	224 x 224
Batch Size	2	16
Epochs	15	25
Early Stopping Monitor	val_loss (Patience = 5)	val_accuracy (Patience = 5)
Learning Rate	0.0001 (Fixed)	Initial LR
LR Schedule/Callback	None	ReduceLROnPlateau (x0.5 reduction, Patience = 3 on val_loss)

Table 4. Summary of the Hyperparameter Configurations in SentimemeNet

Section 5: Results, Evaluation, and Conclusion

This section includes all the training and prediction results, their corresponding evaluation, and an insightful analysis and reflection to conclude the project.

5.1 Final Model Evaluation (Stage 1)

The model of stage 1 has reached an impressive 93.56% training accuracy and 93.07% validation accuracy during the model training. The training history plot of the mini-ResNet model, as shown in Figure 4, for stage 1 confirms that its architecture is stable and learning well with the balanced datasets. It shows that the model reached convergence well within the 15-epoch limit, and that the Early Stopping strategy is proven to be effective based on the stable loss and accuracy curves.

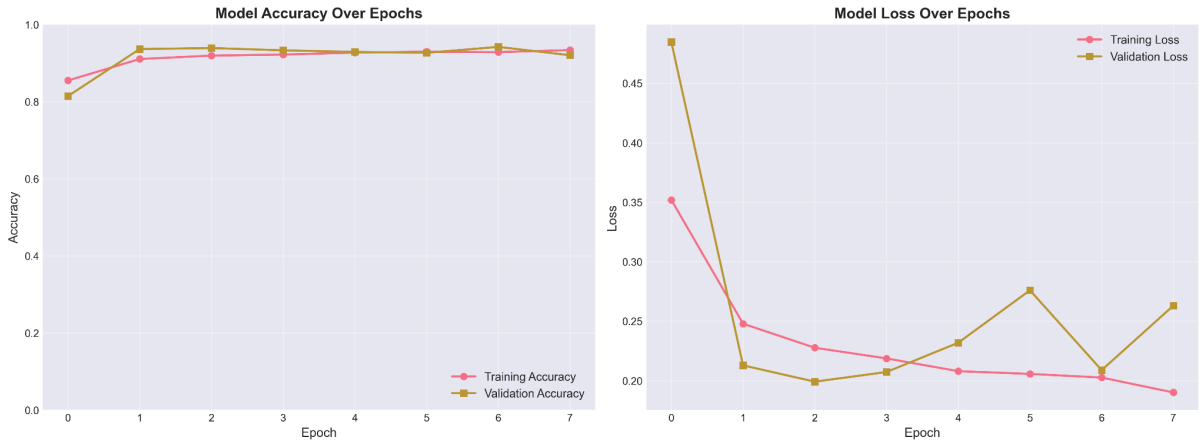

Figure 4. Training History Plot of Mini-ResNet

5.1.1 Test Set Performance

The performance of the Mini-ResNet model for meme detection was evaluated with a test set that consists of approximately 3,000 images (15% of the 20,000 balanced images). Figure 5 shows the sample images used to test the prediction ability of the model for meme vs. non-meme classification. Based on the results, it was able to predict each image correctly and with high confidence scores.

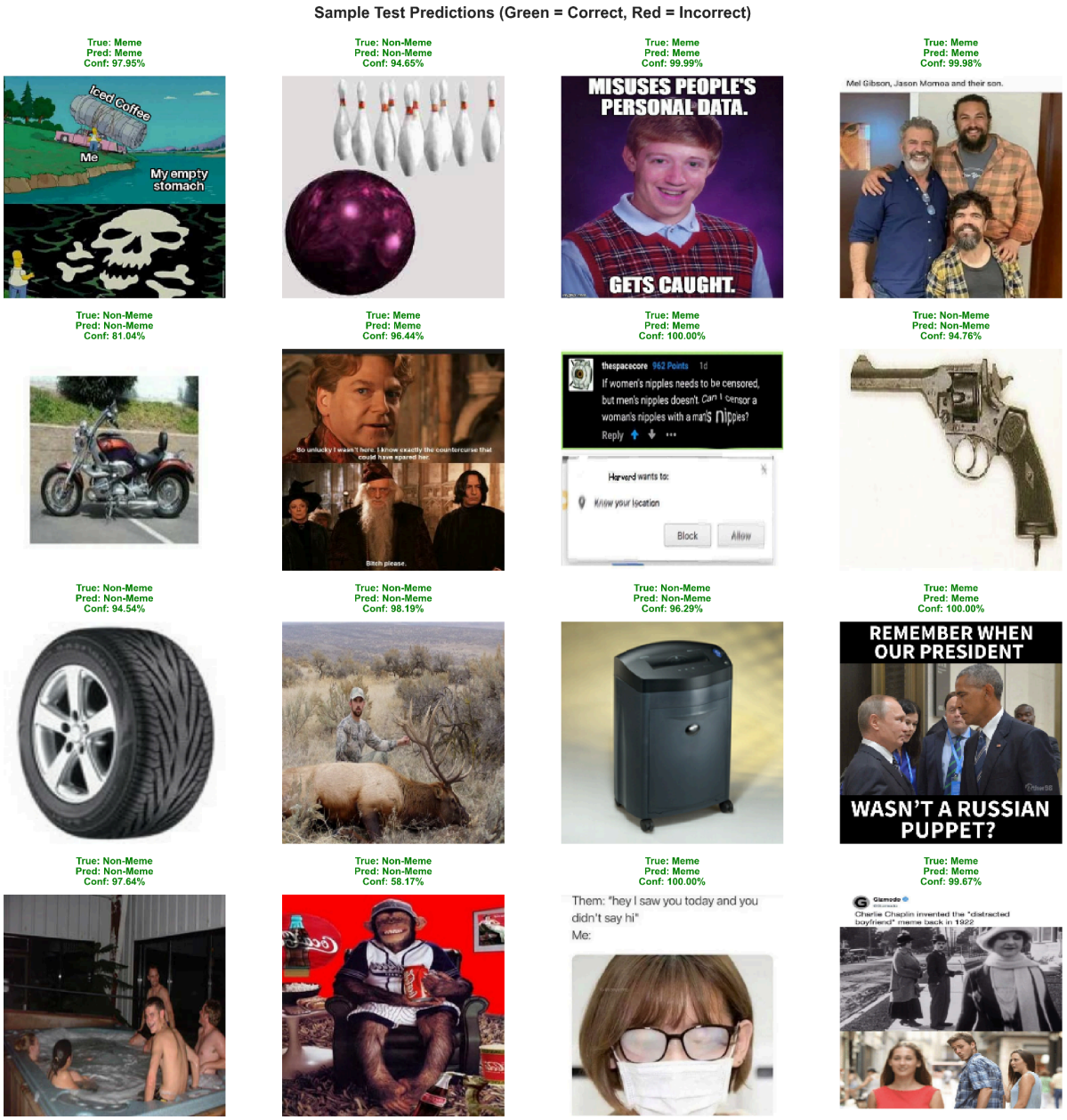

Figure 5. Training History Plot of Mini-ResNet

The mini model demonstrated a high degree of proficiency in doing the binary classification task. Its overall classification performance achieved an impressive Test Accuracy of 94.40% with a low Test Loss of 0.1659. Below are the detailed results of the model evaluation of the mini-ResNet.

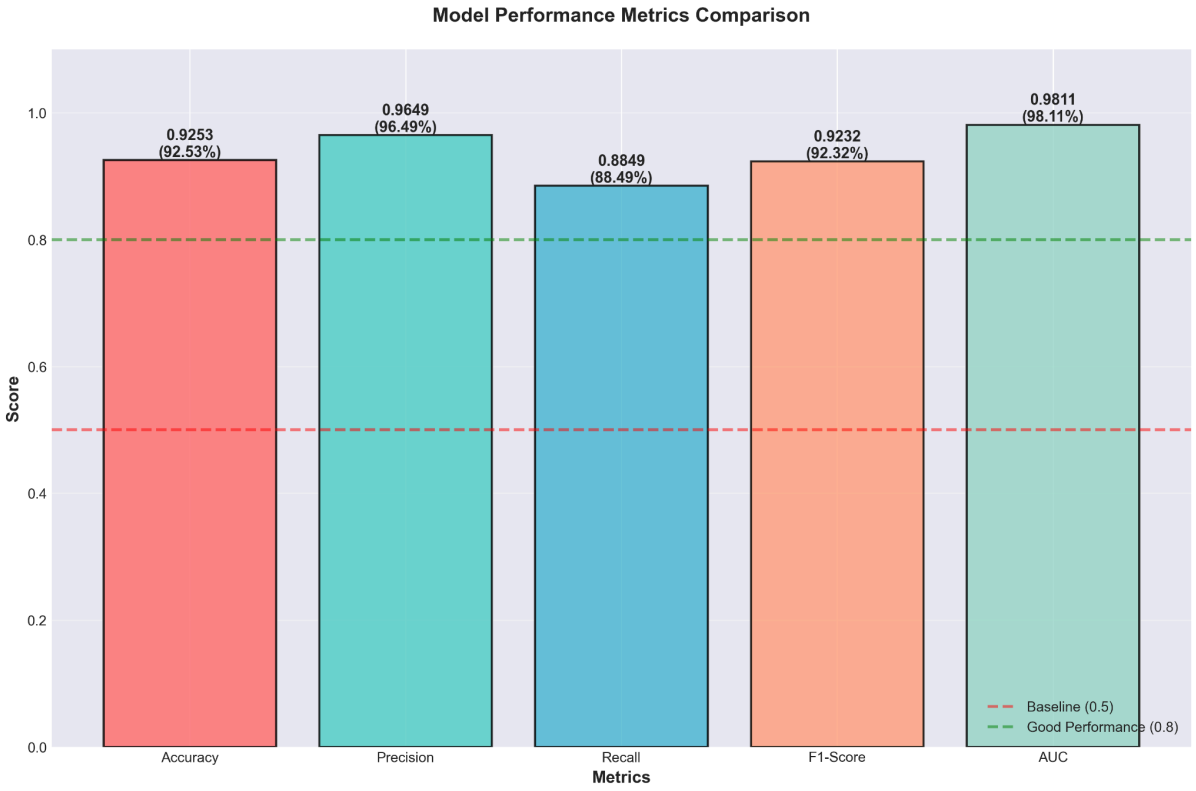

Figure 6. Mini ResNet Performance Metrics Comparison

According to a deeper analysis of the metrics, it is confirmed that the model has an excellent generalization ability. Its overall Accuracy, Precision, Recall, and F1-Score all exceed the defined "Good Performance" baseline of 0.8. The model's 0.9811 Area Under the Curve (AUC) further indicates that the model has high capability in distinguishing between positive and negative classes.

5.1.2 Performance Metrics

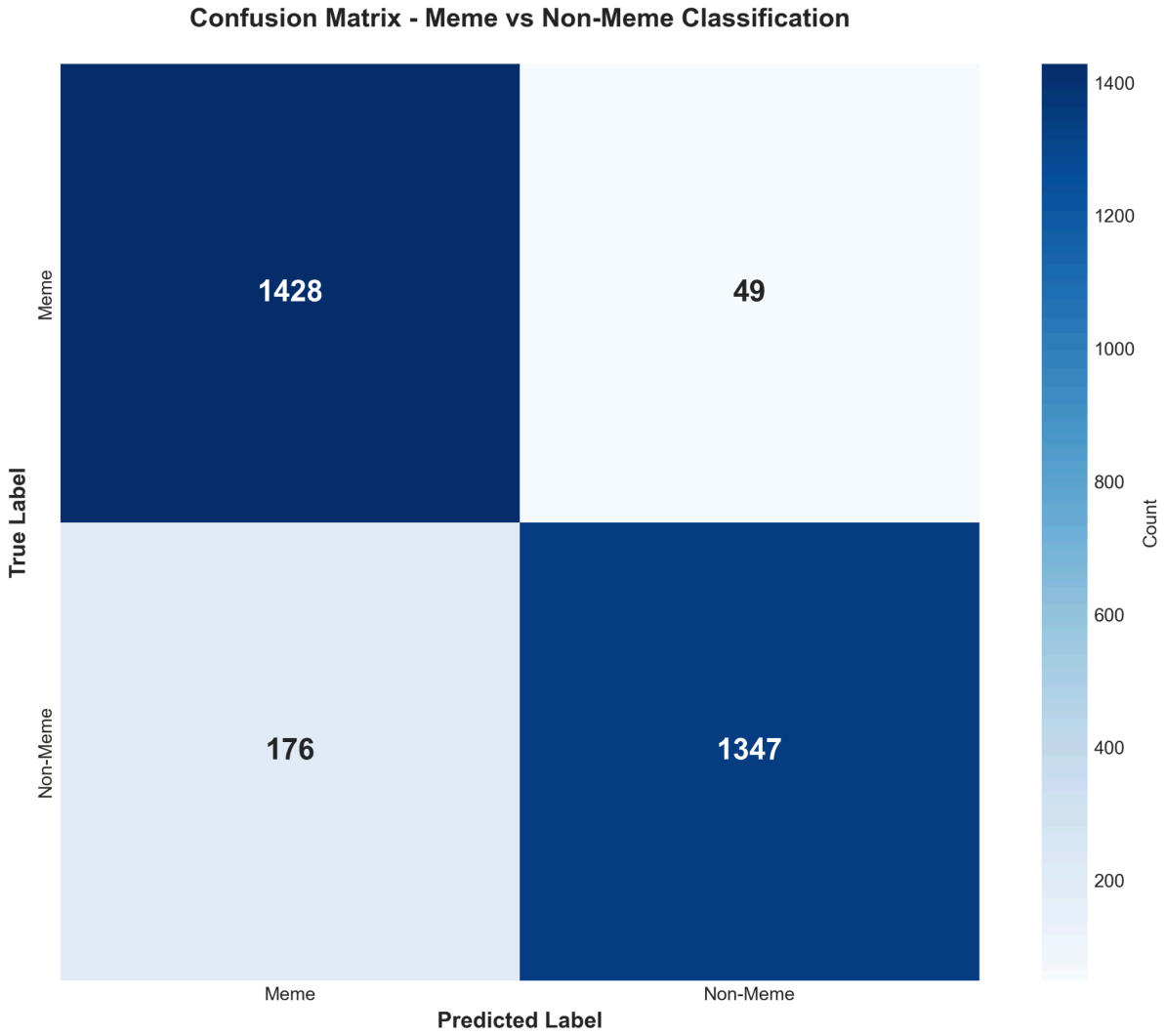

Figure 7. Confusion Matrix of Meme vs. Non-meme Classification

The confusion matrix (Figure 7) also confirms the model's high accuracy. It correctly classified 1428 meme images as true positives, while only 176 non-meme images were incorrectly classified as memes (false positives) out of 1477 images.

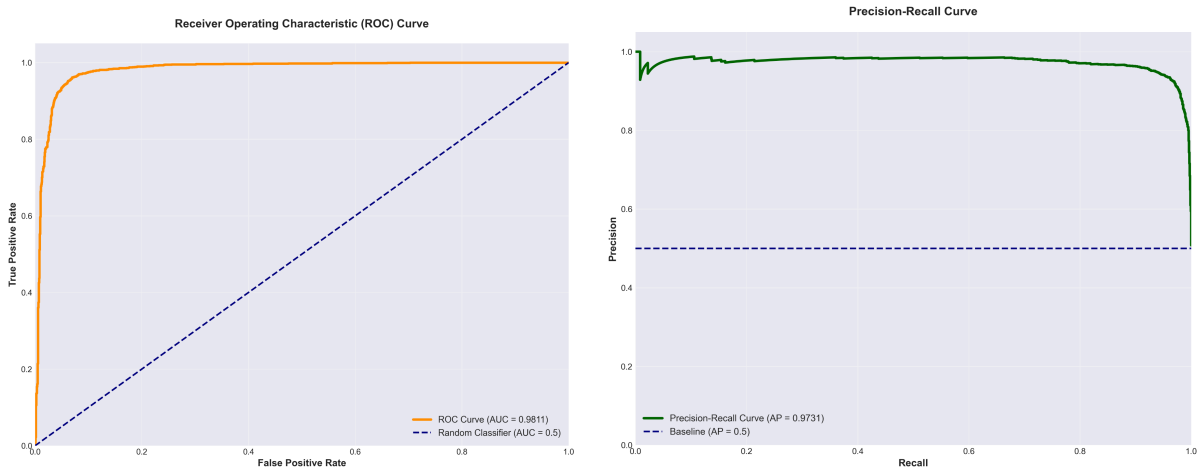

Figure 8. ROC and Precision-Recall Curve Results of Mini-ResNet

The Receiver Operating Characteristic (ROC) Curve of the model visually showcases that the model has a near-perfect ability to distinguish and separate the classes memes and non-memes, while the high Average Precision (AP) of 0.9731 on the Precision-Recall Curve confirms that the model maintains high precision even as the recall increases, guaranteeing that the stage 1 filter is working properly.

5.1.3 Accuracy Requirement Check

Given a strong performance across all metrics, the mini-ResNet model of the meme classification in stage 1 successfully fulfills its design requirement as an efficient and reliable filter for the sentiment analysis of SentimemeNet's two-stage approach.

5.2 Final Model Evaluation (Stage 2)

The high performance of the model in stage 1 cannot be blamed for the poor performance in stage 2. This stage is receiving clean, pre-filtered meme data, so performance issues of the model architecture here are isolated to itself. Therefore, to better evaluate the performance of the multimodal classification model, separate metric evaluations are implemented.

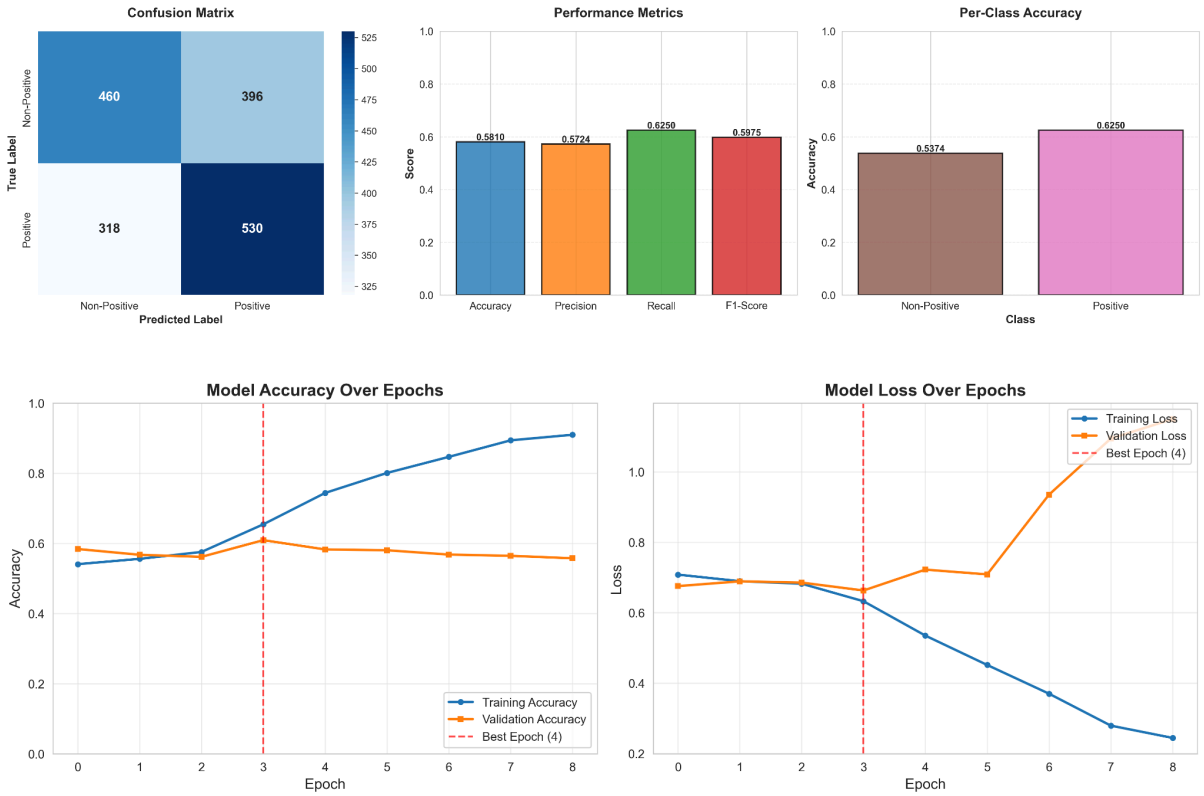

Figure 9. Meme-Sentiment Classification Metrics and Training History

The results of the sentiment classification task show a significant challenge for the multimodal model, as reflected in the overall low performance in accuracy and in the confusion matrix, where it confirms that the model is having a hard time learning the relationship between image and text features required to classify sentiment. It can also be seen that the model is slightly better at identifying Positive sentiment than maintaining overall precision, suggesting metric imbalance and classification bias.

The same goes for the other tasks, as shown in the succeeding figures.

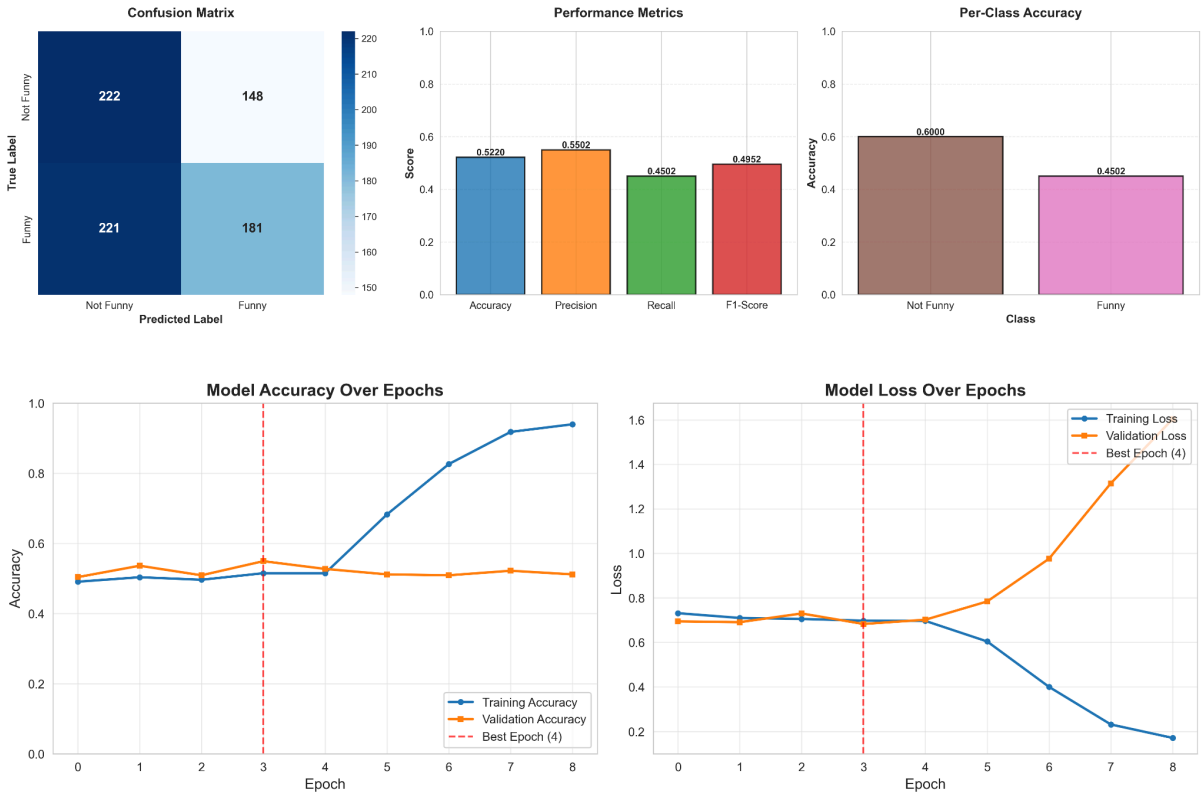

Figure 10. Meme-Humor Classification Metrics and Training History

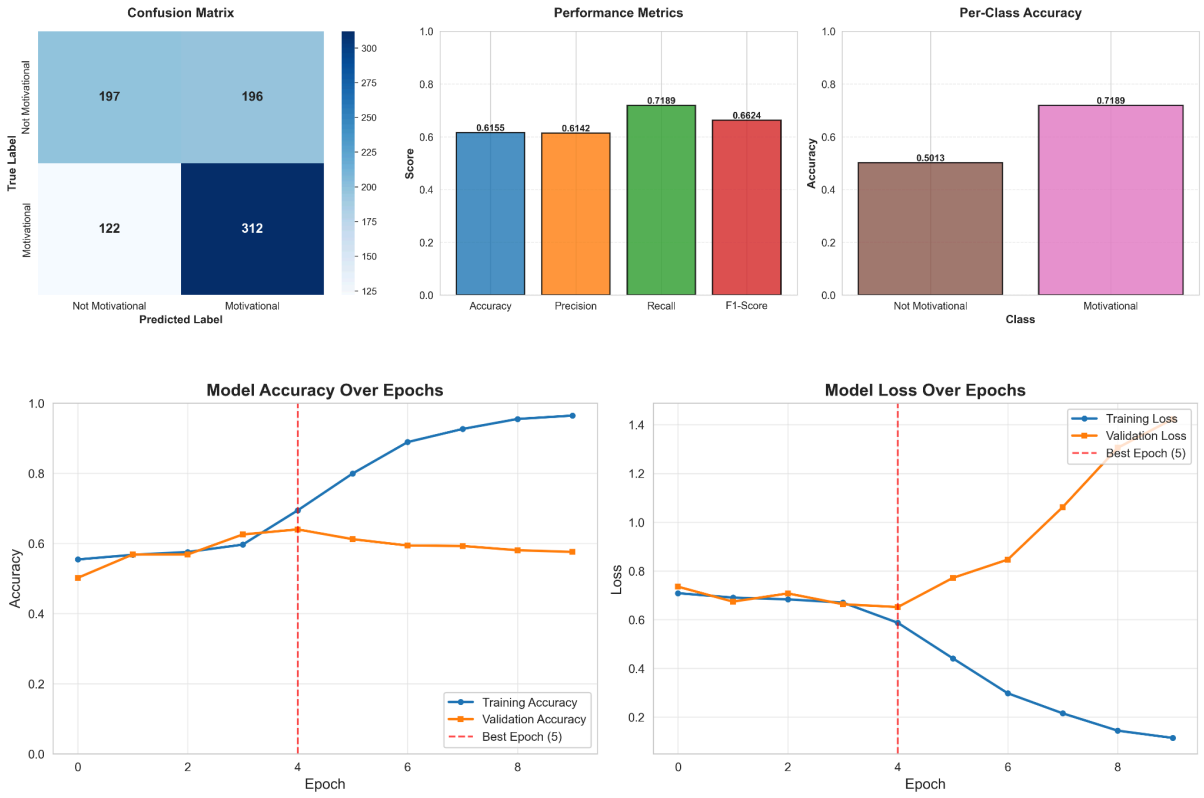

Figure 11. Meme-Motivational Classification Metrics and Training History

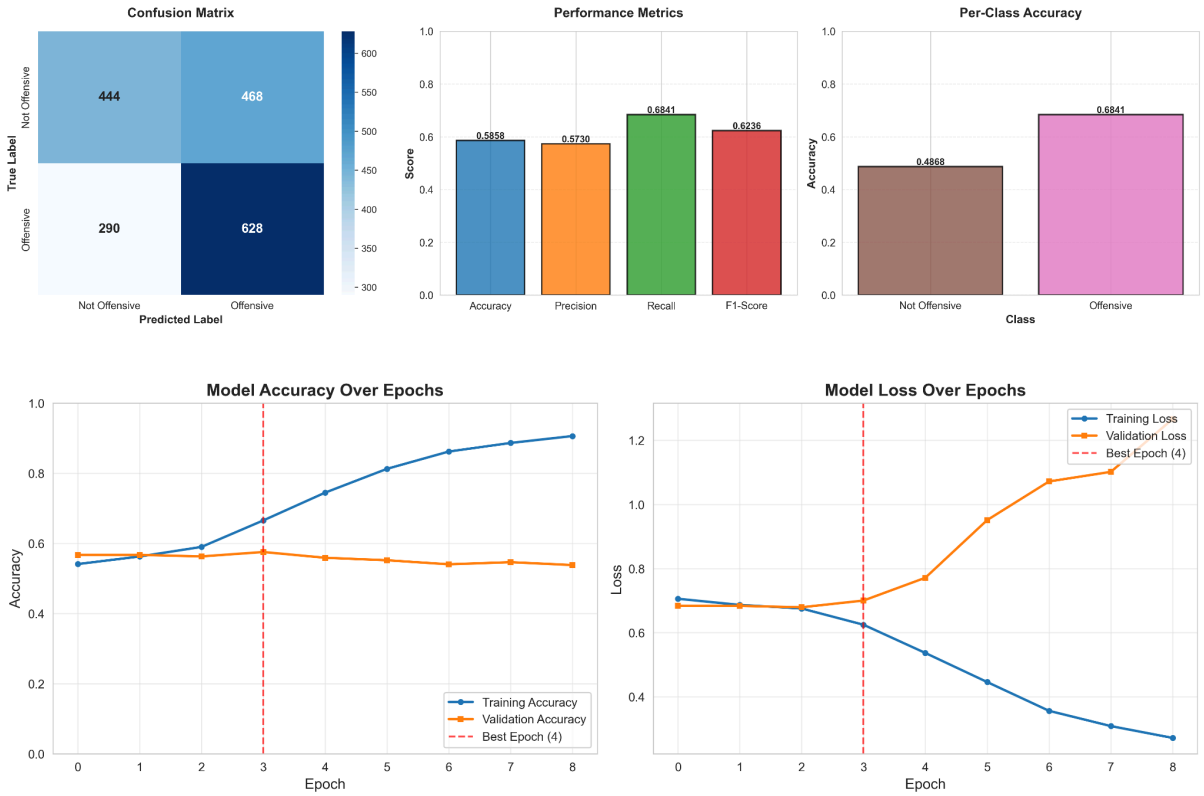

Figure 12. Meme-Offensive Classification Metrics and Training History

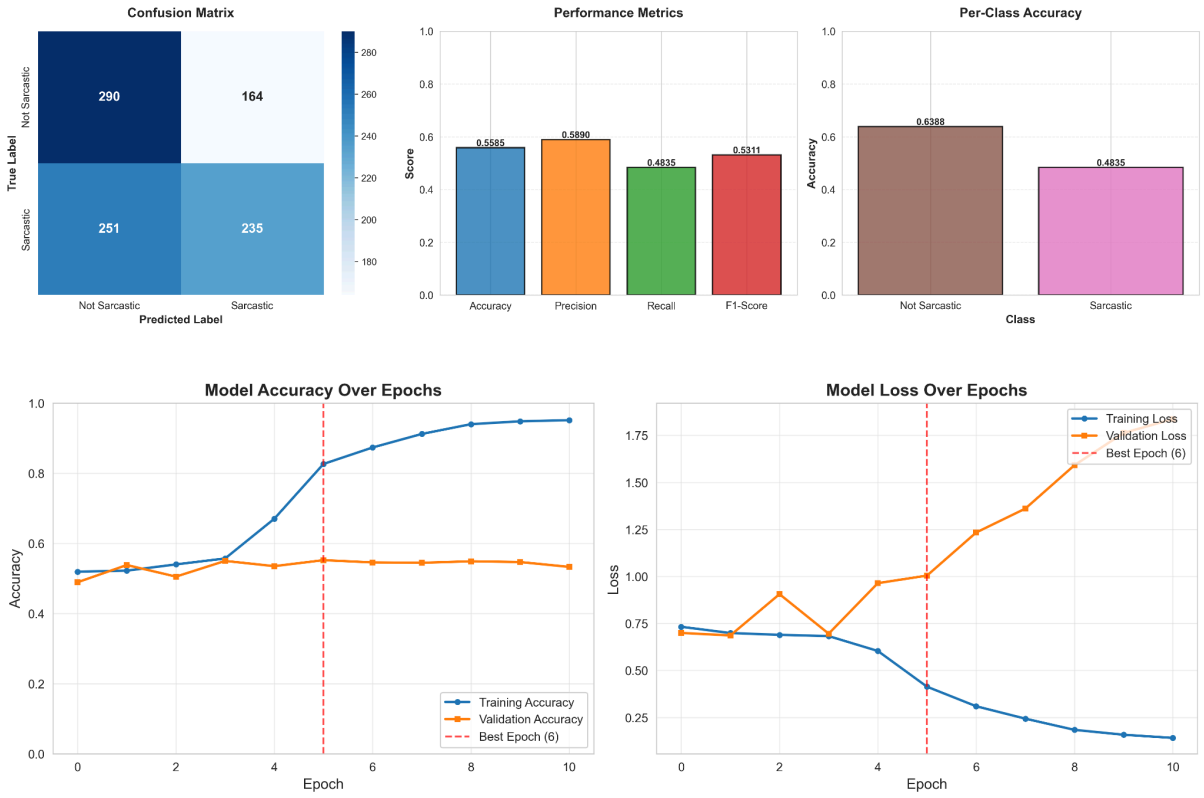

Figure 13. Meme-Sarcasm Classification Metrics and Training History

Based on the figures of the performance metrics of each sentiment type in the classification model, the multimodal model lacks generalization and has low performance overall. None of the models achieves strong performance with all test accuracy and F1-Scores falling in the range of 0.5220 to 0.6624. They are all barely above random, with an accuracy of 0.50% and the models marginally outperform random guessing. This may indicate that the architecture is struggling to extract and fuse the features in stage 2. Furthermore, the contrasting low test performance and high training accuracy seen in the training history plots strongly confirm that the model has severe overfitting.

Task	Test Accuracy	F1-Score	Inference
Sentiment	0.5810	0.5975	Barely exceeds the 0.5 random baseline.
Sarcasm	0.5585	0.5311	Highly difficult task. Often involves text contradicting the image or the context, so it requires more sophisticated cross-modal understanding.
Offensive	0.5858	0.6236	Has metric bias and class imbalance. It is biased towards the predictive Positive class over the non-positive.
Motivational	0.6155	0.6624	Highest Performance. Easiest task maybe because “motivational” means clearer and less subjective cues.
Humor	0.5220	0.4952	Lowest Performance. Hardest task. An F1-score below 0.5 indicates that the model is effectively guessing, reflecting the subjective nature of humor classification.

Figure 14. Summary of Test Accuracy and F1-Scores of the Multimodal Model

5.2 Reflection and Conclusion

The performance of the model in stage 1 confirms the efficacy of the proposed two-stage system for analyzing sentiment context in meme images. The high accuracy of (94.40%) and AUC (0.9811) demonstrates that the model successfully filtered out non-meme images with high reliability. The most critical architectural failure in SentimemeNet is the severe lack of generalization in stage 2. The history plots for all stage 2 tasks show a pattern of massive overfitting. Visual evidence for the sentiment classification history, for example, with training accuracy of 0.90+, while Validation Accuracy plateaus early around epoch 4, and validation loss sharply increases, proves that the model is memorizing the training data. This observed performance gradient reinforces the extreme difficulty of subjective classification tasks in memes. Particularly, SentimemeNet finds the humor task the hardest and performs worse than random guessing. This reflects the highly abstract, context-dependent nature of humor in which the current model fails to capture. Confusion matrices and per-class metrics also reveal a consistent bias toward the positive class in several tasks (offensive and motivational). Hence, even though SentimemeNet validated the effectiveness of implementing a two-stage filtering approach, the performance limitations of the multimodal classification models in Stage 2 were significant. This project, therefore, suggests that future work must focus on addressing the identified architecture and generalization deficiencies in relation to sentiment analysis.

Appendix: Demonstration of SentimemeNet using a Python Application

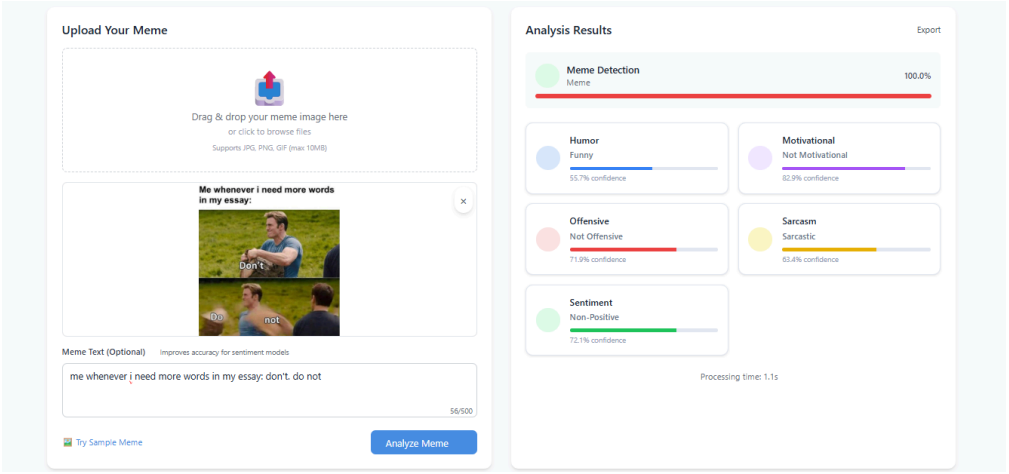

Figure A1. SentimemeNet's analysis of a hilarious essay meme

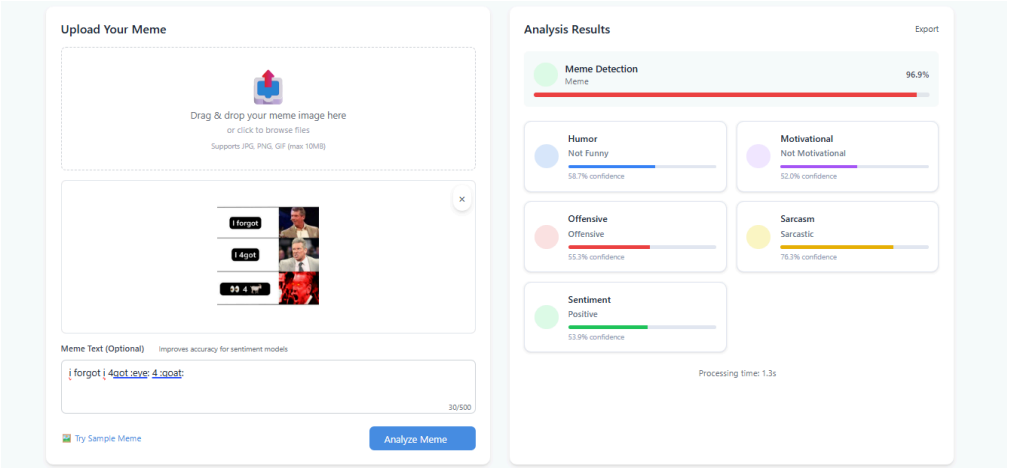

Figure A2. SentimemeNet's analysis results for a funny and sarcastic meme

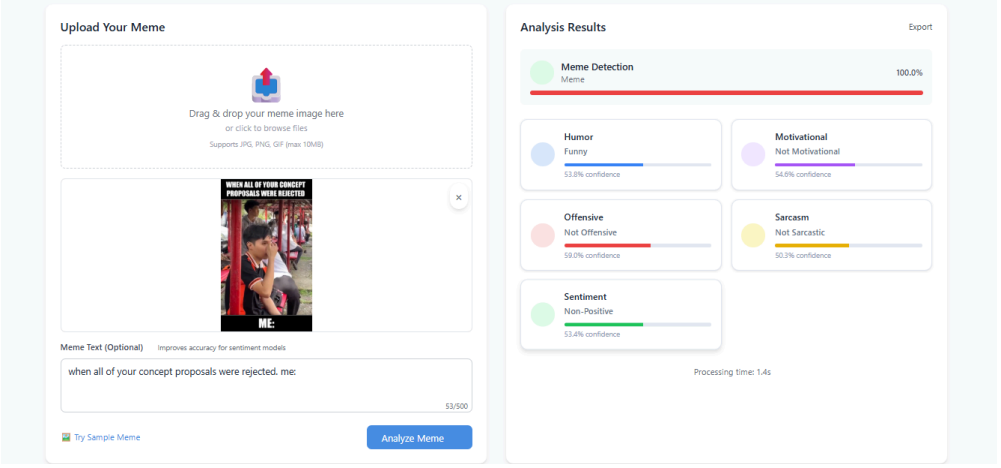

Figure A3. SentimemeNet's analysis results for an Empathetic Humor Meme

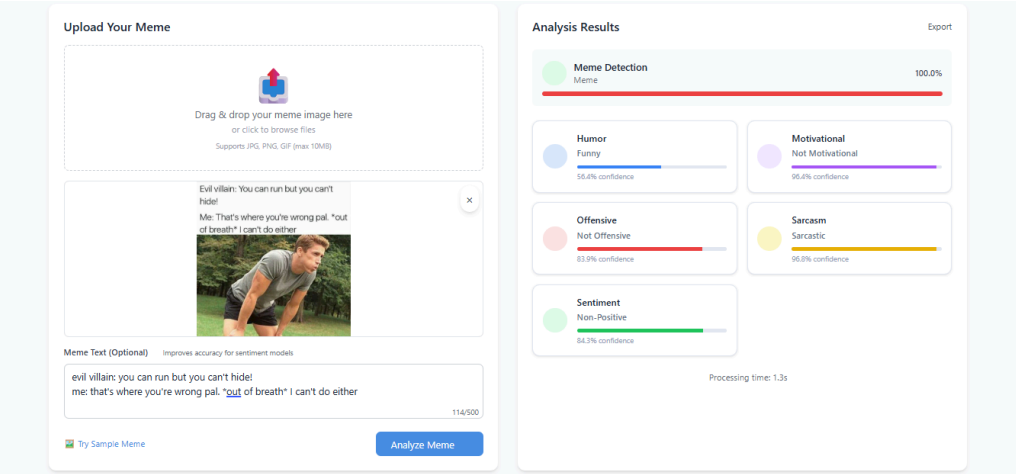

Figure A4. SentimemeNet's analysis results for a self-deprecating meme

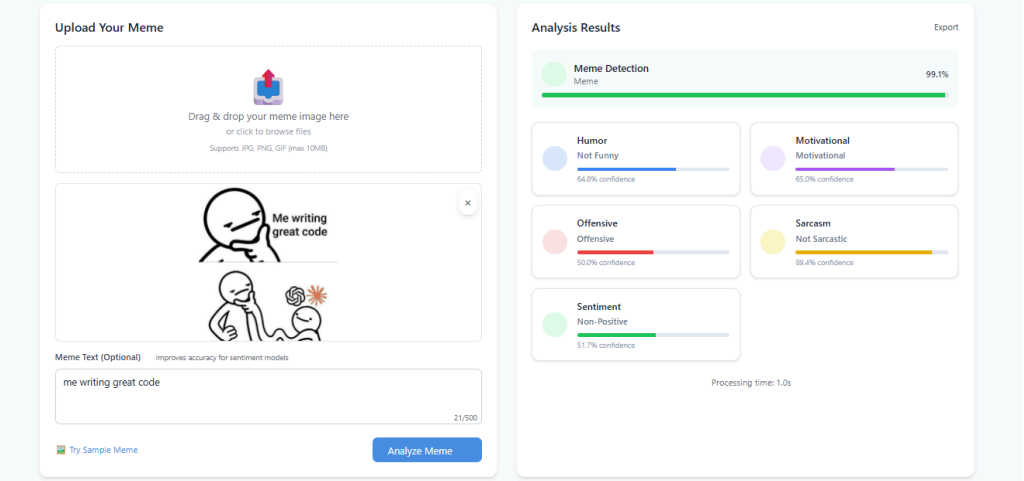

Figure A5. SentimemeNet's analysis results for a funny skepticism meme

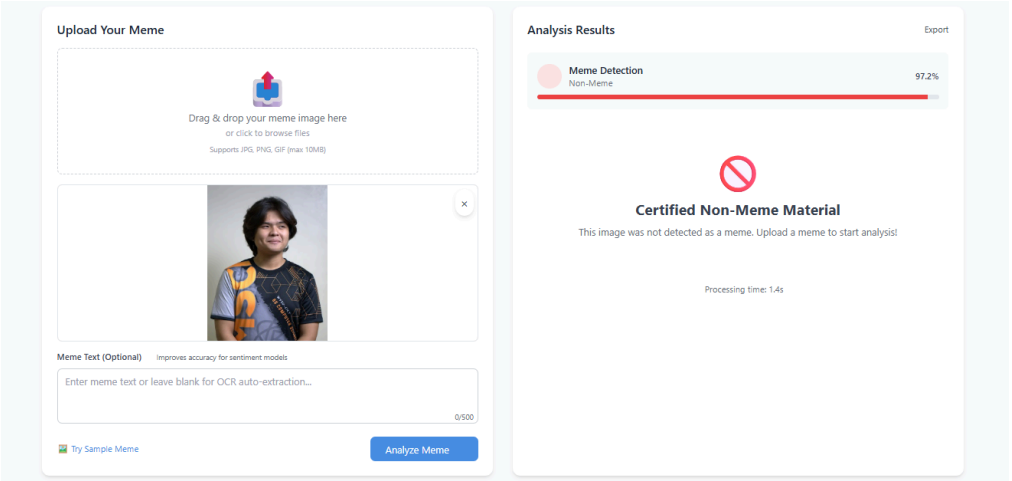

Figure A6. SentimemeNet's analysis results for a non-meme image of Zuriel

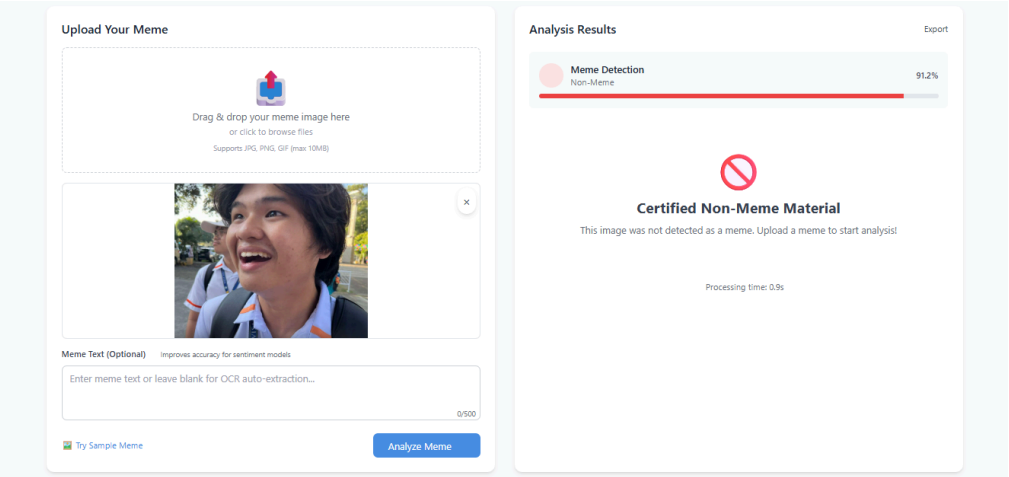

Figure A7. SentimemeNet's analysis results for another non-meme image of Zuriel

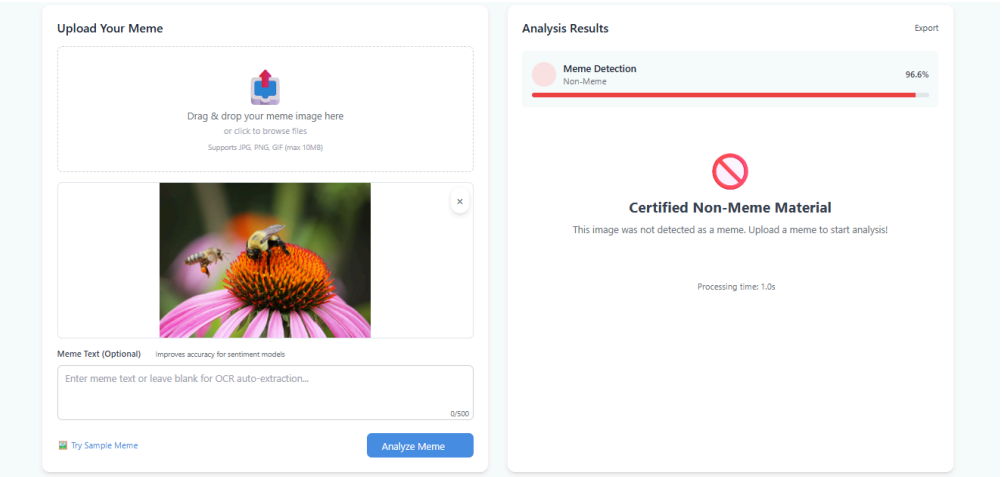

Figure A8. SentimemeNet's analysis results for an image of a bee in a flower

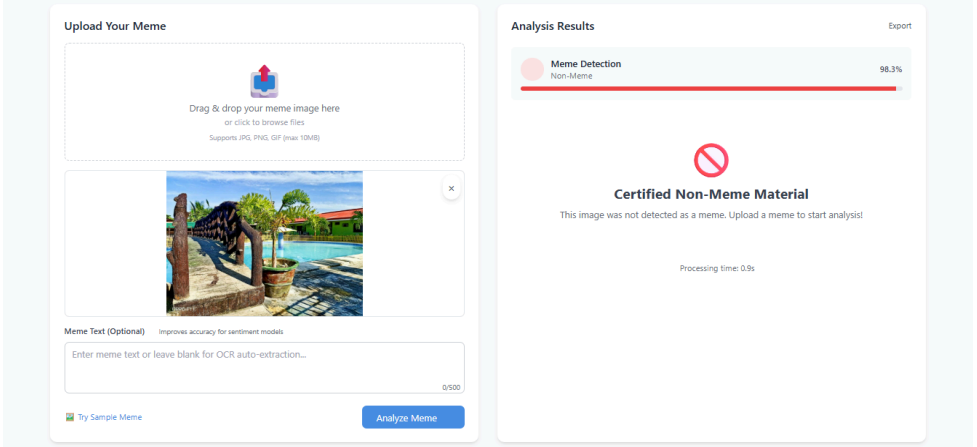

Figure A9. SentimemeNet's analysis results for a landscape image

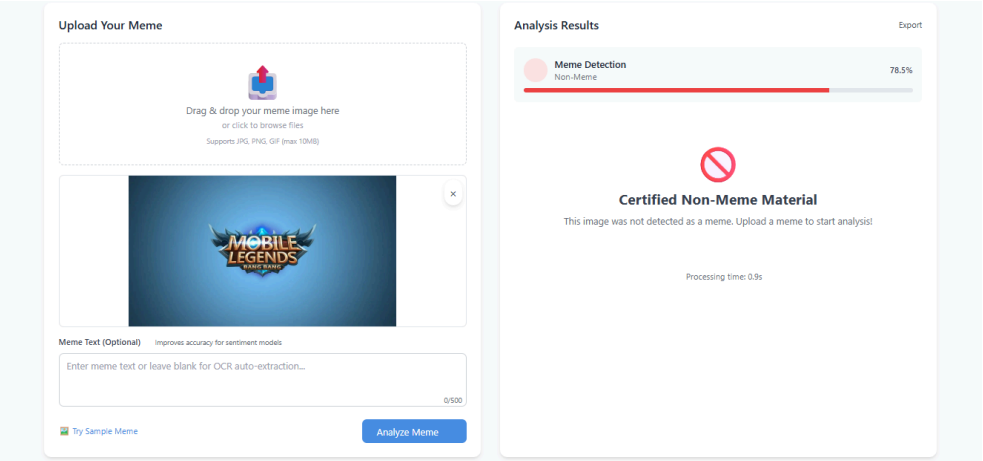

Figure 10. SentimemeNet's analysis results for a mobile game's logo

References

Halversen, A., & Weeks, B. E. (2023). Memeing Politics: Understanding political meme creators, audiences, and consequences on social media. *Social Media + Society*, *9*(4). https://doi.org/10.1177/20563051231205588

Meta AI. (n.d.). Hateful Memes challenge and dataset. Retrieved December 9, 2025, from https://ai.meta.com/tools/hatefulmemes/

Thinkhouse. (n.d.). Top meme moments of summer ’22. The Youth Lab. Retrieved December 9, 2025, from https://www.thinkhousehq.com/the-youthlab/top-meme-moments-of-summer-22

Sharma, C., Paka, S. W., Bhageria, D., Das, A., Poria, S., Chakraborty, T., & Gambäck, B. (2020). *Memotion Dataset 7k [Data set]*. Kaggle. https://www.kaggle.com/datasets/williamscott701/memotion-dataset-7k