1 Problem Statement

Design a CNN-LSTM based Image captioning system.
Condition : Total model parameters cannot be greater than 25 million.
Objective : Maximize the BLEU score on test data.

2 Dataset and Loading

2.1 Dataset

2.1.1 Flickr8k_text

This Dataset contains the following files:

- Flickr8k.token.txt
  - The raw captions of the Flickr8k Dataset.
- Flickr8k.lemma.txt
  - The lemmatized version of the above captions.
- Flickr_8k.valImages.txt
  - The validation images to be used in experiments.
- Flickr_8k.trainImages.txt
  - The training images to be used in experiments.
- Flickr_8k.testImages.txt
  - The test images to be used in experiments.

2.1.2 Flicker8k_Images

There are 8000 images each with 5 captions. These images are divided into 3 sections. Train, test and validation contain 6000, 1000 and 1000 images respectively.

2.1.3 FlickrDataset

We created a custom dataset of our own by taking data from the image and text Flickr8k datasets.

Now, this dataset of ours consists of a vocabulary and a dictionary of embeddings, where the vocabulary is built up after tokenising the words taken from each sentence/caption.
However, our dictionary on the other hand, consists of embeddings of only those words present in the vocabulary which exceed a certain frequency threshold.

We built this custom dataset to make it simpler to associate images to the tensor embeddings of their captions created from the dictionary. Both the image data and the label data (here the tensors), being in the same datset makes it a much easier task to train on.

2.2 Loading

2.2.1 get_loader

Our function we created, which takes the Flicker8k_Images Dataset and the lemma tokens text file as input along with other parameters such as transformation function, batch_size, workers, shuffle and pin_memory, to create and return our custom dataset and the DataLoader created from it.

2.2.2 MyCollate

Custom collate function to modify the data batches created by the DataLaoder.

2.2.3 DataLoader

We are loading the necessary train data by calling the DataLoader function from torch.utils.data onto our custom dataset and collate function.

3 Model Architecture

3.1 Architecture

Total number of parameters = $15,945,662$ (15.95 Million)
Total number of trainable parametes = $4,769,150$ (4.76 Million)
Model memory requirement = 63.8 MB.
Encoder decoder based architecture has been used.

3.1.1 Encoder

Resnet18 model, pre-trained on imagenet data has been used to encode the image. The resnet18 model contains 11 million parameters. The fully connected layer of resnet18 has been instantiated with new Linear layer whose output is is set to 256(image embedding size). Relu activation and dropout technique has been used to create a robust encoder.

3.1.2 Decoder

PyTorch Embedding Module has been used to embed the captions to a finite space. The embeddings of the image and the captions are concatinated and passed to the LSTM module that iterates through the sequence.

The LSTM output is passed through dense layers which finally yields a probability distribution across the vocabulary size.

3.2 Training Methodology

Teacher forcing is a training methodology where the model uses ground truth as the input instead of the output from the previous time step. This method can result in fast training. But the model suffers from poor predictions as the RNN's conditioning context(the sequence of previously generated samples) diverge from sequences seen during training.
We have tried teacher forcing and the normal method. The default lstm module performs teacher forcing when the entire sequence is given initially.

4 Decoding techniques

The Linear layers of our decoder output a score of the likelihood of occurrence of each word in the vocabulary, at each position in the output sequence. The Softmax then converts those scores into probabilities. To obtain a final target sentence , our model should decide which word it should predict for each position in that target sequence. We tried two different approaches for this purpose.

4.1 Greedy Search

In greedy search, we take the word that has the highest probability at each position and predict that. However this might be an issue as we considered each position in isolation. Once we had identified the best word for that position, we did not examine what came before it or after it.

4.2 Beam Search

With Greedy Search, we select the single best word at each position. In contrast, Beam Search expands this and takes the best 'N' words. Beam Search picks the 'N' best sequences so far and considers the probabilities of the combination of all of the preceding words along with the word in the current position. This 'N' is known as

Beam Width. We selected the N = 3.

First we find the top 3 words with the highest probability given the input sentence and then the three best pairs for the first and second words based on conditional probability. We continue this process for 3rd word and so on and keep on selecting the best 3 pairs.

Finally we pick a sentences with the highest probability.

Increasing the beam width gives better results and we also tried increasing it. But as we increase Beam width, memory consumption also increases and for higher beam width, we faced ram issues on testing data.

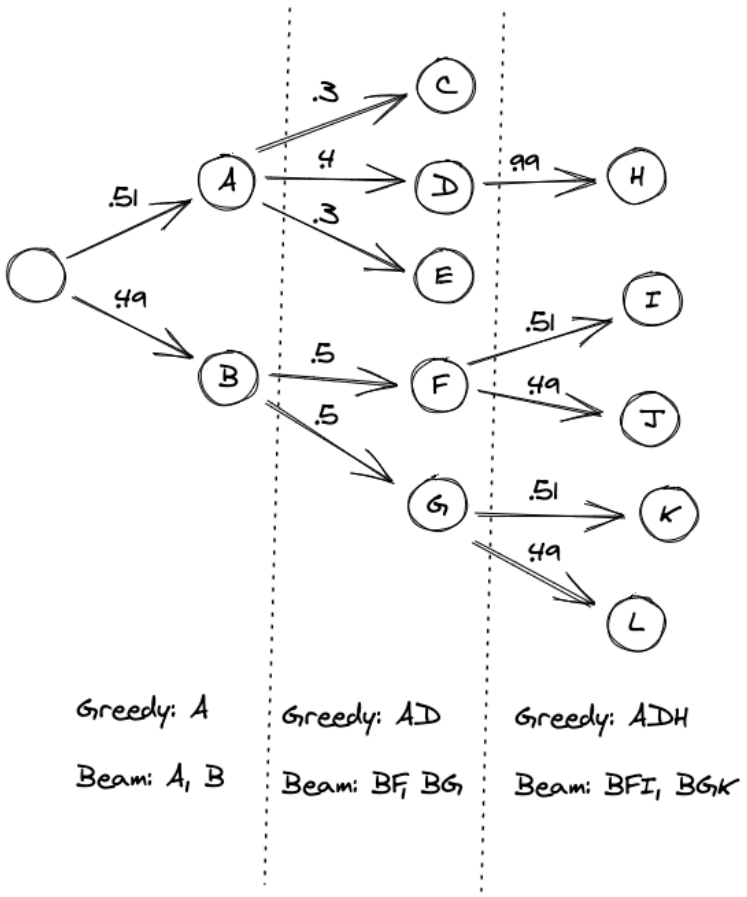

Figure 1: Greedy search vs beam search(N = 2) algorthim

5 BLEU Score

BLEU (BiLingual Evaluation Understudy) is a metric for evaluating text obtained from models. The BLEU score is a number between zero and one that measures the similarity of the machine-translated text to a set of high quality reference translations. A value of 0 means that the machine-translated output has no overlap with the reference translation (low quality) while a value of 1 means there is perfect overlap with the reference translations (high quality).
Train Data Mean BLEU Score = **0.2275**
Test Data Mean BLEU Score = **0.2277**

BLEU Score	Interpretation
< 10	Almost useless
10 - 19	Hard to get the gist
20 - 29	The gist is clear, but has significant grammatical errors
30 - 40	Understandable to good translations
40 - 50	High quality translations

6 Caption results of given sample images

Figure 2: Sample 1

Captions:

————Greedy Search Result————
a man in a red jacket is skateboarding down a railing .
————Beam Search Result————
a man in a red jacket is skateboarding down a railing .

Figure 3: Sample 2

Captions:

————Greedy Search Result————
a man in a red jacket is walking on a dirt path in front of a crowd .
————Beam Search Result————
a man in a red jacket is standing on a rock in front of a large rock .

Figure 4: Sample 3

Captions:

————Greedy Search Result————
a man in a red jacket is standing on a rock overlooking a town .
————Beam Search Result————
a man in a red jacket is standing on a rock in front of a large crowd .

Figure 5: Sample 4

Captions:

————Greedy Search Result————
a man in a red jacket is walking on a dirt path in the woods .
————Beam Search Result————
a man in a red jacket is standing on a rock overlooking a town .

Figure 6: Sample 5

Captions:

————Greedy Search Result————
a man in a red jacket is jumping over a fallen tree .
————Beam Search Result————
a man in a red jacket is skiing down a snowy hill .

References

Sequence-to-sequence RNN models
Teacher Forcing
Image captioning system
Image captioning tutorial
Beam search Implementation
beam search in seq2seq model