MLE Nanodegree

# Capstone Report

# Image Captioning

## Project Definition

### Project Overview

For the purpose of this capstone project, I chose to work on image captioning, which is a supervised machine learning task to automatically generate image captions for an input image. I trained an attention based CNN-RNN model which also uses attention to generate accurate predictions.

I trained my machine learning model on Flickr 8k dataset which consists of 8000 images, along with 5 captions for every image. This is a standard dataset used for image description based tasks and can be found on kaggle[1].

### Problem Statement

Generating sentence based image descriptions or Captions for images as input.

To solve this problem, I trained a CNN-RNN based encoder-decoder model on flickr 8k dataset, the architecture of the model used is similar to[2] which is based on **Show, Attend and Tell: Neural Image Caption Generation with Visual Attention.** The features for every image are generated using a pre-trained CNN model (Inception V3) on image net dataset.

### Metrics

BLEU score is one of the standard metrics to evaluate language generation models, and I will be using that to evaluate the performance of my model.

## Analysis

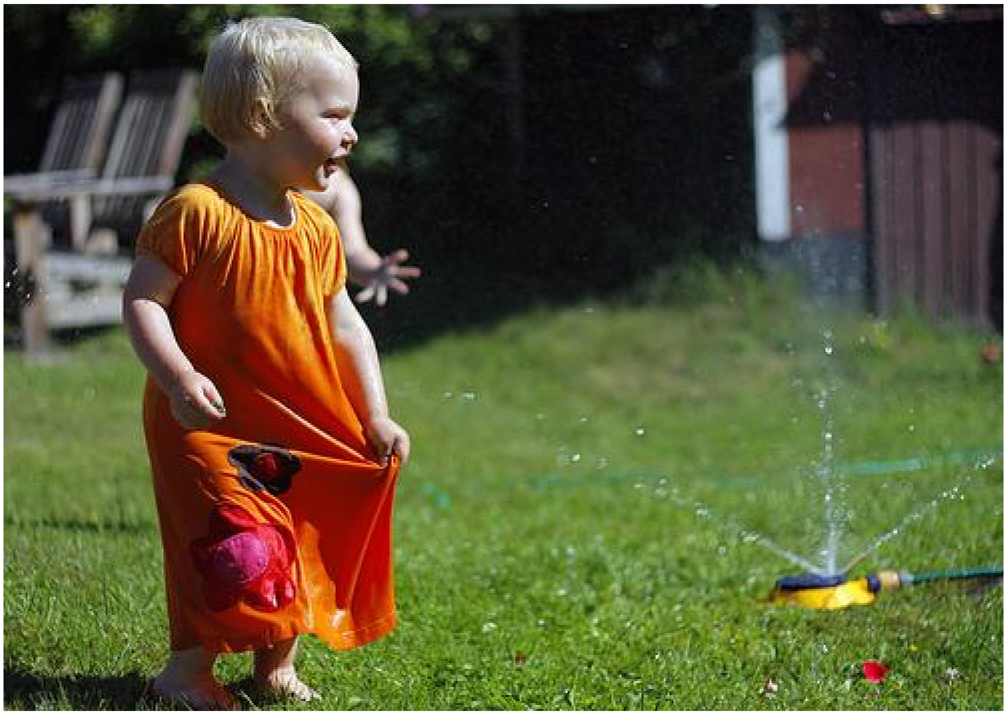
## Data Exploration

The input data as described above is the standard Flickr 8k dataset, the data consists of two files, first is a folder with 8000 different images in jpg format and second is a text file containing image to caption mapping, where for every image 5 captions are present.

The Images are converted into the same size of 299 x 299 pixels for them to be preprocessed by inception v3 to generate features, the captions are preprocessed by using stop words removal and tokenising them.

## Exploratory Visualization

**Sample Image for input into the model:**



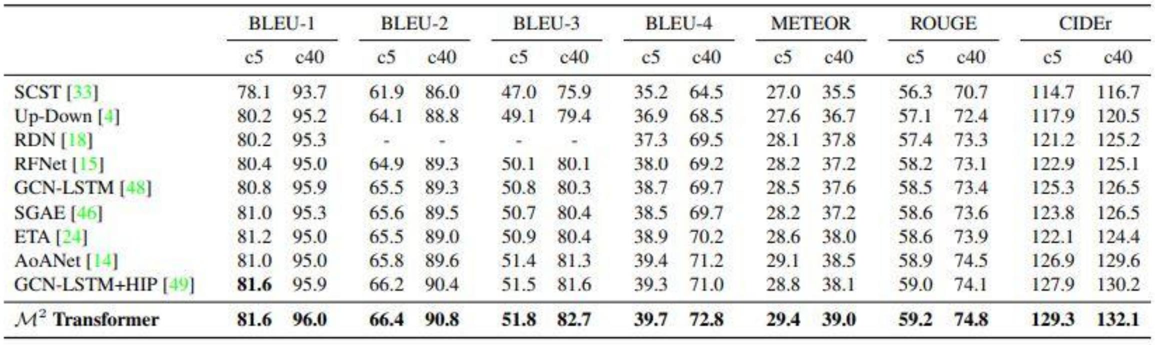
**Sample Output:**

```
Generated Caption: running through the green grass with a blue toy in the
grass
```

The Image is passed through a CNN to get a feature map, the CNN architecture used in this case is the inception net. All but last layer of inception net is used to generate the feature map. As last layer of a CNN is for classification.

## Algorithms and Techniques

### Benchmark



| | BLEU-1 | | BLEU-2 | | BLEU-3 | | BLEU-4 | | METEOR | | ROUGE | | CIDEr | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | c5 | c40 | c5 | c40 | c5 | c40 | c5 | c40 | c5 | c40 | c5 | c40 | c5 | c40 |
| SCST [33] | 78.1 | 93.7 | 61.9 | 86.0 | 47.0 | 75.9 | 35.2 | 64.5 | 27.0 | 35.5 | 56.3 | 70.7 | 114.7 | 116.7 |
| Up-Down [4] | 80.2 | 95.2 | 64.1 | 88.8 | 49.1 | 79.4 | 36.9 | 68.5 | 27.6 | 36.7 | 57.1 | 72.4 | 117.9 | 120.5 |
| RDN [18] | 80.2 | 95.3 | - | - | - | - | 37.3 | 69.5 | 28.1 | 37.8 | 57.4 | 73.3 | 121.2 | 125.2 |
| RFNet [15] | 80.4 | 95.0 | 64.9 | 89.3 | 50.1 | 80.1 | 38.0 | 69.2 | 28.2 | 37.2 | 58.2 | 73.1 | 122.9 | 125.1 |
| GCN-LSTM [48] | 80.8 | 95.9 | 65.5 | 89.3 | 50.8 | 80.3 | 38.7 | 69.7 | 28.5 | 37.6 | 58.5 | 73.4 | 125.3 | 126.5 |
| SGAE [46] | 81.0 | 95.3 | 65.6 | 89.5 | 50.7 | 80.4 | 38.5 | 69.7 | 28.2 | 37.2 | 58.6 | 73.6 | 123.8 | 126.5 |
| ETA [24] | 81.2 | 95.0 | 65.5 | 89.0 | 50.9 | 80.4 | 38.9 | 70.2 | 28.6 | 38.0 | 58.6 | 73.9 | 122.1 | 124.4 |
| AoANet [14] | 81.0 | 95.0 | 65.8 | 89.6 | 51.4 | 81.3 | 39.4 | 71.2 | 29.1 | 38.5 | 58.9 | 74.5 | 126.9 | 129.6 |
| GCN-LSTM+HIP [49] | **81.6** | 95.9 | 66.2 | 90.4 | 51.5 | 81.6 | 39.3 | 71.0 | 28.8 | 38.1 | 59.0 | 74.1 | 127.9 | 130.2 |
| **$\mathcal{M}^2$ Transformer** | **81.6** | **96.0** | **66.4** | **90.8** | **51.8** | **82.7** | **39.7** | **72.8** | **29.4** | **39.0** | **59.2** | **74.8** | **129.3** | **132.1** |

Above is the bleu score of some of the state of the art models for image captioning, though the dataset used is different and training is done on very sophisticated systems for SOTA models, but they provide a benchmark for our bleu scores.

# Methodology

## Data Preprocessing

Preprocessing step for our model can be divided into two parts, preprocessing of the image and the preprocessing of the captions

### Image Preprocessing

The images are preprocessed into the format required by inception v3, that is 299 by 299 pixels, with all the pixel values in the range of -1 to 1. The output of last but one layer of the inception v3 is considered for further tasks, because the last layer is the classification layer.

The model outputs feature maps of size 8x8x2048, this output is saved in the form of numpy files, as this reduces the training time. Since the extraction of features can be done separately so it is done before the training and the features are cached, this removes a major bottleneck.

**Captions Preprocessing**

The preprocessing of captions involves four steps:

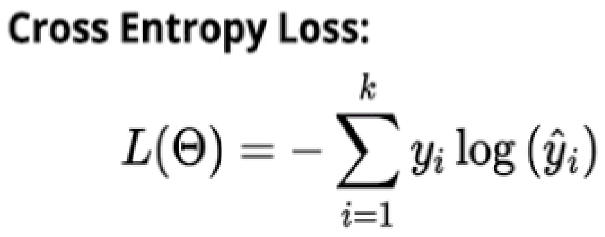
- Tokenize the captions and remove stop words, add rest of the words to vocabulary
- Limit the vocabulary size to top 5000 words and mark all the other words as oov
- Each caption is annotated with a <start> and <end> token to mark the sentence
- All the sequences are padded to the same length for batch processing

## Implementation

The numpy array of feature maps was already extracted and saved to remove any bottleneck from the training. The CNN part of the model just reads those features and passes them through a fully connected layer to get the output.

The encoder output, hidden state and the decoder output is fed to the decoder, the decoder returns the predictions and the decoder hidden state. The decoder hidden state is fed back into the decoder along with decoder input for the next word(teacher forcing) to produce the next output. The output is used for loss calculation. The loss function used is sparse categorical cross entropy.

Sparse categorical cross entropy loss is used in neural networks where outputs are integers, in our case the token numbers are used for calculating the loss.



**Cross Entropy Loss:**

$$L(\Theta) = -\sum_{i=1}^{k} y_i \log(\hat{y}_i)$$

## Refinement

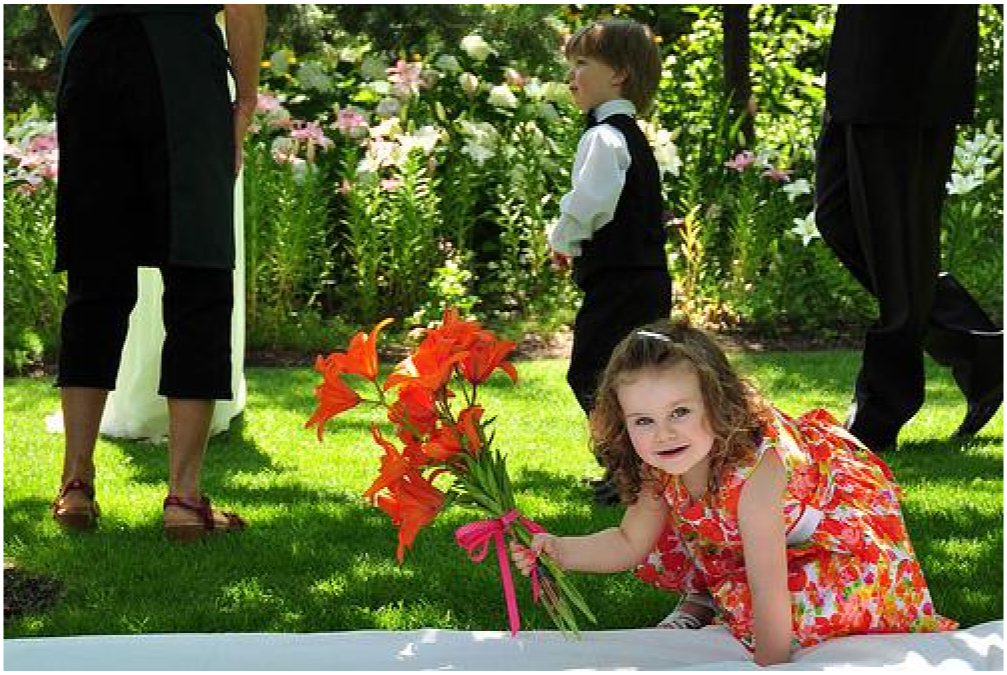
The training for this model was first tried on FLICKR 30k dataset but due to the limitation of AWS subscription, I was not able to complete it, after that I tried to train only on FLICKR 8k dataset.

The BLEU-1 score for the model was earlier coming out to be around 2-4 after that, the prediction which is selected was changed, only caption predictions with length more than 3 were taken and their value was compared to all the 5 reference sentences for that image. The best of the five scores was considered for the final result. This improved the BLEU-1 score to 18.25 but it is still very less than SOTA models, more training on more data will help refine the results further.

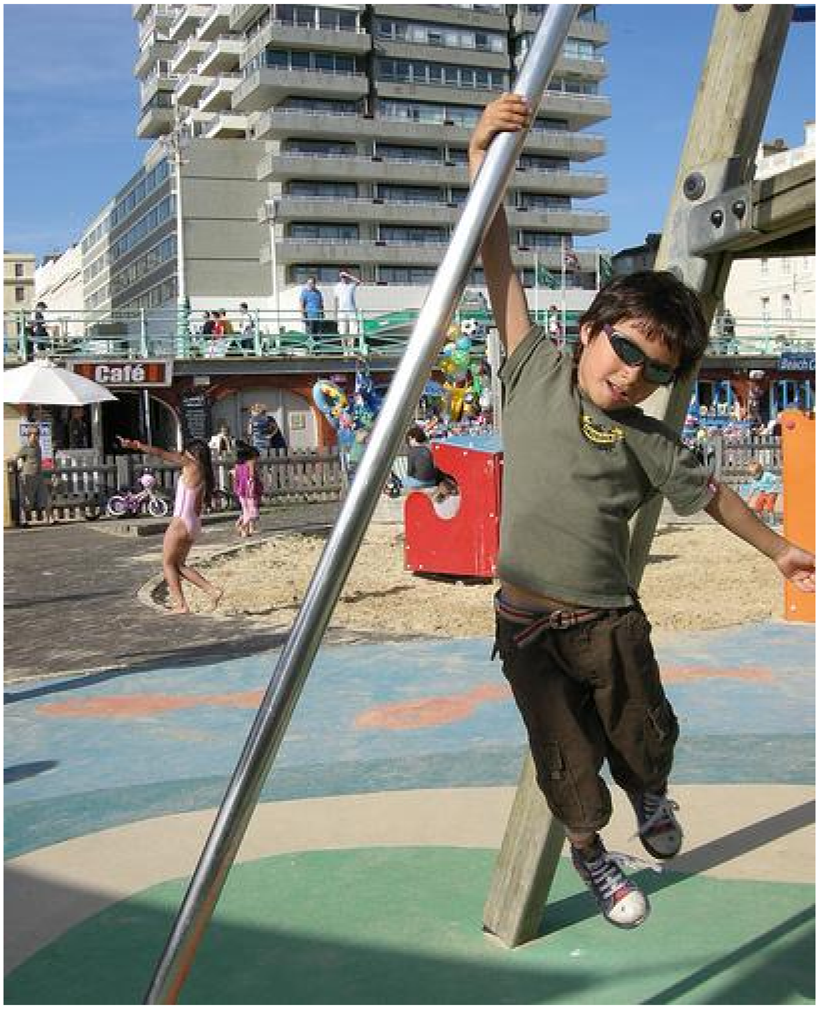
## Results

As the amount of training that I could do reduced significantly so I used only BLEU-1 score for comparison with the state of the art models. The BLEU-1 score that I got was 18.25. The best score against all the 5 references for each image was taken for calculation of the BLEU Score.

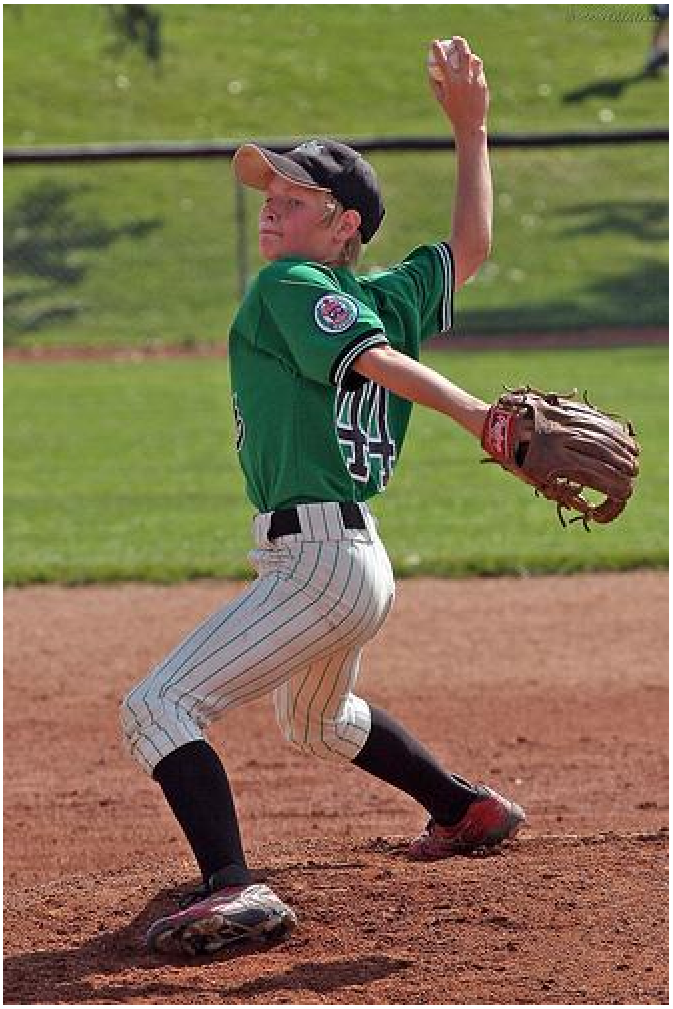
**Example Inputs and Predicted Captions by model:**



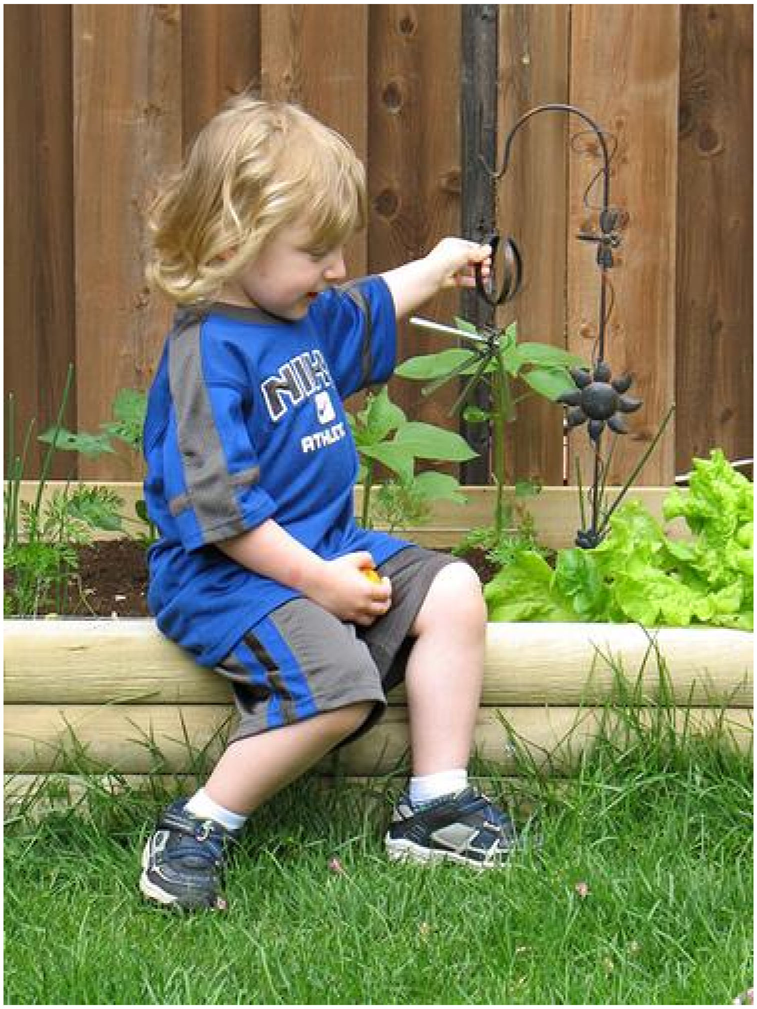
Generated Caption: orange bathing suit and black striped top is looking at the flower bush



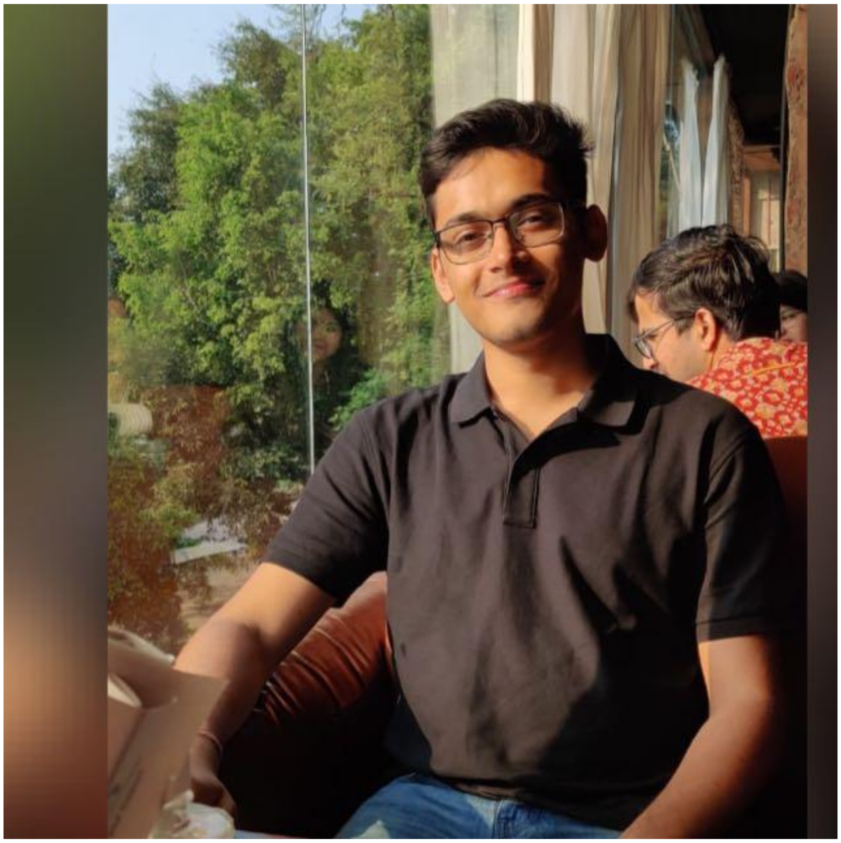
**Generated Caption:** a little boy as he falls over the playground



**Generated Caption:** into field with his mitt out



**Generated Caption:** smiles next to plants



**Generated Caption:** posing for the picture of photo

Even though the BLEU score of the model is not at par with SOTA models, it generates pretty accurate captions, and as is clear from above examples it is able to solve the problem adequately.

With a bit more data and tweaks it will start to perform a lot better.

## References

[1] https://www.kaggle.com/adityajn105/flickr8k

[2] https://www.tensorflow.org/tutorials/text/image_captioning