A Minor Project(CS6490) Report

On

IMAGE CAPTION GENERATION using Deep Learning

Submitted By

Group No. - 37

Md Ishtiyaque Ahmad – 1906146
Kundan Kumar – 1906148
Riyasha Jaiswal – 1906136

Under the Project Guidance of

PROJECT MENTOR

Dr. Ravi Bhushan Mishra
Professor, CSE Dept., NIT PATNA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY PATNA
(An Institute of National Importance)
MAHENDRU, PATNA, BIHAR – 800005

ACKNOWLEDGEMENT

Inspiration and motivation have always played a key role in the success of any venture.
We feel to acknowledge our indebtedness and a deep sense of gratitude to our external guide **Dr. Ravi Bhushan Mishra** Professor, CSE Dept., NIT PATNA. whose valuable guidance and kind supervision gave us throughout the course which shaped the present work as it shows.

We pay our deep sense of gratitude to **Prof. (Dr.) JP Singh**, H.O.D, Computer Science & Engineering Department, National Institute Of Technology Patna to encourage us to the highest peak and to provide us with the opportunity to prepare for the project.

We would like to thank our Computer Science & Engineering Department, National Institute of Technology Patna and to all the faculty members for giving us continuous support and guidance that has helped us in completion of our project.

DOCUMENT CONTROL SHEET

1.	Report Number	Minor Project (CS6490)/37
2.	Title Of Project	IMAGE CAPTION GENERATION using Deep Learning
3.	Type Of Project	Technical
4.	Author(s)	Md Ishtiyaque Ahmad Kundan Kumar Riyasha Jaiswal
5.	Organizing Unit	NIT PATNA
6.	Language Of Document	English

1. DESCRIPTION

1.1 General Overview of the Problem

i. Caption generation is an interesting artificial intelligence problem where a descriptive sentence is generated for a given image.
ii. It involves the dual techniques from computer vision to understand the content of the image and a language model from the field of natural language processing to turn the understanding of the image into words in the right order.
iii. Image captioning has various applications such as recommendations in editing applications, usage in virtual assistants, for image indexing, for visually impaired persons, for social media, and several other natural language processing applications.

1.2 Data Collection

i. For this project we had used Flicker_8k dataset.
ii. The dataset contains over 8091 images, each of which has at least 5 different caption annotations.
iii. For training purpose, we use images as the original size of the dataset is approximately 1 GB.
    a. Approximately each image has 5 captions associated with it, so that will lead to 40,000 captions in the dataset.

1.3 Preprocessing

i. We will be using the InceptionV3 model trained on the image net dataset as the encoder of our model so we will cache the output of the InceptionV3 for the images in our dataset to reduce training time. Also Embedded Glove 6B 200d dataset into our model.
ii. In order to provide a start and end point of the caption we add ‘startseq’ at the beginning of each caption, and ‘endseq’ at the end of each caption.
iii. There are a total of 7578 unique words in our dataset.
iv. Finally we will pad each caption to the length of the largest caption available in the dataset, this way the length of each caption will be same.

2. MODEL

2.1 InceptionV3 Model

Inception v3 is an image recognition model that has been shown to attain greater than 78.1% accuracy on the ImageNet dataset. The model is the culmination of many ideas developed by multiple researchers over the years. It is based on the original paper: "Rethinking the Inception Architecture for Computer Vision" by Szegedy, et. al.

The model itself is made up of symmetric and asymmetric building blocks, including convolutions, average pooling, max pooling, concatenations, dropouts, and fully connected layers. Batch normalization is used extensively throughout the model and applied to activation inputs. Loss is computed using Softmax.

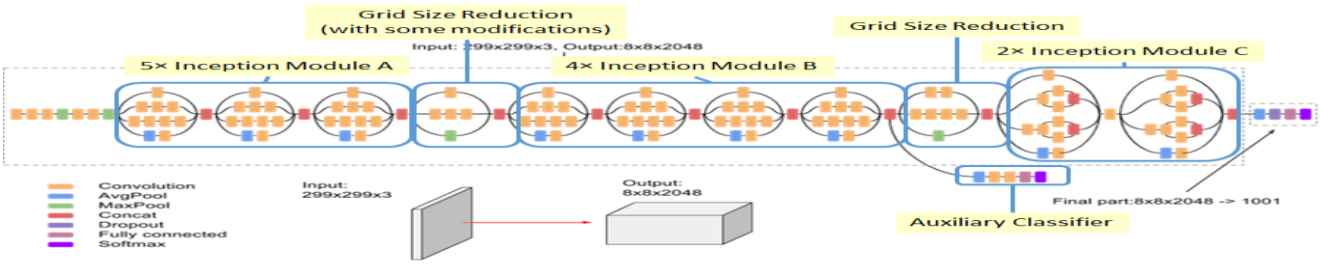


Fig : 0.1

2.2 Convolutional Neural Network

Convolutional Neural Network (CNN) is a Deep Learning algorithm which takes in an input image and assigns importance (learnable weights and biases) to various aspects/objects in the image, which helps it differentiate one image from the other.

One of the most popular applications of this architecture is image classification. The neural network consists of several convolutional layers mixed with nonlinear and pooling layers. When the image is passed through one convolution layer, the output of the first layer becomes the input for the second layer. This process continues for all subsequent layers.

Here, All training Images encoded using InceptionV3 model to extract image features of various sizes and encodes them into vector space which can be fed to our model in a later stage. In this task, CNN is used to encode features instead of classify images. As a result, we removed the fully connected layers and the max pool layers at the end of the network.

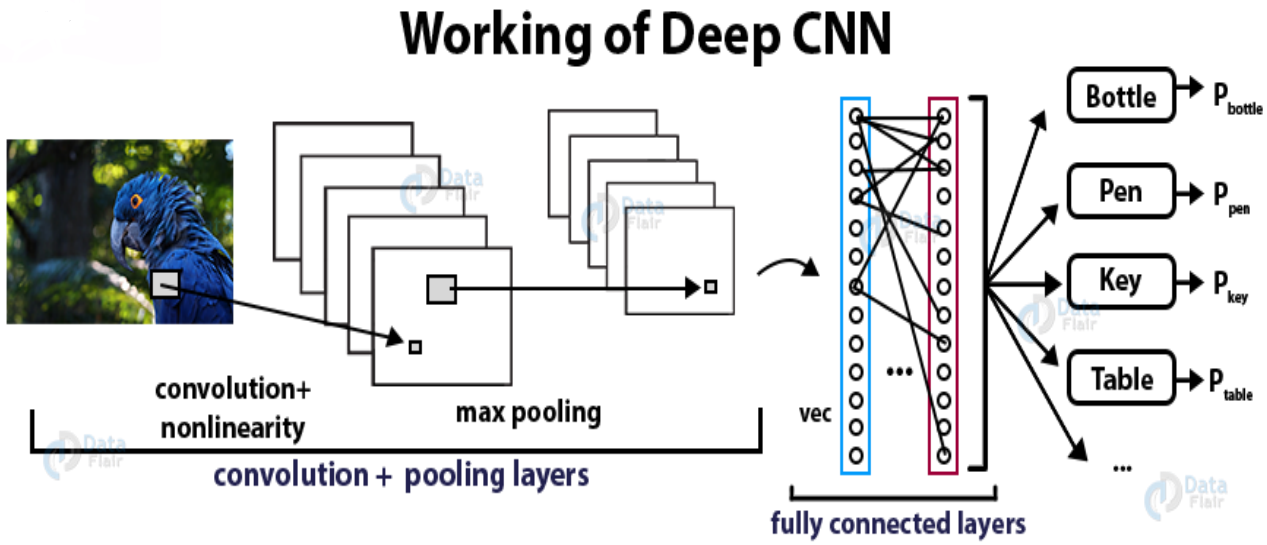


Fig : 0.2

2.3 LSTM Cell

Long Short Term Memory Network is an advanced RNN, a sequential network, that allows information to persist. It is capable of handling the vanishing gradient problem faced by RNN. A recurrent neural network is used for persistent memory.

A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell.

LSTMs take the encoded image feature vectors from CNN and the encoded image captions produced in data preprocessing stage.

The other important thing is the output from RNN network is a series of likelihoods of words (likelihoods). Picking highest likelihood word at each decode step in LSTM tend to yield a sub-optimal result.

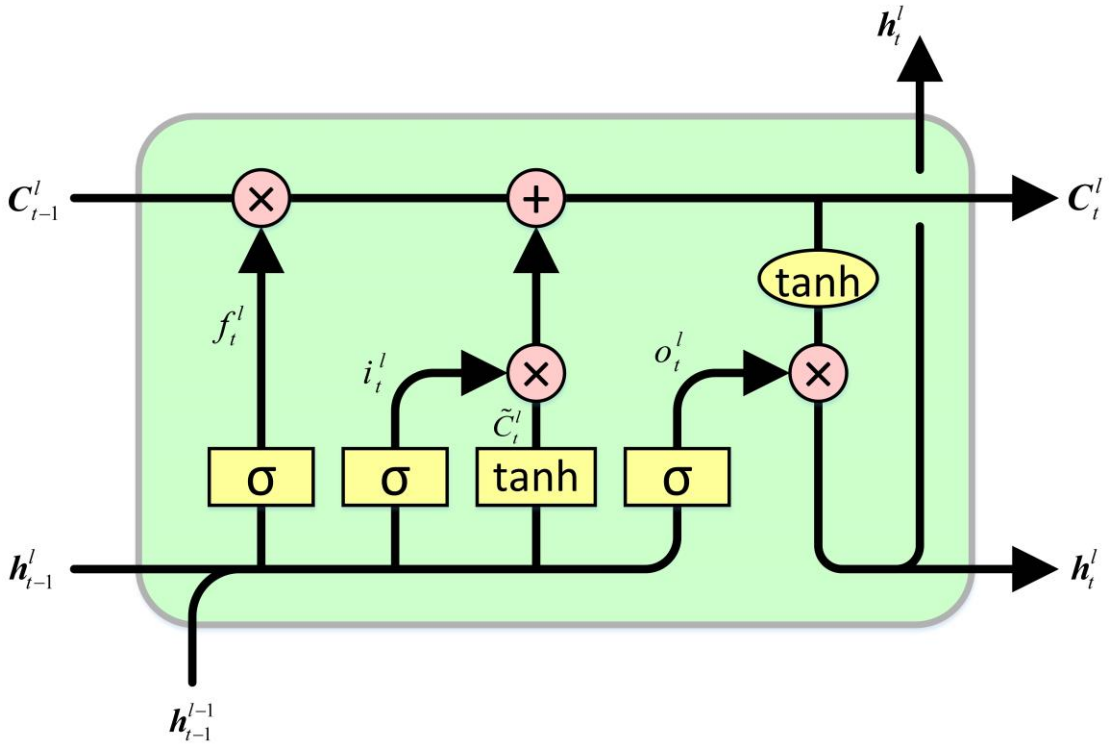


Fig : 0.3

2.4 Final model

So, to make our image caption generator model, we merging above two model. It is also called a CNN-RNN model.

CNN is used for extracting features vectors from the image. And pass these vectors to LSTM model.
LSTM will use the information from CNN to help generate a description of the image.

Final model architecture look like Fig : 0.4

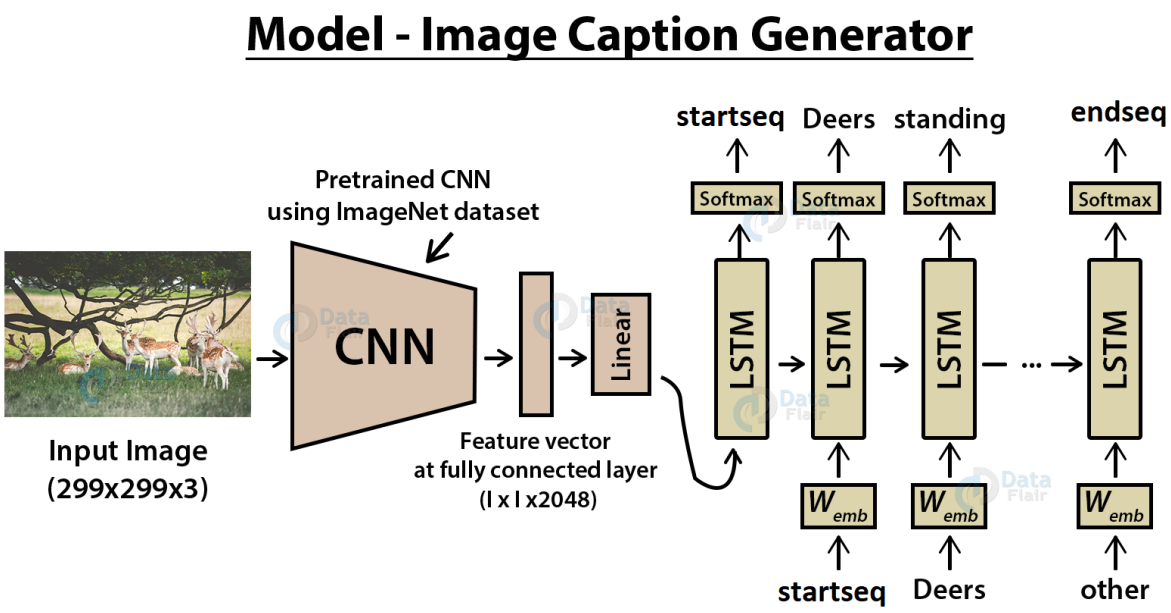


Fig : 0.4

3. Model Execution

Execution of the entire program takes place in 5 major steps. The implementation of the five major modules is as follows:

3.1 Data Cleaning and Preprocessing:

1. Our program starts with loading both, the text file, and the image file into separate variables.
2. The main task of data cleaning involves removing punctuation marks, converting the whole text to lowercase, removing stop words and removing words that contain numbers.
3. Further, a vocabulary of all unique words from all the descriptions is created, which in the future will be used to generate captions to test images.
4. Another aspect of Preprocessing the data involves tokenizing our vocabulary with a unique index value. This is because a computer won't understand regular English words, hence they need to be represented using numbers.
5. We use here glove.6B.200d dataset that is an unsupervised learning algorithm for obtaining vector tokenizing for words. We use represent description.

3.2 Extraction of feature vectors:

1. A feature vector(or simply feature) is a numerical value in the matrix form, containing information about an object's important characteristics
2. In our model we'll be using Transfer Learning, which simply means, using a pretrained model(in our case the inceptionV3 model) to extract features from it.
3. The inceptionV3 model is a Convolutional Neural Network that is 43 layers deep. It is trained on the famous Imagenet dataset which has millions of images and over 1000 different classes to classify from.

4. Python makes using this model in our code extremely easy with keras. To use it in our code, we'll drop the classification layer from it, and hence obtain the 2048 feature vector.

5. Hence weights will be downloaded for each image, and then image names will be mapped with their respective feature array. this mapped values stored in pickle file.

6. This process take a few hours depending on your processor. A feature vector(or simply feature) is a numerical value in the matrix form, containing information about an object's important characteristics.

3.3 Layering the CNN-RNN model:

1. **Feature Extractor:** It will be used to reduce the dimensions from 2048 to 256. We'll make use of a Dropout Layer. One of these will be added in the CNN and the LSTM each. We have pre-processed the photos with the InceptionV3 model (without the output layer) and will use the extracted features predicted by this model as input.
2. **Sequence Processor:** This Embedding layer will handle the textual input, followed by the LSTM layer.
3. **Decoder:** We will merge the output from above two layers, and use a Dense layer to make the final predictions. Both the feature extractor and sequence processor output a fixed-length vector. These are merged together and processed by a Dense layer. The number of nodes in the final layer will be the same as the size of our vocabulary.
4. **Structure of the Neural Network:** Fig : 0.5 is whole structure of neural network.

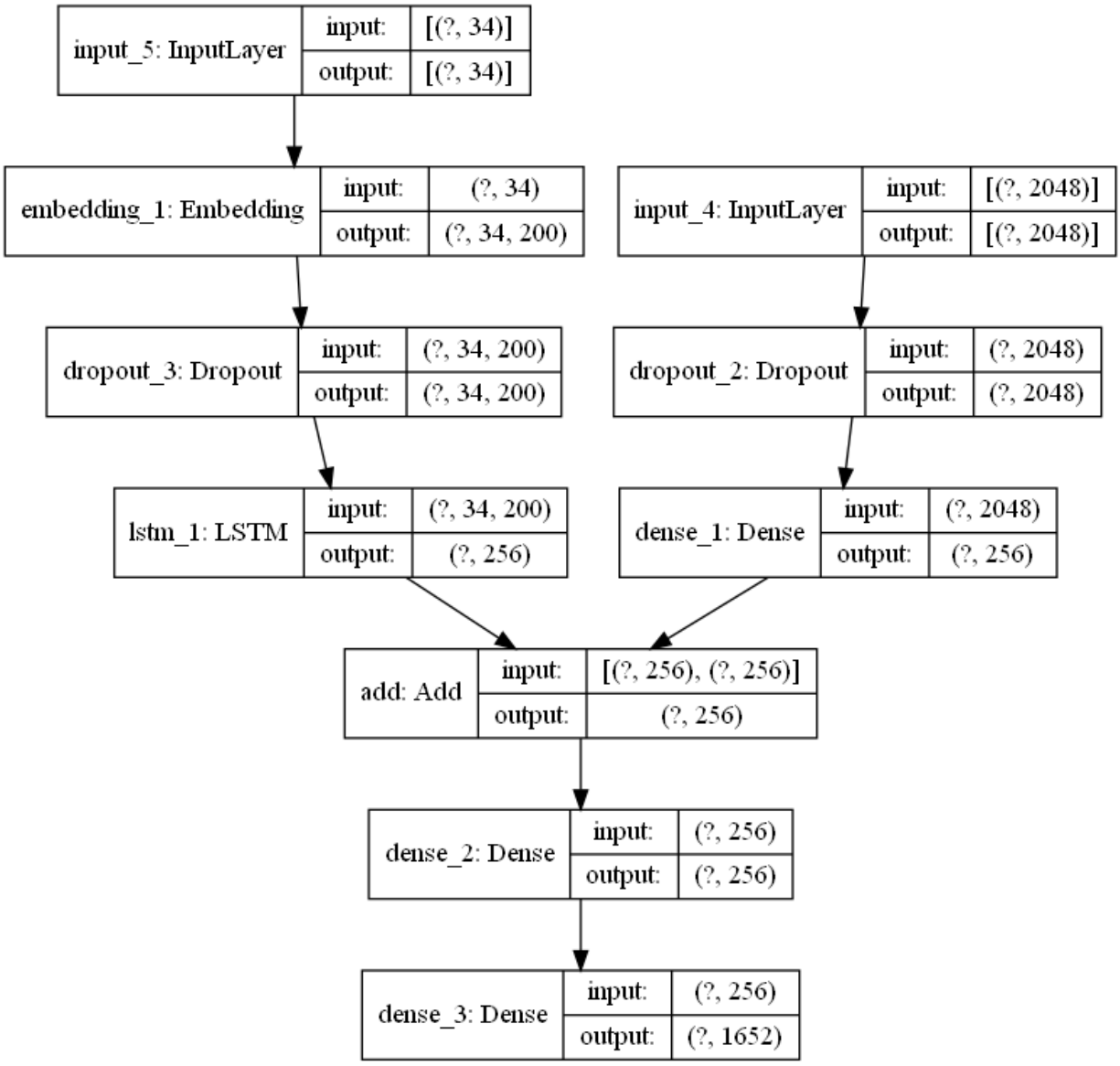


Fig : 0.5

3.4 Training the model:

1. We'll be training our model on 6000 images each having 2048 long feature vectors.

2. Since it is not possible to hold all this data in the memory at the same time, we'll make use of a Data Generator. This will help us create batches of the data, and will improve the speed.

3. Along with this we'll be defining the number of epochs(i.e. iterations of the training dataset) the model has to complete

during its training. This number has to be selected in such a way that our model is neither underfitted nor overfitted.

4. model.fit_generator() method will be used. And this whole process will take some time depending on the processor.

5. The maximum length of descriptions calculated earlier will be used as parameter value here. It will also take as input the cleaned and tokenized data.

6. While training our model, we can use the development dataset(It's provided with the rest of the files), to monitor the performance of the model, to decide when to save the model version to the file.

7. We will proceed to save several models, out of which the final one will be used for testing in future. Fig : 0.6 is training short.

```
for i in range(30):
    model.fit([X1, X2], y, epochs = 1, batch_size = 256)
    if(i%2 == 0):
        model.save_weights("image-caption-weights" + str(i) + ".h5")
```

[43] ✓ 188m 11.7s

```
Output exceeds the size limit. Open the full output data in a text editor
1142/1142 [==============================] - 454s 398ms/step - loss: 3.8974 - accuracy: 0.2611
1142/1142 [==============================] - 656s 574ms/step - loss: 3.1545 - accuracy: 0.3269
1142/1142 [==============================] - 608s 532ms/step - loss: 2.9201 - accuracy: 0.3506
1142/1142 [==============================] - 396s 346ms/step - loss: 2.7692 - accuracy: 0.3676
1142/1142 [==============================] - 405s 355ms/step - loss: 2.6533 - accuracy: 0.3806
...
1142/1142 [==============================] - 341s 299ms/step - loss: 1.8708 - accuracy: 0.4944
1142/1142 [==============================] - 346s 303ms/step - loss: 1.8611 - accuracy: 0.4958
1142/1142 [==============================] - 351s 307ms/step - loss: 1.8475 - accuracy: 0.4988
1142/1142 [==============================] - 349s 305ms/step - loss: 1.8374 - accuracy: 0.5004
```

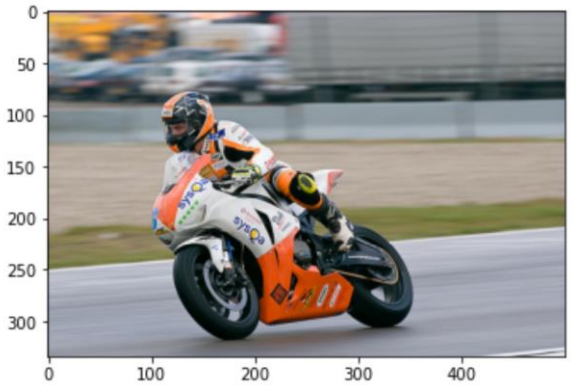
Fig : 0.6

3.5 Testing the model:

1. We loaded last saved model to perform several prediction.
2. The primary step of feature extraction for the particular image under observation will be performed.
3. The path of one of the images from the remaining 2000 test images is passed to the function manually. You can also iterate through the test data set, and store the prediction for each image in a dictionary or a list.
4. The actual functioning behind image generation involves using the start sequence and the end sequence, and to call the model recursively to generate meaningful sentences
5. Four images have been subjected to testing, and the results can be seen in the following images.

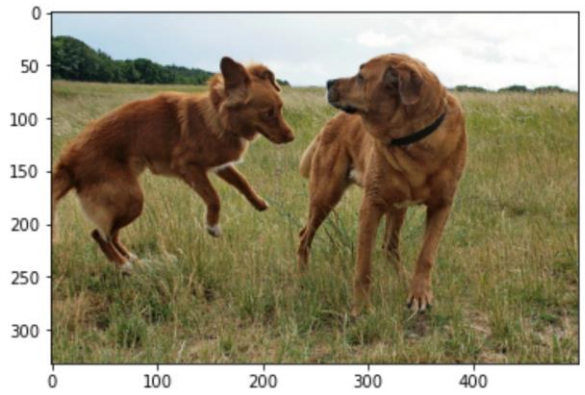
white crane landing over water

Fig : 0.7



motorcycle racer speeds along track

Fig : 0.8



two dogs are running through the grass

Fig : 0.9

dog runs through the water

Fig : 0.10

Image	Original Description	Predicted Description
Fig : 0.7	bird flies above the water	white crane landing over water
Fig : 0.8	motorcyclist races around track	motorcycle racer speeds along track
Fig : 0.9	one dog is jumping up at another dog in grassy field	two dogs are running through the grass
Fig : 0.10	tan dog jumps into water	dog runs through the water

4. CONCLUSION

Based on the results obtained we can see that the deep learning methodology used here bore successful results. The CNN and the LSTM worked together in proper synchronization, they were able to find the relation between objects in images.

Our project aims to implement an Image caption generator that responds to the user to get the captions for a provided image. The ultimate purpose of Image caption generator is to make users experience better by generating automated captions. We can use this in image indexing, for visually impaired persons, for social media, and several other natural language processing applications. What is most impressive about these methods is a single end-to-end model can be defined to predict a caption, given a photo, instead of requiring sophisticated data preparation or a pipeline of specifically designed models. In this an Image caption generator, Basis on our provided image It will generate the caption from our trained model. The basic idea behind this is that users will get automated captions when we use or implement it on social media or on any applications.

References :

1.
Authors - Haoran Wang , Yue Zhang,and Xiaosheng Yu
Topic - An Overview of Image Caption Generation Methods
Hindawi Computational Intelligence and Neuroscience
Volume 2020, Article ID 3062706, 13 pages
https://doi.org/10.1155/2020/3062706

2.
Authors - Megha J Panicker, Vikas Upadhayay, Gunjan Sethi, Vrinda Mathur
Topic - Image Caption Generator
International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-10 Issue-3, January 2021

3.
https://iopscience.iop.org/article/10.1088/1742-6596/1712/1/012015/pdf, Image Captioning Using Deep Convolutional Neural Networks (CNNs)

4.
https://arxiv.org/pdf/1502.03044.pdf Show, Attend and Tell: Neural Image Caption Generation with Visual Attention

5.
B.Krishnakumar, K.Kousalya, S.Gokul, R.Karthikeyan, and D.Kaviyarasu, “IMAGE CAPTION GENERATOR USING DEEP LEARNING”, (international Journal of Advanced Science and Technology- 2020)

6.
MD. Zakir Hossain, Ferdous Sohel, Mohd Fairuz Shiratuddin, and Hamid Laga, “A Comprehensive Survey of Deep Learning for Image Captioning” ,(ACM-2019) 4. Rehab Alahmadi, Chung Hyuk Park, and James Hahn, “Sequence-tosequence image caption generator”, (ICMV-2018)