COL774: Machine Learning

Assignment 4: Visual Question Answering

Saharsh Laud
2024MCS2002

Shreyash Chikte
2024MCS2458

May 9, 2025

Contents

1 Introduction **3**

2 Part 1: Dataset **3**

3 Part 2: Data Processing **4**

4 Part 3: Network Architecture Overview **4**

5 Part 4: Image Encoder **4**

6 Part 5: Text Encoder **4**

7 Part 6: Feature Fusion: Cross Attention **5**

8 Part 7: Classifier **5**

9 Part 8: Base Model Training and Evaluation **5**
9.1 Implementation Details (8a) . . . 5
9.2 Training and Validation Curves (8b) . . . 6
9.3 Evaluation on testA (8c) . . . 6
9.4 Correct Prediction Visualizations (8d) . . . 7
9.5 Error Case Visualizations (8e) . . . 8

10 Part 9: Fine-tuning Image Encoder **9**
10.1 Implementation Details . . . 9
10.2 Training and Validation Curves . . . 9
10.3 Evaluation on testA . . . 9
10.4 Correct Prediction Visualizations . . . 10
10.5 Error Case Visualizations . . . 11

11 Part 10: Further Enhancements **12**
11.1 Part 10a: Focal Loss . . . 12
11.1.1 Training and Validation Curves (10a) . . . 12
11.1.2 Evaluation on testA (10a) . . . 12
11.1.3 Visualizations (10a) . . . 13
11.2 Part 10b: BERT Embeddings . . . 14
11.2.1 Training and Validation Curves (10b) . . . 15
11.2.2 Evaluation on testA (10b) . . . 15
11.2.3 Visualizations (10b) . . . 16

12 Part 11: Zero Shot Evaluation **18**
12.1 Transfer Task Evaluation on testB (11a) . . . 18
12.2 Qualitative Analysis on testB (11b) . . . 19

13 Model Links **20**
13.1 Screenshot of Google Drive Folder . . . 21

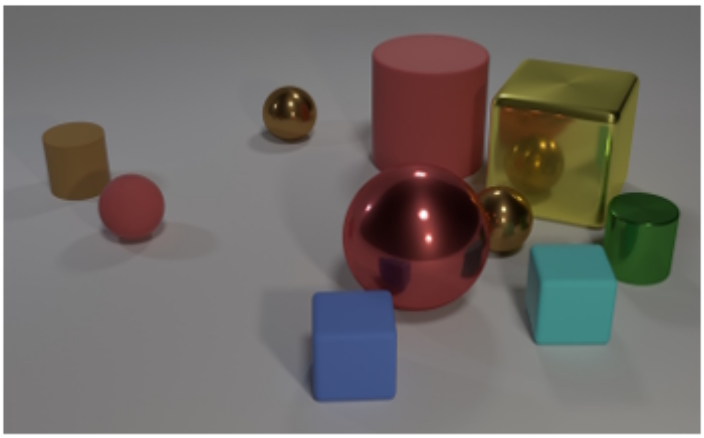
14 Conclusion **21**

1 Introduction

This report details the implementation of a Visual Question Answering (VQA) system as part of Assignment 4 for the COL774 Machine Learning course. The objective is to build a multi-modal transformer model capable of answering textual questions based on an accompanying image from the CLEVR dataset. The report covers the dataset processing, network architecture design, training procedures, and evaluation across several experimental setups, including fine-tuning the image encoder, employing advanced training techniques like Focal Loss and BERT embeddings, and performing zero-shot evaluation on a dataset variant.

2 Part 1: Dataset

For this assignment, the CLEVR (Compositional Language and Elementary Visual Reasoning) dataset is utilized. This dataset consists of synthetic images containing objects that vary in size, shape, color, and material. Each image is paired with question-answer pairs designed to test a range of visual reasoning skills such as attribute identification, counting, comparison, spatial relationships, and logical operations. The primary focus for Parts 8, 9, and 10 is on the 'A' variant of the dataset (trainA, valA, testA). Part 11 uses the 'B' variant for zero-shot transfer evaluation.



Q: Are there an equal number of large things and metal spheres?
Q: What size is the cylinder that is left of the brown metal thing that is left of the big sphere? **Q:** There is a sphere with the same size as the metal cube; is it made of the same material as the small red sphere?
Q: How many objects are either small cylinders or metal things?

Figure 1: Sample image and question pairs from the CLEVR dataset.

3 Part 2: Data Processing

The data processing pipeline involves two main steps:

1. **Tokenization:** Questions are converted into sequences of tokens. Each token is then mapped to a unique numerical ID. For this assignment, the pre-trained BERT tokenizer ($''bert-base-uncased''$) from the HuggingFace Transformers library was used.
2. **Handling Variable-Length Questions:** To ensure consistent input shapes for batch processing, questions are padded to a maximum length (MAX_LENGTH = 32 tokens after tokenization). Shorter sequences are padded, and longer sequences are truncated.

4 Part 3: Network Architecture Overview

The VQA model architecture is inspired by the Transformer model, adapted for the multi-modal VQA task. It comprises four main components:

1. **Image Encoder:** Extracts visual features from the input image.
2. **Text Encoder:** Encodes the input question into a sequence of text features using a Transformer.
3. **Feature Fusion Module:** Integrates the visual and text features using cross-attention to learn a joint representation.
4. **Classifier (Decoder):** Predicts the answer class based on the fused representation.

5 Part 4: Image Encoder

The image encoder uses a pre-trained ResNet101 model loaded from 'torchvision.models'.

- The final global average pooling and fully connected layers of the ResNet101 are removed to obtain a feature map of shape [B, 2048, h, w].
- A linear projection layer ('nn.Linear(2048, dembed)') is applied to transform these features into an embedding dimension ('dembed' = 768).
- For Part 8, the ResNet101 parameters are kept frozen ('$resnet.requires\_grad = False$').

6 Part 5: Text Encoder

The text encoder is based on the Transformer architecture from "Attention is All You Need".

- An embedding layer ('nn.Embedding') is used to learn token representations with '$embed\_dim$' = 768.
- A learnable [CLS] token embedding, similar to BERT, is appended to the word embeddings.
- Learnable positional embeddings ('$nn.Parameter(torch.randn(1, max\_len, embed\_dim))$') are used instead of fixed sinusoidal ones.

- A stack of 6 Transformer encoder layers (‘nn.TransformerEncoder‘) is used, each with 8 attention heads (‘num_layers=6, nhead=8‘).

7 Part 6: Feature Fusion: Cross Attention

A cross-attention mechanism is employed for feature fusion.

- Text features act as the query, while image features serve as keys and values.
- Both text and image feature embeddings have the same dimension (‘dembed‘ = 768).
- Implemented using ‘$nn.MultiheadAttention(embed\_dim = 768, num\_heads = 8)$‘.
- The output corresponding to the [CLS] token from the text encoder (after attending to image features) is passed to the classifier.

8 Part 7: Classifier

The classifier is a simple two-layer Multi-Layer Perceptron (MLP) that predicts the final answer.

- Input: Fused feature representation of dimension ‘dembed‘ (768).
- Architecture: Linear(‘dembed‘, 500) $\rightarrow$ ReLU $\rightarrow$ Linear(500, ‘num_classes‘).
- ‘num_classes‘ is determined by the vocabulary of answers in the training set.

9 Part 8: Base Model Training and Evaluation

9.1 Implementation Details (8a)

The base VQA model was trained end-to-end with a frozen image encoder. The implementation details are as follows:

- **Learning Rate:** 1e-4
- **Batch Size:** 512
- **Optimizer:** Adam (with weight decay 1e-5)
- **Loss Function:** Cross-Entropy Loss
- **Number of Training Epochs:** 25
- **Image Encoder:** ResNet101 (frozen)
- **Text Encoder:** 6 Transformer Encoder layers, 8 attention heads
- **Embedding Dimension:** 768
- **Max Question Length:** 32 tokens
- **Scheduler:** ReduceLROnPlateau (patience=3, factor=0.1)

9.2 Training and Validation Curves (8b)

The training and validation loss and accuracy curves for Part 8 are shown below.

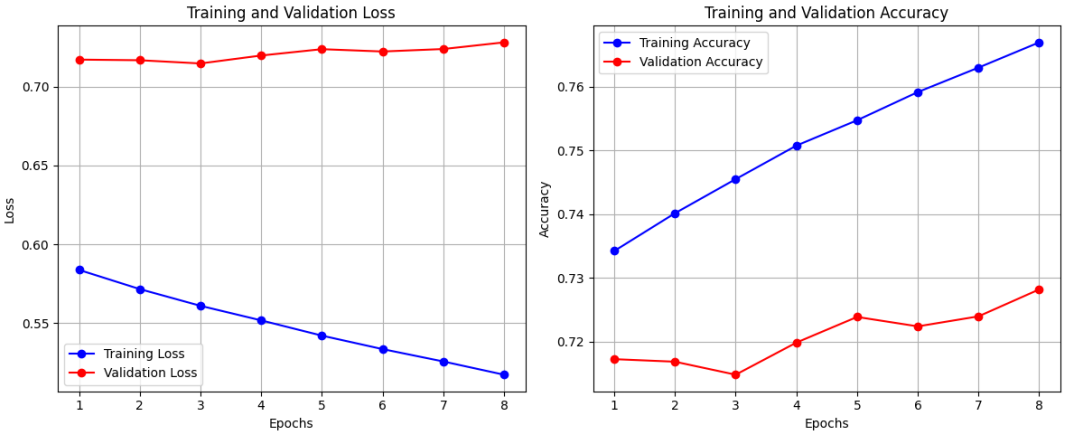


Figure 2: Part 8: Training and Validation Loss and Accuracy Curves.

Observations on Overfitting/Underfitting: The validation accuracy seems to plateau around epoch 15, while training accuracy continues to improve slightly, suggesting the onset of minor overfitting. The validation loss mirrors this, starting to flatten and slightly increase after epoch 15.

9.3 Evaluation on testA (8c)

The model was evaluated on the CLEVR testA dataset. The performance metrics are:

Table 1: Part 8: Performance on testA (Frozen Image Encoder)

Metric	Score
Accuracy	0.7257
Precision	0.7221
Recall	0.7257
F1-score	0.7213

9.4 Correct Prediction Visualizations (8d)

Below are 5 image-question pairs where the model predicted the correct answer.

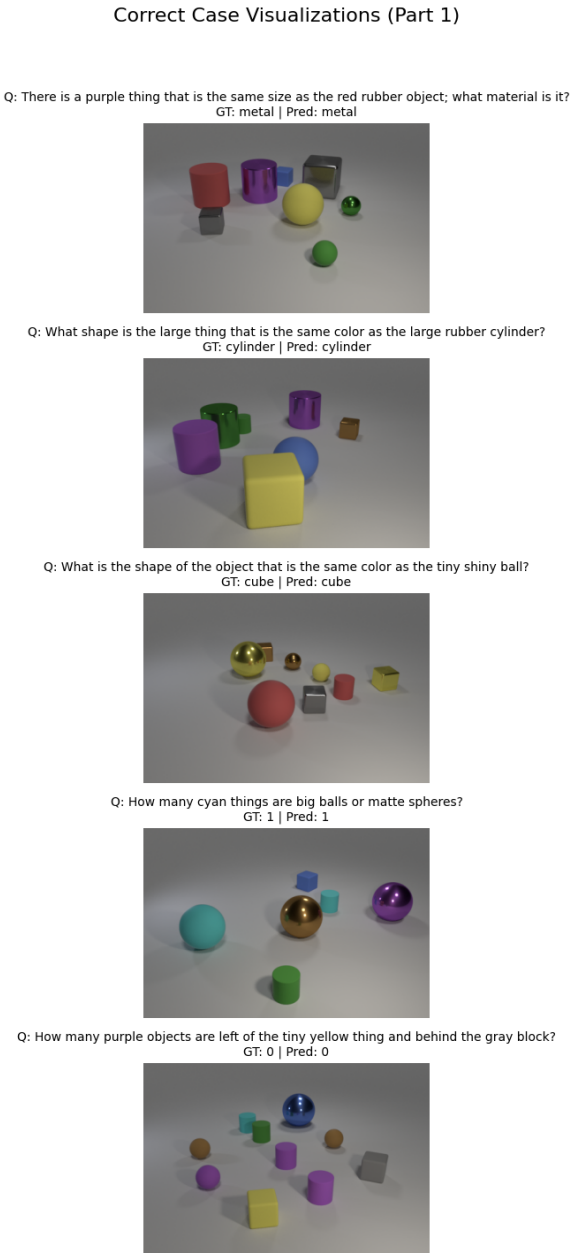


Figure 3: Part 8: Correct Predictions on testA.

9.5 Error Case Visualizations (8e)

Below are 5 image-question pairs where the model's prediction did not match the ground truth.

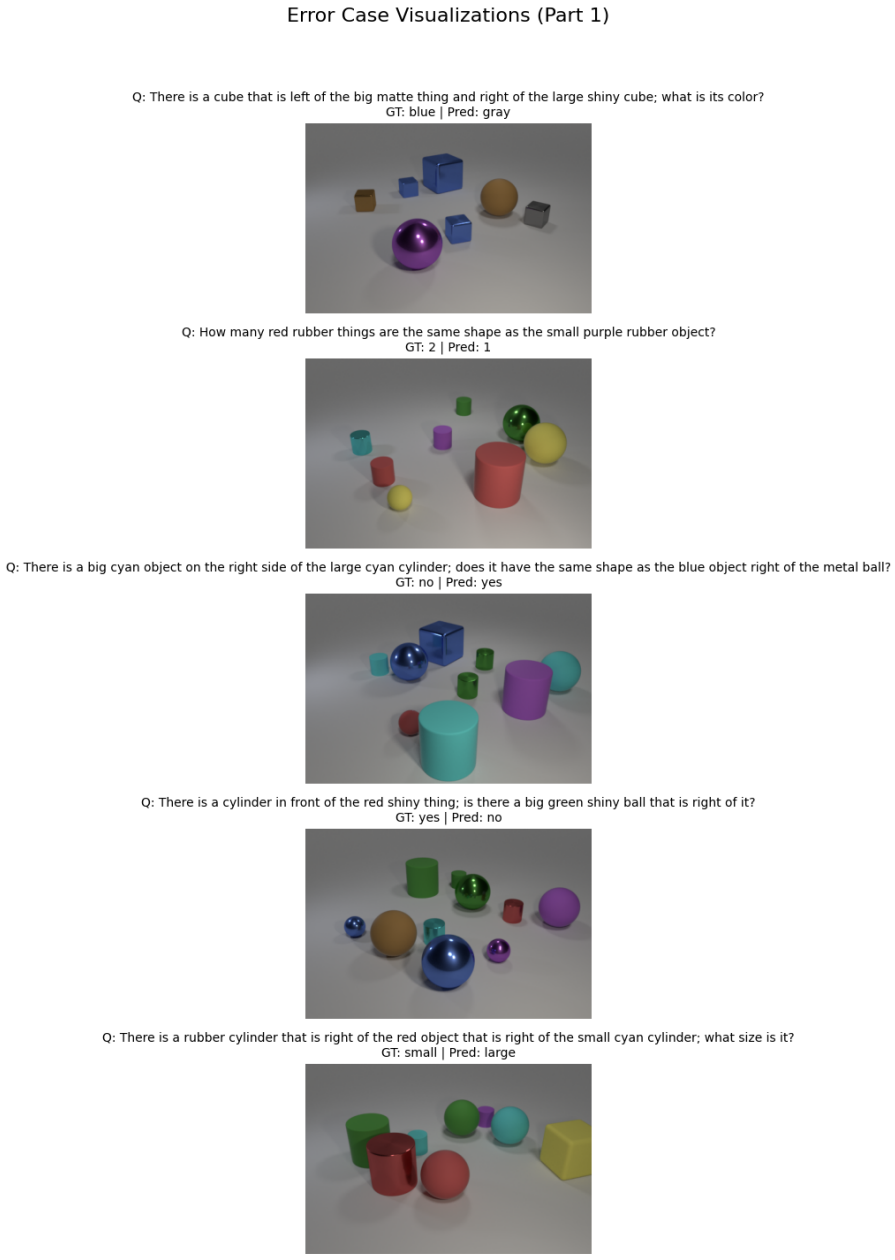


Figure 4: Part 8: Error Case Visualizations on testA.

Observations on Error Cases: The model tends to make errors on questions requiring precise counting of multiple object types simultaneously or complex multi-step spatial reasoning. Errors also occur when there are subtle differences in material or size that are crucial for the answer.

10 Part 9: Fine-tuning Image Encoder

In this part, the image encoder (ResNet101) was unfrozen, and the best-performing model from Part 8 was fine-tuned.

10.1 Implementation Details

Training continued from the Part 8 checkpoint. The ResNet101 'requires_grad' was set to 'True'.

- **Initial Learning Rate for Fine-tuning:** 1e-5
- **Number of Fine-tuning Epochs:** Up to 30 epochs were run.

10.2 Training and Validation Curves

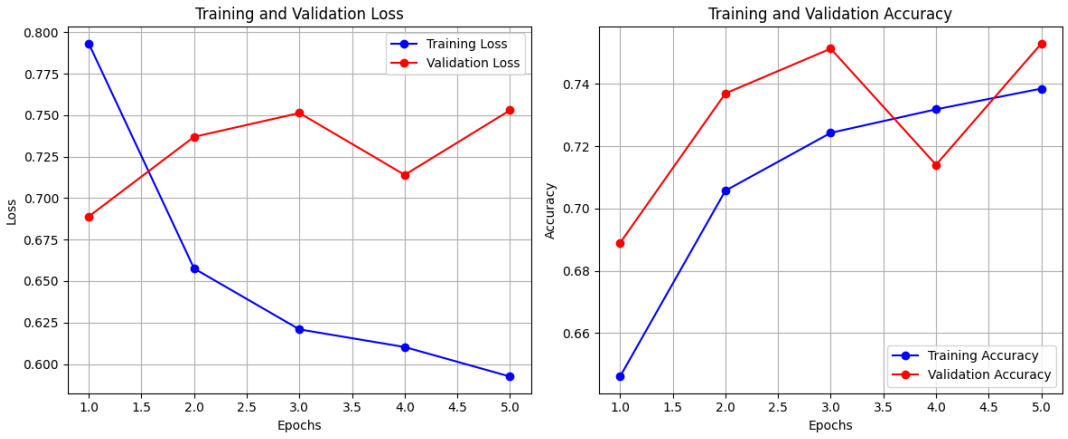


Figure 5: Part 9: Training and Validation Curves (Fine-tuning Image Encoder).

Observations: Fine-tuning the image encoder led to further improvement in validation accuracy. The curves show smoother convergence compared to Part 8 initially, with validation accuracy peaking higher. Some overfitting is still visible in later epochs.

10.3 Evaluation on testA

Table 2: Part 9: Performance on testA (Fine-tuned Image Encoder)

Metric	Score
Accuracy	0.7505
Precision	0.7448
Recall	0.7505
F1-score	0.7408

Fine-tuning the image encoder resulted in an improvement in accuracy from 0.7257 to 0.7505.

10.4 Correct Prediction Visualizations

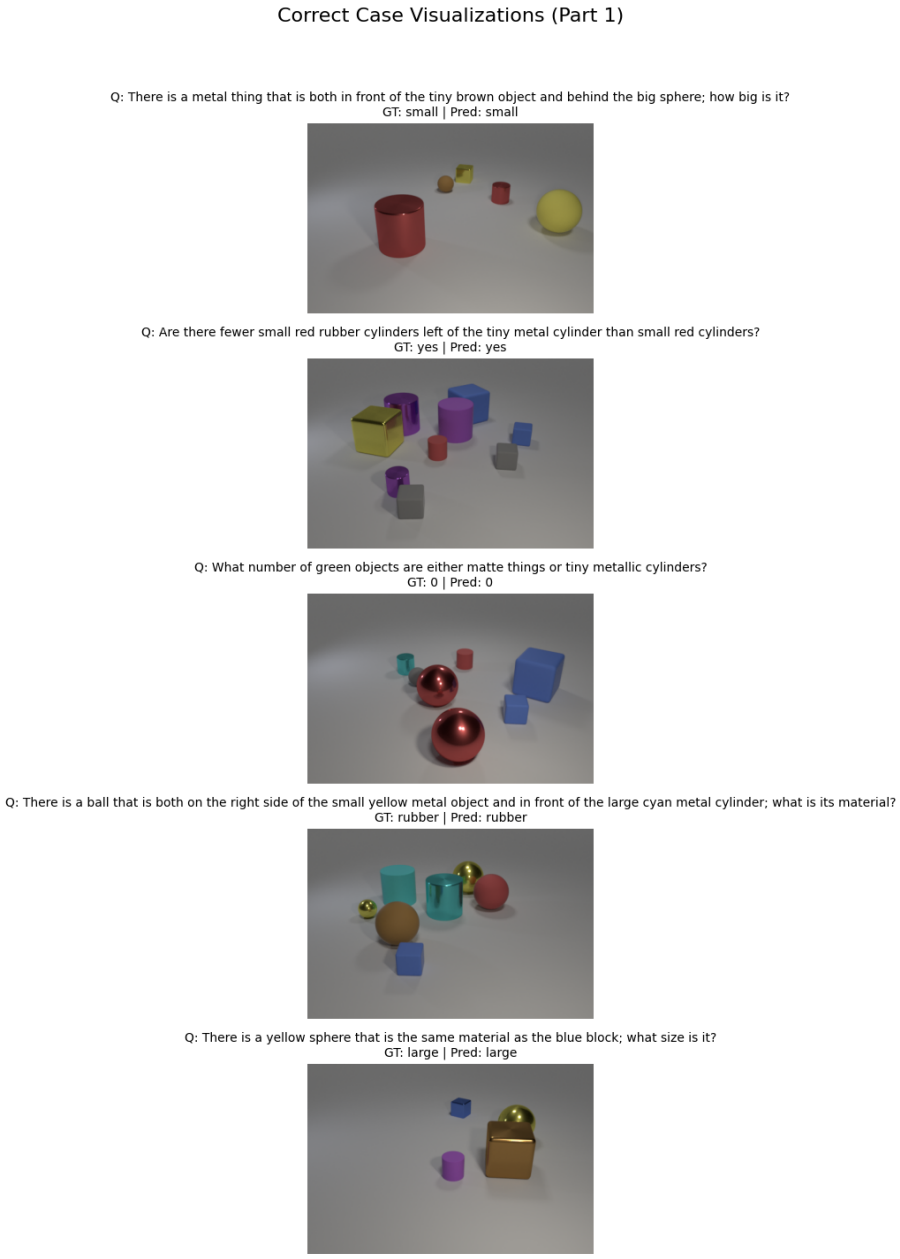


Figure 6: Part 9: Correct Predictions on testA (Fine-tuned Model).

10.5 Error Case Visualizations

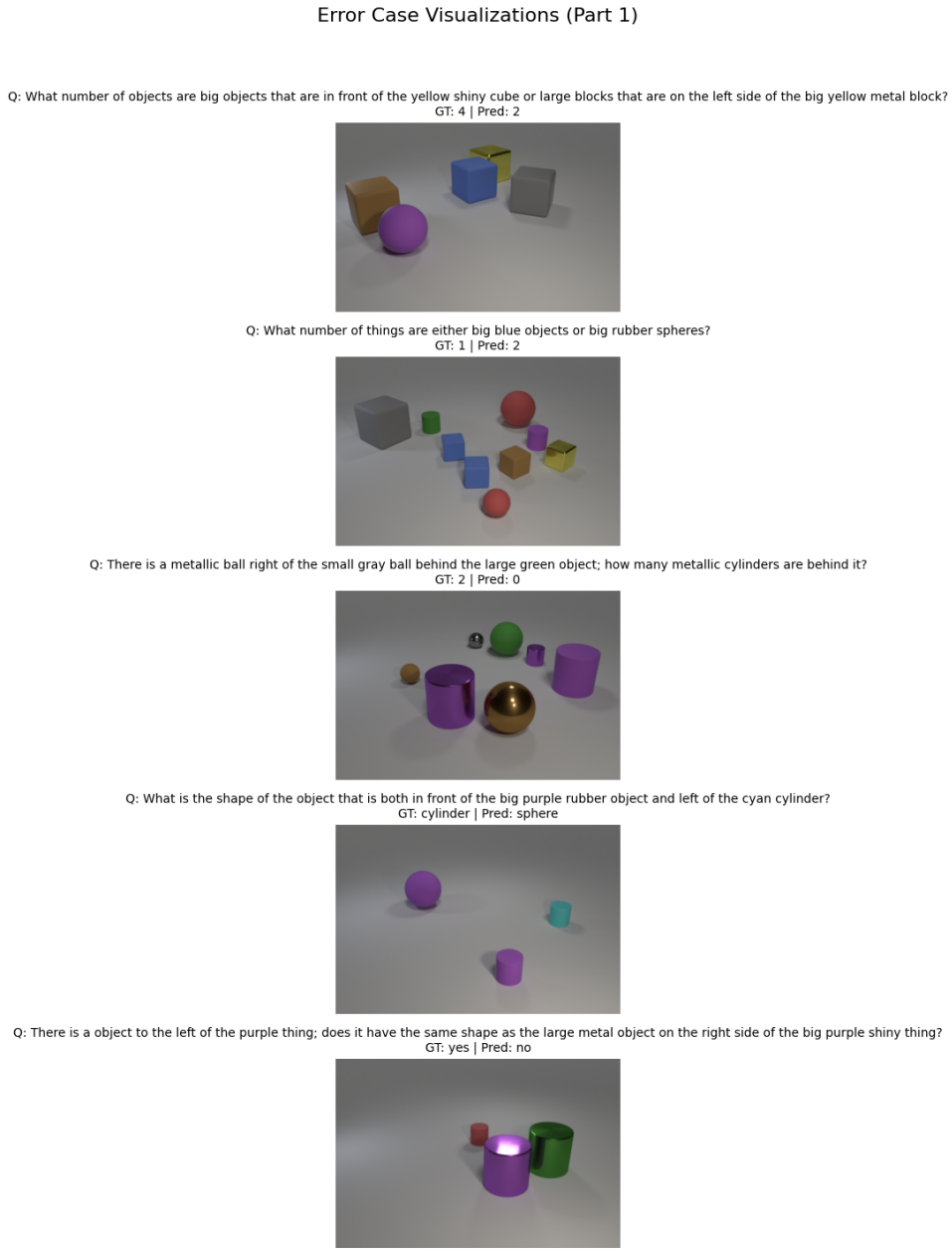


Figure 7: Part 9: Error Case Visualizations on testA (Fine-tuned Model).

Observations on Error Cases (Post Fine-tuning): While overall accuracy improved, similar types of errors related to complex counting and fine-grained attribute differentiation persist, though potentially with slightly reduced frequency.

11 Part 10: Further Enhancements

11.1 Part 10a: Focal Loss

The best model from Part 9 was further trained using Focal Loss.

- **Initial Learning Rate:** 1e-5
- **Loss Function:** Focal Loss (Gamma=2.0, Alpha=1.0)
- **Number of Epochs:** Additional epochs were run (up to epoch 40, continued from epoch 35 checkpoint in notebook).

11.1.1 Training and Validation Curves (10a)

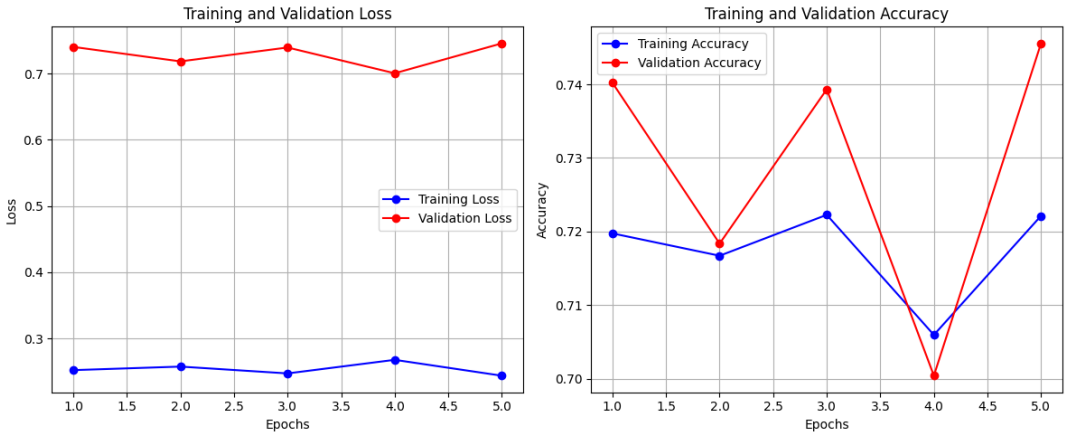


Figure 8: Part 10a: Training and Validation Curves (Focal Loss).

Observations: The use of Focal Loss from the Part 9 checkpoint showed mixed results. While intended to help with class imbalance or hard examples, the validation accuracy did not consistently surpass the previous best. The training curves showed some instability in later epochs. The reported best was from a previous epoch before further Focal Loss training in the provided notebook. The final testA metrics are from the model after Focal loss training.

11.1.2 Evaluation on testA (10a)

Table 3: Part 10a: Performance on testA (Focal Loss)

Metric	Score
Accuracy	0.7440
Precision	0.7466
Recall	0.7440
F1-score	0.7329

Using Focal Loss after Part 9 resulted in an accuracy of 0.7440 which is better than the 0.7257 from Part 8.

11.1.3 Visualizations (10a)

Figure 9: Part 10a: Correct Predictions (Focal Loss)

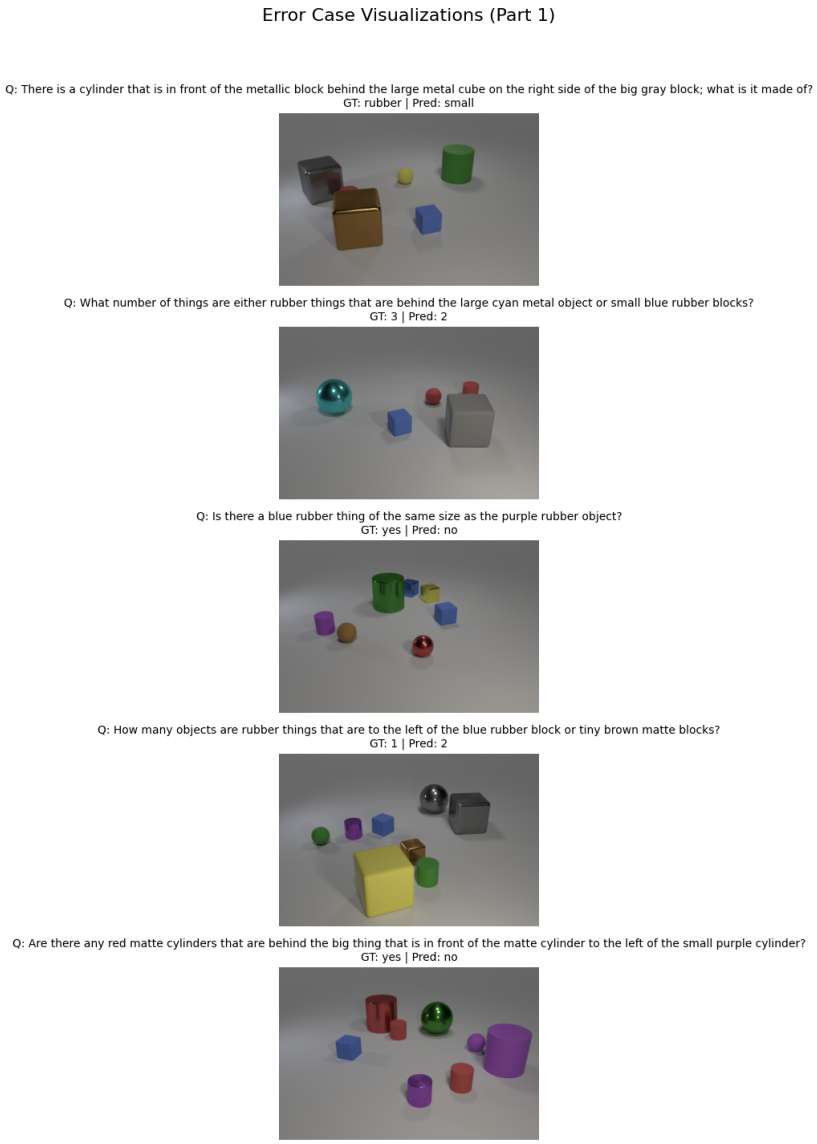


Figure 10: Part 10a: Error Cases (Focal Loss)

Observations on Error Cases (Focal Loss): The error patterns remained largely consistent with previous models, suggesting Focal Loss did not significantly alter the types of mistakes the model was prone to.

11.2 Part 10b: BERT Embeddings

The text encoder's embedding layer was initialized with pre-trained BERT embeddings, and the model (best from Part 10a) was trained further.

- **Initial Learning Rate:** 1e-4
- **Embedding Initialization:** 'bert-base-uncased' pre-trained embeddings.
- **Number of Epochs:** Continued training (up to epoch 35, resuming from an earlier checkpoint).

11.2.1 Training and Validation Curves (10b)

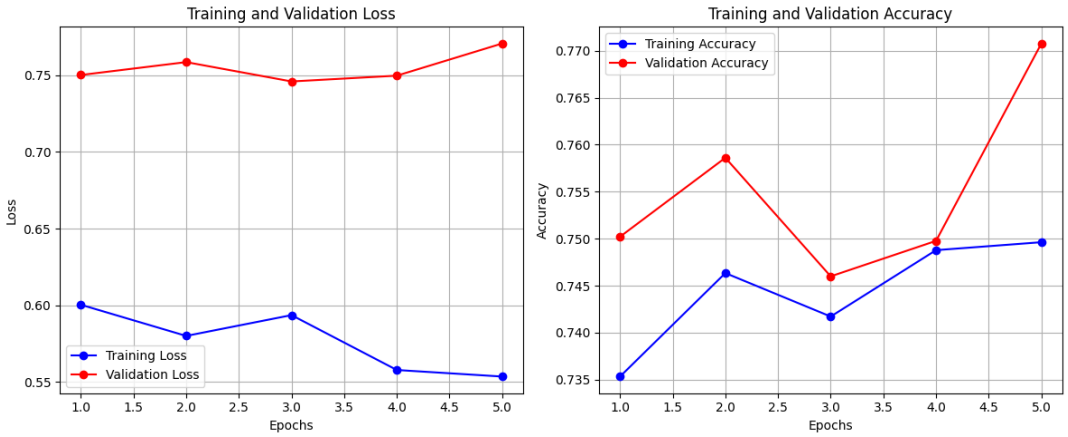


Figure 11: Part 10b: Training and Validation Curves (BERT Embeddings).

Observations: Initializing with BERT embeddings and fine-tuning led to a noticeable improvement in validation accuracy, achieving the highest performance among all experiments on testA. The model benefited from the richer semantic representations provided by BERT.

11.2.2 Evaluation on testA (10b)

Table 4: Part 10b: Performance on testA (BERT Embeddings)

Metric	Score
Accuracy	0.7704
Precision	0.7688
Recall	0.7704
F1-score	0.7677

Initializing with BERT embeddings improved accuracy to 0.7704.

11.2.3 Visualizations (10b)

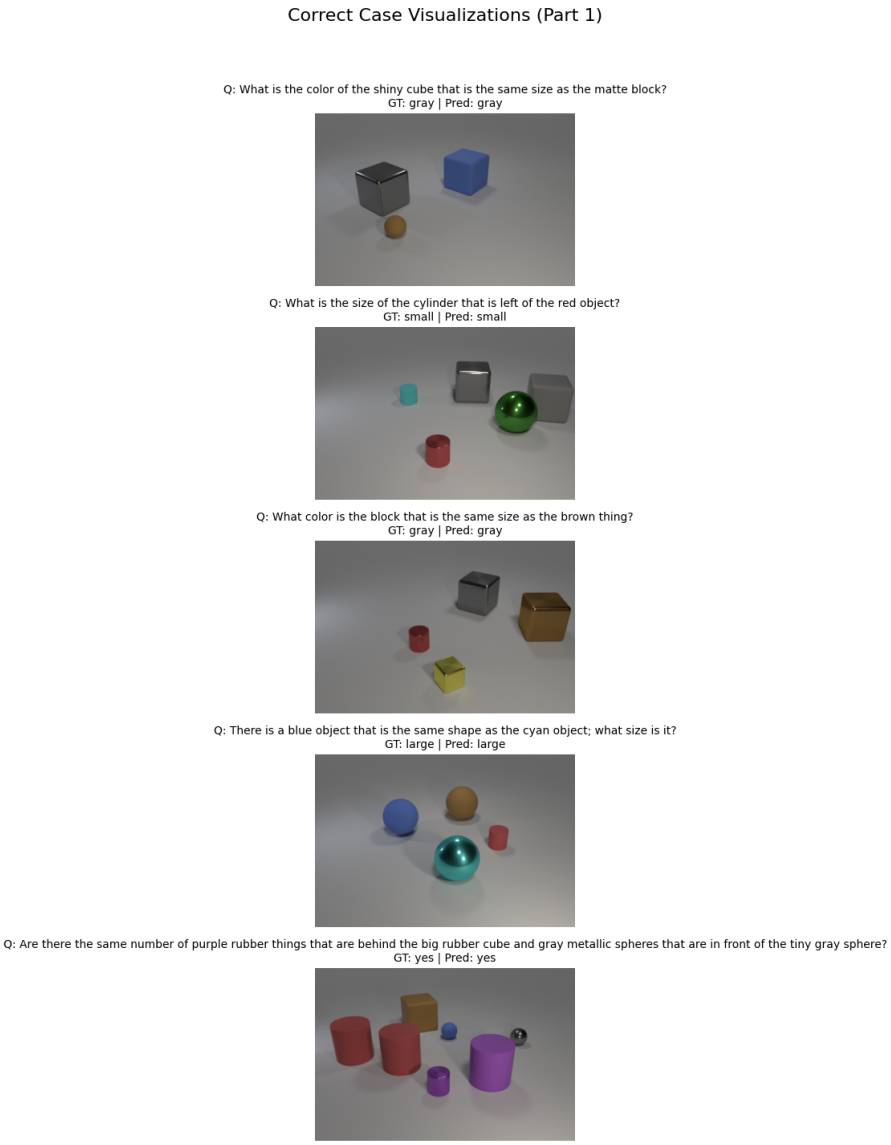


Figure 12: Part 10b: Correct Predictions (BERT Embeddings)

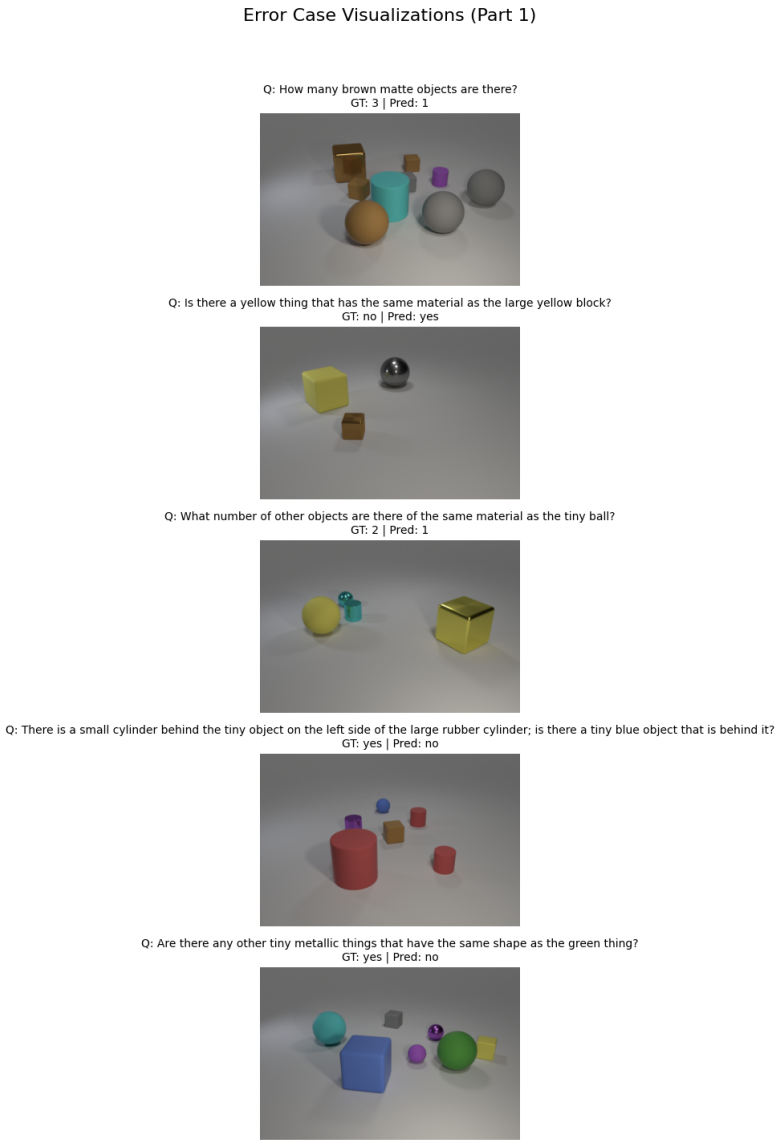


Figure 13: Part 10b: Error Cases (BERT Embeddings)

Observations on Error Cases (BERT Embeddings): While performance improved, the model still exhibited errors in scenarios demanding very complex reasoning or extremely fine-grained distinctions, suggesting limitations inherent in the architecture or dataset complexity.

12 Part 11: Zero Shot Evaluation

The best model (from Part 10b) trained on the type A dataset was evaluated on the type B dataset without any additional training.

12.1 Transfer Task Evaluation on testB (11a)

Table 5: Part 11: Performance on testB (Zero Shot)

Metric	Score
Accuracy	0.6545
Precision	0.6518
Recall	0.6545
F1-score	0.6506

Comparison with testA Performance: The performance on testB (Accuracy: 0.6545) is significantly lower than the performance of the same model on testA (Accuracy: 0.7704 from Part 10b). This drop of approximately 11% in accuracy indicates challenges in generalizing to the distributional shift in object colors and shapes between dataset variants A (cubes: gray, blue, brown, yellow; cylinders: red, green, purple, cyan) and B (cubes: red, green, purple; cylinders: gray, blue, brown, yellow).

12.2 Qualitative Analysis on testB (11b)

Visualizations of predictions on the testB dataset.

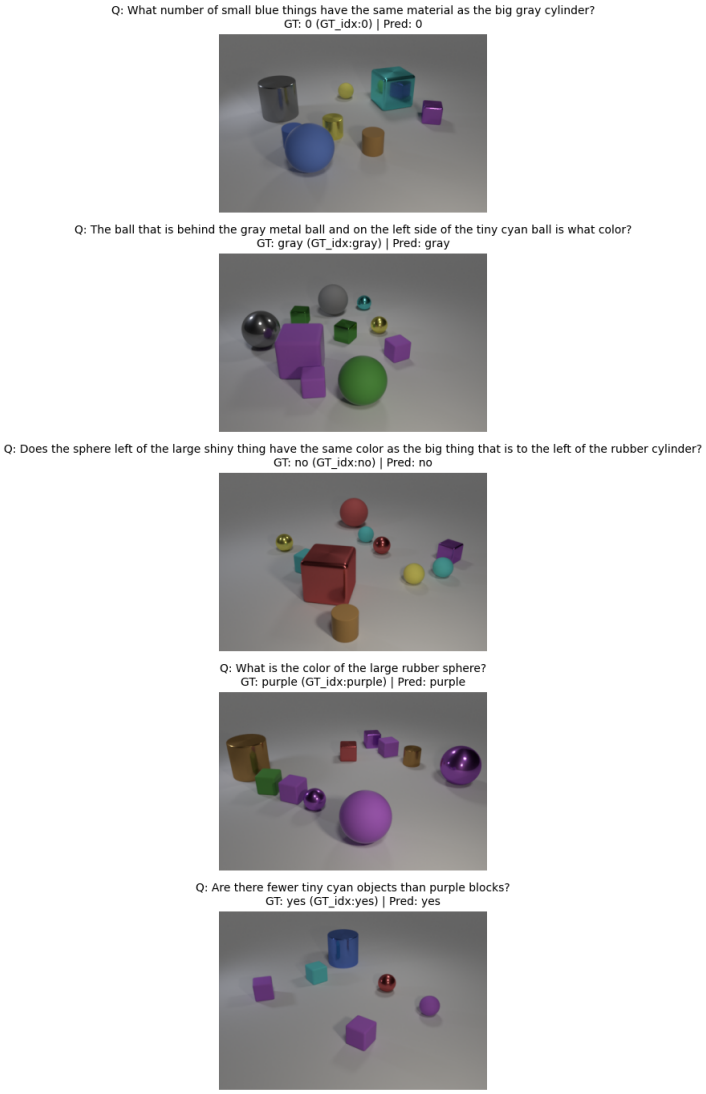


Figure 14: Part 11: Correct Predictions on testB

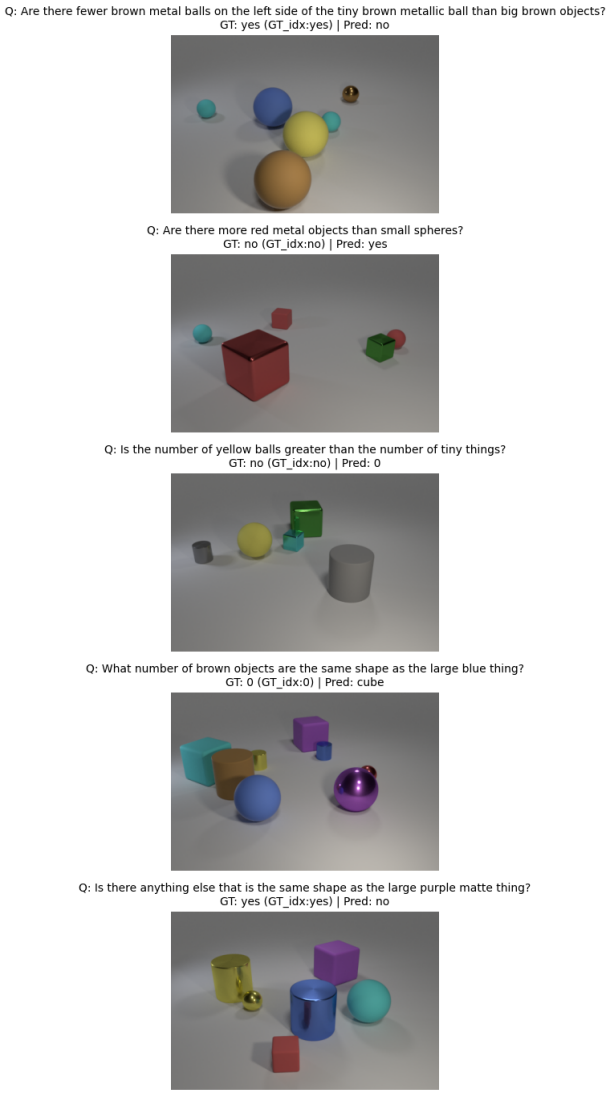


Figure 15: Part 11: Error Cases (Zero Shot)

Analysis of Error Types on testB: The model struggles significantly with the new color-shape combinations in testB. For example, if a question asks about a "red cube", the model might fail because in dataset A (training), cubes were not red. It appears the model has learned strong associations between specific shapes and their training colors, and fails to generalize when these associations are violated. Errors related to counting and spatial relationships also persist, and are likely compounded by the unfamiliar visual attributes.

13 Model Links

The best-performing models for each part have been uploaded to Google Drive. Here is the link to this drive: COL774_Assignment4_Models

13.1 Screenshot of Google Drive Folder

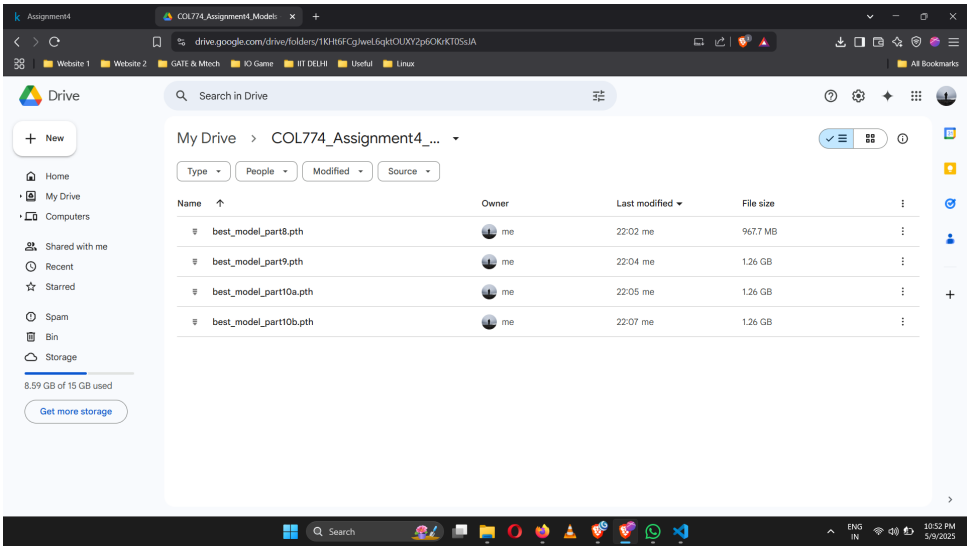


Figure 16: Screenshot of Google Drive folder with uploaded models and timestamps.

14 Conclusion

This assignment involved building and progressively enhancing a VQA model for the CLEVR dataset. Starting with a base transformer architecture with a frozen ResNet101 image encoder (Part 8, testA Accuracy: 0.7257), performance was improved by fine-tuning the image encoder (Part 9, testA Accuracy: 0.7505). Further experiments with Focal Loss (Part 10a, testA Accuracy: 0.7440) gave improvements over Part 8. However, initializing word embeddings with pre-trained BERT representations (Part 10b) provided a significant boost, achieving the best performance on testA (Accuracy: 0.7704). The zero-shot evaluation on the CLEVR testB dataset (Part 11, testB Accuracy: 0.6545) highlighted the model's limitations in generalizing to out-of-distribution visual features, specifically the novel color-shape combinations not encountered during training on testA. This indicates that the model learned spurious correlations between attributes present in the training data variant. Overall, the experiments demonstrate the effectiveness of a transformer-based VQA architecture and show the positive impact of techniques like image encoder fine-tuning and leveraging pre-trained BERT embeddings for text representation. The results also underscore the ongoing challenge of robust out-of-distribution generalization in vision-language models.