Multimodal Sentiment Analysis with Vision Transformers and BERT

Brent Weiffenbach Mauricio Mergal Jeffrey Li Tanveer Kaur

Abstract

Image-Text Sentiment Analysis requires understanding emotional cues from multiple sources to determine an overall sentiment. In this project, we implement a multimodal sentiment classification model from scratch in PyTorch. The approach uses a Vision Transformer to encode vision features, and BERT to encode textual features. The encoded representations are positionally aligned and passed to a fusion module to extract a multimodal feature space. A fully connected layer with a softmax activation function acts as a prediction head to predict positive, negative, or neutral sentiment for the image-text pair.

1. Introduction

Multimodal Sentiment Analysis (MSA) aims to bridge the gap between the separate aspects that make up our everyday conversations like tone, imagery, and context by integrating this type of information to produce a more accurate understanding of sentiment. In this project, we will focus on building a model capable of jointly analyzing images and text to predict sentiment labels (positive, neutral, negative). Our approach leverages a transformer-based architecture to learn shared representations between visual and textual modalities, allowing the model to capture the subtle cross-modal relationships that single-modal models often overlook. We initially planned to train and evaluate our model using the MVSA-Multiple dataset, which contains approximately 19,600 image–text pairs annotated by three independent raters. However, due to practical constraints, we ultimately shifted to primarily using T4SA dataset along with the MVSA-Single, both of which provide sufficiently large and well-annotated multimodal samples for supervised training. Building on prior work such as CTMWA and TOM, our goal is to implement an MSA model entirely from scratch. This design choice allows us to investigate how multimodal representations can be learned end-to-end and to assess whether such a model can approach the baseline accuracy of approximately 74% reported for CTMWA.

2. Related Work

2.1. CTMWA

The Crossmodal Translation-Based Meta Weight Adaption (CTMWA) model advances Image-Text Sentiment Analysis by tackling two key challenges: capturing both shared (modality-invariant) and unique (modality-independent) emotional cues, and ensuring robustness when one modality is missing. CTMWA's first core component is a Cross-modal Translation Network, which learns to translate text features into an image-like space and image features into a text-like space. This not only helps the model capture shared semantics but also enables it to reconstruct a missing modality. The second innovation is the Unimodal Weight Adaption (UWA) strategy, which dynamically adjusts the importance of text versus vision for each input during training. Using a meta-learning approach, CTMWA trains a small weight-generation network on a curated meta set (128 instances per dataset) with manually annotated unimodal labels (text-only and image-only sentiment). This allows the model to adaptively emphasize the more reliable modality - for instance, trusting text in sarcastic posts or images in cases with strong visual emotion - leading to more accurate and context-aware predictions. Evaluated on three benchmarks, CTMWA achieved state-of-the-art results: 73.34% on MVSA-Single, 74.02% on MVSA-Multiple, and 75.75% on the fine-grained TumEmo dataset.

2.2. TOM

Weakly Correlated Multimodal Sentiment Analysis: One New Dataset and One Topic-oriented Model is a model focused on predicting sentiment labels for their own collected dataset, RU-Senti, which contains all data samples related to the Russia-Ukraine conflict. This model proposes a transformer focused approach to focus on weakly correlated samples. This approach uses pretrained ViT and BERT modules, with their novel topic-oriented feature alignment, with a multimodal feature fusion module and classification head to predict sentiment of image-text pairs. TOM is able to learn topic information from the text and use that to extract information from images semantically related to the text. The goal is to align the two modalities before feature fusion and sentiment prediction. This model achieves 75.32% accuracy on the MVSA dataset, and 85.44% on their own RU-Senti dataset (Liu

et al., 2024; Ren, 2024). Other relevant work includes sentiment analysis on multi-view social data (Niu et al., 2016) and recent advances in multimodal sentiment analysis using BERT and ResNet (Ren, 2024).

3. Proposed Method

3.1. ViT

The visual encoder is responsible for transforming raw image data into high-level semantic representations relevant to sentiment, such as color, facial expression, and compositional cues. We implement this as a Vision Transformer (ViT), which models an image as a sequence of patch tokens rather than relying on convolutional filters. Each image is divided into fixed-size patches, 16 x 16 pixels, flattened, and projected into a latent embedding space through a learnable linear projection. Positional encodings and a learnable classification (CLS) token are added to the sequence, which is then processed by the stack of Transformer encoder layers. Each Transformer layer consists of multi-headed self-attention and a feed-forward network with residual connections and layer normalization. This architecture enables the model to capture global contextual relationships across the entire image, allowing it to detect sentiment cues that emerge from broader spatial structure rather than small-scale details. The final visual embedding is taken from the CLS token output. Dropout and weight decay are applied throughout the encoder for regularization, and all parameters are updated end-to-end using gradients from the downstream sentiment classification objective. Prior to downstream training, the ViT encoder is pretrained using a Masked Autoencoder (MAE) to improve generalization and stability. During MAE pretraining, approximately 75% of patch images are randomly masked. Only the remaining visible patches are passed through the ViT, while learned mask tokens are injected later and combined with encoder outputs. A lightweight decoder then attempts to reconstruct the missing patches. The model is trained using mean squared error computed solely on masked patches. This encourages the encoder to learn robust and generalizable representations of image structure. Gradients are propagated through both the encoder and decoder during this stage. After pretraining, the decoder is discarded and the pretrained encoder weights are retained for integration into the multimodal sentiment analysis model.

3.2. BERT

The text encoder is implemented using the BERT architecture. BERT is a transformer-based language model that processes text bidirectionally, meaning it uses context from both the left and the right. It uses Masked Language Modeling (MLM) which works by hiding some of the words in a sentence and making BERT guess them, which helps the model learn how words relate to each other in context, and Next Sentence Prediction (NSP), which trains BERT to decide whether one sentence logically follows another. In our project, we implemented a simplified "mini" BERT encoder with fewer layers and smaller hidden dimensions. Our encoder starts with token embeddings, where each word ID is mapped to a 256-dimensional vector. We also add position and token embeddings so the model is aware of the positioning and order of the words in the sentence. After embeddings, the input passes through four Transformer encoder layers. Each layer contains multi-head self-attention which lets every word attend to every other word in the sentence, and a feed-forward network that transforms that representation, followed by layer normalization with residual connections. At the end, we use CLS pooling, where the first token of the sequence is used as the overall sentence representation. Masked language modeling (MLM) is used in order to learn proper representations. During training, about fifteen percent of the tokens in a sentence are replaced with a mask symbol. The model's job is to predict the original words that were replaced. The hidden states produced by our BERT encoder are first passed through a dense layer, followed by a GeLU activation and LayerNorm. Then, a final linear decoder maps each hidden state to a probability distribution over the entire vocabulary. We also use weight tying, which reduces parameters, overfitting and keeps word representations consistent. The loss function is the standard cross-entropy loss. When computing the loss, only the masked positions are significant, as all other unmasked tokens receive a label of -100.

3.3. Fusion and Classification

Our multimodal sentiment analysis framework employs a structured pipeline where features from separate visual and textual encoders are fused and classified to produce the final sentiment prediction. The fusion and classification stages are designed to be efficient and effective, leveraging the powerful representational capabilities of the Transformer architecture. The Fusion Module is implemented as a dedicated MultimodalFeatureFusion class. It receives as input the full sequence of feature vectors from the Vision Transformer (ViT) encoder, which represents the individual patches of the input image, and the full sequence of feature vectors from the BERT encoder, which represents the tokens of the input text. These two sequences, which may have different lengths and feature dimensions, are first projected independently into a common `fusion_dim` space using linear layers. The projected sequences are then concatenated along the sequence dimension to form a single, unified sequence that contains all information from both modalities. This concatenated sequence is then processed through a series of standard Transformer en-

coder blocks. Each block consists of a Multi-Head Self-Attention (MHSA) layer followed by a Multi-Layer Perceptron (MLP). The MHSA mechanism is the core of this fusion process, as it allows every element in the combined sequence - whether it originated from the image or the text - to attend to every other element. This global, dense interaction enables the model to learn complex, contextual dependencies and relationships between the visual and textual content, effectively creating a rich, joint multimodal representation. The output of this Transformer stack is a new sequence of the same length, where each element's representation has been updated based on its interactions with all other elements in the sequence. To prepare this fused representation for the final classification step, a simple yet effective pooling strategy is applied. The entire fused sequence is reduced to a single, fixed-size vector by taking the mean (average) across the sequence dimension. This pooled vector, which encapsulates the globally-interacted, multimodal understanding of the input, serves as the final input to the classification head. The Classification Head is a lightweight module implemented as a ClassificationHead class. It consists of a dropout layer for regularization, followed by a single linear layer. This linear layer maps the dimension of the pooled fused vector directly to the number of output classes (three, for positive, neutral, and negative sentiment). The output of this linear layer is a vector of raw scores, or logits, for each sentiment class. These logits are then passed through a standard cross-entropy loss function during training to compute the final prediction loss. This simple two-component structure (dropout + linear layer) provides a direct and efficient pathway from the complex, fused representation to a concrete sentiment prediction, completing the model's forward pass. This design - using a Transformer-based fusion to enable rich cross-modal interaction followed by mean pooling and a simple linear classifier - provides a robust and scalable foundation for multimodal sentiment analysis, balancing representational power with architectural simplicity.

4. Experiment

Our team designed experiments to determine the best techniques and setups to use with the raw pytorch implementation. The first experiment was to verify if our architecture can learn and theoretically get reasonable results by using Hugging Face pretrained ViT and BERT modules fed into our fusion module and classification head trained on the T4SA dataset. Once the fusion module and classification head were verified, the goal is to replace the Hugging Face modules with our own pytorch implementation. Pretraining the BERT and VIT modules using the T4SA dataset helps us get good feature representations for the downstream tasks to use for sentiment analysis. During these experiments we try changing embedding dimension sizes to implement smaller versions of the standard models. Finally, we have our own custom encoder implementations that have been pretrained on a large amount of data, a fusion module, and classification head ready to be fully trained on the final dataset. We train with the encoders frozen and available to finetune to compare final results. With the completed trained model, we can test the model on other datasets like MVSA-Single and compare accuracy.

5. Results

Initially, the custom PyTorch implementation was trained on the full dataset without pretraining. Validation loss exploded and accuracy was unstable, showing that training from scratch on a small dataset was ineffective (see Appendix A).

After verifying the architecture, we ran full online training with early stopping (patience 5). Both models stopped improving after about 10 epochs, regardless of encoder freezing. Figure 1 shows the frozen encoder case, reaching about 94% accuracy after 13 epochs.

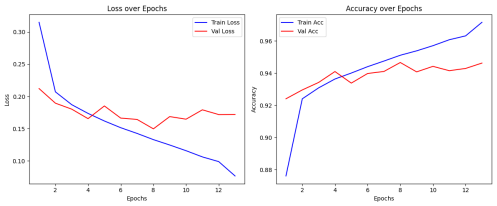

Figure 1. Training and validation accuracy for full training with frozen encoders.

With finetuned encoders, training is less stable but validation accuracy is higher (about 97.5%), as shown in Figure 2.

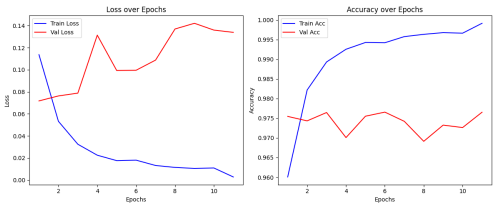

Figure 2. Training and validation accuracy for full training with finetuned encoders.

With pretrained encoders, we repeated full training (LR=5e-05, BS=64, Dropout=0.1, early stopping). Frozen encoders gave stable training and 94.65% accuracy; finetuning was less stable but reached 97.66% after 7 epochs (see Figure 3 and Figure 4).

On MVSA-Single, the model achieved 56.89% test accuracy (61.46% for consistent samples, 51.69% for incon-

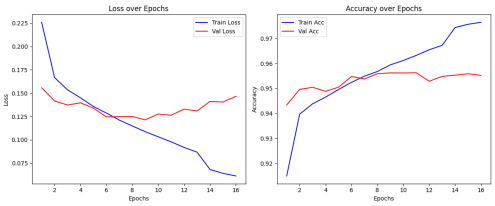

Figure 3. Training and validation accuracy for full training with custom PyTorch encoders frozen.

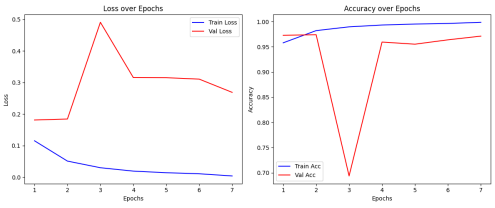

Figure 4. Training and validation accuracy for full training with custom PyTorch encoders unfrozen.

sistent). Accuracy vs text label: 61.74%; vs image label: 41.34%. The model predicts text sentiment better, likely due to dataset label structure. See Appendix B for more results and examples.

6. Discussion

This project set out to investigate whether a multimodal sentiment analysis architecture could be trained entirely from scratch, using custom PyTorch implementations of BERT and ViT, while still approaching the performance of systems built on pretrained encoders. The progression of experiments provides several clear lessons about model design, the role of pretraining, and the practical challenges of multimodal learning. A key takeaway is that training a full multimodal transformer from scratch was not feasible on our datasets. Using uninitialized encoders for both modalities led to unstable optimization, exploding validation loss, and no meaningful learning signal. This confirmed that the complexity of modern vision and language architectures far exceeds what can be learned from MVSA or T4SA alone without strong initialization. A second lesson appeared when comparing frozen and fine-tuned encoders. Across both the pretrained Hugging Face models and our custom-pretrained models, frozen encoders produced more stable learning curves, while fine-tuning significantly increased the validation instability within the large networks. However, we did find that fine-tuning consistently offered a small but meaningful accuracy benefit (e.g., from 94% to 97.5% with Hugging Face models and from 95.45% to 96.12% with our own encoders). This highlights a core trade-off within multimodal systems: fine-tuning improves performance but destabilizes optimization, which in turn requires careful scheduling and regularization. Thirdly, the experiments with MAE-pretrained ViT and MLM-pretrained BERT demonstrated that even smaller, simplified transformer encoders can learn useful feature representations when given appropriate pretraining objectives. Both pretraining procedures showed stable decreases in training and validation loss, indicating successful learning of generalizable representations. However, despite the success of our custom pretraining, the encoders still underperformed relative to the powerful pretrained Hugging Face counterpart - an expected outcome given their significantly smaller size and the limited scale of our pretraining datasets. Finally, qualitative testing on our own real-world examples demonstrated that the model can still make coherent multimodal predictions outside the training distribution, offering some evidence that the learned fusion module generalizes in a practical sense, even if the dataset transfer remains limited. Overall, this does raise several broader questions. How large must a dataset be to train a full multimodal transformer end-to-end from scratch? Could other pretraining methods (like contrastive) produce stronger shared visual and text representations than reconstruction based objectives like MAE and MLM?

7. Conclusions and Future Work

Multimodal Sentiment Analysis remains a challenging problem, and our work highlights several limitations that future research should continue to address. First, transformer-based encoders such as BERT and ViT require large amounts of pretraining data to learn strong representations, and smaller datasets can often limit their effectiveness. This is especially true for larger encoders, which tend to overfit and struggle to capture meaningful patterns when training data is scarce or homogeneous. The quality, diversity, and distribution of the dataset also play a critical role, as imbalanced or noisy samples can greatly affect model performance. Furthermore, generalizing to new domains or datasets remains difficult, as models often learn specific patterns that do not transfer well. Finally, sentiment itself can be inherently subjective and ambiguous; even humans frequently disagree on labels. Future work should potentially explore stronger pretraining strategies, more robust fusion mechanisms, and the collection of larger and more diverse multimodal datasets to better capture the complexity of human sentiment.

References

Liu, Wuchao, Li, Wengen, Ruan, Yu-Ping, Shu, Yulou, Chen, Juntao, Li, Yina, Yu, Caili, Zhang, Yichao, Guan, Jihong, and Zhou, Shuigeng. Weakly correlated multimodal sentiment analysis: New dataset and topic-oriented model. *IEEE Transactions on Affective Computing*, 15(4):2070–2082, 2024. doi: 10.1109/TAFFC.2024.3396144.

Niu, T., Zhu, S. A., Pang, L., and Saddik, A. El. Sentiment analysis on multi-view social data. In *MultiMedia Modeling (MMM)*, pp. 15–27, Miami, 2016.

Ren, J. Multimodal sentiment analysis based on bert and resnet. *arXiv preprint arXiv:2412.03625*, 2024.

A. Initial Architecture Experiments and Pretraining Results

To verify our architecture we first do a short experiment with Hugging Face encoders 'bert-base-uncased' and 'google/vit-base-patch16-224' with our custom pytorch fusion module and classification heads. Figure 5 shows the hugging face models with only a classification head and without the fusion module. The model learns for 10 epochs and gets to a reasonable 75% accuracy.

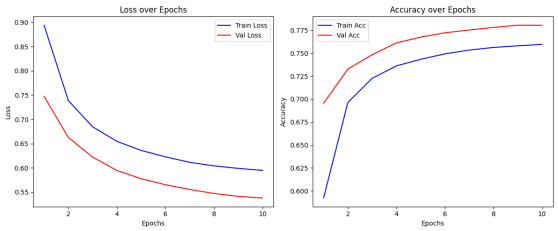

Figure 5. Training accuracy for Hugging Face encoders with classification head only.

Then we implement the fusion module with the hugging face encoders and classification head to get Figure 6. This model immediately achieves higher accuracy after a small number of epochs, verifying the need for the fusion module to increase our accuracy.

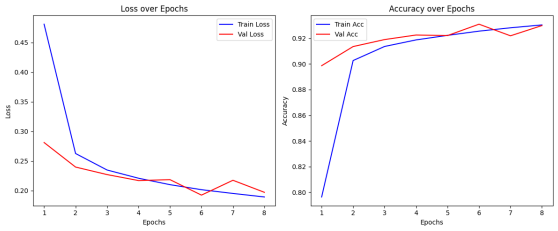

Figure 6. Training accuracy for Hugging Face encoders with fusion module.

Now that the architecture has achieved great accuracy with existing encoders, it's time to implement our own custom pytorch BERT and ViT. First, we can pretrain the ViT encoder with a strategy called Masked Autoencoder reconstruction to get weights for the encoder that output good feature representations. Figure 7 shows validation and training loss decreasing, evidently suggesting that the model is learning and achieves a good feature representation.

The text autoencoder, BERT, also needs to be pretrained to get good text features from the data. To do this, we use the masked language modeling pretraining technique which randomly masks some input tokens (about 15%) and the model must predict the original masked tokens. Figure 8 shows training and validation loss decreasing over

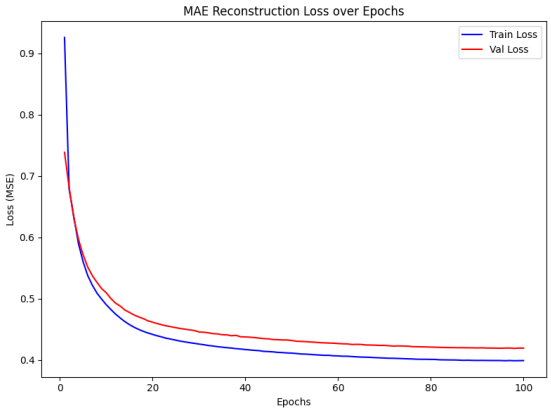

Figure 7. Training and validation loss for ViT encoder pretraining (Masked Autoencoder).

epochs which implies the model has learned good text features.

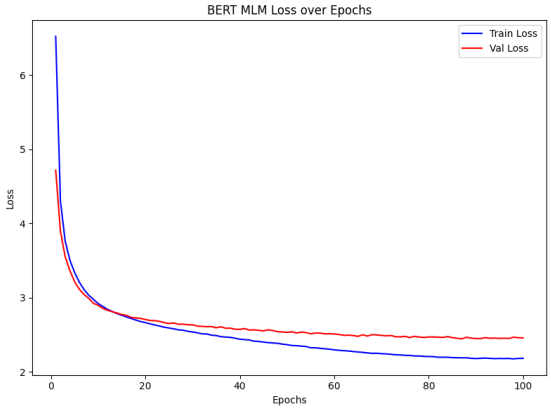

Figure 8. Training and validation loss for BERT encoder pretraining (Masked Language Modeling).

B. Additional Results

B.1. MVSA-Single Test Results

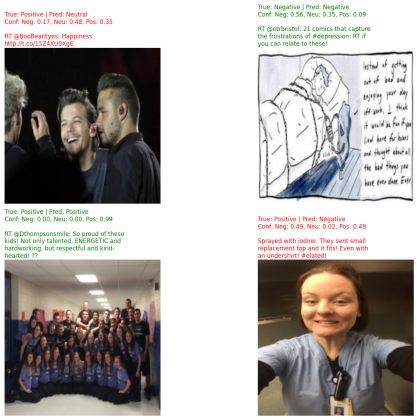

Figure 9. Test accuracy results for MVSA-Single dataset.

B.2. Model Predictions for Team Images

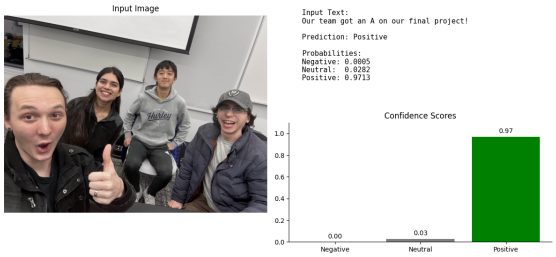

Figure 10. Model prediction for positive tweet about the project.

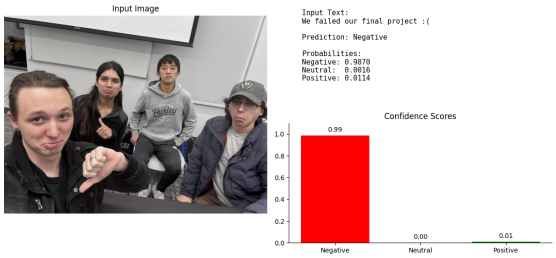

Figure 11. Model prediction for negative tweet about the project.