VISUAL RECOGNITION
Mini Project Report

Submitted By-
Manan Bansal (IMT2018039)
Tanishq Jaswani (IMT2018077)
Avik Bhatnagar (IMT2018505)
Unique Group code: MAT

Introduction

Image Captioning is the process of generating textual description of an image. It uses both Natural Language Processing and Computer Vision to generate the captions. Image captioning models typically follow an encoder-decoder architecture which uses abstract image feature vectors as input to the encoder. Improved convolutional neural networks, object detection architectures and more sophisticated sequential models, such as attention-based recurrent neural networks have contributed to improved and accurate image captioning systems.

Implementation Details

- First we read the image data and their corresponding captions from the **flick 8k dataset** folder provided to us (consisting of 8,000 images that are each paired with five different captions which provide clear descriptions of the salient entities and events).
- **Writing the dataloader:**
  - We implemented a custom dataset class so that we can abstract out the data loading steps during the training and validation process
  - Here we have created a dataloader which gives the batch of image and its captions with following processing done:
    - Caption words are tokenized to unique numbers.

- Vocabulary instance created to store all the relevant words in the datasets.For each word in the text corresponding index token for that word form the vocab built as list.
- Each batch and caption is padded to have the same sequence length.
- Images are resized to the desired size and are then converted into captions.

➢ After the data processing is done, we have implemented the model to train on the GPU and predict the captions of the given images.

➢ The **architecture of the model** as follows:

- **Encoder:**
  - The Convolutional Neural Network(CNN) can be thought of as an encoder. The input image is given to CNN to extract the features. The features of the last fully connected layer or convolutional layer are extracted as features of the image.
  - Here we used **ResNet50** as our model. It consists of 5 stages each with a convolution and Identity block. Each convolution block has 3 convolution layers and each identity block also has 3 convolution layers. The ResNet-50 has over 23 million trainable parameters.

- **Attention:**
  - The attention mechanism allows the neural network to have the ability to focus on its subset of inputs (or features)—to select specific inputs or features.
  - The main part of the attention mechanism is the following two aspects: the decision needs to pay attention to which part of the input; the allocation of limited information processing resources to the important part.

- For this project, we used the **Bahdanau Attention** method. So the method is as follows:
    - First it carries the hidden states that the Encoder produces to the next step of Alignment Scores calculation. *(1) in Figure 2*
    - Now the alignment score is calculated between the previous decoder hidden state and each of the encoder's hidden states are calculated. *(2) in the Figure 2*
    - Now the obtained scores are combined, represented in a single vector and then passed through the softmax function. *(3) in Figure 2*
    - The encoder hidden states and their respective alignment scores are then multiplied to form the context vector. *(4) in Figure 2.*
    - The context vector is then concatenated with the previous decoder output and fed into the Decoder RNN for that time step along with the previous decoder hidden state to produce a new output. *(5) in Figure 2*
    - The above steps are repeated for each time step of the decoder until an token is produced or output is past the specified maximum length

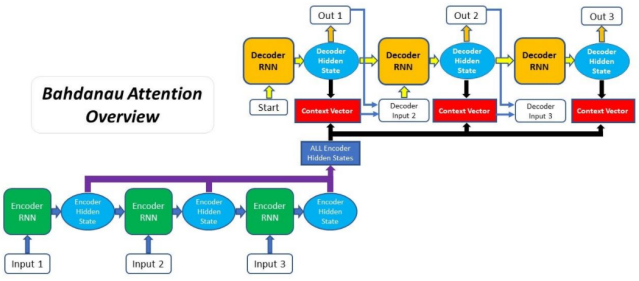


Figure 1

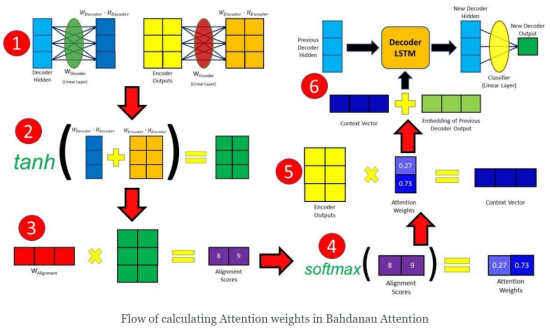


Figure 2

- **Decoder:**
    - The LSTM (Long-Short Term Memory) network is used as decoder.
    - We'll give the decoder RNN a special <start> token to indicate the start of the sentence and <end> token to indicate end of sequence.
    - In addition to taking two weight matrices i.e. the input at the current time-step and the hidden state at the previous time-step thus combining those to get next hidden states, we need to add image information i.e. third weight matrix. Then, we sample the vocabulary at every time-step.
    - In this Decoder Class, we have implemented the generate caption function which generates captions using the image features.

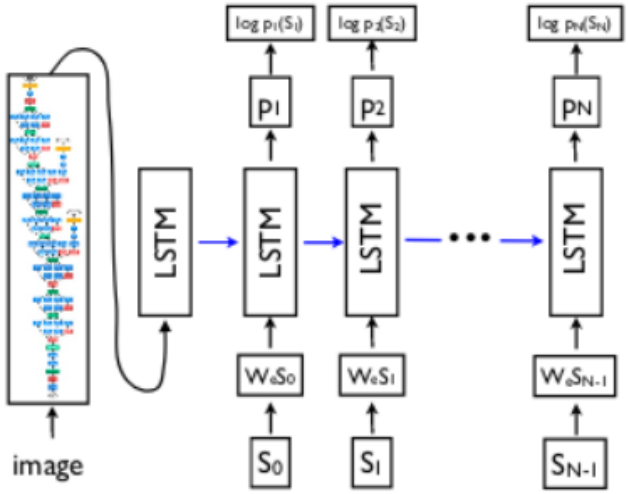


Fig3: Encoder decoder model for image captioning

- **Scoring:**
    - Our project required us to use the Bleu Scoring method. It is used to analyze the correlation of n-gram between the translation statement to be evaluated and the reference translation statement. Its core idea is that the closer the machine translation statement is to a human professional translation statement, the better the performance. In this task, the processing is the same as machine translation: multiple images are equivalent to multiple source language sentences in the translation. The Python Natural Language Toolkit library, or NLTK, provides an implementation of the BLEU score that we used to evaluate your generated text against a reference.

During the training process we experimented with various hyperparameters and pre-trained CNN weights. However, this resulted in us overshooting the CNN-LSTM model upper limits and we stuck to the following model hyperparameters:

- Embed size equal to 100
- Vocab size equal to 2994
- Attention dimension equal to 64
- Encoder dimension equal to 2048
- Decoder dimension equal to 128
- Learning rate equal to 3e-4

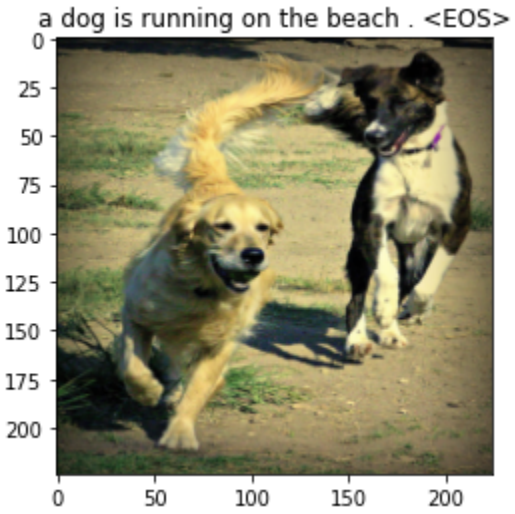
We trained our model for several epochs, we were able to achieve our best obtained results after training the model for 15 epochs.

Since it was taking a very long time to train the model, we opted for saving the model and then using that to make predictions on the images.

Results:

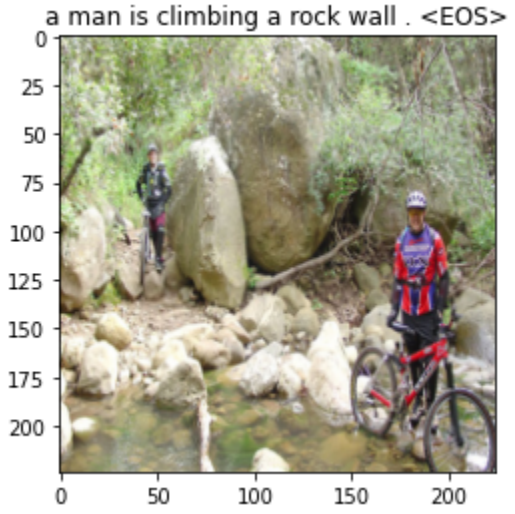
- We obtained the following results for the testing data :

1)




```
Ground truth Caption:  ['two', 'dogs', 'are', 'running', 'in', 'a', 'field', '.']
Predicted Caption:  ['a', 'dog', 'is', 'running', 'on', 'the', 'beach', '.', '<EOS>']
BLEU score:  0.6905911470987942
```

2)



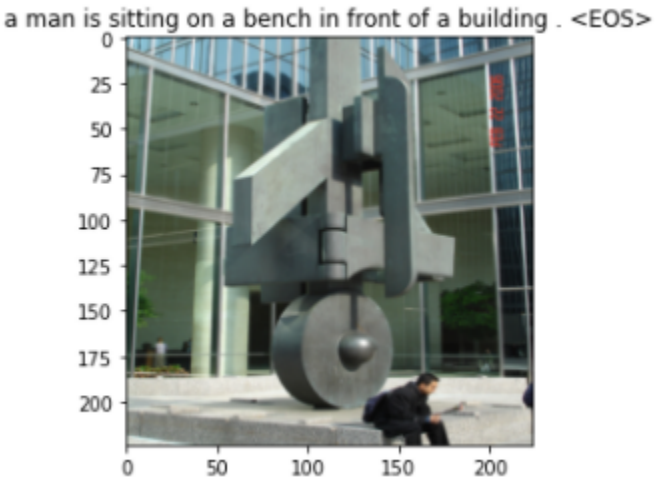
```
Ground truth Caption:  ['person', 'standing', 'beside', 'bike', 'in', 'stream', '.']
Predicted Caption:  ['a', 'man', 'is', 'climbing', 'a', 'rock', 'wall', '.', '<EOS>']
BLEU score:  0.46199933699457096
```

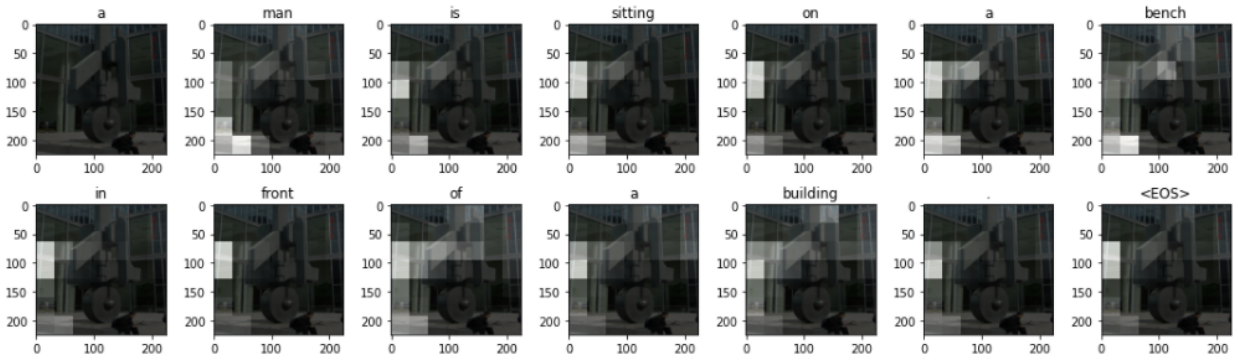
3)


```
Ground truth Caption:  ['a', 'man', 'on', 'a', 'bike', 'tries', 'to', 'do', 'a', 'trick', 'on', 'the', 'railing', 'of', 'an', 'outdoor', 'fountain', '.']
Predicted Caption:  ['a', 'man', 'is', 'doing', 'a', 'trick', 'on', 'a', 'bike', '.', '<EOS>']
BLEU score:  0.3575297164449809
```

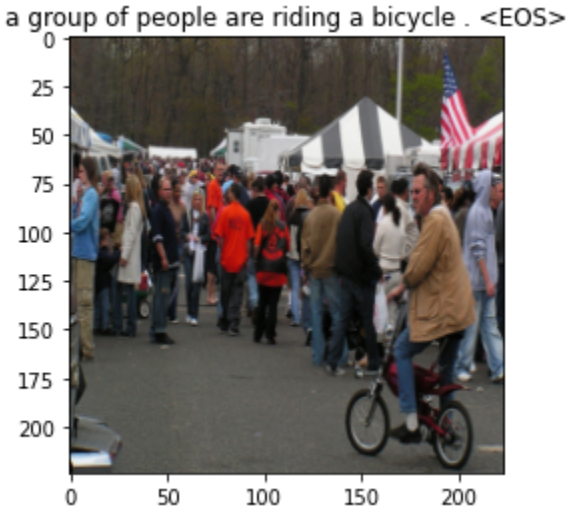
- The next set of photos are the ones that were given to us for predicting the caption (First is the image with it's predicted caption and the second one is the attention plot corresponding to that image).

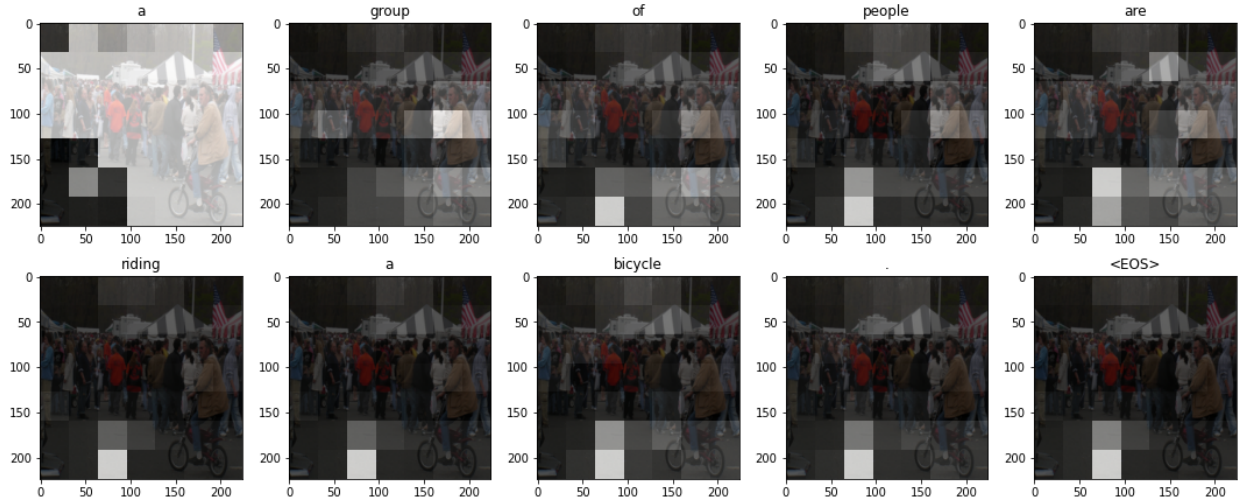
1)




a
man
is
sitting
on
a
bench
in
front
of
a
building
.
<EOS>

2)


a group of people are riding a bicycle . <EOS>

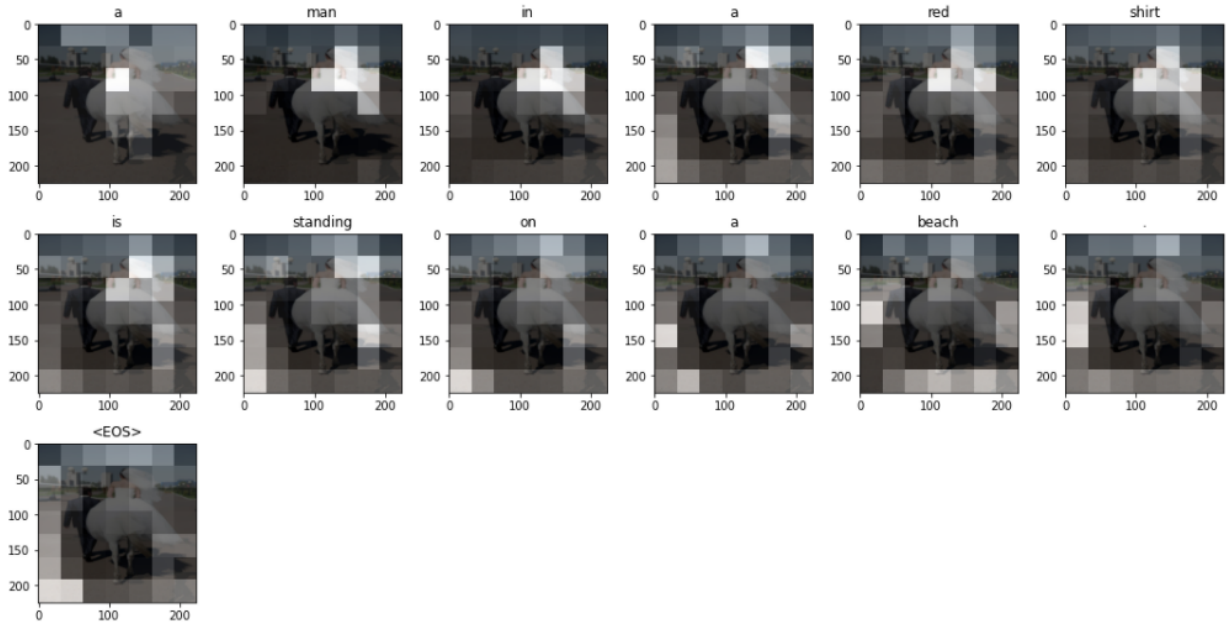

a
group
of
people
are
riding
a
bicycle
.
<EOS>

3)

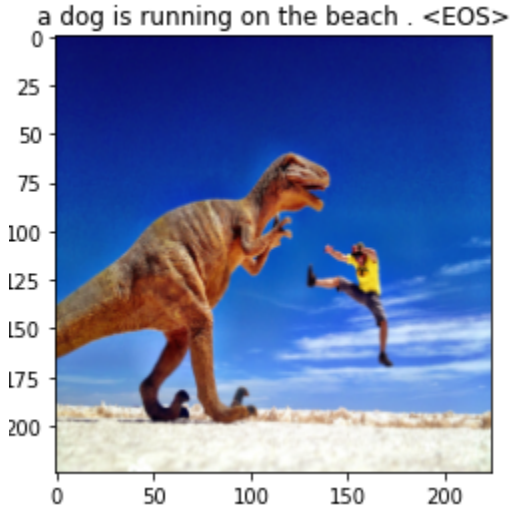

a group of people are standing on a city street . <EOS>
a
group
of
people
are
standing
on
a
city
street
.
<EOS>

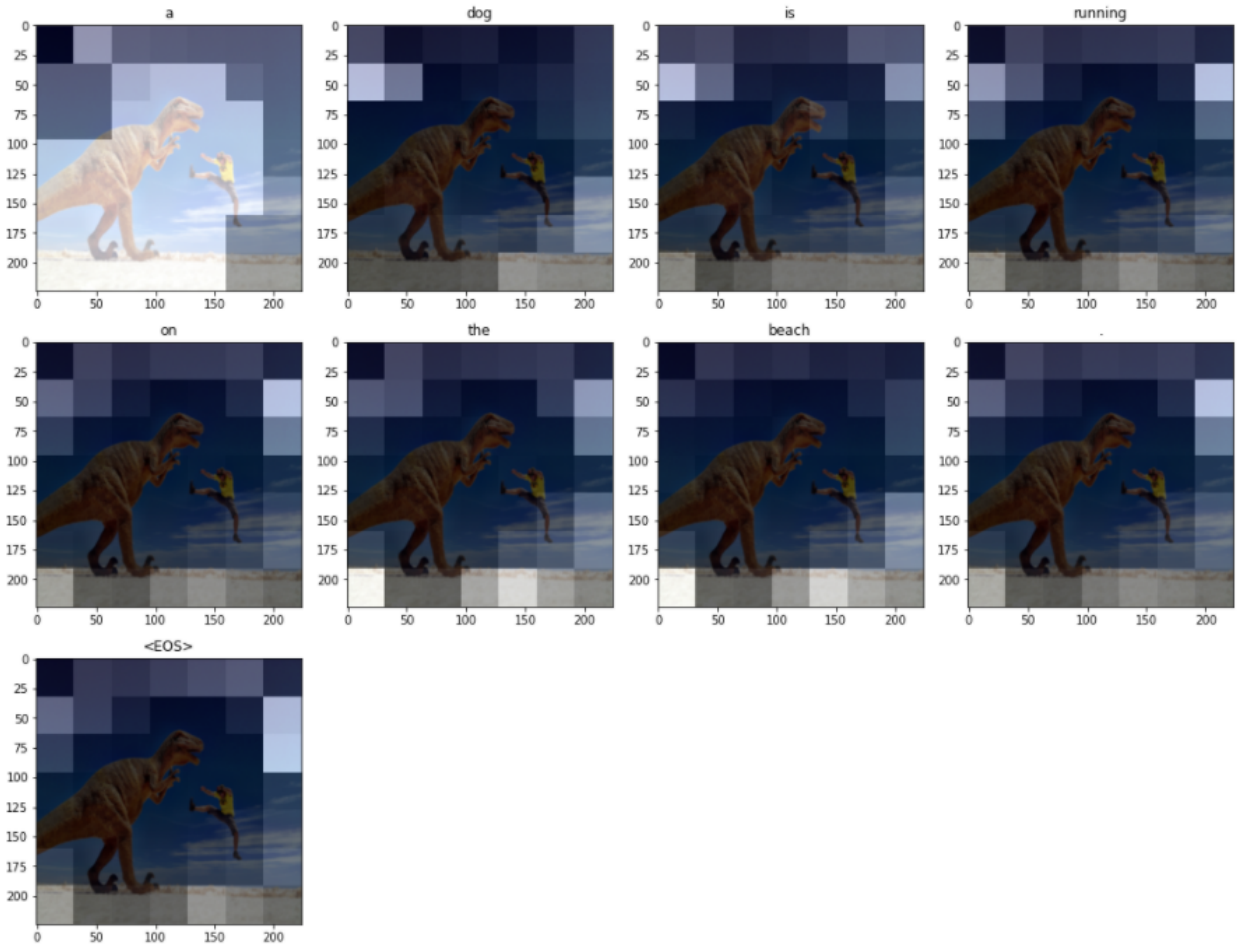
4)


a man in a red shirt is standing on a beach . <EOS>


a
man
in
a
red
shirt
is
standing
on
a
beach
.
<EOS>

5)


a dog is running on the beach . <EOS>



REFERENCES:

- https://blog.floydhub.com/attention-mechanism/
- An Overview of Image Caption Generation Methods (hindawi.com)
- python - NLTK: corpus-level bleu vs sentence-level BLEU score - Stack Overflow
- Image Captioning in Deep Learning | by Pranoy Radhakrishnan | Towards Data Science
- 10.4. Bahdanau Attention — Dive into Deep Learning 0.16.4 documentation (d2l.ai)
- Understanding and Coding a ResNet in Keras | by Priya Dwivedi | Towards Data Science