Image-Sentence Pair Matching

Chirobocea Mihail-Bogdan (507)

Abstract

This study addresses the Image-Sentence Pair Matching task by proposing two distinct model pipelines that combine textual and visual features to make accurate predictions. In the first model, image and text data are processed separately using convolutional backbones, with the image processed through 2D convolutions and the text through 1D convolutions. The outputs of these separate streams are fused in a shared fully connected layer for final prediction. The model is trained using a learning rate scheduler, AdamW optimizer, and Binary Cross-Entropy loss. In the second model, multi-modal features such as image encodings from an autoencoder, HOG descriptors, TF-IDF, and Word2Vec embeddings are extracted and concatenated. These features are passed through a fully connected network to make predictions. Both models are optimized using the AdamW optimizer with weight decay, and training is performed with a learning rate scheduler. The results show an improvement in prediction accuracy for the second model, achieving 77.3% accuracy, compared to 62.5% in the first model, highlighting the benefits of multi-modal fusion for this task.

Data Processing

This section describes the data processing pipeline used for the Image-Sentence Pair Matching task. Two separate models were employed in this study, each with distinct preprocessing techniques for the textual and visual data.

For both models, I made the following preprocessing for text:

1. Conversion of numbers into their word equivalents.
2. Removal of stop words.
3. Lemmatization of words to their base forms.

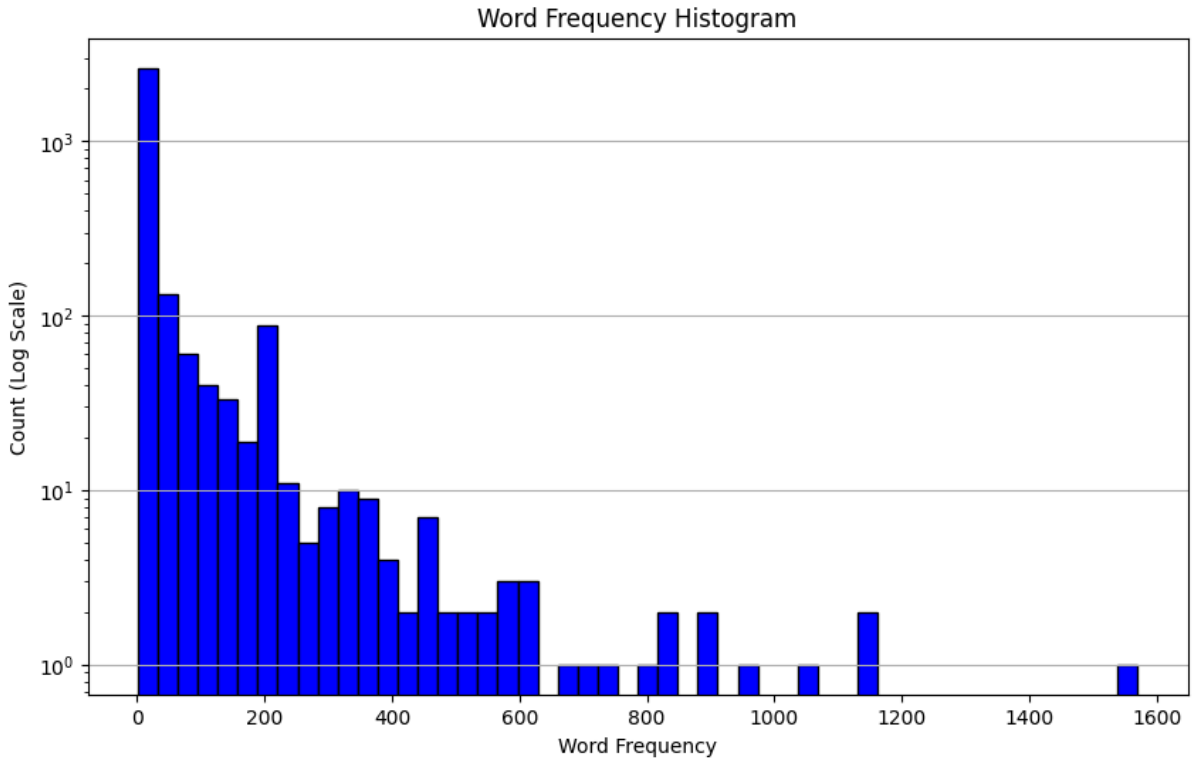
Model 1: Tokenizer-Based Text Processing and Augmented Image Tensors

Text Preprocessing

For the first model, I trained a Byte Pair Encoding (BPE) tokenizer on the text corpus to convert textual data into a numeric format suitable for input to the model. The tokenizer was configured with the following parameters:

- Vocabulary size: 1500 words
- Minimum word frequency: 2

The vocabulary size of 1500 was determined based on a histogram of word frequencies, ensuring coverage of the most relevant terms while reducing noise. Additionally, a maximum token length of 32 was selected based on the histogram of token lengths, ensuring the input sequences fit the majority of the corpus.



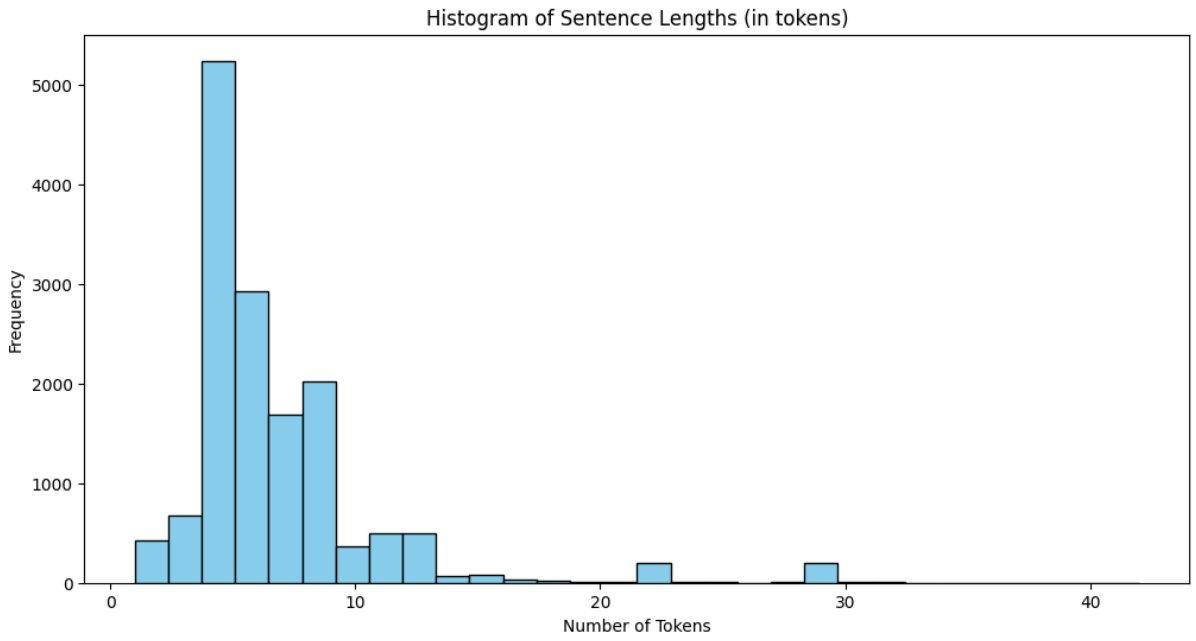


Image Preprocessing

The images were directly passed to the network as tensors. To enhance the robustness of the model, the following data augmentation techniques were applied:

- **Random Horizontal Flip**
- **Random Rotation**: Limited to 15 degrees.
- **Color Jitter**: Adjustments made to brightness (0.2), contrast (0.2), saturation (0.2), and hue (0.1).
- **Random Resized Crop**: Output size of 100x100 pixels with a scale range of (0.8, 1.0).

The augmented image tensors were subsequently input to the network for training.

Model 2: Feature-Based Representation for Images and Text

Image Preprocessing

For the second model, image features were extracted using a combination of two methods:

1. **Autoencoder**
    - An autoencoder was trained on the corpus images with the same augmentations as the first model.
    - After training, only the encoder component was retained to extract feature vectors of size 2000 from the input images.

2. **HOG Descriptors**
    - The Histogram of Oriented Gradients (HOG) descriptors were calculated for the resized images (100x100 pixels). The following parameters were used for HOG computation:
        - Window size: 64x64
        - Block size: 32x32
        - Block stride: 16x16
        - Cell size: 16x16
        - Number of orientation bins: 6
    - This process produced feature vectors of size 1944 for each image.

Text Preprocessing

The textual data was processed using two feature extraction techniques:

1. **TF-IDF**
    - Term Frequency-Inverse Document Frequency (TF-IDF) was applied to the text corpus.
    - A maximum of 2000 features were retained to represent the corpus in a high-dimensional space.
2. **Word2Vec**
    - A Word2Vec model was trained on the tokenized corpus with the following parameters:
        - Vector size: 100 (dimensionality of word vectors)
        - Context window size: 5
        - Minimum word frequency: 1
        - Number of worker threads: 4
        - Skip-Gram architecture

Model Architecture

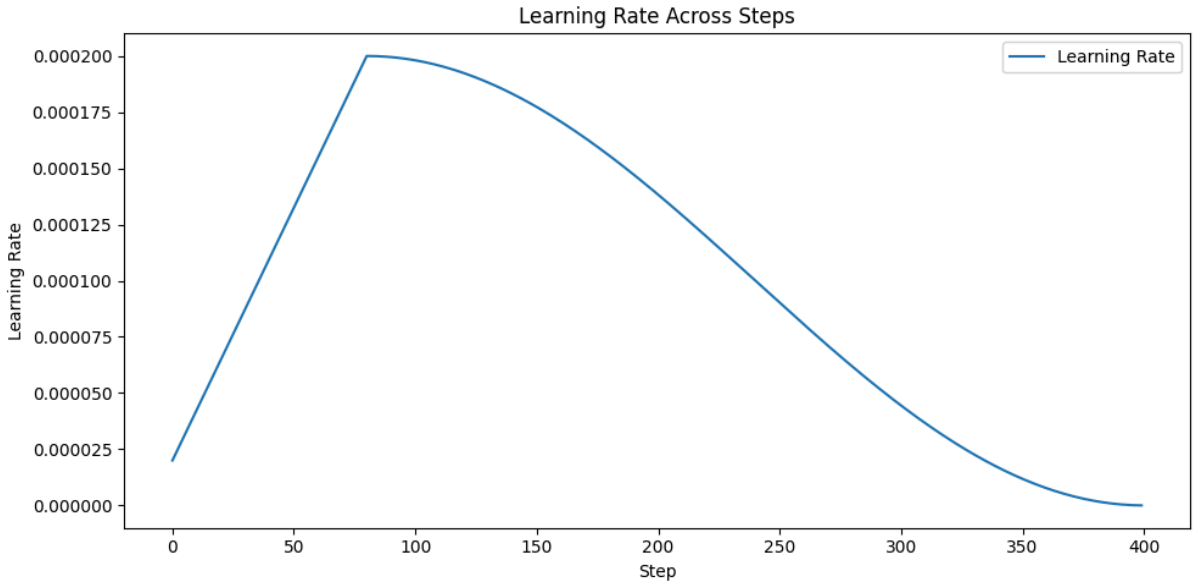
In this study, I explore two distinct pipelines for solving the Image-Sentence Pair Matching task, with a focus on utilizing both image and text features for prediction. The following sections describe the architecture and specific design choices for each model pipeline.

Model 1: Convolutional Backbone for Image and Text

The first approach leverages a dual-stream architecture where both the image and text are processed separately before being merged for final prediction.

1. **Input Representation:**
    - Image Input: The input image undergoes augmentation.
    - Text Input: A sequence of 32 tokens is used, which is processed using a text-based model.

2. **Image Processing Backbone:**
    - The image is passed through a convolutional neural network (CNN) with a kernel size of 3 for the 2D convolutions. The network is designed to capture spatial patterns and hierarchical features.
    - The CNN is followed by fully connected layers that reduce the image feature map size.
3. **Text Processing Backbone:**
    - The text sequence is processed using a 1D convolutional network with a kernel size of 7. This network captures sequential patterns and dependencies between the tokens.
    - Similar to the image backbone, fully connected layers are used to reduce the feature dimensionality.
4. **Fusion and Final Prediction:**
    - The outputs of both the image and text backbones are combined into a shared fully connected layer, where they are fused and passed through to produce the final prediction.
5. **Activation Functions and Regularization:**
    - GELU activations are applied across all layers to introduce non-linearity, with weight normalization used for better training stability.
    - The model is regularized using dropout, and Batch Normalization is applied in appropriate locations to stabilize training and improve convergence.
6. **Optimization and Training Configuration:**
    - The model is trained using the AdamW optimizer with a learning rate of 2e-4 and a batch size of 256. A learning rate scheduler is implemented, increasing the learning rate linearly in the first 20% of training steps and then decaying it with a cosine schedule to the final learning rate.
    - The Binary Cross-Entropy (BCE) loss function is used for training the model for 10 epochs.

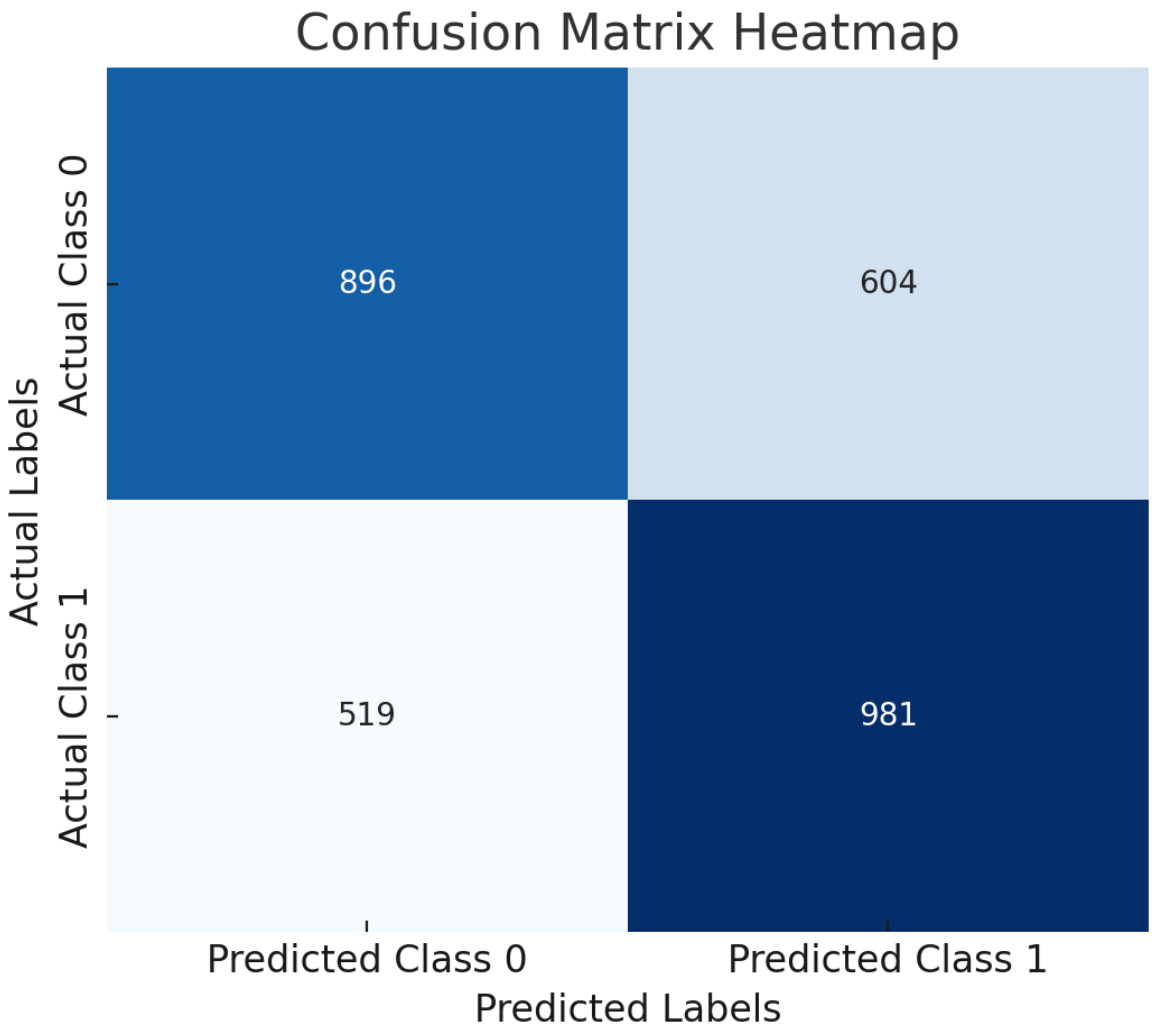


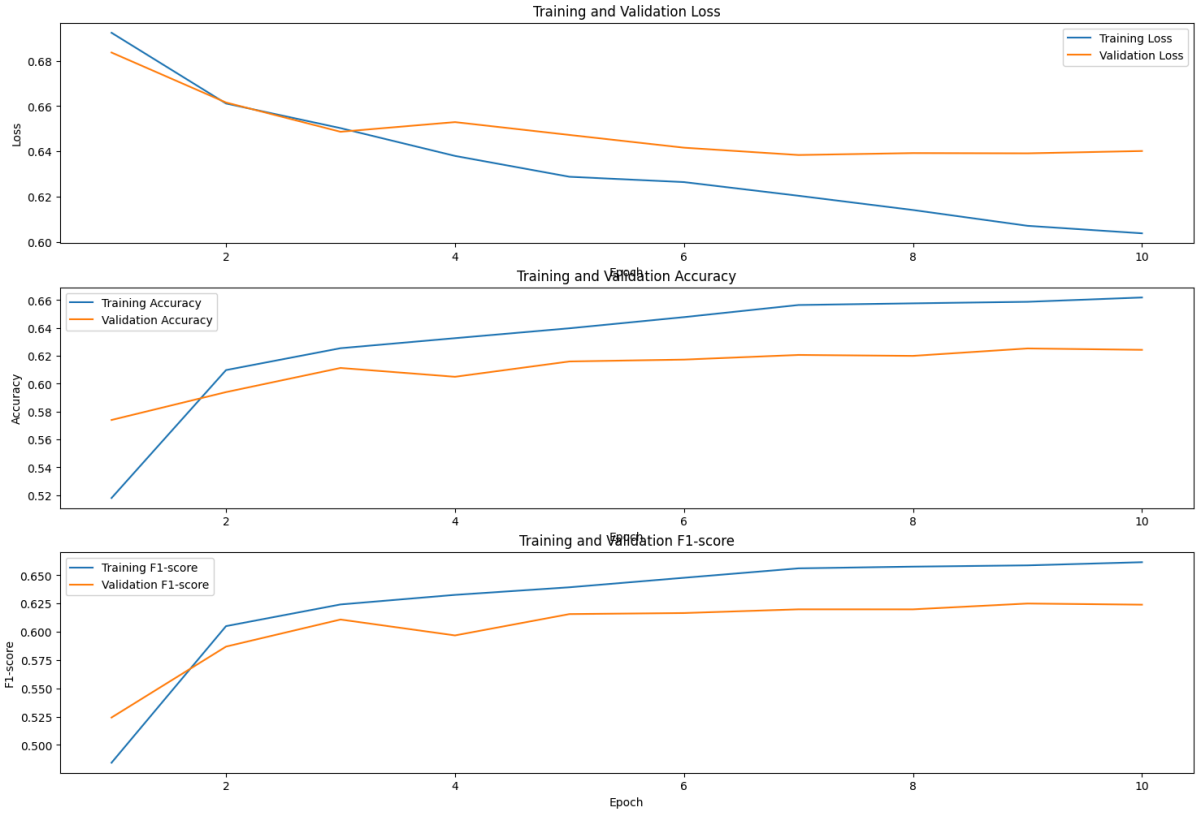
Results

	precision	recall	f1-score	support
Class 0	0.633	0.597	0.614	1500.000
Class 1	0.619	0.654	0.636	1500.000
accuracy	0.625	0.625	**0.625**	0.625
macro avg	0.626	0.625	0.625	3000.000
weighted avg	0.626	0.625	0.625	3000.000

Validation Loss: 0.6392

Best model saved with accuracy: **0.6253**





Model 2: Multi-Modal Feature Fusion

In the second approach, a variety of features is extracted from both the image and textual sources, followed by a multi-modal fusion strategy to make predictions.

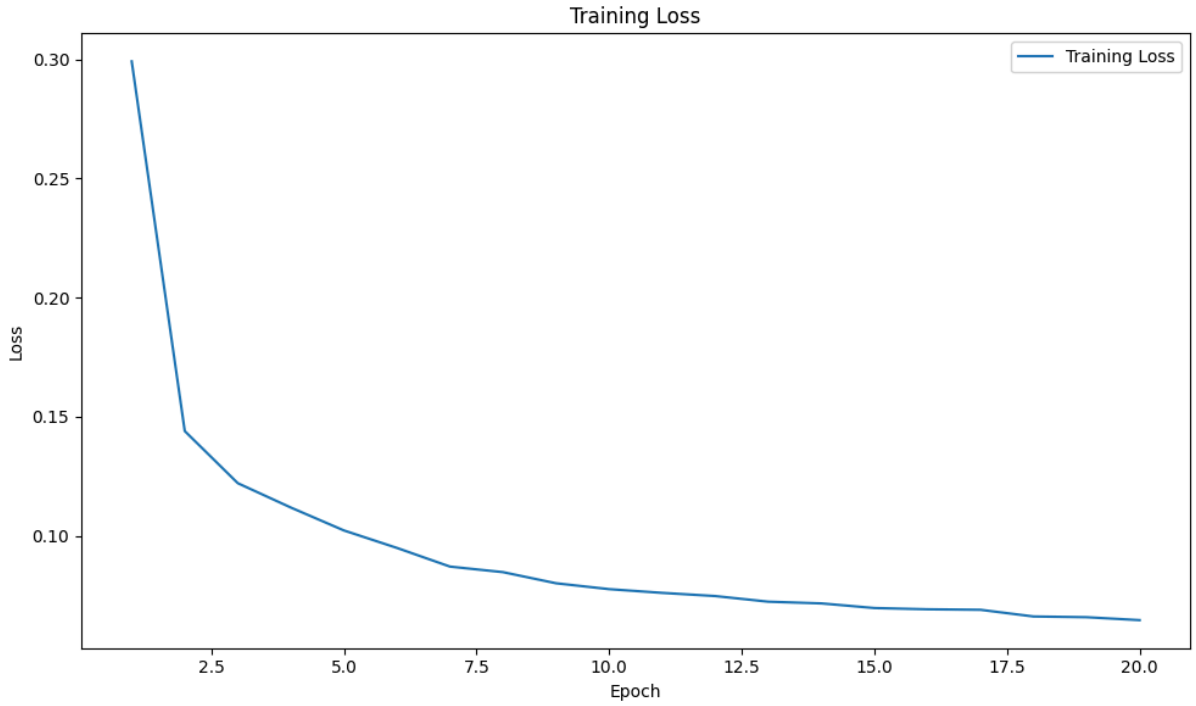
1. **Input Representation:**
    - Image Encoding (Size 2000): Encoded using a pre-trained autoencoder to extract high-level image representations.
    - HOG Features (Size 1944): Histogram of Oriented Gradients (HOG) features extracted from the image.
    - TF-IDF Features (Size 2000): Term Frequency-Inverse Document Frequency features derived from the associated text.
    - Word2Vec (W2V) Features (Size 100): Pre-trained word embeddings used to represent the text. I used average over token size in order to remove length biases.

    To balance the feature sizes, the W2V features are concatenated with themselves 20 times, resulting in a final size of 2000 for all feature vectors.

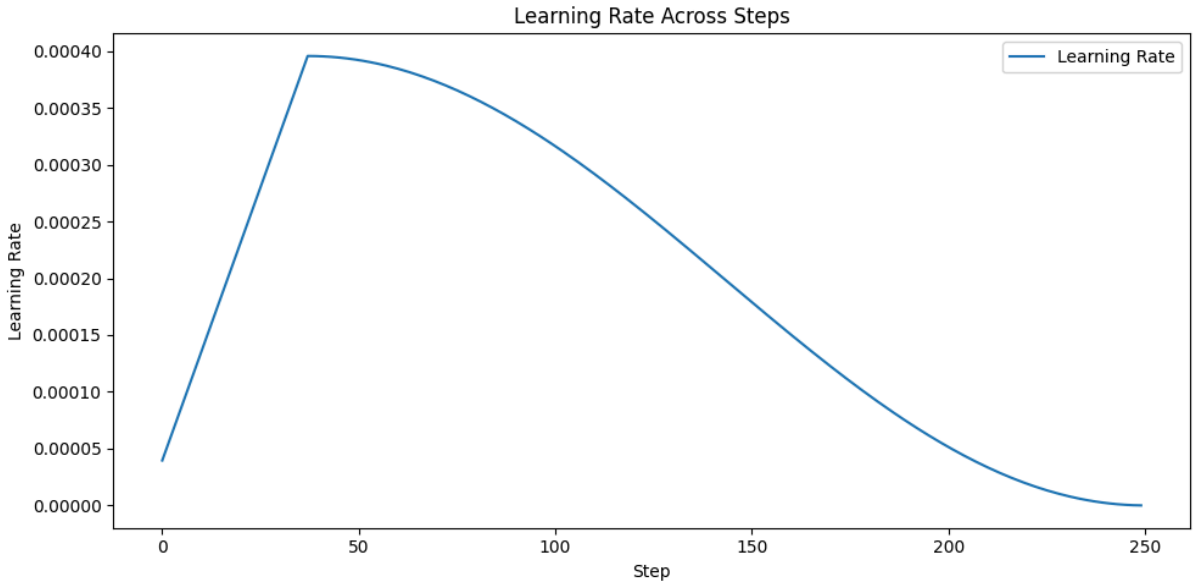
2. **Autoencoder Architecture & Configuration:**

The autoencoder architecture used in this study consists of an encoder and decoder network. The encoder employs four convolutional layers with a stride of 2 and a kernel size

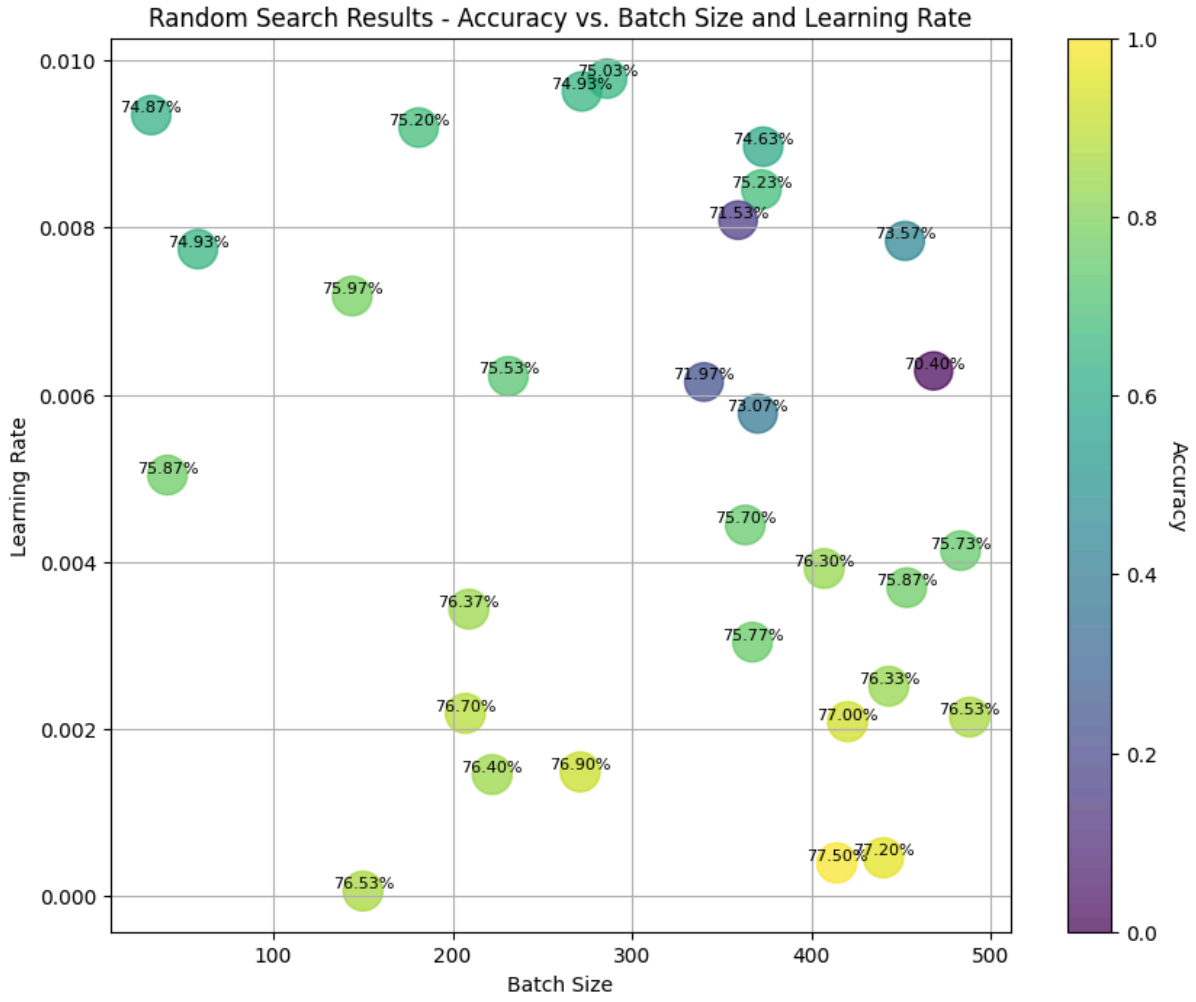
of 3, progressively reducing the spatial dimensions of the image to a feature map of size 7x7x256. A fully connected layer then projects this feature map to a 2000-dimensional vector. For decoding, the network uses a fully connected layer followed by bilinear interpolation for upsampling, with convolutional layers to refine the output. The model is trained using ReLU activations, the AdamW optimizer with a learning rate of 1e-3 and a weight decay of 0.01, MSE loss, a batch size of 256, and trained for 20 epochs. I used gaussian noise over input with mean 0 and std 0.05 for training.



3. **Fusion Architecture:**
    - All extracted features (image encoding, HOG, TF-IDF, and W2V) are concatenated into a single feature vector, which is passed through a fully connected network.
    - This architecture uses weight normalization and GELU activations to ensure stable learning and effective non-linearity.
    - Dropout is applied to prevent overfitting during training.
4. **Optimization and Training Configuration:**
    - Similar to the first pipeline, the model is trained using the AdamW optimizer with a weight decay of 0.1 to regularize the network.
    - A learning rate scheduler is used with the same strategy as the first pipeline (linear warm-up followed by cosine decay).
    - The learning rate and batch size are selected through random search with 30 iterations, where learning rates are randomly chosen between 1e-2 and 1e-6, and batch sizes range from 16 to 512


Learning Rate Across Steps
Learning Rate
Learning Rate
Step
0.00040
0.00035
0.00030
0.00025
0.00020
0.00015
0.00010
0.00005
0.00000
0
50
100
150
200
250

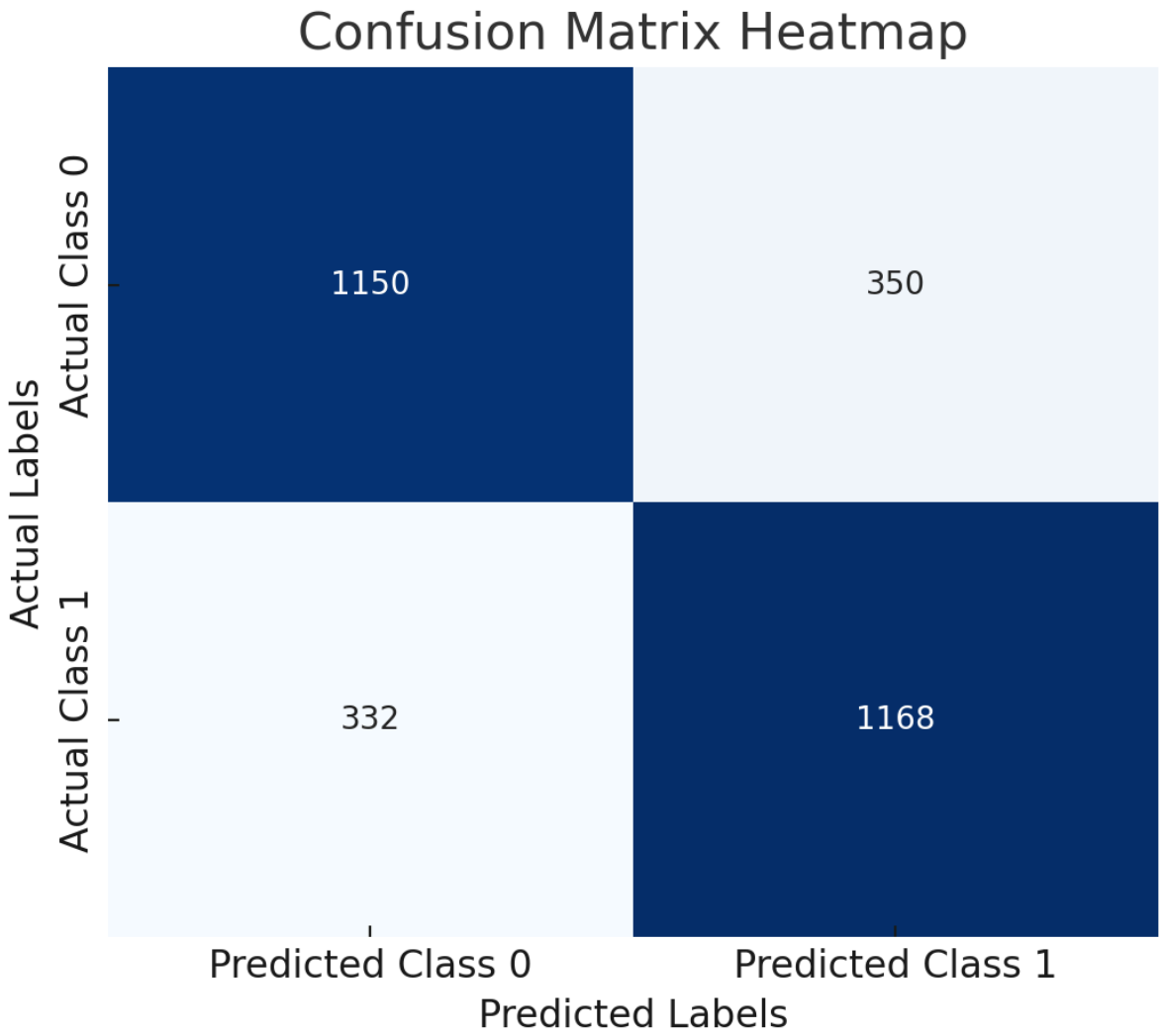
Results

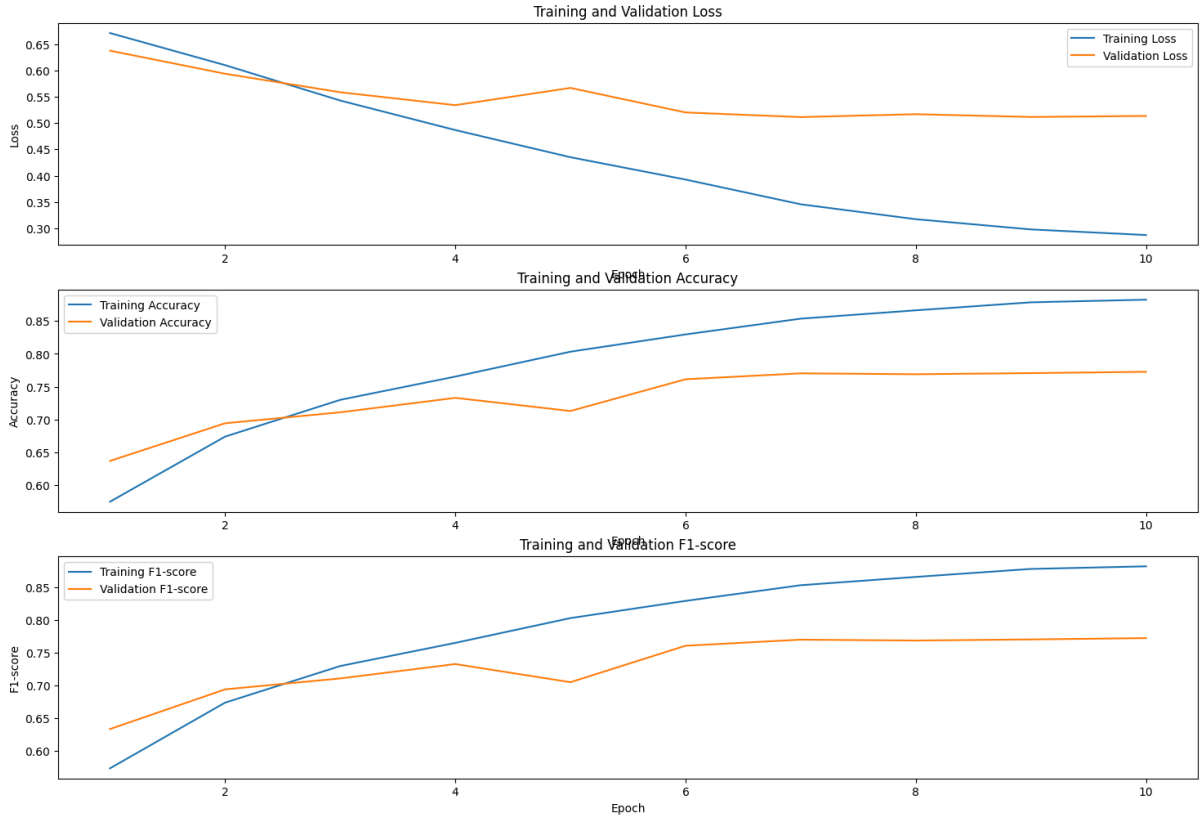

Random Search Results - Accuracy vs. Batch Size and Learning Rate
Learning Rate
Batch Size
Accuracy
0.010
0.008
0.006
0.004
0.002
0.000
100
200
300
400
500
1.0
0.8
0.6
0.4
0.2
0.0
74.87%
75.03%
74.93%
75.20%
74.63%
75.23%
71.53%
74.93%
73.57%
75.97%
75.53%
71.97%
70.40%
73.07%
75.87%
75.70%
75.73%
76.30%
75.87%
76.37%
75.77%
76.33%
76.70%
77.00%
76.53%
76.40%
76.90%
77.50%
77.20%
76.53%

	precision	recall	f1-score	support
Class 0	0.776	0.767	0.771	1500.000
Class 1	0.770	0.778	0.774	1500.000
accuracy	0.773	0.773	**0.773**	0.773
macro avg	0.773	0.773	0.773	3000.000
weighted avg	0.773	0.773	0.773	3000.000

Validation Loss: 0.5135

Best model saved with accuracy: **0.7727**





Conclusions

The results of this study indicate that the second model, which utilizes pre-trained feature extractors for both images and text, significantly outperforms the first model. The higher accuracy achieved by the second model suggests that leveraging pre-extracted features, such as image encodings from an autoencoder, HOG descriptors, and pre-trained word embeddings, provides valuable information that aids in the task of image-sentence pair matching. This is likely due to the complexity of the task and the relatively small size of the dataset, which makes it challenging for the model to learn effective features from scratch. The use of pre-trained knowledge appears to mitigate this issue, allowing the model to better capture the relevant relationships between images and text. Therefore, the success of the second model highlights the importance of feature extraction in tasks involving limited data and complex multi-modal input.