Visual Question Answering

Constructing a multimodal model with BERT and ViT

Dibyarup Pal Roll Number: MT2023090
Subhankhi Maiti Roll Number: MT2023113
Saanvi Vishal Roll Number: IMT2021043
International Institute of Information Technology Bangalore, Bangalore

***Abstract*—This paper presents a novel approach to Visual Question Answering (VQA) by integrating multimodal models using Bidirectional Encoder Representations from Transformers (BERT) and Vision Transformers (ViT). VQA is a complex task that requires a deep understanding of both visual and textual information to provide accurate answers to questions about images. Our proposed model leverages the strengths of BERT in natural language processing and ViT in computer vision to create a unified framework that effectively combines textual and visual data.**

***Index Terms*—BERT, ViT, LoRA, Visual Question Answers, Computer Vision**

I. Introduction

MULTIMODAL models have put a huge buzz in the Computer Vision world. Models that can work with two types of data and integrate them to enhance the performance analysis. By leveraging the complementary strengths of various modalities, these models can capture more nuanced context and deliver more accurate results. In Visual Question Answering (VQA), for instance, multimodal models use both visual features from images and semantic features from text to generate precise and contextually relevant answers to questions about the images. This approach significantly improves the model's ability to comprehend and respond to complex queries, showcasing the potential of multimodal learning in artificial intelligence.

BERT (Bidirectional Encoder Representations from Transformers) is a pre-trained language model developed by Google that has revolutionized natural language processing (NLP) by enabling a deep bidirectional understanding of context. By training on large text corpora and leveraging transformers, BERT captures nuanced meanings of words based on their surrounding words in sentences, significantly enhancing the performance of various NLP tasks. In the realm of Visual Question Answering (VQA), BERT has shown promising results by effectively understanding and generating responses to questions about images. When combined with visual features extracted from images, BERT's powerful contextual understanding allows it to interpret and generate accurate answers, pushing the boundaries of VQA performance and demonstrating the model's versatility and strength in integrating textual and visual data.[1] [2]

The Vision Transformer (ViT) is a groundbreaking model that adapts the transformer architecture, originally designed for natural language processing, to image recognition tasks. ViT divides an image into patches, processes these patches as sequences of tokens (akin to words in a sentence), and applies transformer layers to capture global context and relationships between patches. This method allows ViT to leverage the powerful self-attention mechanism of transformers to achieve state-of-the-art performance in image classification tasks. In the domain of Visual Question Answering (VQA), ViT has demonstrated remarkable capabilities by effectively integrating visual data with textual questions. By providing a comprehensive understanding of images and their components, ViT enables more accurate and contextually relevant answers, showcasing its potential to enhance VQA systems through improved visual representation and analysis.[3]

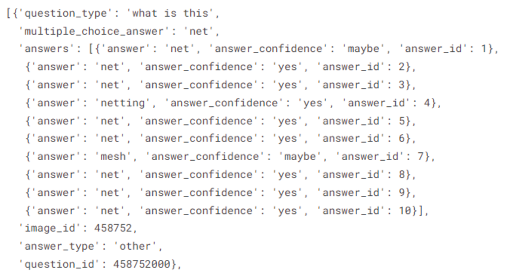
LoRA (Low-Rank Adaptation) is a method used to fine-tune large-scale pre-trained models with a significantly reduced number of trainable parameters by leveraging low-rank decomposition techniques. This approach maintains the core structure of the pre-trained model while injecting task-specific knowledge, making it efficient in terms of both computational resources and storage requirements. In the context of Visual Question Answering (VQA), LoRA enables the adaptation of large vision-language models to the VQA task with minimal computational overhead. By focusing on the most relevant parameters, LoRA enhances the model's ability to understand and respond to questions about the images effectively. This method allows the fine-tuning of powerful models on VQA datasets, resulting in improved performance and accuracy while maintaining efficiency and scalability. [4]

II. Data

We worked with the VQA Dataset released by Georgia Tech: the Balanced Real Dataset [5]. Due to its large size and computational constraints, we will be using only one-fourth of the total training images. We sampled it by taking random images from the total training images.
The dataset consists of three components: VQA Annotations, VQA Input Questions, and VQA Input Images. The Annotation Dataset consists of various answers to a particular question about a specific image.

A. Making our required dataset

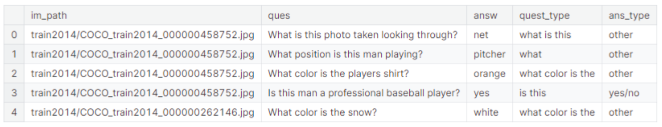
In this research paper, we have made our dataset by combining the VQA Annotations dataset and the VQA Questions dataset. It now consists of the following columns: im_path,



```
[{'question_type': 'what is this',
  'multiple_choice_answer': 'net',
  'answers': [{'answer': 'net', 'answer_confidence': 'maybe', 'answer_id': 1},
   {'answer': 'net', 'answer_confidence': 'yes', 'answer_id': 2},
   {'answer': 'net', 'answer_confidence': 'yes', 'answer_id': 3},
   {'answer': 'netting', 'answer_confidence': 'yes', 'answer_id': 4},
   {'answer': 'net', 'answer_confidence': 'yes', 'answer_id': 5},
   {'answer': 'net', 'answer_confidence': 'yes', 'answer_id': 6},
   {'answer': 'mesh', 'answer_confidence': 'maybe', 'answer_id': 7},
   {'answer': 'net', 'answer_confidence': 'yes', 'answer_id': 8},
   {'answer': 'net', 'answer_confidence': 'yes', 'answer_id': 9},
   {'answer': 'net', 'answer_confidence': 'yes', 'answer_id': 10}],
  'image_id': 458752,
  'answer_type': 'other',
  'question_id': 458752000},
```

Fig. 1: VQA Annotations dataset

ques, answ, ques_type, answ_type which describes its image's corresponding path, Question, Answer, Question Type and Answer type respectively.



	im_path	ques	answ	quest_type	ans_type
0	train2014/COCO_train2014_000000458752.jpg	What is this photo taken looking through?	net	what is this	other
1	train2014/COCO_train2014_000000458752.jpg	What position is this man playing?	pitcher	what	other
2	train2014/COCO_train2014_000000458752.jpg	What color is the players shirt?	orange	what color is the	other
3	train2014/COCO_train2014_000000458752.jpg	Is this man a professional baseball player?	yes	is this	yes/no
4	train2014/COCO_train2014_000000262146.jpg	What color is the snow?	white	what color is the	other

Fig. 2: Required Data

B. Insight into the Data

The dataset consists of 443757 rows. An example image with question and answer is as follows:

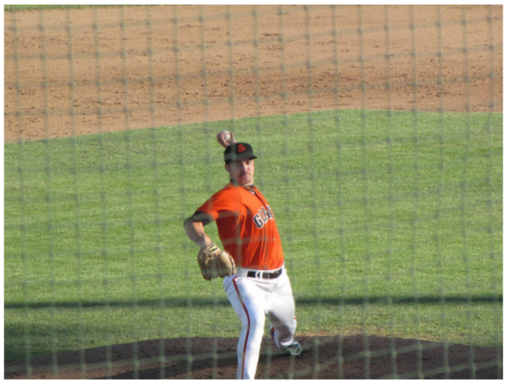
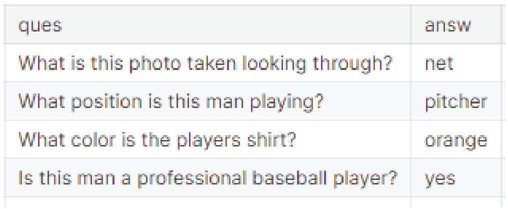


Fig. 3: Example Image



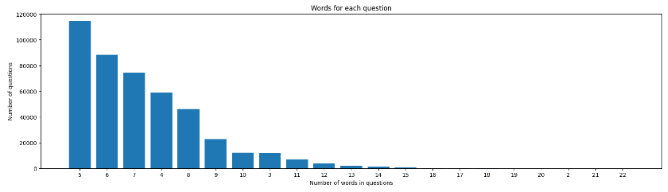
ques	answ
What is this photo taken looking through?	net
What position is this man playing?	pitcher
What color is the players shirt?	orange
Is this man a professional baseball player?	yes

Fig. 4: Fig 3's respective question answer

Before starting the Explanatory Data Analysis(EDA), we expanded the contracted words in questions, words or combinations of words that are shortened by dropping letters and replacing them with an apostrophe, mainly "we'll" to "we will", "can't" to "can not" and such.

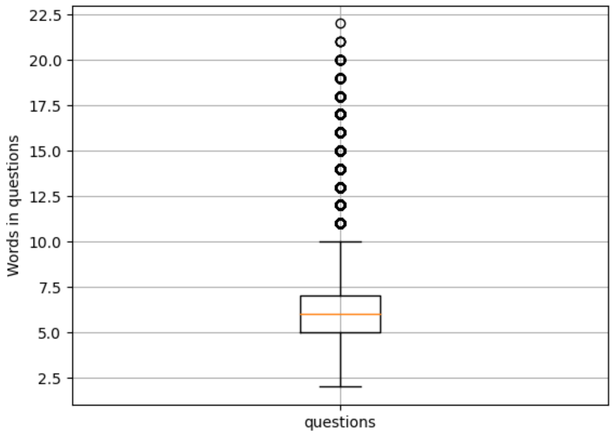
C. Explanatory Data Analysis

- **Inspecting Distribution of questions with the number of words in it**



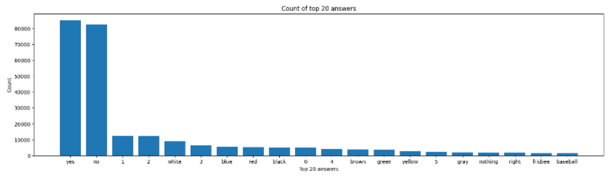
Observation: Maximum questions are in 4-8 words

- **Percentile Distribution of questions with the number of words in it**



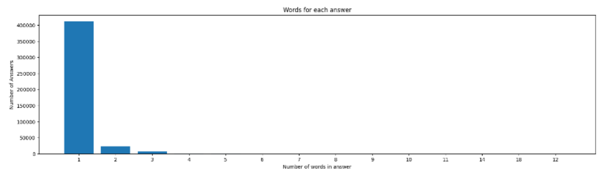
Observation: 50% of questions have 5 to 8 words.

- **Distribution of answers**



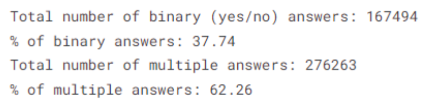
Observation: Most of the answers are one-worded

- **Top 20 answers' distribution**



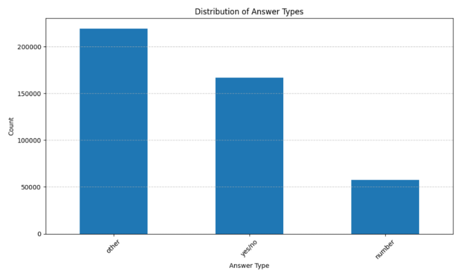
Observation: A very large proportion of the answer has a yes/no answer. Apart from binary answers, most common answers consist of either colours or numbers

- **Inspecting Binary answers v/s multiple answers**



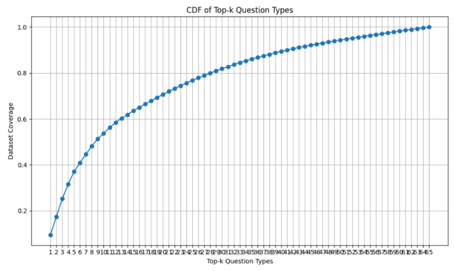
```
Total number of binary (yes/no) answers: 167494
% of binary answers: 37.74
Total number of multiple answers: 276263
% of multiple answers: 62.26
```

- **Distribution of Answer Types**



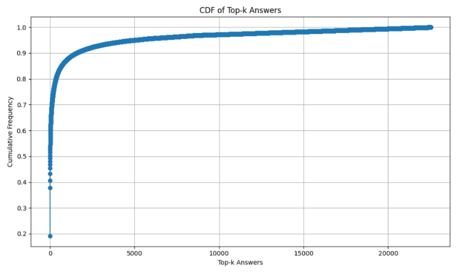
Observation: Other type of answers are more than yes/no and number answer types.

- **CDF of Question Types**



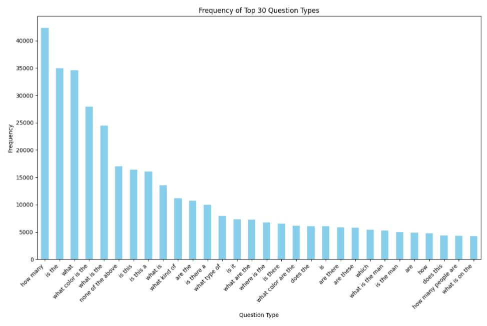
Observation: There isn't a notable bias in the distribution of question types, suggesting that the majority of questions are spread out across various types rather than being concentrated in a few specific categories.

- **CDF of Answer Types**



Observation: Almost 95% of answers are one-word answers although few questions have 2,3 and 4-word answers.

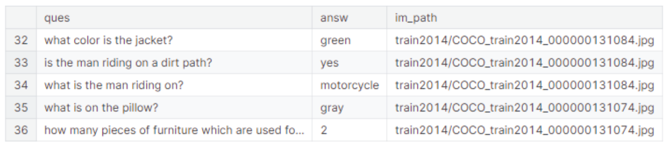
- **Frequency of Top 30 Question Types**



Observation: Question type *How many* has the highest frequency, which implies that the majority of the answer involves numbers.

D. Data Preprocessing

As we have seen almost 95 % of the data has one-worded answers, we will filter out them and work with only those datasets. We also will remove the ques_type column and answ_type column as we have dived into the EDA and got a very good idea of our dataset, so these columns are now unnecessary. The final dataset consists of 102937 rows. Our final dataset looks like this:



	ques	answ	im_path
32	what color is the jacket?	green	train2014/COCO_train2014_000000131084.jpg
33	is the man riding on a dirt path?	yes	train2014/COCO_train2014_000000131084.jpg
34	what is the man riding on?	motorcycle	train2014/COCO_train2014_000000131084.jpg
35	what is on the pillow?	gray	train2014/COCO_train2014_000000131074.jpg
36	how many pieces of furniture which are used fo...	2	train2014/COCO_train2014_000000131074.jpg

Fig. 5: Final Dataset

E. Test-Train-Split and Label Encoding

We will split the dataset into 3 parts: train_df, test_df, and val_df which are our train dataset, test dataset and validation dataset respectively. Train data contains 74114 rows, Train data has 18529 and Validation data has 10294 rows. With the help of a label encoder, we converted all answer columns of all three datasets into numerical values.

III. Methodology

A. Model Architecture

We designed a Visual Question Answering (VQA) model that integrates both visual and textual features to predict answers to questions about images. The VQAModel class is implemented using the PyTorch framework. The model architecture is comprised of the following components:

1) **Vision Transformer**: We utilised a pre-trained Vision Transformer (ViT) model (vit-base-patch16-224-in21k) from Google's library to extract high-dimensional visual features from input images. The ViT model processes the image and produces a hidden representation[3].
2) **Text Transformer**: For the text component, we employed a pre-trained BERT model (bert-base-uncased) to extract semantic features from the input question text. The BERT model encodes the text and provides a contextualised hidden representation[1][2].
3) **Fully Connected Layers**: The outputs from both the vision and text transformers are further processed through fully connected layers:

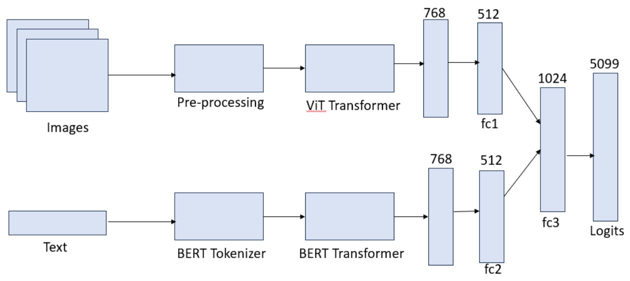


Fig. 6: Model Architecture

- A linear layer (fc1) is defined to process the features from the vision transformer and reduce the dimension of the vision transformer's hidden state from 768 to an output size of 512.
- Another linear layer (fc2) is defined to similarly process the hidden state from the text transformer to map to an output size of 512.
- These processed features are concatenated and these concatenated features(of size 1024) are passed through a final linear layer (fc3) that maps the combined features to the output space corresponding to the number of possible answers.

Algorithm 1 VQA Multimodal model

```
Require: Dataset consisting of Image, Question corresponding to the image and
Ensure: Give answers to the given question based on the provided image
1: Model Initialization:
     • vision_transformer: Load pre-trained Vision Transformer (vit-base-patch16-224-in21k)
     • text_transformer: Load pre-trained BERT model (bert-base-uncased)
     • Define fully connected layers:
         I   fc1: Linear layer for vision features (input size: vision transformer hidden size, output size: 512)
         II  fc2: Linear layer for text features (input size: text transformer hidden size, output size: 512)
         III fc3: Linear layer for combined features (input size: 1024, output size: num_answers)
2: Forward Pass:
     • Input: Image and text
     • Vision Transformer:
         I Pass the image through the Vision Transformer to obtain vision_outputs
     • Text Transformer:
         I Pass text through the BERT model to obtain text_outputs
     • Feature Extraction:
         I  Apply fc1 to the vision transformer's last hidden state to get vision_feats
         II Apply fc2 to the text transformer's last hidden state to get text_feats
     • Feature Combination:
         I Concatenate vision_feats and text_feats to form combined_feats
     • Prediction:
         I Pass combined_feats through fc3 to get logits..
     • Return: logits
```

B. Model Architecture using LoRA

We incorporated LoRA in the above model to optimise the fine-tuning process. Our model architecture is designed as follows:

Algorithm 2 VQA Multimodal model with LoRA

```
Require: Dataset consisting of Image, Question corresponding to the image and
Ensure: Give answers to the given question based on the provided image
  Model Initialization:
     • vision_transformer: Load pre-trained Vision Transformer (vit-base-patch16-224-in21k)
     • text_transformer: Load pre-trained BERT model (bert-base-uncased)
     • Apply LoRA to both transformers with the specified rank.
     • Define fully connected layers:
         I   fc1: Linear layer for vision features (input size: vision transformer hidden size, output size: 512)
         II  fc2: Linear layer for text features (input size: text transformer hidden size, output size: 512)
         III fc3: Linear layer for combined features (input size: 1024, output size: num_answers)
2: Forward Pass:
     • Input: Image and text
     • Vision Transformer:
         I Pass the image through the Vision Transformer to obtain vision_outputs
     • Text Transformer:
         I Pass text through the BERT model to obtain text_outputs
     • Feature Extraction:
         I  Apply fc1 to the vision transformer's last hidden state to get vision_feats
         II Apply fc2 to the text transformer's last hidden state to get text_feats
     • Feature Combination:
         I Concatenate vision_feats and text_feats to form combined_feats
     • Prediction:
         I Pass combined_feats through fc3 to get logits..
     • Return: logits
```

1) **Vision Transformer**:
   - We utilise a pre-trained Vision Transformer (ViT) model (vit-base-patch16-224-in21k) to extract high-dimensional visual features from input images[3].
   - LoRA (Low-Rank Adaptation) is applied to the Vision Transformer to enable efficient fine-tuning by significantly reducing the number of trainable parameters. We set the LoRA rank to 16[4].
2) **Text Transformer**:
   - A pre-trained BERT model (bert-base-uncased) is employed to extract semantic features from input questions[1][2].
   - LoRA is similarly applied to the BERT model, with a rank of 16, to facilitate efficient fine-tuning[4].
3) **Fully Connected Layers**: The outputs from both the

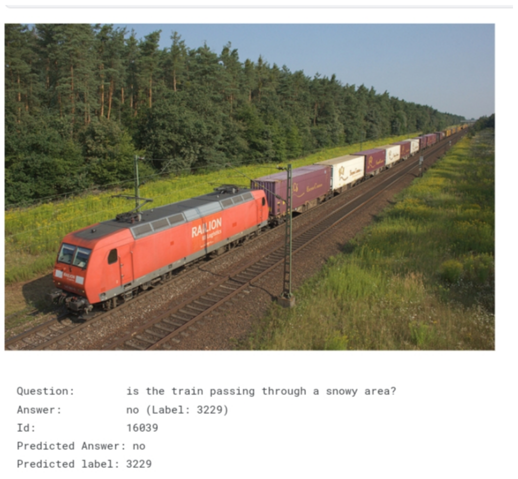
vision and text transformers are further processed through fully connected layers:

- A linear layer (fc1) that processes the hidden state output from the Vision Transformer, reducing its dimensionality from 768 to 512
- Another linear layer (fc2) processes the hidden state output from the BERT model, similarly reducing its dimensionality from 768 to 512.
- A final linear layer (fc3) that combines the features from both modalities to create features of size 1024 and maps them to the output space, corresponding to the number of possible answers (num_answers)

IV. Testing Real-Time Input

We tested two models with real-time input, using input images accompanied by questions. Below are a few random examples provided for inference.

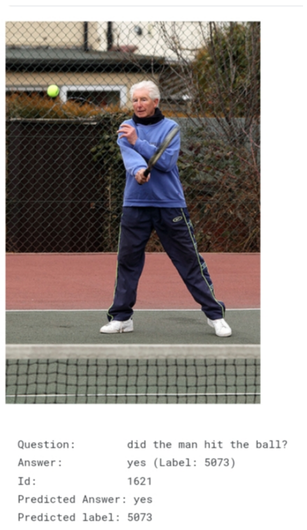
1) Testing images for the model without LoRA



```
Question:        is the train passing through a snowy area?
Answer:          no (Label: 3229)
Id:              16039
Predicted Answer: no
Predicted label: 3229
```

Fig. 7: Testing image 1

The model predicted the answer correctly exactly matching with the given output answer.

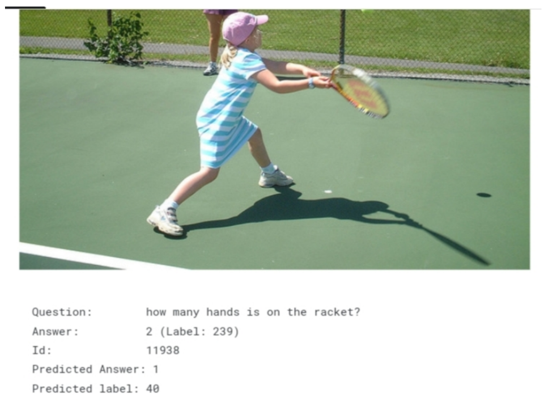


```
Question:        did the man hit the ball?
Answer:          yes (Label: 5073)
Id:              1621
Predicted Answer: yes
Predicted label: 5073
```

Fig. 8: Testing image 2

The model predicted the answer correctly exactly matching the given output answer

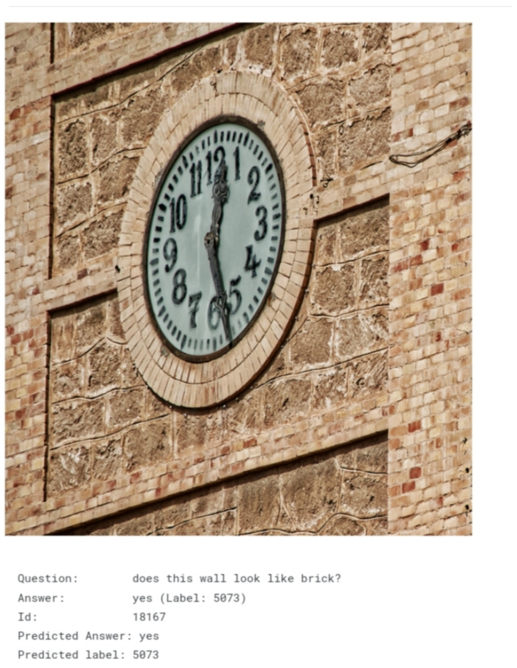
2) Testing images for the model with LoRA



```
Question:        how many hands is on the racket?
Answer:          2 (Label: 239)
Id:              11938
Predicted Answer: 1
Predicted label: 40
```

Fig. 9: Testing Image 3

The model couldn't predict the answer correctly, mismatching with the given output answer.



```
Question:        does this wall look like brick?
Answer:          yes (Label: 5073)
Id:              18167
Predicted Answer: yes
Predicted label: 5073
```

Fig. 10: Testing Image 4

The model predicted the answer correctly matching the t given output answer.

V. Results

The visual question-answering (VQA) task involves several key stages, including data preprocessing, model training, and inference, each contributing to the overall performance and accuracy of the model. Our metrics experiment involves comparative analysis based on Time taken for Training, Accuracy, F1 score, Precision and Recall.

A. Training Time

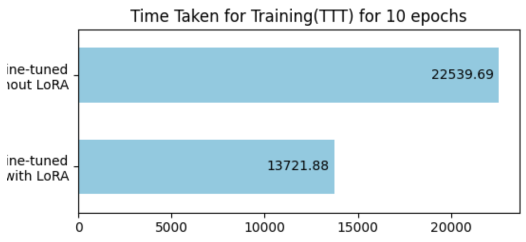


Fig. 11: Time Taken while training

Time taken for training without Lora is 22539.69 seconds while that of LoRA is 13721.88 seconds. This is because LoRA uses a reduced number of trainable parameters and it efficiently updates parameters and preserves pre-trained weights.

B. Accuracy without LoRA

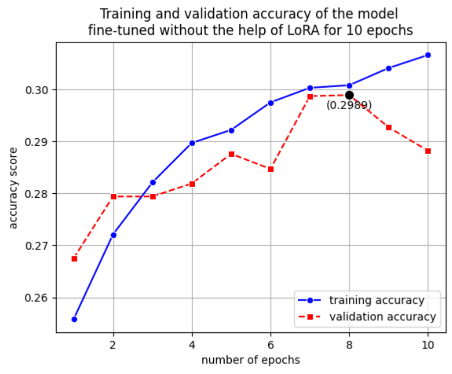


Fig. 12: Accuracy of training and validation without LoRA

We ran our model for 10 epochs for both the training and validation datasets. We can see with each increasing epoch, the accuracy for the training dataset is increasing but that of validation is fluctuating. We are taking the weights of that epoch where the validation accuracy is the highest and saving it. Here the highest accuracy for validation (marked by the black dot) is 0.2989 at epoch 8.

C. Accuracy with LoRA

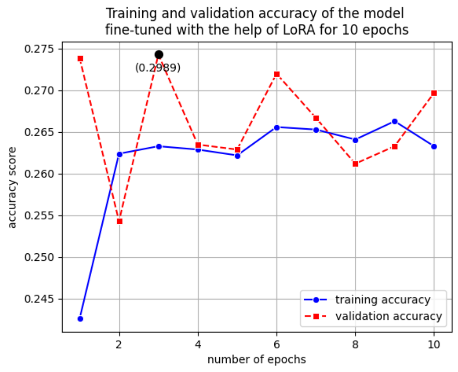


Fig. 13: Accuracy of training and validation with LoRA

We ran our model with LoRA for 10 epochs on both the training and validation datasets. With each successive epoch, the accuracy of the training dataset increased, while the validation accuracy fluctuated. We saved the weights from the epoch with the highest validation accuracy, which was 0.2989 at epoch 3 (marked by the black dot).

D. Performance for Different Metrics for Both Our Models

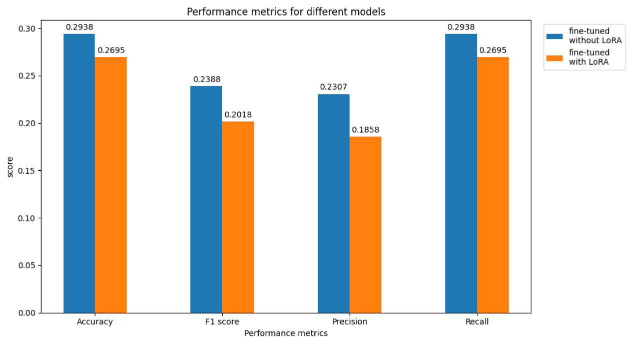


Fig. 14: Performance Metrics

Metric	Value
Accuracy	0.2938
F1 Score	0.2388
Precision	0.2307
Recall	0.2938

TABLE I: Model without LoRA Performance Metrics

1) Performance Metrics for Model without LoRA: The Table I shows the key metrics used to evaluate the model's performance without LoRA.

Metric	Value
Accuracy	0.2695
F1 Score	0.2018
Precision	0.1858
Recall	0.2695

TABLE II: Model with LoRA Performance Metrics

2) Performance Metrics for Model with LoRA: The Table II shows the key metrics used to evaluate the model's performance with LoRA.

E. Analysis of the Performance

- **Accuracy**: The model without LoRA achieves higher accuracy (0.2938) compared to the model with LoRA (0.2695). This indicates that the model without LoRA is better at correctly predicting the overall outcomes.
- **F1 Score**: The F1 score is also higher for the model without LoRA (0.2388) compared to the model with LoRA (0.2018). The F1 score is a harmonic mean of precision and recall, which means the balance between precision and recall is better in the model without LoRA.
- **Precision**: The model without LoRA shows better precision (0.2307) compared to the model with LoRA (0.1858). Precision measures the number of true positive

predictions relative to the number of positive predictions made, indicating that the model without LoRA is better at minimizing false positives.
- **Recall**: Both models have the same recall values as their accuracy, which is a consistent behaviour in the metrics. The recall for the model without LoRA (0.2938) is higher than that of the model with LoRA (0.2695), indicating that the model without LoRA is better at identifying true positives.

VI. Conclusion

Fine-tuning the model without LoRA took nearly twice as long as fine-tuning with LoRA. However, it is notable that the accuracy of the model without LoRA is only slightly higher than that of the model with LoRA.
The overall performance of the model without LoRA is superior across all key metrics, including accuracy, F1 score, precision, and recall. This suggests that for this particular task and dataset, incorporating LoRA does not enhance the model's performance. Instead, it results in a slight decrease in effectiveness.
This conclusion indicates that while LoRA can be useful for reducing the number of trainable parameters and computational overhead, it might not always lead to improved model performance. Therefore, careful consideration and evaluation are necessary when choosing to implement LoRA or similar techniques in machine learning models.

Acknowledgment

The authors would like to thank Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova for their seminal work on BERT, which has been a significant foundation for our research **devlin2018bert**. Their contributions to the field of natural language processing are greatly appreciated.

We also express our gratitude to Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby for their innovative work on transformers for image recognition **dosovitskiy2020image**, which has provided valuable insights.

Additionally, we acknowledge Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen for their research on Low-Rank Adaptation of Large Language Models **hu2021lora**, which has informed and guided aspects of our methodology.

References

[1] Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. Google AI Language

[2] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805*.

[3] Alexey Dosovitskiy, Lucas Beyer Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby *AN IMAGE IS WORTH 16X16 WORDS:TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE.*

[4] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen *LoRA: Low-Rank Adaptation of Large Language Models*

[5] Yash Goyal, Tejas Khot, Douglas Summers Stay, Dhruv Batra, Devi Parikh Making the V in VQA Matter: Elevating the Role of Image Understanding in Visual Question Answering, Conference on Computer Vision and Pattern Recognition (CVPR), Dataset available at https://visualqa.org/