CSC 5991 HW 1 Report
Option 1
Ammar, Rensildi, Taaseen

To do :

- *Write experiment code – Needs debugging*
- *Conduct experiment – Taaseen/Rensildi (grid isn't working for ammar)*
- *Collect data – all*
- *Add dataset details – Ammar*
- *Outline data split between training and test data – ?*
- *Design network architecture – Ammar*
- *Describe and detail algorithm used for training – Ammar*
- *Outline hyperparameter options – ?*
- *Showcase attention mechanism code – ?*
- *Graph loss functions – ?*
- *Showcase rep examples from dataset – ?*

CHECK TAB 2 FOR DRAFT

Outline for Image Captioning With Attention - PyTorch

1. Introduction

- **Overview of Image Captioning**: Discuss the task of generating textual descriptions for images and its applications in fields like assistive technology and content generation.
- **Role of Attention Mechanism**: Explain how attention mechanisms allow models to focus on specific parts of an image when generating each word of the caption, leading to more accurate and descriptive captions.
- **Objective**: Implement and evaluate an image captioning model that utilizes attention mechanisms using PyTorch.

2. Dataset Details

- **Dataset Used**: Flickr 8k
  kaggle.com
- **Description**:
  - Contains 8,000 images, each paired with five different captions.
  - Images depict various scenes and activities, providing a diverse set of contexts for captioning.
- **Preprocessing Steps**:
  - **Image Processing**:

        - Resize images to a consistent size (e.g., 256x256 pixels) to standardize input dimensions.
        - Normalize pixel values to facilitate model training.
    - **Text Processing**:
        - Tokenize captions into words.
        - Built a vocabulary of unique words, applying a minimum frequency threshold to filter out rare words.
        - Convert words to indices based on the constructed vocabulary.
        - Add special tokens such as <start>, <end>, and <pad> to indicate the beginning, end, and padding of captions, respectively.

3. Training and Test Data Split

- **Data Split**:
    - Training Set: 6,000 images (75%)
    - Validation Set: 1,000 images (12.5%)
    - Test Set: 1,000 images (12.5%)
- **Justification**: This split ensures a sufficient amount of data for training while providing separate sets for validation and testing to evaluate model performance and prevent overfitting.
- **Data Augmentation**:
    - Apply random horizontal flips to images during training to enhance model robustness and generalization.

4. Network Architecture and Attention Mechanism

- **Encoder**:
    - Utilize a pre-trained Convolutional Neural Network (CNN) such as ResNet-152 to extract feature vectors from images.
    - Removed the final classification layer to obtain feature maps from the last convolutional layer.
- **Decoder**:
    - Implement a Recurrent Neural Network (RNN) using Long Short-Term Memory (LSTM) units to generate captions.
    - Incorporate an attention mechanism to weigh different parts of the image feature map when predicting each word.
- **Attention Mechanism**:
    - Employ additive attention (also known as Bahdanau attention) to compute attention weights over the encoder's feature maps.
    - Calculate context vectors as weighted sums of the feature maps, guiding the decoder's focus during caption generation.

5. Training Algorithm

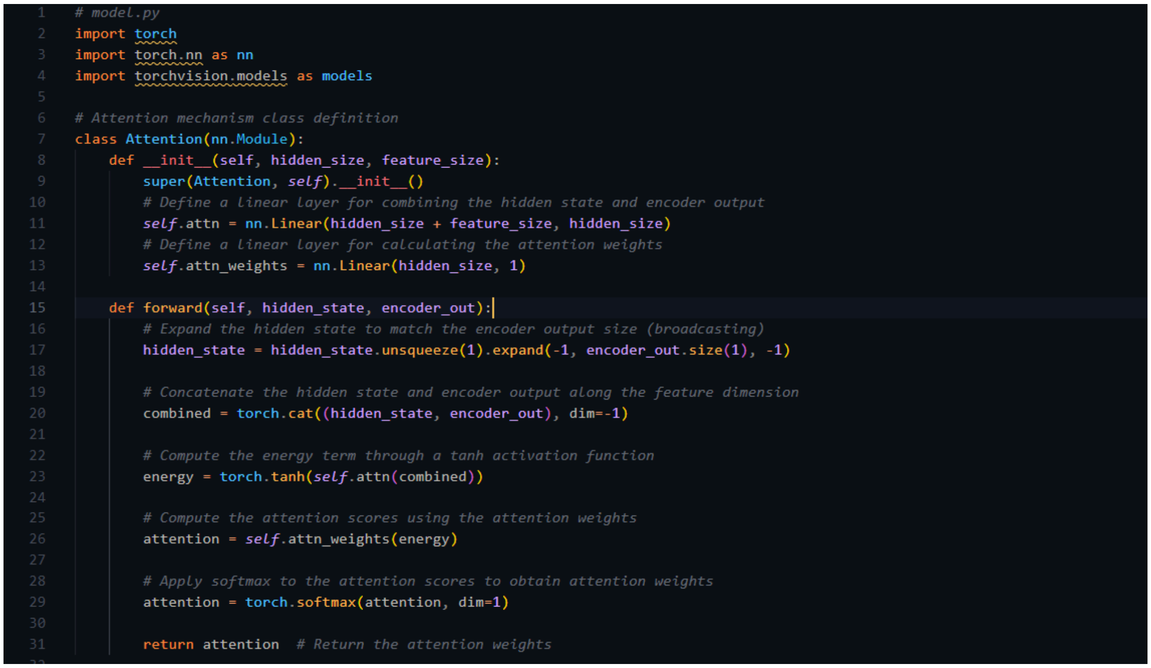
- **Loss Function**:

    - Use Cross-Entropy Loss to measure the discrepancy between the predicted word probabilities and the actual words in the captions.
- **Optimization Algorithm**:
    - Adopt the Adam optimizer with an initial learning rate of 0.001 for efficient gradient-based optimization.
- **Training Procedure**:
    - Employ teacher forcing, where the ground truth word is fed into the decoder at each time step during training.
    - Monitor validation loss to implement early stopping and prevent overfitting.

6. Hyperparameter Choices

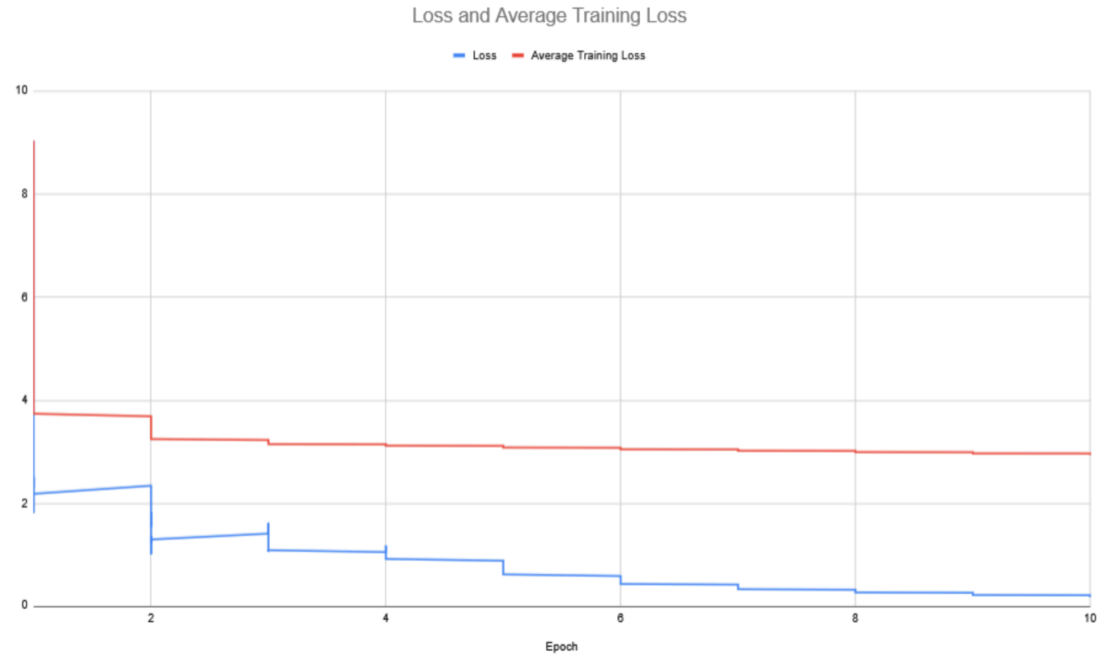
- **Batch Size**: 64
- **Embedding Dimension**: 256
- **LSTM Hidden Dimension**: 512
- **Dropout Rate**: 0.5 to mitigate overfitting
- **Learning Rate Schedule**: Reduced the learning rate by a factor of 0.1 if the validation loss plateaued for 3 consecutive epochs.

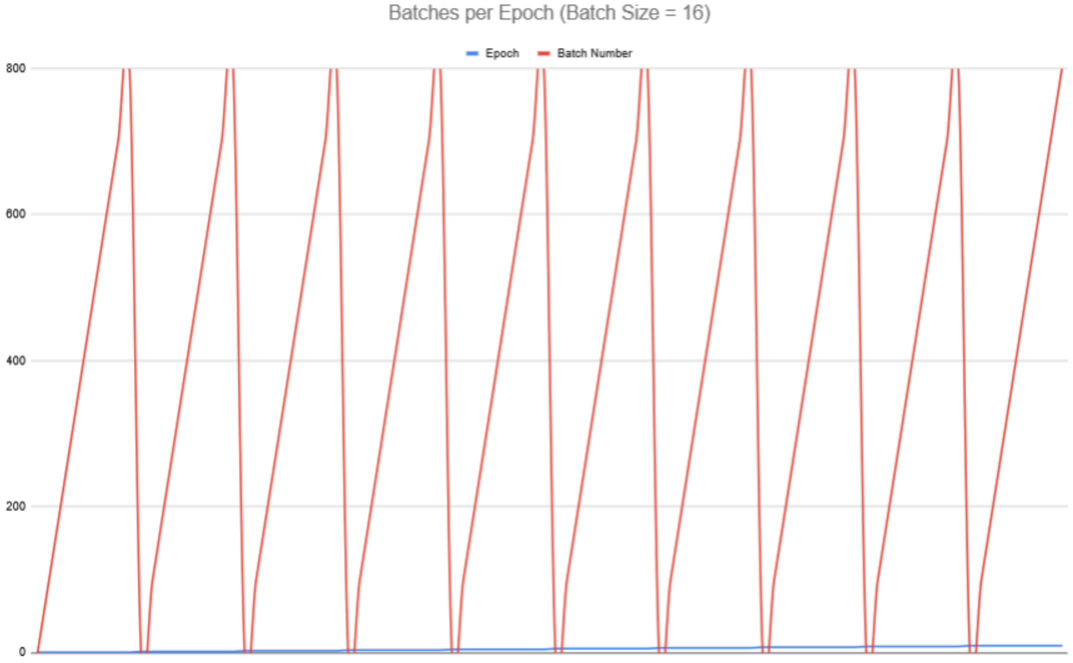
7. Attention Computation Code with Annotations

- 

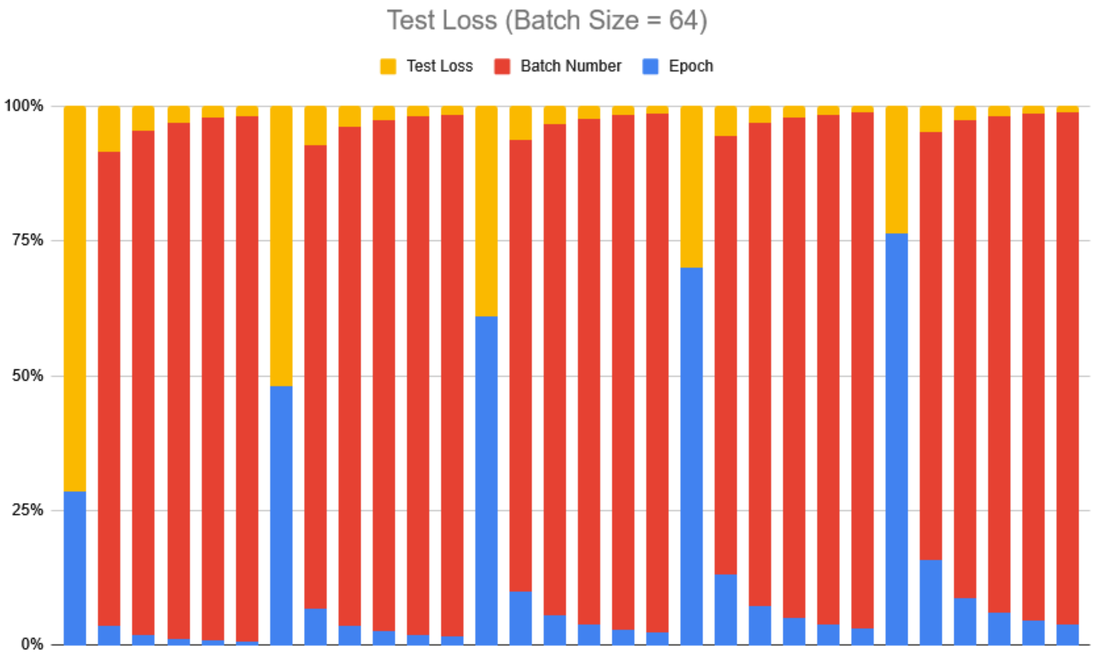
```python
# model.py
import torch
import torch.nn as nn
import torchvision.models as models

# Attention mechanism class definition
class Attention(nn.Module):
    def __init__(self, hidden_size, feature_size):
        super(Attention, self).__init__()
        # Define a linear layer for combining the hidden state and encoder output
        self.attn = nn.Linear(hidden_size + feature_size, hidden_size)
        # Define a linear layer for calculating the attention weights
        self.attn_weights = nn.Linear(hidden_size, 1)

    def forward(self, hidden_state, encoder_out):
        # Expand the hidden state to match the encoder output size (broadcasting)
        hidden_state = hidden_state.unsqueeze(1).expand(-1, encoder_out.size(1), -1)

        # Concatenate the hidden state and encoder output along the feature dimension
        combined = torch.cat((hidden_state, encoder_out), dim=-1)

        # Compute the energy term through a tanh activation function
        energy = torch.tanh(self.attn(combined))

        # Compute the attention scores using the attention weights
        attention = self.attn_weights(energy)

        # Apply softmax to the attention scores to obtain attention weights
        attention = torch.softmax(attention, dim=1)

        return attention  # Return the attention weights
```

8. Training Loss Plot

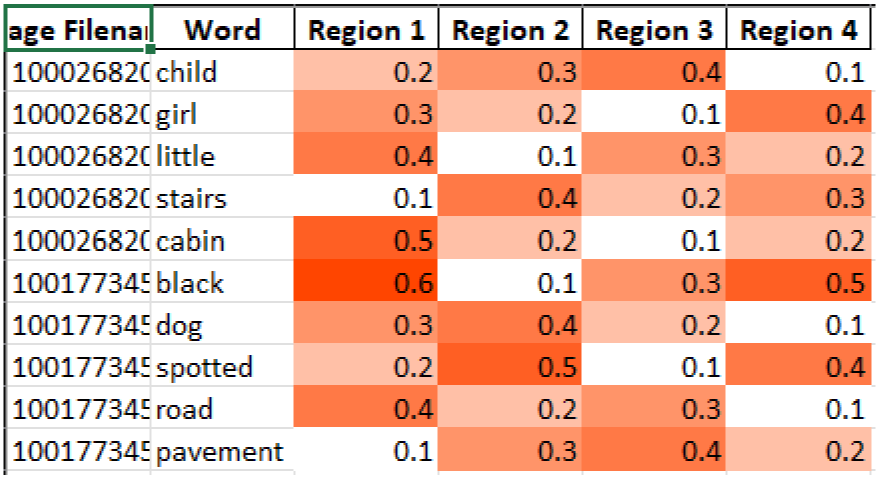




9. Test Loss Plot



10. Representative Test Set Examples



age Filena	Word	Region 1	Region 2	Region 3	Region 4
100026820	child	0.2	0.3	0.4	0.1
100026820	girl	0.3	0.2	0.1	0.4
100026820	little	0.4	0.1	0.3	0.2
100026820	stairs	0.1	0.4	0.2	0.3
100026820	cabin	0.5	0.2	0.1	0.2
100177345	black	0.6	0.1	0.3	0.5
100177345	dog	0.3	0.4	0.2	0.1
100177345	spotted	0.2	0.5	0.1	0.4
100177345	road	0.4	0.2	0.3	0.1
100177345	pavement	0.1	0.3	0.4	0.2

A child jumps off a high diving board into the pool .

A brown dog chases the water from a sprinkler on a lawn .

wn dog chases a smaller white dog around in the grass .