REPORT (DL TERM PROJECT)

Team Details

- **Team Name**: F_Grade (Team Id: 10)
- GOURAV SARKAR (23CS60R01)
- RAJDEEP GHOSH (23CS60R10)
- ROHIT DUTTA (23CS60R19)

Running Instructions (For both task A and B)

- There will a folder named 'custom_captions_dataset` and it will consist of the exact files and folders as provided in the question.
- Now for each taks, there are a single individual file named `F_Grade_10_a.ipynb` and `F_Grade_10_b.ipynb` respectively.
- Run the respective file for the task you want to execute (Collab or Jupyter Notebook).

Task A (Image Captioning using CNN (VGG16) and LSTM)

Methodology

In this project, we aim to build an image captioning model using a combination of Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks. The model will take an image as input and generate a descriptive caption for the image.

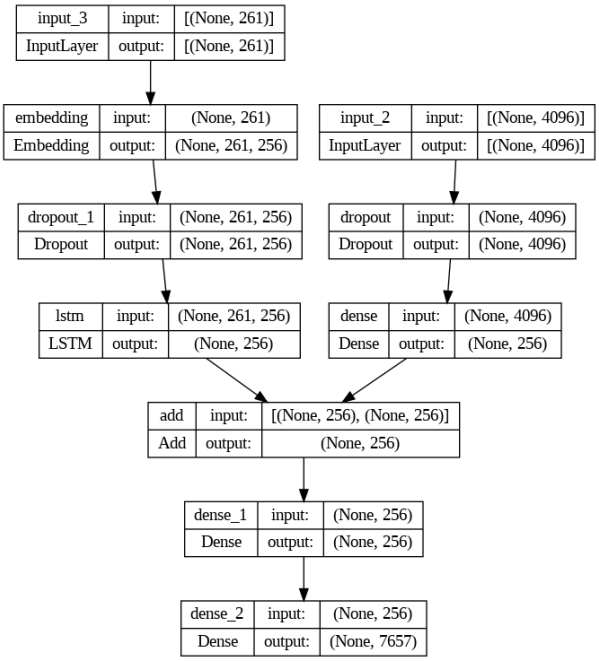
Data Preprocessing

- Preprocess captions by removing unnecessary characters, converting to lowercase, and adding start and end tokens.
- Load images and captions, preprocess them, and store them in suitable data structures(dictionary).

Model Architecture

Encoder:

The encoder processes the input image and extracts its features. It utilizes the **VGG16 pre-trained convolutional neural network (CNN)** model to extract features from images. The last fully connected layer of VGG16 is removed, leaving the feature extraction layers intact. The output shape of the encoder is (None, 4096), representing the extracted image features.



Decoder:

The decoder generates captions based on the features extracted by the encoder.It consists of two parts:

Sequence Feature Layers:

- Receives input in the form of sequences representing captions and embedded into dense vectors using an embedding layer.
- The embedded sequences are fed into an LSTM (Long Short-Term Memory) layer and it processes the sequential information and generates a fixed-size output.

Image Feature Layers:

- The features extracted by the encoder are combined with the output of the LSTM layer.
- These combined features are then passed through a fully connected (Dense) layer.
- The output of the decoder is a probability distribution over the vocabulary of words, representing the likelihood of each word being the next word in the caption and the final dense layer produces a softmax output

Model Compilation:

- The model is compiled with categorical cross-entropy loss, which is suitable for multi-class classification problems like language generation.
- The Adam optimizer is used for training, which is an adaptive learning rate optimization algorithm.

Training and Evaluation:

- **Data Generator** - Implement a data generator function to generate data in batches during training, ensuring efficient memory usage.

- **Training** - Train the model using the training data, iterating over 8 epochs and generating batches of data using the data generator.

- **Evaluation** - Define evaluation metrics such as ROUGE-L to assess the performance of the model and evaluate the model on the validation set after each epoch and save the model with the best validation score as `best_model_t1.pickle`.
- **Testing** - Generate captions for all the images in the test set using the trained modela and save the predicted captions along with the corresponding image IDs in a CSV file named `t1_predictions_v3.csv`
- **Evaluation Metrics** - Calculate ROUGE-L, CIDEr, and SPICE scores for the predicted captions compared to the ground truth captions in the test set.
- **Results Display** -Display the original and predicted captions for a all images from the test set and print the evaluation scores obtained for the model.

Results

```
- ROGUE-L: 0.35167331036641325
- CIDEr : 0.05834273241798831
- SPICE : 0.137427499450287
```

Analysis

The Image Captioning model utilizing CNN (VGG16) and LSTM achieves moderate performance, indicating potential for improvement in semantic understanding, diversity, and contextual coherence of generated captions

Task B (Image Captioning Using ViT Encoder and GPT-2 Decoder)

Methodology

Dataset Preprocessing:

- Captions are preprocessed exactly as in Task A.
- Images are resized to a fixed size (e.g., 224x224 pixels) to match the input size expected by the ViT encoder.

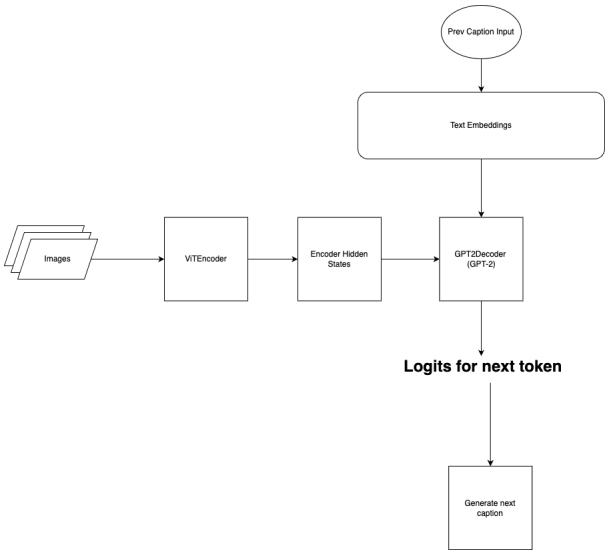
Model Architecture:

Encoder (ViT):

- Utilizes the Vision Transformer (ViT) model and we have used `"google/vit-base-patch16-224"` pretrained ViT model.
- Extracts global image features, represented by the last hidden state of the [CLS] token.

Decoder (GPT-2):

- Employs the GPT-2 language model for generating captions conditioned on image features.
- Fine-tuned to incorporate image features provided by the encoder and generates captions autoregressively, one token at a time, until an end token is predicted.



Training and Evaluation:

- **Training** - AdamW optimizer with a learning rate of 5e-5 is used for model optimization.Cross-entropy loss function minimizes the difference between predicted and ground truth captions.
- **Epochs** - Training is performed for 5 epochs, with model performance evaluated on the validation set after each epoch and the model with the best validation score (e.g., based on ROUGE-L metric) is saved as `best_model_t2.pickle`.
- **Evaluation** -Best model is loaded to generate captions for images in the test set.
- **Beam Search** - Caption generation employs beam search decoding strategies (beam size = 5) to improve caption quality.
- **Testing** - Generate captions for all the images in the test set using the trained model and Save the predicted captions along with the corresponding image IDs in a CSV file named `t2_predictions_v3.csv`
- **Evaluation Metrics** - Calculate ROUGE-L, CIDEr, and SPICE scores for the predicted captions compared to the ground truth captions in the test set.
- **Results** -Display the original and predicted captions for a all images from the test set and Print the evaluation scores obtained for the model.

Results

```
- ROGUE-L: 0.3060691691828152
- CIDEr : 0.03435318968165713
- SPICE : 0.08774878153608791
```