VR Part 2: Mini Project - Visual Question Answering

Objectives

- To build a model for Visual Question Answering which takes as input an image and a corresponding question and gives an answer to the question.
- Fine-tune pre-trained models with and without the help of LoRA and compare the training time and performance of the fine-tuned models.

Dataset

For this project, you will be using the VQA Dataset released by Georgia Tech: https://visualqa.org/download.html. Specifically, you will be using the **Balanced Real Images** part of the dataset. Given that the dataset size is very large, you will be using only **1/4th of the training images** for this project. You can decide the way to sample the dataset. Loading this dataset is not trivial, hence we recommend starting as soon as possible.

Tasks

1. Create a model that takes as an input an image and a corresponding question and gives an answer to the question. You are **not allowed** to use multimodal models like CLIP, BLIP, etc for this. However, using the BERT family of models and the VIT family of models **is allowed**.

2. First, fine-tune the model you created on the dataset above. Note down the time taken to train this model along with the evaluation metrics. This is the baseline. Next, use LoRA to fine-tune the same model and again note down the time taken and the metrics. You are free to choose the different layers and ranks for the LoRA fine-tuning. Bonus points will be given if you can come up with more optimisations.
3. The metrics to be reported are - Accuracy, F1 Score, Precision, Recall and Time Taken for Training (TTT).

Hints and Recommendations

- Carefully study the data before building the model. Understand the format of the answers and then build the model.
- Use Kaggle for training. Wisely use the GPU credits amongst your teammates. Use functionalities like scheduled notebooks, etc for stability.
- Use huggingface for both getting the models and for LoRA fine-tuning. Use PyTorch for better compatibility with other libraries and faster computation.

Deliverables

- All the code written along with the environment.yaml file that is required to run the code. We expect **at least** three different files, more files can be added as necessary:
  - Notebook/Python Script to load and preprocess the data
  - Notebook/Python Script for training

    - Notebook with example inferences and the evaluation - this should be able to load the model weights given as input.
- Model Weights - the parameters of the model that you build should be saved and uploaded to GDrive/OneDrive and submitted.
- Project Report (preferably in Latex, using IEEE two-column format) - The report should contain your approach in detail along with the architecture diagram of the model you created. The metrics listed above should be reported for every model that you report.