AUTOMATED CAPTION GENERATOR FOR THE VISUALLY IMPAIRED

Anchal Awasthi
Sailee Chitnis
Alisha Fal Dessai
Chitrangi Jog
Sneha Katasani

Dissertation submitted in partial fulfillment of the requirements for the degree of

BACHELOR OF ENGINEERING

IN

INFORMATION TECHNOLOGY

OF GOA UNIVERSITY

2020-2021

INFORMATION TECHNOLOGY
PADRE CONCEICAO COLLEGE OF ENGINEERING
VERNA GOA – 403722

AUTOMATED CAPTION GENERATOR FOR THE VISUALLY IMPAIRED

Anchal Awasthi
Sailee Chitnis
Alisha Fal Dessai
Chitrangi Jog
Sneha Katasani

Dissertation submitted in partial fulfillment of the requirements for the degree of

BACHELOR OF ENGINEERING

IN

INFORMATION TECHNOLOGY

OF GOA UNIVERSITY

2020-2021

INFORMATION TECHNOLOGY

PADRE CONCEICAO COLLEGE OF ENGINEERING

VERNA GOA – 403722

PADRE CONCEICAO COLLEGE OF ENGINEERING

VERNA GOA – 403722

AUTOMATED CAPTION GENERATOR FOR THE VISUALLY IMPAIRED

Bona fide record of work done by

Name	Roll Number	PR Number
Anchal Awasthi	17IT02	201703278
Alisha Fal Dessai	17IT15	201702251
Sailee Chitnis	17IT10	201702252
Chitrangi Jog	16IT12	201608212
Sneha Katasani	17IT27	201702234

Dissertation submitted in partial fulfillment of the requirements for the degree of Bachelor of engineering in Information Technology under of Goa University

2020-2021

Approved By:

…………..……………….

[Prof. Razia de Loyola Furtado Sardinha]

…………..……………..

[Dr. Terence Johnson]
Head of the Department

…………..………………

[Dr. Mahesh B. Parappagoudar]
Principal

Examined By:

Examiner 1: …………………………………...……………………….
(Name / Signature / Date)

Examiner 2: …………………………………...……………………….
(Name / Signature / Date)

CONTENTS

CHAPTER **Page No.**

ACKNOWLEDGMENT

We would like to thank our guide Mrs. Razia Loyola Furtado Sardinha, Our project would not have been possible without her exceptional support. Her enthusiasm, knowledge, and exacting attention to detail have been an inspiration and kept our work on track from the very start. We would like to show our greatest appreciation to our reviewers Mr. Shaba Desai, Mrs. Ravina Quadros and Mr. Siddesh Savant who provided us with their revered time and assistance during the course of our project and provided vital inputs which helped us immensely in successful completion of our project.

ABSTRACT

Image Captioning is the process of generating a textual description according to the contents observed in an image. The core idea of our method is to translate the given visual query into a distributional semantics based form, which is generated by the average of the sentence vectors extracted from the captions of images visually similar to the input image. Image captioning models typically follow an encoder-decoder architecture which uses abstract image feature vectors as input to the encoder. Two kinds of discriminator architectures (CNN and RNN based structures) are introduced since each has its own advantages. We show that our approach provides more accurate results compared to the state-of-the art data-driven methods in terms evaluation.

LIST OF FIGURES

LIST OF ABBREVIATIONS

1. CNN:	Convolution Neural Networks
2. RNN:	Recurrent Neural Networks
3. LSTM:	Long Short Term Memory
4. VGG:	Visual Geometry Group
5. ResNet:	Residual Neural Network
6. BLEU:	Bilingual Evaluation Understudy
7. gTTS:	Google Text-to-Speech

CHAPTER 1

OVERVIEW

1.1 INTRODUCTION

Auto image captioning is the process to automatically generate human like descriptions of the images. It is very dominant task with good practical and industrial significance. Auto Image captioning has a good practical use in industry, security, surveillance, medical, agriculture and many more prime domains. It is not just very crucial but also very challenging task in computer vision. The object detection and image classification tasks just needed to identify objects within the image. However, the task of auto image captioning is not just identifying the objects but also identifying the relationships between them. The idea is to create a total scene understanding of the image. After understanding the scene, it is also required to generate a human like description of that scene. Auto image captioning is performed by carrying out certain key tasks. First, the input photo is fed to the system and an encoded representation is generated using CNN. This representation captures all the essential features in the image. The encoded representation is then fed to a sequence decoder. The output of the decoder is a sequence of tokens that describes the photo. Once the objects are detected and relationships are identified, it is required to generate the text description, i.e. a sequence of words that describes the relationship between the image objects. The sequence of words must be a well-formed sentence.

1.1.1 Problem Statement

To develop a neural network model which generates accurate descriptive captions of any scenario based image fed to it. The model allows the user to understand its surrounding objects as well as the relation between them.

1.2 MOTIVATION

Globally, at least 2.2 billion people have a vision impairment or blindness, out of which at least 1 billion have a vision impairment that is yet to be addressed. Living with visual impairment can be challenging since many daily life situations are difficult to understand without good visual accuracy. To automatically describe the content of an image with properly formed English sentences is a challenging task, but if implemented properly will have great impact by helping visually impaired people to better understand their surroundings.

Let us see few applications where caption generator could be best solution:

a) Self-driving cars: Automatic driving is one of the biggest challenges and if we can properly caption the scene around the car, it can give a boost to the self-driving system.

b) Automatic Captioning can help, make Google Image Search as good as Google Search, as then every image could be first converted into a caption and then search can be performed based on the caption.

c) In Web Development, it is good practice to provide a description for any image that appears on the page so that an image can be read or heard as opposed to just seen. This makes web content accessible.

d) CCTV cameras (closed-circuit television cameras) are everywhere today but along with viewing the world, if we can also generate relevant captions, then we can raise alarms as soon as there is some malicious activity going on somewhere. This could probably help reduce some crime and/or accidents.

e) It can be used to describe video in real time.

All the above scenarios show cast how important vision is and how it can help people with vision impairment.

CHAPTER 2

LITERATURE SURVEY

Most work in visual recognition has originally focused on image classification, i.e. assigning labels corresponding to a fixed number of categories to images. Great progress in image classification has been made over the last couple of years, especially with the use of deep learning techniques. Nevertheless, a category label still provides limited information about an image, and especially visually impaired people can benefit from more detailed descriptions. Some initial attempts at generating more detailed image descriptions have been made, for instance by Farhadi et al. and Kulkarni et al. but these models are generally dependent on hard-coded sentences and visual concepts. In addition, the goal of most of these works is to accurately describe the content of an image in a single sentence.

Many early neural models for image captioning encoded visual information using a single feature vector representing the image as a whole and hence did not utilize information about objects and their spatial relationships. Generating sentences that describe the content of images has already been explored. Several works attempt to solve this task by finding the image in the training set that is most similar to the test image and then returning the caption associated with the test image. Jia et al., Kuznetsova et al., and Li et al. find multiple similar images and combine their captions to generate the resulting caption.

Kuznetsova et al. and Gupta et al. tried using a fixed sentence template in combination with object detection and feature learning. They tried to identify objects and features contained in the image and based on the identified objects contained in the image they used their sentence template to create sentences describing the image. Nevertheless, this approach greatly limits the output variety of the model.

The issue of picture depiction using natural language processing has picked up importance in current trends. In recent days natural language processing methodologies were used to identify objects using attributes and locations. Farhadi et al. states the process involved in converting image to text. Li et al. describes the object as a sentence by dividing the objects and matches the corresponding phrases, finally, all phrases are joined and form a sentence.

CHAPTER 3

TOOLS AND TECHNOLOGIES

3.1 GOOGLE COLAB

- Colaboratory is a free Jupyter notebook environment that requires no setup and runs entirely in the cloud.
- With colaboratory you can write and execute code, save and share your analyses, and access powerful computing resources, all for free from your browser.
- Colab supports many popular machine learning libraries which can be easily loaded in your notebook.
- Another attractive feature that Google offers to the developers is the use of GPU. Colab supports GPU and it is totally free.

3.2 PYTHON

- Python can be used on a server to create web applications.
- Python can be used alongside Software to create workflows.
- Python can be used to handle big data and perform complex mathematics.
- Python can be used for rapid prototyping, or for production-ready software development.
- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc.).
- Python has a simple syntax similar to the English language.
- Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.

3.3 TENSORFLOW

- TensorFlow is an open-source library developed by Google and has become very popular with Machine Learning.
- TensorFlow offers APIs that facilitates Machine Learning. TensorFlow also has a faster compilation time than other Deep Learning libraries such as Keras and Touch.
- TensorFlow supports both CPU and GPU computing devices.
- The execution mechanism is in the form of graphs so that makes it easier to execute the code.
- It takes a long time to train the models in deep learning because of the large amount of data and using TensorFlow makes it easier to write the code and then execute it in a distributed manner.

3.4 MACHINE LEARNING

Machine learning is a subfield of artificial intelligence that uses algorithms in order to learn from data. Basically, the models find patterns on data without explicitly coding what these patterns are.

3.5 DEEP LEARNING

In practical terms, deep learning is just a subset of machine learning that uses artificial neuronal networks must deeper and in a more complex way (with more neurons and more hidden layers) than the original machine learning artificial neuronal networks.

CHAPTER 4

DESIGN

4.1 DATASET

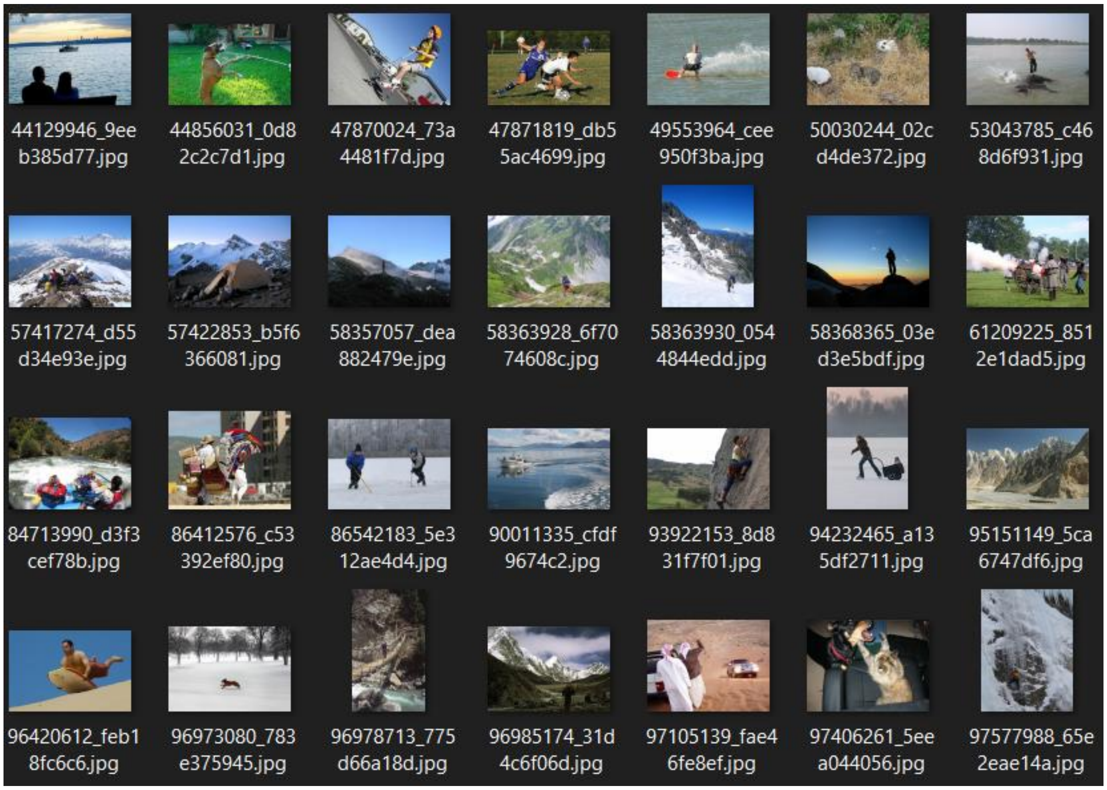
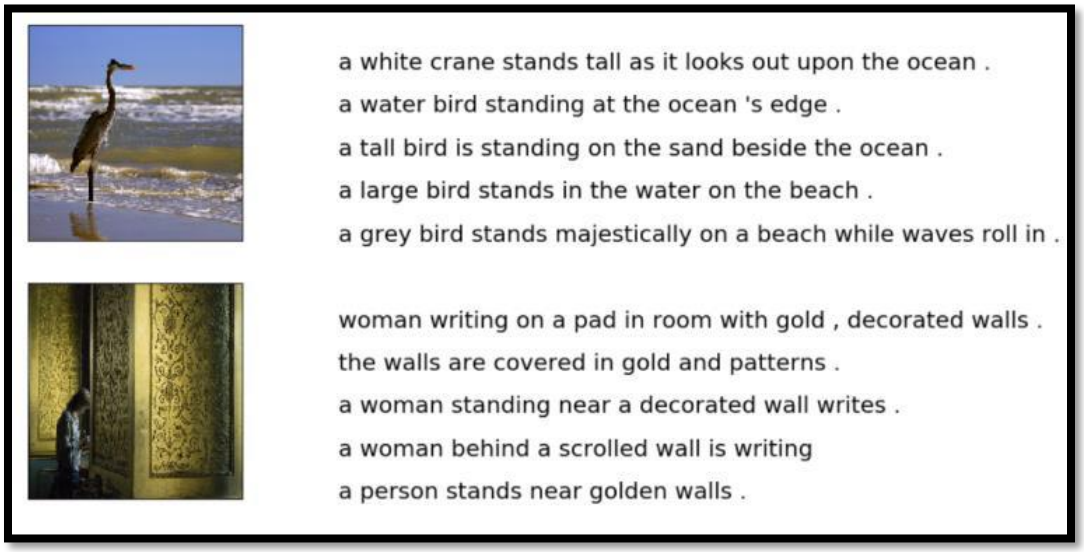


Figure 1 Flickr8k Dataset

For this model we are using Flickr 8k dataset. It is a dataset for sentence-based image description and search, consisting of 8,000 plus images a sample of which can be observed in figure 1 and with each image paired with five different captions which provides clear descriptions of salient entities and events as can be observed in Figure 2. The images in this dataset were chosen from six different Flickr groups, and tend not to contain any well-known people or locations but were selected to depict a variety of scenes and situations. This dataset was selected as it is freely available, realistic, and relatively small so that we can download it and build models on our workstation using a CPU. By associating each image with multiple independently produced sentences, the dataset captures some of the linguistic varieties that can be used to describe the same image. The dataset depicts a variety of events and scenarios and doesn’t include images containing well-known people and places which makes the dataset more generic.



a white crane stands tall as it looks out upon the ocean .
a water bird standing at the ocean 's edge .
a tall bird is standing on the sand beside the ocean .
a large bird stands in the water on the beach .
a grey bird stands majestically on a beach while waves roll in .

woman writing on a pad in room with gold , decorated walls .
the walls are covered in gold and patterns .
a woman standing near a decorated wall writes .
a woman behind a scrolled wall is writing
a person stands near golden walls .

Figure 2. Example images and captions from the Flickr 8k Caption dataset.

4.2 DATA PREPROCESSING

4.2.1. DATA CLEANING

Data pre-processing is a process of preparing the raw data and making it suitable for a machine leaning model. It is the crucial step while creating a machine learning model. It increases the accuracy and efficiency of a machine learning model. The captions contain regular expressions, numbers and other stop words which need to be cleaned before they are fed to the model for further training. The cleaning part involves removing punctuation, single character and numerical values. We clean the captions as it removes major errors and inconsistencies that are inevitable when multiple sources of data are getting pulled into one dataset. The data needs to be converted into vectors before feeding it to the model, and only words can be converted into vectors. The various pre-processing techniques used are Converting text to lower case, punctuation and non-alphanumeric character removal, stopwords, tokenization, parts of speech tagging, named entity recognition, stemming and lemmatization.

4.2.2. TOKENIZING

Tokenization is essentially splitting a phrase, sentence, paragraph, or an entire text document into smaller units, such as individual words or terms. Each of these smaller units is called Tokens. It is one of the most foundational NLP task. The tokens could be words, numbers or punctuation marks. In tokenization, smaller units are created by locating word boundaries. These tokens help in understanding the context or developing the model for the NLP. Tokenization helps in interpreting the meaning of the text by analyzing the sequence of the words. for example, the text "it is raining" can be tokenized into 'it', 'is', 'raining'.

4.3 FEATURE EXTRACTION

Feature extraction is a process of dimensionality reduction by which an initial set of raw data is reduced to more manageable groups for processing. It is the name for methods that select and/or combine variables into features, effectively reducing the amount of data that must be processed, while still accurately and completely describing the original data set.

Feature extraction is performed for the following reasons:

- Feature set reduction, to save resources in the next round of data collection or during utilization.
- To limit storage requirements and increase algorithm speed.
- Performance improvement, to gain in predictive accuracy.
- Data understanding, to gain knowledge about the process that generated the data or simply visualizes the data.

4.3.1. CONVOLUTION NEURAL NETWORKS (CNN)

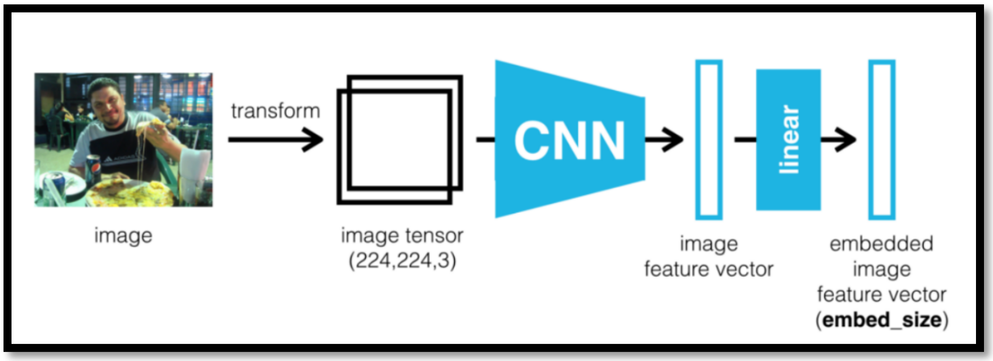


Figure 3. CNN working

A convolutional neural network (CNN) is a type of artificial neural network that has one or more convolutional layers. It is mainly used for image processing, classification, segmentation. Here we take an input image, assign importance (learnable weights and biases) to various objects in the image and be able to differentiate one from the other, the same can be observed in figure 3.

There are four different layers in CNN :

Layer 1:

- It takes features and moves it across the input image.
- While we are move we will multiply the pixel values from the input image with the corresponding pixel values in the output image.

Layer 2:

- Relu is an activation function.
- In this layer, we remove the negative values from the filtered image and replace it by 0.

Layer 3:

- We reduce the size of the image.

Layer 4:

- The actual classification is performed here.
- We take our filtered and shrunk images and put them into a single list.

Role of CNN in Image Captioning

- Image is passed to the first Convolutional layer. The filters applied in the convolution layer extract relevant features from input image to pass further.
- Then next the pooling layer reduces the number of the parameter when the image is too large. Later several convolution and pooling layers are added before prediction is made.
- Convolution layer helps in extracting features. As we go deeper in the network more specific features are extracted as compared to a shallow network.
- The fully connected layer is a layer where the input from other layers will be depressed into the vector.
- The output is then generated through the output layer and is compared to the actual output for error generation which is then back propagated.

4.3.1.1. VGG-16

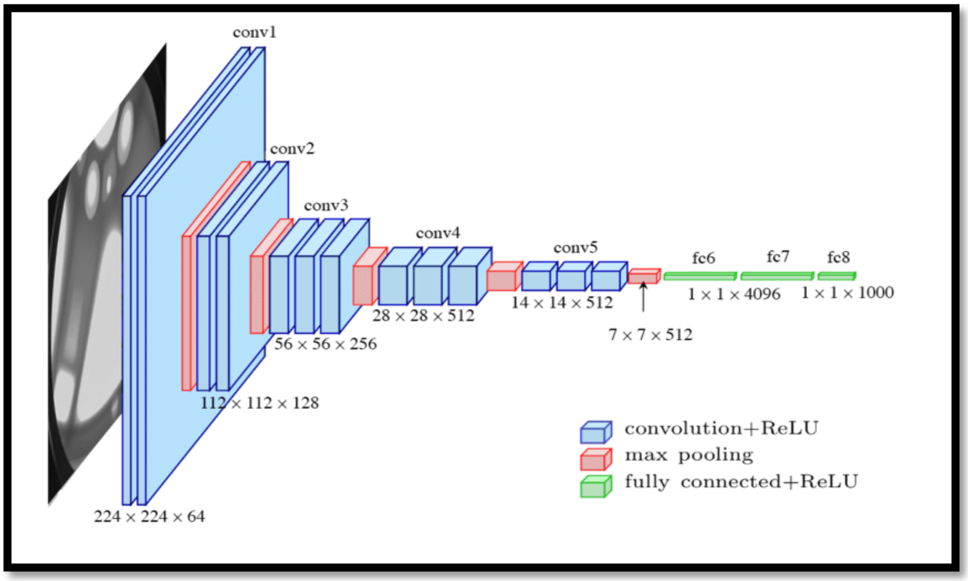


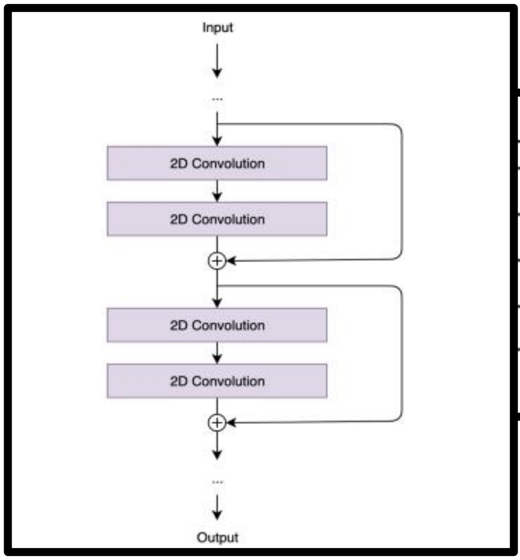
Figure 4. VGG Architecture

VGG-16 is Convolutional neural network architecture; its name VGG-16 comes from the fact that it has 16 layers. Its layers consist of Convolutional layers, Max Pooling layers, Activation layers, fully connected layers. There are 13 Convolutional layers, 5 Max Pooling layers and 3 Dense layers which sums up to 21 layers but only 16 weight layers. The input layers accept color images as an input with the size 224 x 224 and 3 channels i.e. Red, green, and blue. The images pass through a stack of convolution layers. Every convolution kernel uses padding so the resolution after the convolution is performed remains the same. Soft-max layer: is the final layers and it outputs a vector that represents the probability distributions of a list of potential outcomes.VGG-16 network is trained on ImageNet dataset which has over 14 million images and 1000 classes, and achieves 92.7% top-5 accuracy.

4.3.1.2. RESNET

ResNet, short for Residual Networks is a classic neural network used as a backbone for many computer vision tasks. This model was the winner of ImageNet challenge in 2015. While AlexNet has only 5 convolutional layers, the VGG network and GoogleNet have 16 and 22 layers respectively. However, increasing network depth does not work by simply stacking layers together. Deep networks are hard to train because of the notorious vanishing gradient problem — as the gradient are back-propagated to earlier layers, repeated multiplication may make the gradient infinitively small. As a result, as the network goes deeper, its performance gets saturated or even starts degrading rapidly.

The fundamental breakthrough with ResNet was it allowed users to train extremely deep neural networks with 150+layers successfully. Prior to ResNet training very deep neural networks was difficult due to the problem of vanishing gradients. The ResNet architecture consists of blocks, where each block is composed of 2D Convolution. The weights of a neural network are updated using the back propagation algorithm.



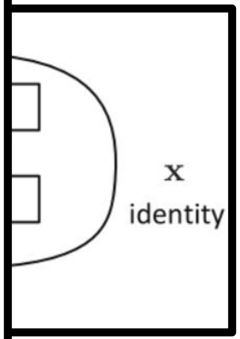


Figure 5. ResNet Architecture

Figure 6. Residual block

The core idea of ResNet is introducing a so-called "identity shortcut connection" that skips one or more layers, as shown in figure 6. Stacking layers with residual blocks (layer that doesn't do anything) shouldn't degrade the network performance. This indicates that the deeper model should not produce a training error. ResNet can be thought of as a special case of Highway which introduces gated shortcut connections. These parameterized gates control how much information is allowed to flow across the shortcut.

4.4. CAPTION GENERATION

4.4.1. LONG SHORT-TERM MEMORY (LSTM)

LSTM is a variant of RNN. It is better than simple RNN because it solves the issues faced by simple RNN. Two major issues faced by simple RNN are the vanishing gradient and the long term dependency problem. LSTM is explicitly designed to avoid the long-term dependency problem, remembering information for long periods of time is their default behavior and have a chain like structure.

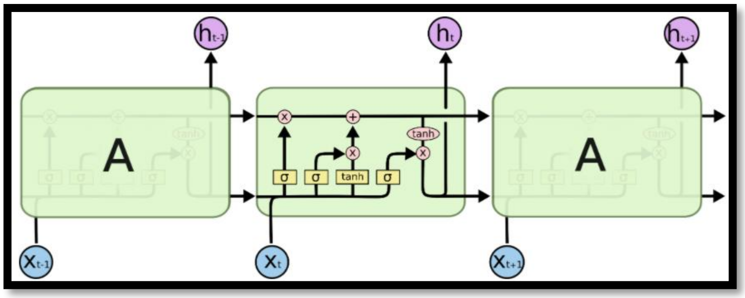


Figure 7. LSTM Architecture

LSTM uses gates to remember the past and gates are the heart of LSTM. Gates which are available in LSTM are (i) input gate (ii) forget gate and (iii) output gate. They all are sigmoid activation function. Sigmoid means output between 0 and 1, mostly 0 or 1. When output is 0, it means gate is blocking. If output is 1 then pass everything.

Role of LSTM in Image Captioning

- CNN encodes the contents of the image into a smaller feature vector. this vector is used as an initial input to LSTM.
- The job of the RNN is to decode the process vector and turn it into a sequence of words. The RNN is trained on the captions in the dataset.
- The decoder is made of LSTM cells which are good for remembering the lengthy sequences of words.
- When in training, first word it produces should always be the <start> token and the next word should be those in the training caption.
- We continue to feed the next word in the caption to the network and so on until we reach the <end>token.
- Once the training cycle is complete and the model is trained, it will have learned from many image caption pairs and should be able to generate captions for new image data.

4.5. EVALUATION

4.5.1. BLEU

BLEU stands for Bilingual Evaluation Understudy. It is an algorithm, which has been used for evaluating the quality of machine translated text. We can use BLEU to check the quality of our generated caption.

Features of BLEU are:

- BLEU is language independent
- Easy to understand
- It is easy to compute.
- It lies between [0,1]. Higher the score better the quality of caption

Calculation of BLEU score is done as follows

Predicted caption = “the weather is good”

References:

1. the sky is clear
2. the weather is extremely good

$$\text{modified ngram precision} = \frac{\text{max number of times ngram occurs in referance}}{\text{total number of ngrams in hypothesis}}$$

- First, convert the predicted caption and references to unigram/bigrams.

Predicted: (the, weather) , (weather, is) , (is, good)

References: (the, sky) , (sky, is) , (is, clear)

(the, weather) , (weather, is) , (is, extremely) , (extremely, good)

BLEU $\Rightarrow$ $\frac{1}{3}+\frac{1}{3}+\frac{0}{3}=\frac{2}{3}$ =0.666

4.6. TEXT-TO-SPEECH

Once the captions are generated by the LSTM model they are fed to the NLP model. In Text-To-Speech conversion the input text is analyzed and then this text is converted into its audio version to play. This functionality has an effective advantage when a person understands a language but is not fluent with reading and writing in that language and is also useful for the people who are visually impaired as they cannot read but understand the message by hearing it. The TTS process is performed step wise, first the text is prepared for audio conversion by performing pre-processing and text normalization. To generate the Waveform of the text message the linguistic analysis and prosodic predictions are done in series. For achieving the above conversation we are using a python library called gTTS. gTTS is a very easy-to-use tool that converts the text entered, into audio which can be saved as an mp3 file. The gTTS API supports several languages including English, Hindi, Tamil, French, German, and many more. The speech can be delivered at any one of the two available audio speeds, fast or slow.

CHAPTER 5

IMPLEMENTATION

Model Flow

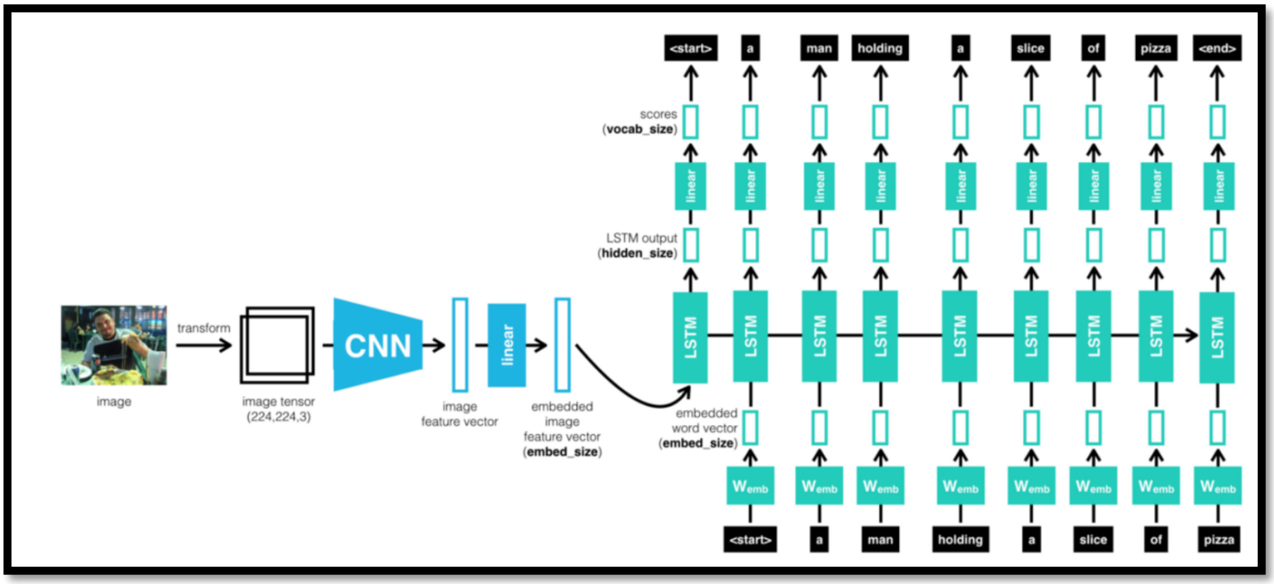


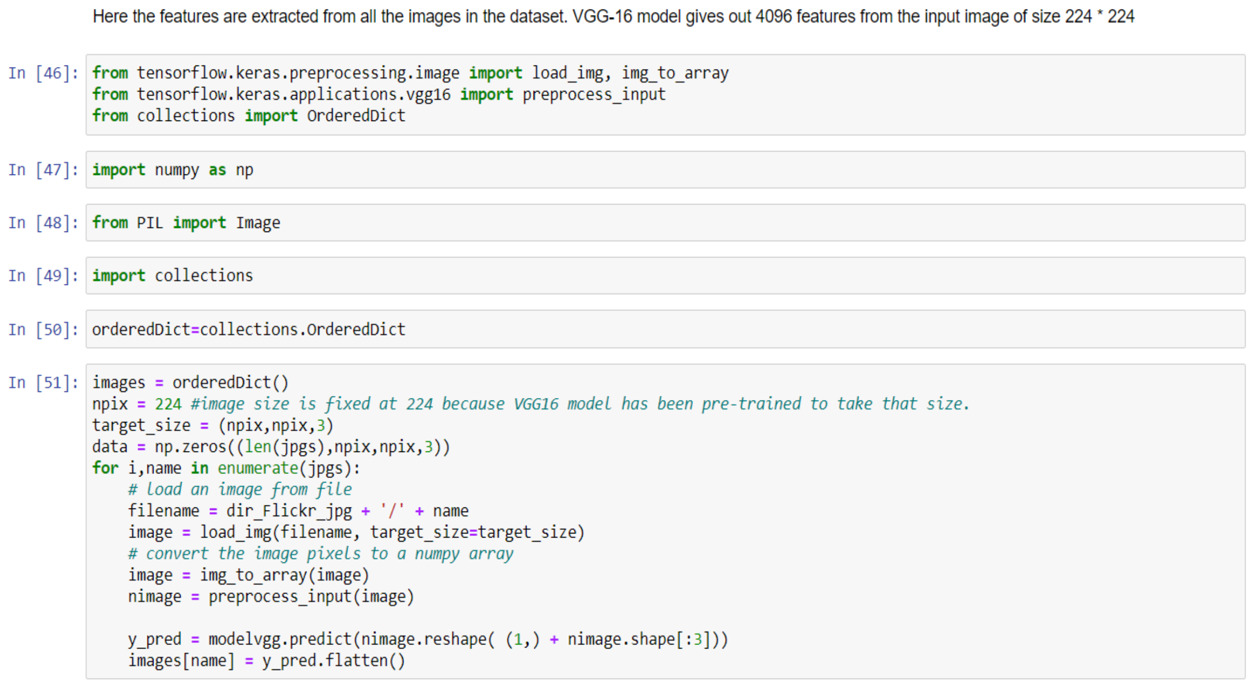
Figure 8. Model Approach

A captioning model is a combination of two separate architecture that is CNN & RNN and in this case LSTM. We want our captioning model to take in an image as input and output a text description of that image. We're using the CNN as a feature extractor that compresses the huge amount of extraction contained in the original image into a smaller representation.

This CNN is often called the encoder because it encodes the content of the image into a smaller feature vector. Then we can process this feature vector and use it as an initial input to the following RNN. The job of the RNN is to decode the process vector and turn it into a sequence of words. Thus, this portion of the network is often called a decoder.

5.1 IMAGE FEATURE EXTRACTION

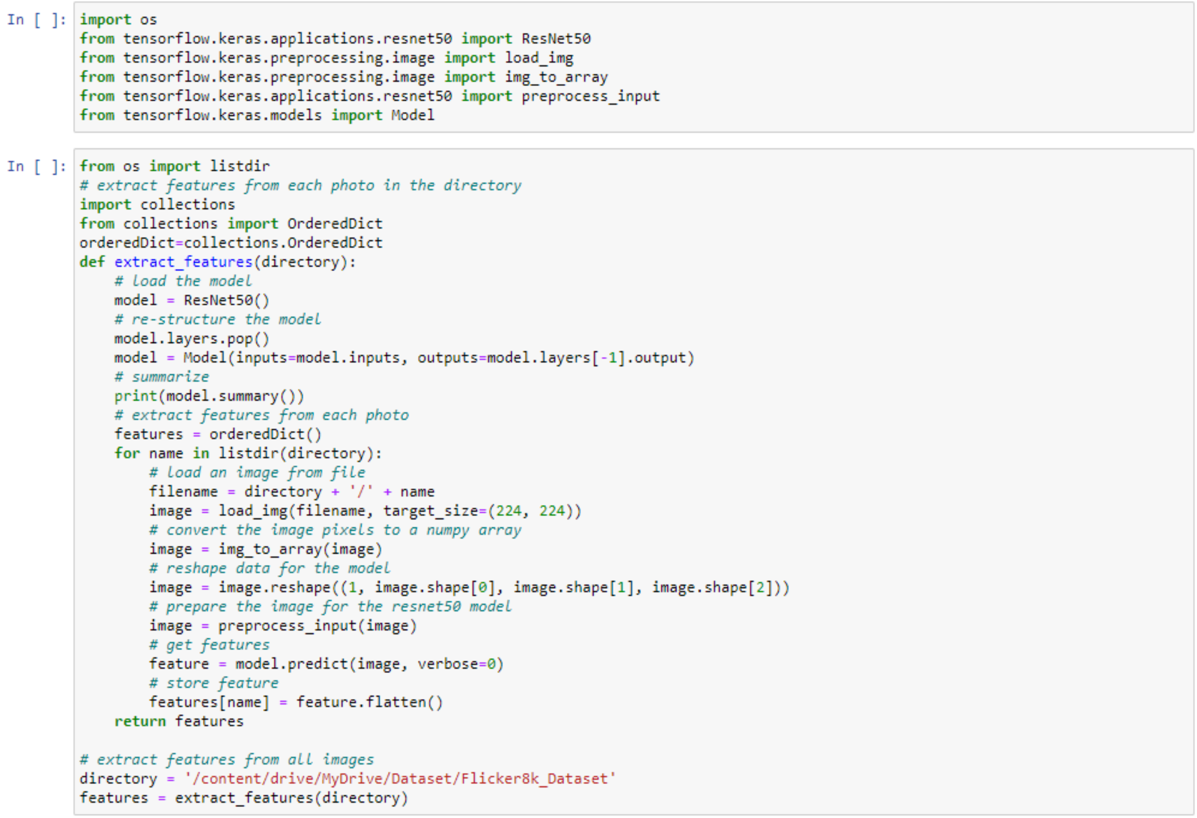
The features of the images from the Flickr 8K dataset are extracted using the VGG 16 model due to the performance of the model in object identification. The VGG is a convolutional neural network which consists of consists of 16 layer which has a pattern of 2 convolution layers followed by 1 dropout layers until the fully connected layer at the end. The dropout layers are present to reduce overfitting the training dataset, as this model configuration learns very fast. These are processed by a Dense layer to produce a 4096-vector element representation of the photo and passed on to the LSTM layer.



Here the features are extracted from all the images in the dataset. VGG-16 model gives out 4096 features from the input image of size 224 * 224

```
In [46]: from tensorflow.keras.preprocessing.image import load_img, img_to_array
         from tensorflow.keras.applications.vgg16 import preprocess_input
         from collections import OrderedDict

In [47]: import numpy as np

In [48]: from PIL import Image

In [49]: import collections

In [50]: orderedDict=collections.OrderedDict

In [51]: images = orderedDict()
         npix = 224 #image size is fixed at 224 because VGG16 model has been pre-trained to take that size.
         target_size = (npix,npix,3)
         data = np.zeros((len(jpgs),npix,npix,3))
         for i,name in enumerate(jpgs):
             # load an image from file
             filename = dir_Flickr_jpg + '/' + name
             image = load_img(filename, target_size=target_size)
             # convert the image pixels to a numpy array
             image = img_to_array(image)
             nimage = preprocess_input(image)

             y_pred = modelvgg.predict(nimage.reshape( (1,) + nimage.shape[:3]))
             images[name] = y_pred.flatten()
```

Figure 9. VGG Approach

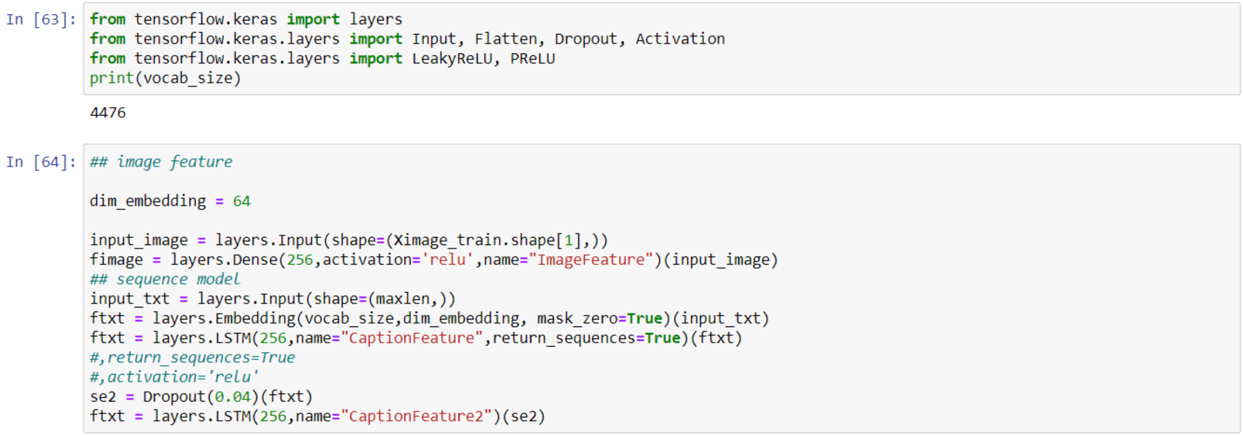


```
In [ ]: import os
        from tensorflow.keras.applications.resnet50 import ResNet50
        from tensorflow.keras.preprocessing.image import load_img
        from tensorflow.keras.preprocessing.image import img_to_array
        from tensorflow.keras.applications.resnet50 import preprocess_input
        from tensorflow.keras.models import Model
```

```
In [ ]: from os import listdir
        # extract features from each photo in the directory
        import collections
        from collections import OrderedDict
        orderedDict=collections.OrderedDict
        def extract_features(directory):
            # load the model
            model = ResNet50()
            # re-structure the model
            model.layers.pop()
            model = Model(inputs=model.inputs, outputs=model.layers[-1].output)
            # summarize
            print(model.summary())
            # extract features from each photo
            features = orderedDict()
            for name in listdir(directory):
                # load an image from file
                filename = directory + '/' + name
                image = load_img(filename, target_size=(224, 224))
                # convert the image pixels to a numpy array
                image = img_to_array(image)
                # reshape data for the model
                image = image.reshape((1, image.shape[0], image.shape[1], image.shape[2]))
                # prepare the image for the resnet50 model
                image = preprocess_input(image)
                # get features
                feature = model.predict(image, verbose=0)
                # store feature
                features[name] = feature.flatten()
            return features

        # extract features from all images
        directory = '/content/drive/MyDrive/Dataset/Flicker8k_Dataset'
        features = extract_features(directory)
```

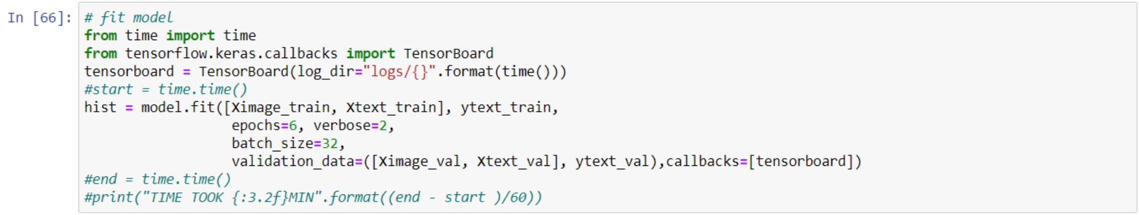
Figure 10. ResNet Approach

5.2 BUILDING AND TRAINING LSTM



```
In [63]: from tensorflow.keras import layers
         from tensorflow.keras.layers import Input, Flatten, Dropout, Activation
         from tensorflow.keras.layers import LeakyReLU, PReLU
         print(vocab_size)

         4476
```

```
In [64]: ## image feature

         dim_embedding = 64

         input_image = layers.Input(shape=(Ximage_train.shape[1],))
         fimage = layers.Dense(256,activation='relu',name="ImageFeature")(input_image)
         ## sequence model
         input_txt = layers.Input(shape=(maxlen,))
         ftxt = layers.Embedding(vocab_size,dim_embedding, mask_zero=True)(input_txt)
         ftxt = layers.LSTM(256,name="CaptionFeature",return_sequences=True)(ftxt)
         #,return_sequences=True
         #,activation='relu'
         se2 = Dropout(0.04)(ftxt)
         ftxt = layers.LSTM(256,name="CaptionFeature2")(se2)
```

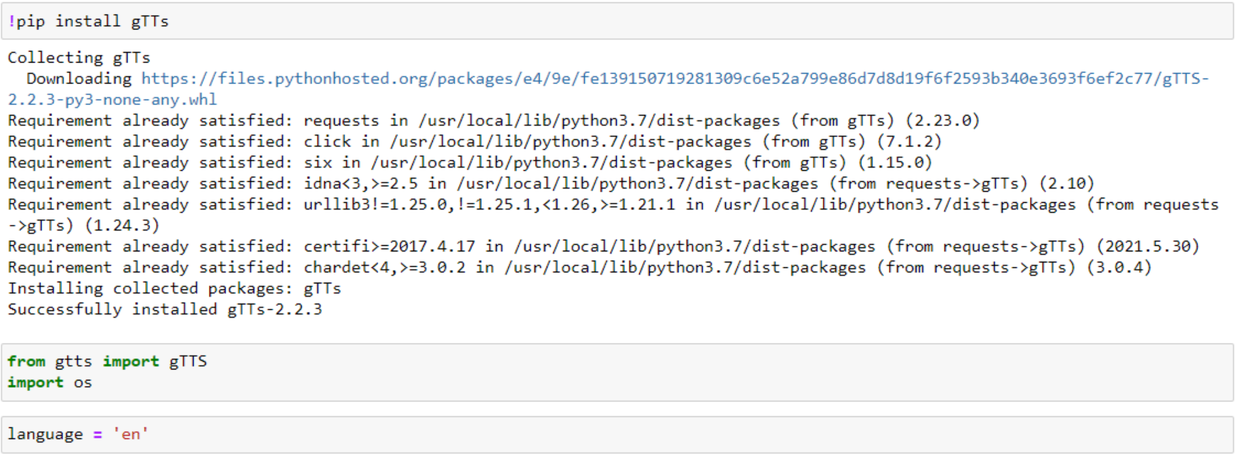


```
In [66]: # fit model
         from time import time
         from tensorflow.keras.callbacks import TensorBoard
         tensorboard = TensorBoard(log_dir="logs/{}".format(time()))
         #start = time.time()
         hist = model.fit([Ximage_train, Xtext_train], ytext_train,
                          epochs=6, verbose=2,
                          batch_size=32,
                          validation_data=([Ximage_val, Xtext_val], ytext_val),callbacks=[tensorboard])
         #end = time.time()
         #print("TIME TOOK {:3.2f}MIN".format((end - start )/60))
```

Figure 11. Building and Training LSTM

The function of a sequence processor is for handling the text input by acting as a word embedding layer. The embedded layer consists of rules to extract the required features of the text and consists of a mask to ignore padded values. The network is then connected to a LSTM for the final phase of the image captioning.

5.3 TEXT TO SPEECH



```
!pip install gTTs
```

```
Collecting gTTs
  Downloading https://files.pythonhosted.org/packages/e4/9e/fe139150719281309c6e52a799e86d7d8d19f6f2593b340e3693f6ef2c77/gTTS-
2.2.3-py3-none-any.whl
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages (from gTTs) (2.23.0)
Requirement already satisfied: click in /usr/local/lib/python3.7/dist-packages (from gTTs) (7.1.2)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from gTTs) (1.15.0)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests->gTTs) (2.10)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python3.7/dist-packages (from requests
->gTTs) (1.24.3)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (from requests->gTTs) (2021.5.30)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from requests->gTTs) (3.0.4)
Installing collected packages: gTTs
Successfully installed gTTs-2.2.3
```

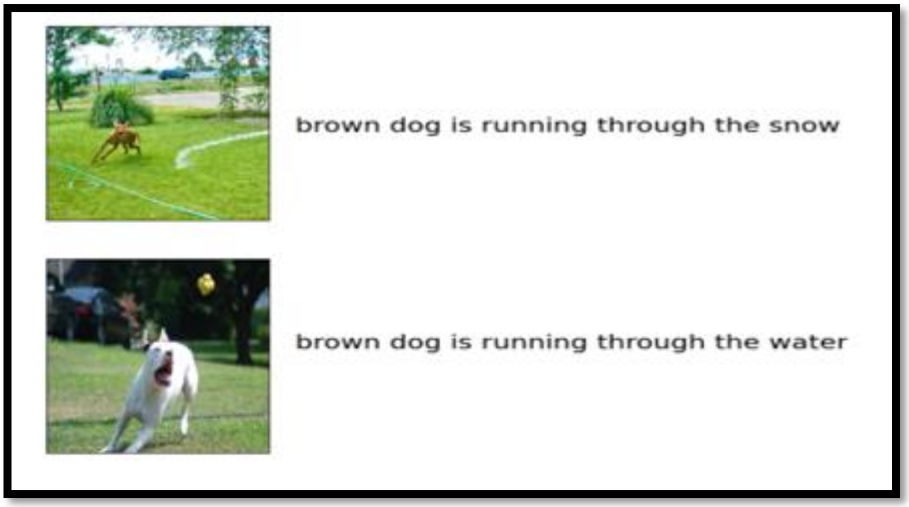
```
from gtts import gTTS
import os
```

```
language = 'en'
```

Figure 12. Converting Text-to-Speech

The captions generated by the LSTM model are then converted to speech using Google Text to Speech (gTTS) API. gTTS is a very easy to use tool which converts the text entered, into audio which can be saved as an mp3 file. The gTTS API supports several languages including English, Hindi, Tamil, French, German and many more.

OUTPUT



brown dog is running through the snow

brown dog is running through the water

CHAPTER 6

RESULT AND ANALYSIS

The image captioning model was implemented and we were able to generate moderately comparable captions with compared to human generated captions. The CNN model first assigns probabilities to all the objects that are possibly present in the image. The model converts the image into word vector. This word vector is provided as input to LSTM cells which will then form sentence from this word vector. The generated sentences are shown in Fig 13 for VGG and ResNet respectively which compare the evaluation scores. For VGG the generated sentence is **"Brown dog is running through the snow",** for this situation we observed a BLEU score of 75. However, For ResNet the generated sentence is **"Brown dog is running through the water"**, in this scenario we got a BLEU score of 84.Consdering the table it is very evident that ResNet produces better results as compared to VGG for our model. Also, using NLP technology we were able to convert the captions generated by the model to speech.

Our model generates sensible descriptions of images in valid English. As can be seen from example groundings, the model discovers interpretable visual semantic correspondences, even for relatively small object. The generated descriptions are accurate enough to be helpful for visually impaired to better understand their surroundings. In general, we find that a relatively large portion of generated sentences (70%) can be found in the training data. However the average accuracy is around 50%.

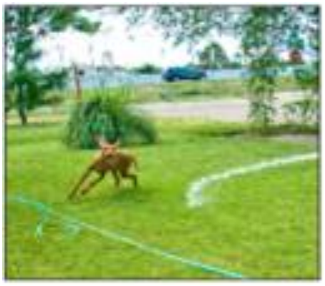
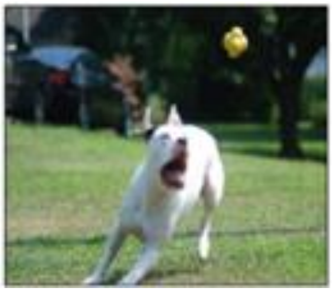
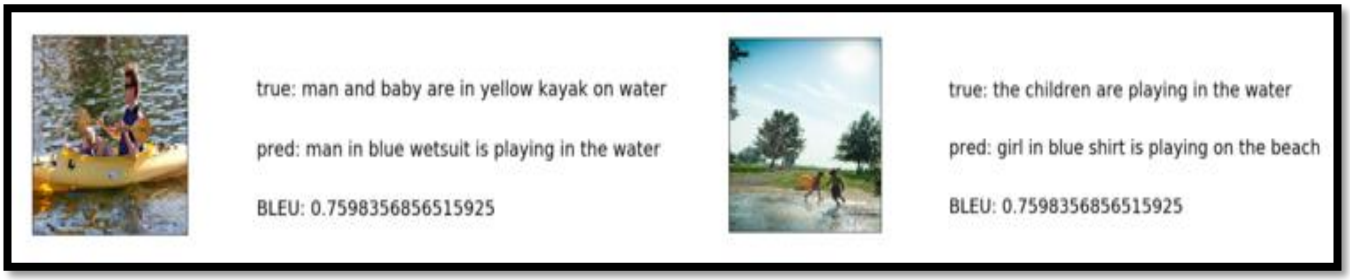
Models		
VGG-16	Brown dog is running through the snow **BLEU**: 0.753	Brown dog is running through the water **BLEU**: 0.723
ResNet	Brown dog is running through the water **BLEU**: 0.841	Dog is running through the air **BLEU**: 0.872

Figure 13. Generated Output & Evaluation score for VGG and ResNet



true: man and baby are in yellow kayak on water

pred: man in blue wetsuit is playing in the water

BLEU: 0.7598356856515925

true: the children are playing in the water

pred: girl in blue shirt is playing on the beach

BLEU: 0.7598356856515925

Figure 14. Example image descriptions generated using LSTM structure

CONCLUSION

Image captioning has made significant advances in recent years. Recent work based on deep learning techniques has resulted in a breakthrough in the accuracy of image captioning. We have presented a deep learning model that automatically generates image captions with the goal of helping the visually impaired better understand their environments. Our described model is based on a CNN that encodes an image into a compact representation, followed by a LSTM that generates corresponding sentences based on the learned image features. The model has been successfully trained to generate the captions as expected for the images.

We showed that both VGG and ResNet models achieve comparable to state-of-the-art performance, and that the generated captions are highly descriptive of the objects and scenes depicted on the images, however ResNet produces better quality captions, that are higher in evaluation score as well. ResNet is able to do so because it overcomes the disappearing gradient issue. Thus, because of the high quality of the generated image descriptions, visually impaired people can greatly benefit and get a better sense of their surroundings. Furthermore, we converted the generated captions, produced by the respective models into speech using the NLP technology.

Considering the future aspects of the project, we intend to introduce a lot of new functionalities such as Path Navigation which could guide the visually impaired to their desired destinations safely, Self-Driving Car wherein the scenes around the car could be captioned to boost the self-driving experience for the visually impaired and Video Description wherein the CCTV cameras incorporated with our model could raise alarms when a malicious activity is spotted, thus reducing crimes.

BIBLIOGRAPHY

1. https://towardsdatascience.com/image-captioning-with-keras-teaching-computers-to-describe-pictures-c88a46a311b8

2. lamri, Christopher and Teun de Planque. “Automated Neural Image Caption Generator for Visually Impaired People.” (2016).

3. Farhadi, Ali, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. “Every Picture Tells a Story: Generating Sentences from Images.” Computer Vision ECCV 2010 Lecture Notes in Computer Science (2010): 15-29. Web. 5 Apr. 2016

4. Makav, B. and Kılıç, V., 2019, November. A New Image Captioning Approach for Visually Impaired People. In 2019 11th International Conference on Electrical and Electronics Engineering (ELECO) (pp. 945-949). IEEE.

5. Kulkarni, Girish, Visruth Premraj, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C. Berg, and Tamara L. Berg. “Baby Talk: Understanding and Generating Simple Image Descriptions.” Cvpr 2011 (2011). Web. 27 May 2016

6. Kumar, N.K., Vigneswari, D., Mohan, A., Laxman, K. and Yuvaraj, J., 2019, March. Detection and recognition of objects in image caption generator system: A deep learning approach. In 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS) (pp. 107-109). IEEE.

7. Fidler, Sanja, Abhishek Sharma, and Raquel Urtasun. “A Sentence Is Worth a Thousand Pixels.” 2013 IEEE Conference on Computer Vision and Pattern Recognition (2013). Web. 18 May 2016