IMAGE CAPTIONING SYSTEM

A
Report submitted
in the partial fulfilment of the requirements for the award of the degree of

Bachelor of Technology
in
Computer Science and Engineering

BY:

Ansh Lehri (1805210008)
Ashutosh Kumar (1805210013)
Himanshu Verma (1805210022)

Under the guidance of

PROF. GIRISH CHANDRA
MR. DEEPANSHU SINGH YADAV

Department of Computer Science and Engineering
Institute of Engineering and Technology, Lucknow
Dr. A.P.J. Abdul Kalam Technical University, Lucknow, Uttar Pradesh

TABLE OF CONTENTS

DECLARATION

We hereby declare that this submission is our own work and that, to the best of our belief and knowledge, it contains no material previously published or written by another person or material which to a substantial error has been accepted for the award of any degree or diploma of university or other institute of higher learning, except where the acknowledgement has been made in the text. The project has not been submitted by us at any other institute for the requirement of any other degree.

Date: 25th May 2022

Submitted by: -

(1) Name:Ansh Lehri

Roll No.: 1805210008

Branch: CSE

Signature: Ansh Lehri

(2) Name:Ashutosh Kumar

Roll No.: 1805210013

Branch: CSE

Signature: Ashutosh Kumar

(3) Name:Himanshu Verma

Roll No.: 1805210022

Branch: CSE

Signature: himanshu

CERTIFICATE

This is to certify that the project report entitled “Image Captioning System” presented by Ansh lehri, Ashutosh Kumar and Himanshu Verma in the partial fulfillment for the award of Bachelor of Technology in Computer Science and Engineering, is a record of work carried out by them under my supervision and guidance at the Department of Computer Science and Engineering at Institute of Engineering and Technology, Lucknow.

It is also certified that this project has not been submitted at any other Institute for the award of any other degrees to the best of my knowledge.

03/06/2022

(Prof. Girish Chandra)

Department of Computer Science and Engineering

Institute of Engineering and Technology, Lucknow

ACKNOWLEDGEMENT

I am highly indebted to Prof. Girish Chandra and Mr. Deepanshu Singh Yadav, and I want to thank them for giving us the freedom to operate and experiment with new ideas. I want to make a move to our significant thanks to them for their educational direction and advantage in our task and steady help combined with certainty boosting and propelling meetings that demonstrated extremely productive and were instrumental in injecting confidence and trust inside us. The sustaining and blooming of the current work is primarily because of their significant direction, ideas, adroit judgment, productive analysis, and an eye for flawlessness. Our mentor consistently addressed a horde of our questions with grinning thoughtfulness and enormous tolerance. They never make us feel like we're on our backsides by constantly listening to our perspectives, respecting and developing them, and allowing us a free hand in our project. It is simply because of their staggering interest and accommodating disposition; the current work has achieved its stage. Finally, I am grateful to our Institution and colleagues whose constant encouragement served to renew our spirit, refocus our attention and energy, and carry out this work.

Ansh Lehri

Ashutosh Kumar

Himanshu Verma

ABSTRACT

Picture subtitling is a course of consequently depicting a picture with at least one regular language sentences. As of late, picture inscribing has seen fast advancement, from introductory layout based models to the ongoing ones, in view of profound brain organizations. This paper gives an outline of issues and late picture subtitling research, with a specific accentuation on models that utilization the profound encoder-decoder architecture.Recent propels in profound learning techniques on perceptual undertakings, for example, picture order and protest discovery have urged specialists to handle considerably more troublesome issues for which acknowledgment is only a stage towards to more mind boggling thinking about our visual world. Picture inscribing is one of such undertakings. The point of picture inscribing is to consequently depict a picture with at least one normal language sentences. This is an issue that coordinates PC vision and regular language handling, so its principal challenges emerge from the need of deciphering between two particular, yet generally matched, modalities. In the first place, it is important to recognize objects on the scene and decide the connections among them and afterward, express the picture content accurately with appropriately framed sentences. The produced portrayal is still very different from the manner in which individuals depict pictures since individuals depend on sound judgment and experience, call attention to significant subtleties and overlook items and connections that they suggest. In addition, they frequently use creative mind to make portrayals distinctive and fascinating.

LIST OF FIGURES

Chapter 1

Introduction

1.1 Background Information

Image Captioning is the system of producing a textual description for given images.It has been an extremely important and basic endeavor in the Deep Learning space. Picture subtitling has a major amount of use. NVIDIA is the usage of picture captioning applied sciences to create an software to assist human beings who have low or no eyesight.

Picture inscribing can be considered as a start to finish Sequence to Sequence issue, as it changes over pictures, which is considered as a grouping of pixels to a succession of words. For this reason, we need to methodology each the language or explanations and the pictures. For the Language part, we utilize intermittent Neural Networks and for the Image part, we use Convolutional Neural Networks to individually accomplish the capacity vectors.

Before moving to further chapters let's understand the about digital imagers and their advantages.An image is visual representation of object. It can be anything from paintings, sculptures, photos etc. The images are in existence from a very long time now. As computers cannot understand images, it became necessary to develop special methods to represent images in computers. These days, they are addressed as a succession of 0s and 1s in PC.

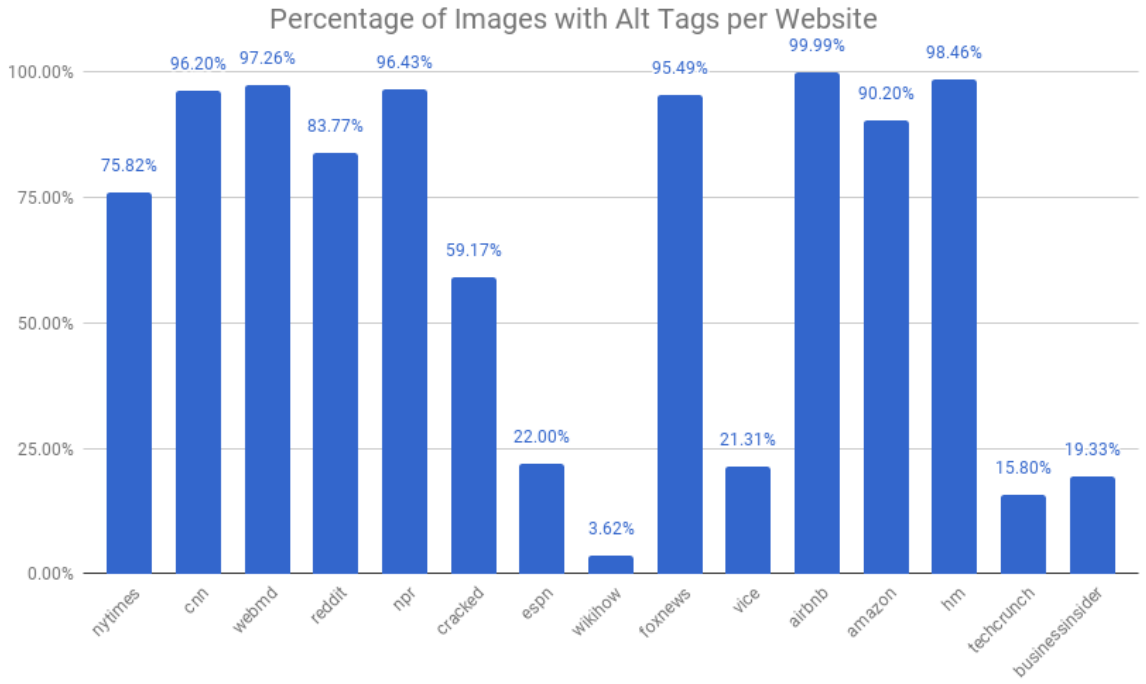
Digital images[4] are of 4 types:

- ☐ Highly contrasting Images, pixel have esteems high or low.
- ☐ Grayscale Images, pixel esteem is range limited showing dull or light.
- ☐ RGB Images, every pixel power is addressed in Red, Blue, Green tone.
- ☐ RGBA pictures, every pixel alonwith RGB part has an extra alpha part.

1.2 Motivation

As per WHO, near about a billion people have some disability and some 280 million have visual impairments. many of them use some devices like screen readers which help them to understand digital text.

As of 2022, about 60% of world has internet access. This increase internet usage has disrupted the market as many content creators are shifting to internet and web. This pose challenge to provide similar level of service to visually impaired people and also a challenge to maintain large amount of images and their text.



1.3 Project Objective

We are going to develop a web application that helps us in identifying the action of an image. We aim at creating a Neural Network Model to analyze the images and create captions using transformer models.

1.4 Report Layout

- Literature Review describes all the previous works done in this field.
- Methodology describes the architecture of the project.
- Implementation Details describes the tools that are used and the process that needs to be followed in each mode.
- Results shows the final outputs that are done in the course of this project.
- End indicates the impediments and future extent of this undertaking.

Chapter 2
Literature Review

One of the most striking notice is the ImageNet project, where they publicly supported huge number of named pictures and prepared models for the last ten years to perceive objects in the picture. Beginning around 2010, the yearly ImageNet Large Scale Visual Recognition Challenge (ILSCRC) holds a contention consistently, to vie for most elevated precision on different visual acknowledgment undertakings. Presently the profound CNN[2] networks have more exactness than people in acknowledgment. Anyway Captioning pictures could be a lot testing task, since it includes object acknowledgment and tracking down connections among them. This has been unthinkable as of not long ago, attributable to gigantic improvement in computational power[13]. Despite the fact that there are different scientists taking care of on a similar issue, there are two groups that stood apart with their calculations. One from Google, and the other from Stanford University. Google delivered a paper "Sharing time: A Neural Image Caption Generator" in 2014 [6]. Their model is prepared to expand the probability of the objective portrayal sentence, given the picture. The model is prepared on different datasets like Flickr30K, SBU, MSCOCO and has accomplished human level execution in creating subtitles. At the point when google originally delivered a paper in 2014, the framework utilized the "Commencement V1" picture characterization model which accomplished 89.6% exactness. The most recent delivery in 2016 utilized "Commencement V3" model, which accomplishes 93.9% precision[3]. Before Google, picture subtitling was conceivable utilizing DistBelief sotware system. Then Google delivered TensorFlow execution, which utilizes GPU power and contrasted with before executions, the preparation time is decreased by a variable of 4. The other group that accomplished well in taking care of the issue is from Stanford University - Fei Li and Andrej Karpathy. Their paper "Profound Visual-Semantic Alignments for Generating Image Descriptions" which delivered in 2015 [7], use pictures and depictions to find out about multi-modular correspondences among language and visual information. They've utilized RNN and CNN to accomplish the errand. Their execution is clustered. It utilizes Torch library, which runs on GPU and upholds CNN finetuning, which sped up by immense component

Chapter 3
Methodology

The methodology is a relevant structure for research. We have multiple sections that cover the Architecture, Working, and Tools Used in the project.

System Architecture

The accompany addresses the system architecture and the essential working of Web Application of Image caption system.

The framework is intended to give subtitles of a picture.

3.1 RNN[1,5]: Recurrent Neural Networks (RNNs) are perhaps the most pervasive engineering in light of the capacity to deal with variable-length texts. They are networks with different circles in them, permitting data to proceed.

3.2 LSTM[5,11]: LSTMs are unequivocally intended to avoid the drawn out reliance issue. Recalling records for an extended time frame is practically their default conduct, at this point not something they battle to learn. Long non super durable memory (LSTM) is a counterfeit repetitive brain local area (RNN) structure utilized in the circle of profound learning. Not at all like chic feedforward brain organizations, LSTM has remarks associations. LSTM networks are appropriate to ordering, handling, and making forecasts upheld measurement measurements on account that there could likewise be slacks of obscure period between essential events in a very measurement.

3.3 Transformers: The figure given below depicts transformers and which is also called a sequence-to-sequence architecture. Sequence-to-Sequence architecture[4] is a neural network which changes a specified succession of components, like grouping words in a sentence, into another grouping. These models are admissible for interpretation, in which the grouping in words from one language is changed to a series of different words in some other dialect.

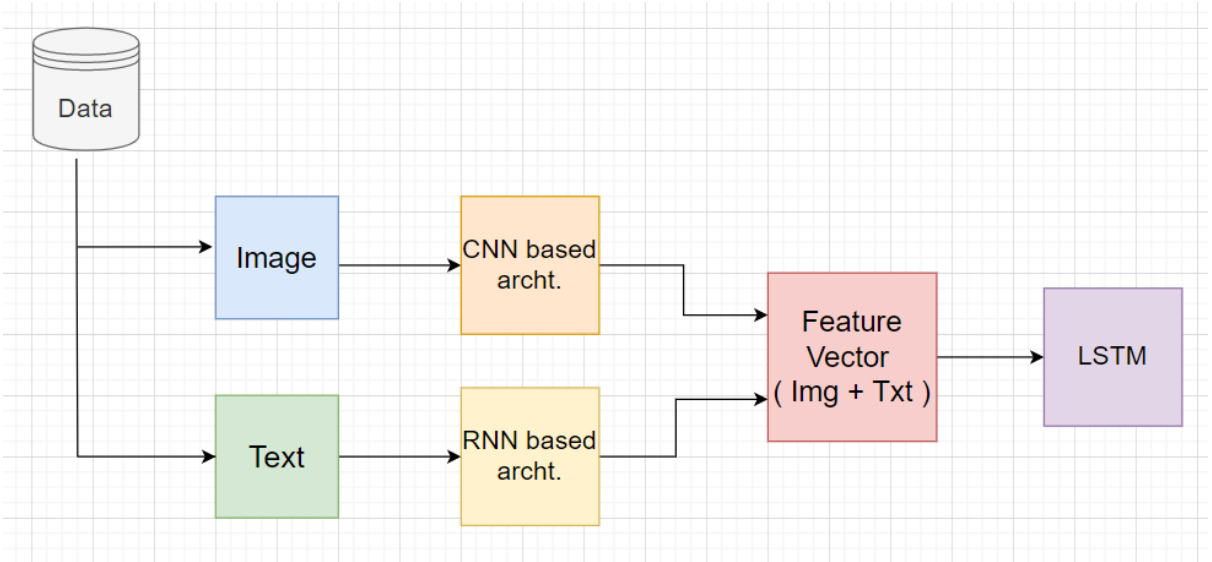


Fig: System Architecture

Above figure proposes the System Architecture of our task that arrangements with the text and pictures.

The walkthrough of the Architecture is as follows:

- The Text Data is cleaned and pre-processed. This Bag Of Words or Embedding Matrix is created to passed through RNN architecture which converts text into its feature vector.

- The image data is pre-processed and size is adjusted to 248*248 to be input into Resnet50 model. The Resnet50 model outputs a feature vector describing the image mathematically.

- Feature vectors created in above steps are concatenated and passed through LSTM model, which train on data and for a particular image learns suitable text.

Chapter 4
Proposed Model

The system developed is a caption generation system. It contains components of pre-processing, caption generation and a caption decoder. During training and testing phase, images are read from the disk. This image is converted from JPEG/PNG to array representation. The image is resized to an array of shap 224X224X3. The image pre-processor normalizes the image to array of shape 224X224. The image is passed through a Renet50 model, which generates a compressed gist of information contained within the image.
For training phase only, text inputs are preprocessed as well. The text is changed to lower case and each sentence is appended with 'startofseq' and 'endofseq'. After this each word is broken down into words and frequency of each word is stored temporarily in form of dictionary. A new dictionary with an indexing of every word is generated which is used to replace words in a sentence with their index. Then a mapping of every image to its index-positioned sentences is maintained.

Vectors obtained for image (in testing) or image and text in training is passed through an architecture of LSTM which generates text caption for the image.

4.1 Architecture

Engineering of the model basically include 2 kinds of brain organizations, RNN and CNN. RNN is Recurrent brain Network and CNN is Convolutional Neural Network. Insights concerning them are examined as follows:

Convolutional Neural Network:

Neural Networks are utilized for Image Recognition. However, the problem with simple Neural Networks is, in the event that the photo is of huge pixels, the no.of.parameters for a Neural people group increments. This makes Neural organizations progressive and consumes a great deal of computational power.

To overcome this problem, CNN[8] are used. The convolutional neural community is a distinctive kind of feed ahead neural network. Convolutional neural networks ingest and procedure pix as tensors, and tensors are matrices of numbers with extra dimensions.

There are three primary kinds of layers to fabricate CNN structures:

- Convolutional layer
- Pooling layer
- Fully-connected layer

The completely associated layer is very much like the standard brain organizations. The convolutional layer can be considered as playing out the convolution activity commonly on the past layer. The pooling layer can be however as downsampling by the limit of each block of the past layer. We stack these three layers to develop the full CNN engineering.

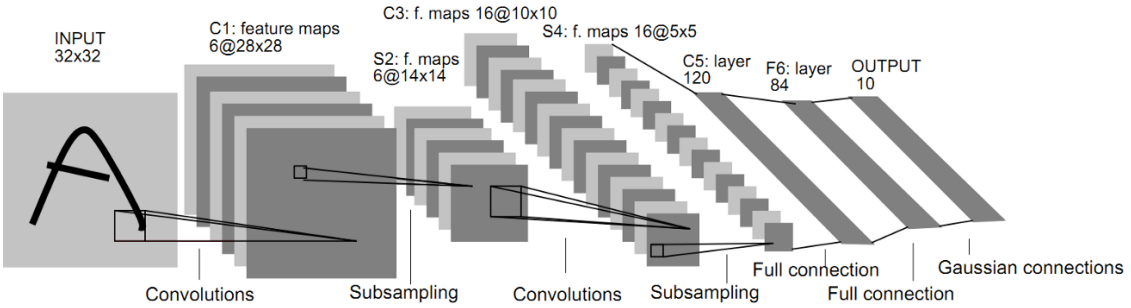


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

An example of CNN architecture (Image from datasciencecentral)

Convolutional Neural Networks have 2 main components.

- **Feature learning**: We have convolution, ReLU,Pooling layer stages here. Edges,shades,lines,curves, in this Feature learning step are get extricated.
- **Classification**: There is Fully Connected(FC) layer[9] in this stage. They will relegate a likelihood for the item on the picture being what the calculation predicts it is.

For our use-case, we are only interested in Feature learning component of CNN.

Feature learning:

- Convolution: Has following parts :
    - **Input image:** Each photograph can be viewed as a network of pixel values. Consider a 5 x 5 picture whose pixel values are exclusively zero and 1.
    - **Filter**: Input photograph is expanded with the guide of a channel to get the Convolved layer. these channel varies in shapes and in values to get exceptional focuses like edges, bends, lines. this channel once in a while referred to as Kernel, Feature finder.
    - **Convolved layer:** each worth in every pixel of info picture is duplicated with separate worth and pixel of channel that gives Convolved layer here. this Convlayer here now and again called as Convolutional Feature, Feature map, Filter map here.

ReLU(Rectified Linear Unit):

An additional an activity known as ReLU[13] has been utilized after every single Convolution activity. Relu is a non-direct activity. ReLU is an issue savvy activity (applied per pixel) and replaces all awful pixel values in the limit map by using zero. The justification for ReLU is to introduce non-linearity in our ConvNet, when you consider that the greater part of this present reality measurements we would favor our ConvNet to look at would be non-straight.

Pooling layer:

In this segment the dimensionality of convlayer or trademark map gets diminished protecting the imperative data. from time to time this spatial pooling is furthermore known as Downsampling or subsampling. this pooling layers could likewise be Max pooling, Avg pooling, total pooling[14]. frequently we see Max pooling is utilized most.

Recurrent Neural Network:

RNN[1] represents Recurrent Neural Network. It is a sort of brain local area which conveys memory and good OK for consecutive information. RNN is utilized by utilizing Apples Siri and Googles Voice Search. We should discuss a few essential norms of RNN.

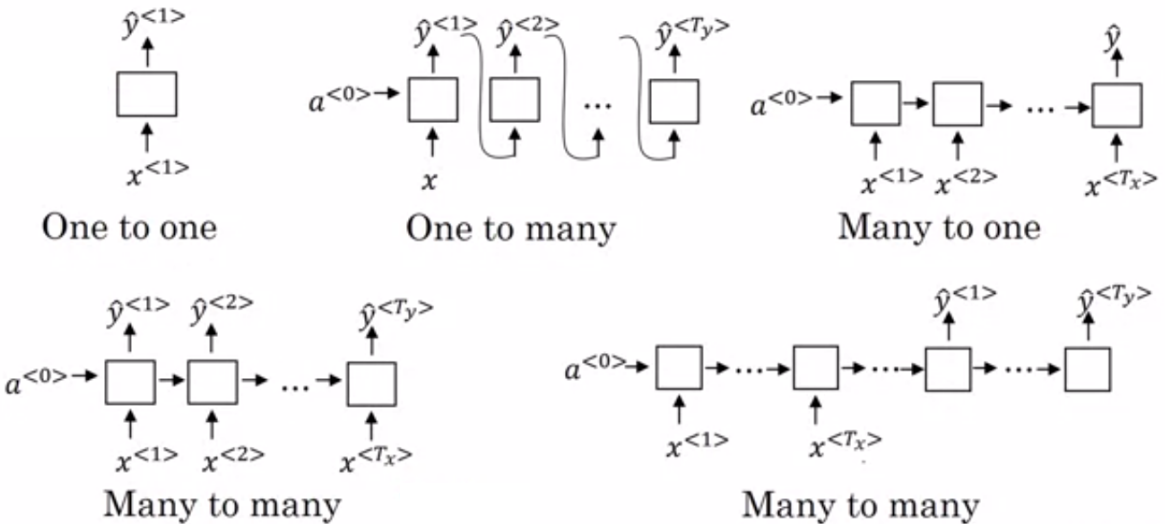
It is a speculation of feed-forward brain local area that has an inside memory. RNN repetitive in nature as it plays out the equivalent component for every single enter of data while the result of the present day enter depends upon on the past one calculation. In the wake of delivering the result, it is duplicated and despatched lower once more into the repetitive organization[5]. For going with a choice, it considers the current day enter and the result that it has found from the previous info.

The contrast among RNN[10] and feed forward brain network is that RNN can utilize inward memory to handle arrangement of data sources. It is reasonable for successive information where result of one info relies upon past conditions of info and result.

RNN has ability to retain past information. While chipping away at current information, it additionally thinks about what it has gained from past conditions of information and result. In this way, it ascertains its present status utilizing set of current info and the past state. Along these lines, the data burns through a circle.

There are different types of RNN:

- one-to-one RNN
- one-to-many RNN
- many-to-many RNN
- many-to-one RNN

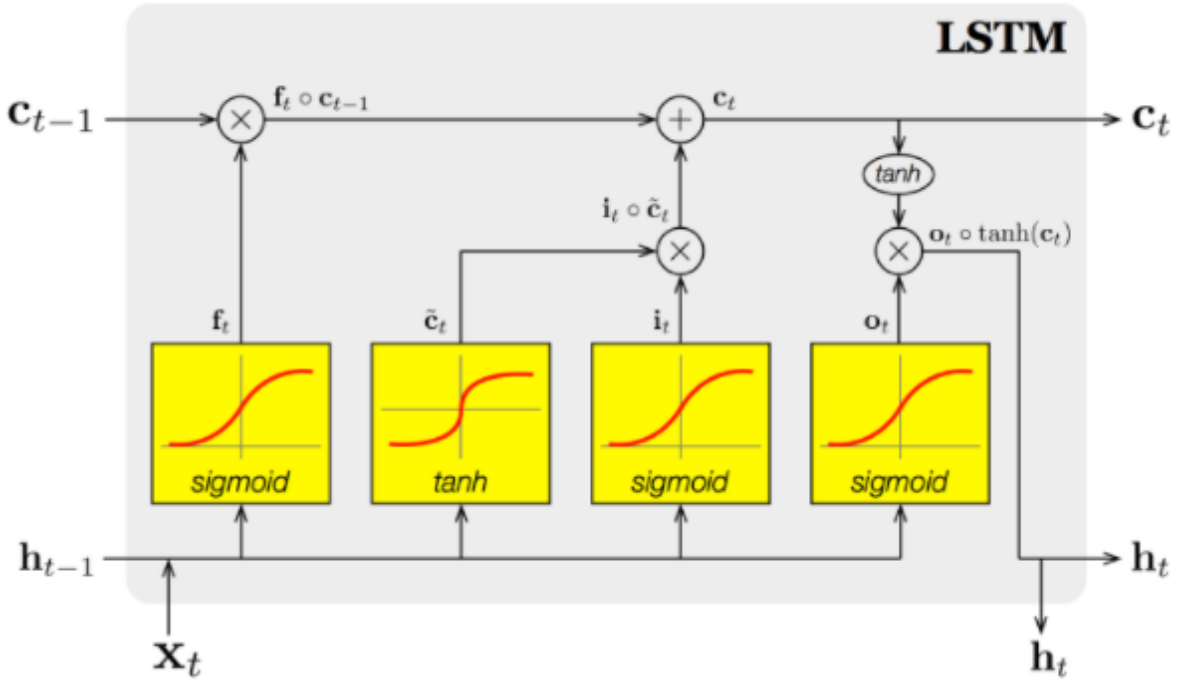


Types of RNN (Image from opengenus)

RNNs are powerful, however are tough to train. The major cause is "vanishing gradient problem". While theoritically RNNs can make use of facts in arbitrarily lengthy sequences, in exercise they are constrained to searching again solely a few steps. This skill in exercise the vary of contextual records that popular RNNs can get admission to are limited[3].To overcome this issue, new type of structures were introduced, one out of which is LSTM.

LSTM (Long Short Term Memory):

LSTM[1,5] is a RNN structure that take note values over arbitrary intervals. It is used to classify, procedure and predict time sequence given time lags of unknown duration. Relative insensitivity to hole size offers an gain to LSTM over choice RNNs, hidden Markov fashions and different sequence gaining knowledge of methods.
The drawn out memory is regularly known as the telephone state. Each telephone has a recursive nature which allows in measurements from going before stretches to be put away in the LSTM cell[10]. Cell country is adjusted by utilizing the disregard entryway situated under the telephone realm and moreover alter via the enter regulation door.
The consider vector is by and large known as the disregard entryway. The result of the disregard entryway advises the cellphone realm which measurements to ignore through duplicating zero to a job in the framework. Assuming the result of the disregard door is 1, the measurements is saved in the mobilephone state.
The store vector is by and large alluded to as the enter entryway. These entryways conclude which information need to enter the versatile country/long haul memory. The important parts are the enactment highlights for each door. The enter entryway is a sigmoid trademark and have a shift of [0,1].
The focal point of consideration vector is regularly alluded to as the result entryway.
The working reminiscence is normally known as the hidden state.



LSTM(Image from Stackoverflow)

4.2 Hardware Requirements

- 16 GB RAM
- NVIDIA GPU
- M1 core
- 50 GB physical storage

4.3 Software Requirements

- MACOS (X)
- Python3
- CUDA 9
- Tensorflow
- Keras
- Numpy
- Matplotlib

4.4 Dataset Sources

We picked our dataset from Kaggle. It is a web-based local area foundation of information researcher and AI lovers. It permits clients to team up on projects, find publicly released datasets, access GPU note pads and so forth.

Our dataset is Flickr images dataset which has about 8000 images. Each image is described in about 5 captions, which give different perspectives to an image. This is a standard dataset and is used by many researchers and developers to develop image captioning model.

.

Chapter 5
Implementation Details

5.1 Languages And Libraries

- OpenCv[11]: OpenCv is a ML package library and associates ASCII text file laptop vision. It's a library of programming functions chiefly geared toward period laptop vision. We have utilized this for the most part in pre-handling pictures.

- HTML: HTML or HyperText Markup Language is a markup language that permits web clients to make and design different pieces of a page like headers, tables and links using elements, tags, and attributes. It tends to be helped by innovations like Cascading Style Sheets (CSS) and prearranging dialects like JavaScript[14]. This would be based on the language for my website that I would create to deploy my three models of Sentiment Analysis.

- NumPy[9]: It is a open source Python library. NumPy works with Python objects called multi-dimensional arrays. Arrays are basically collections of values, and they have one or more dimensions. NumPy array data structure is also called *ndarray*, short for n-dimensional array. Datasets are usually built as matrices and it is much easier to open those with NumPy instead of working with lists. Numpy here is used for many processes. This library does all the numerical calculation in my undertaking.

- Tensorflow[2]: Tensorflow is an open source framework. It was initially designed to be a neural network library but with advancement it can perform much more functions. It is a machine learning library. It is the base library that we have used to create our model; this is the cover of Keras that helped in model designing and fitting the values.

5.2 Setup Used

- Flask[16]: It is a little net design written in Python. It's named a microframework. As a result of it doesn't need explicit tools or libraries. It's no information abstraction layer, type validation, or the other parts wherever pre-existing third- party libraries give standard functions. It has been used to create an interface between the website and models and also is responsible for returning the HTML pages accordingly to the output

- Kaggle Notebooks[17]: Kaggle Notebbok is a free Jupyter notebook climate that runs altogether in the cloud. In particular, it doesn't need an arrangement, and the notebooks that you make can be at the same time altered by your colleagues - how you vary reports in Google Docs. The free GPU of kaggle is used in this project for the training purpose of the Video and Audio modes for providing fast results.

- Spyder[18]: It is a free and open-source logical climate written in Python and planned by and for researchers, specialists, and information examiners. It includes a remarkable mix of an exhaustive advancement instrument's high-level altering, examination, troubleshooting, and profiling usefulness. I have used this to work on all my python files, and all the mathematical work is done over here.

- VS Code[19]: VS Code is a lightweight text editor, one of the best for coding in all most all languages. It provides you to code in any programming language for example Python, Java, C++, JavaScript, and more. Visual Studio Code is a source code editor, which helps businesses build and debug web applications running on Windows, Linux, and macOS. It is a source-*code* editor text editor program de

5.3 Model Training Implementation

The steps that we went through to train our model:

1. As our dataset is divided into 2 files of images and their corresponding text, we first worked on image dataset.
2. We read pictures and went them through picture pre-handling step.
3. In picture pre-handling, pictures are resized to suitable element of 224x224 and reshaped to 224x224x3. The third aspect address RGB part.
4. Once pre-handled, picture is gone through Renet50 model to create an element vector of length 2048.
5. These vectors are stored against their image.
6. Next we pre-process our text dataset.
7. In this, we first read captions for an image which have been pre-processed to generate feature vectors.
8. Once read, captions are first converted to lowercase.
9. Then strings like 'startofseq' and 'endofseq' are added in start and end of captions respectively.
10. 'startofseq' is added so that we can have a starting point when we use deployed model.
11. 'endofseq' is added to show that prediction is over.
12. These captions are then broken into words and frequency of these words is counted.
13. These words/tokens are then indexed and a dictionary is created.
14. The tokens are replaced by their indices in captions.
15. Image feature vectors and the tokenized captions are then mapped together.
16. Next we create architecture for our model, which is divided into 2 parts, image-model and language-model.
17. These models are concatenated with each other and a Dense LSTM layer with softmax activation layer is appended as an output layer to above architecture.
18. In this we use categorical_crossentropy to quantify misfortune, RMSprop as enhancer and exactness as metric.
19. We then passed our pre-processed data to our architecture and train the model.

5.4 Web Site Implementation

The steps that we went through to develop the website:

1. We have used flask to develop our website as it is lightweigh.
2. On hitting our endpoint 'http:127.0.0.1:5000/', user getsredirected to a home page.

3. There user is asked to upload any image they want to see caption for.
4. Once uploaded, user hits predict button and a caption is generated for their image.
5. Under the hood, once predict button is pressed, image is passed as a parameter head of a user request to our server (localhost).
6. We capture the user image and pre-process it.
7. Image pre-processing happens in a similar manner as it happened during model training.
8. Once preprocessed, the image is passed through similar model architecture as that used in mode training.
9. The only difference in input is in textual data. During model training we passed entire captions, with 'startofseq' and 'endofseq' as prefix and suffix respectively. During user testing, we only pass 'startofseq' as text input.
10. 'startofseq' is used to create a sequence of textual predictions.
11. Once entire caption is predicted, it is returned to the user.
12. In model used to predict, we used trained model_vocabulary and model_weights in our architecture.

5.5 Deployment

The project's primary purpose is to have a place where we can test all the capabilities of a project. This deployment is the last stage that will help us to do this work.

We created a website on the local server that will run all the entire project. the model that we have created during the training time will be used up by Flask which will help us to run our project on Local Network so that all the dependencies can be used in one go.

All the steps discussed here were implemented and below are the results of that implementation with the final local web server.

Chapter 6
Result And Analysis

The Web-App is deployed the models on a local server and ran it using Flask , results of models is shown in below :-

6.1 Flow of Web

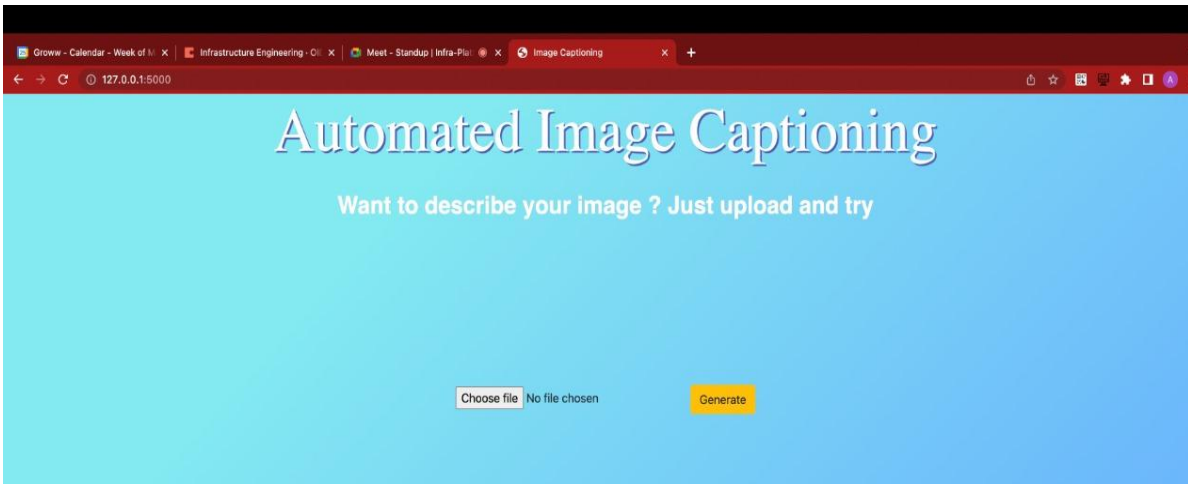


Fig. 6.1: Home Page

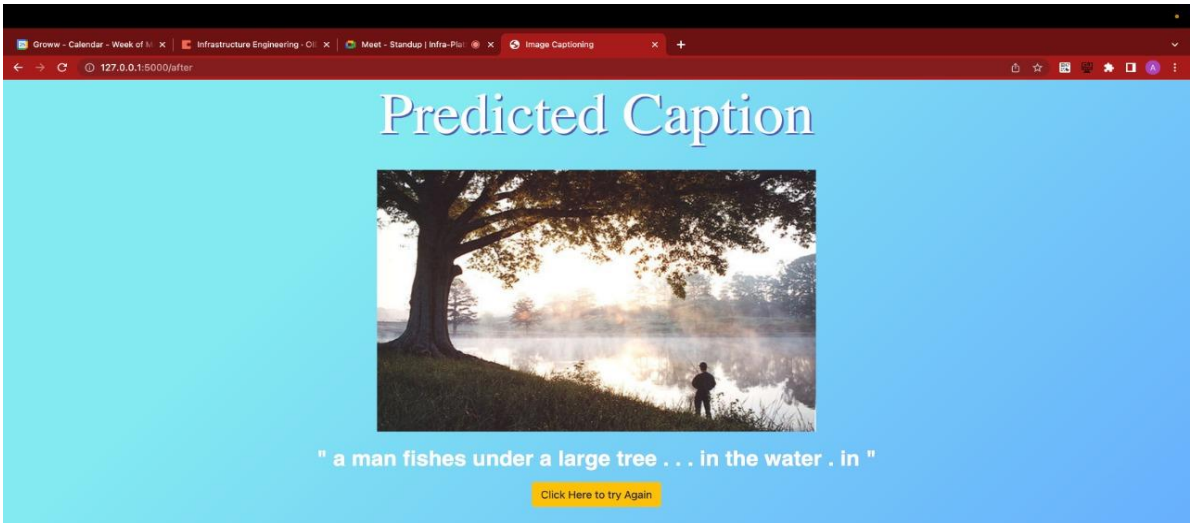
The generated sentence are shown in Fig 6.2 Generated sentences are “ several people are standing around in a fish fountain “ , while actual humans read as “ several people are standing around fish fountain and watching them “ .

Fig 6.2 shows image of captions generated for the image

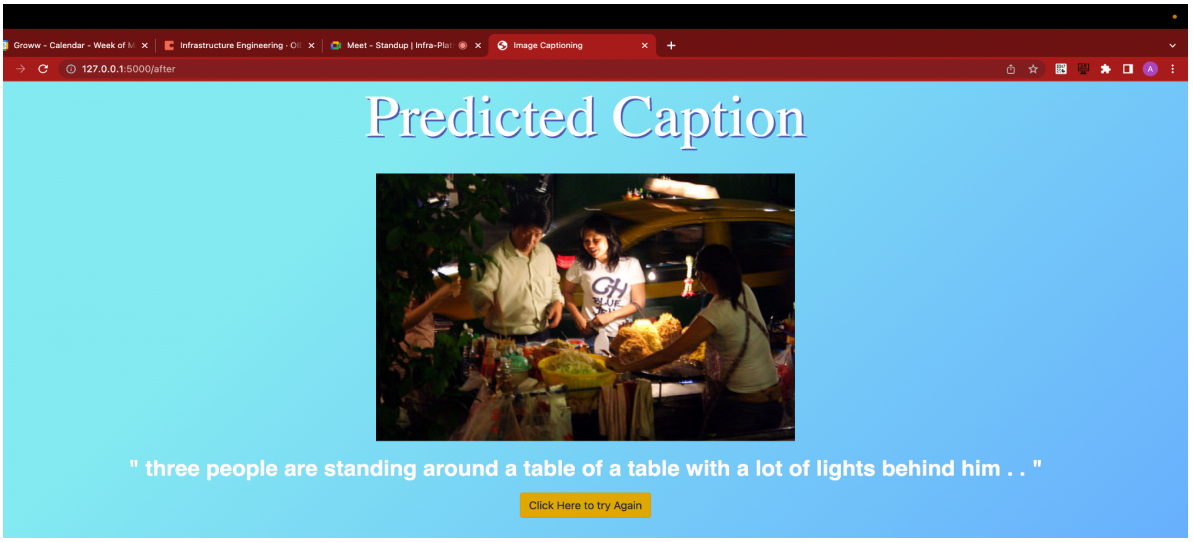
6.2 Caption Accuracy

We are dividing our captions into 3 categories of High, Medium and Low accuracy. Following are examples of each category:

High Accuracy Captions:

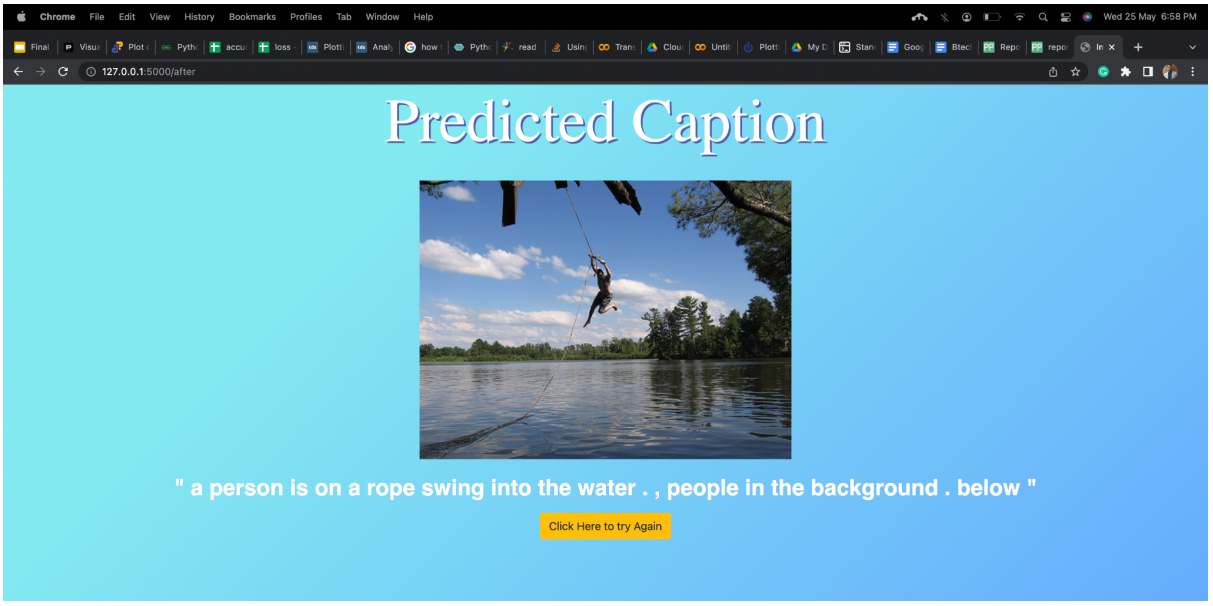


For this image, model generated caption “a man fishes under a large tree… in the water . in”. In this we can see that model captured the scenario of image accurately and selected appropriate words to give an idea. Only shortcoming is syntax of the sentence.

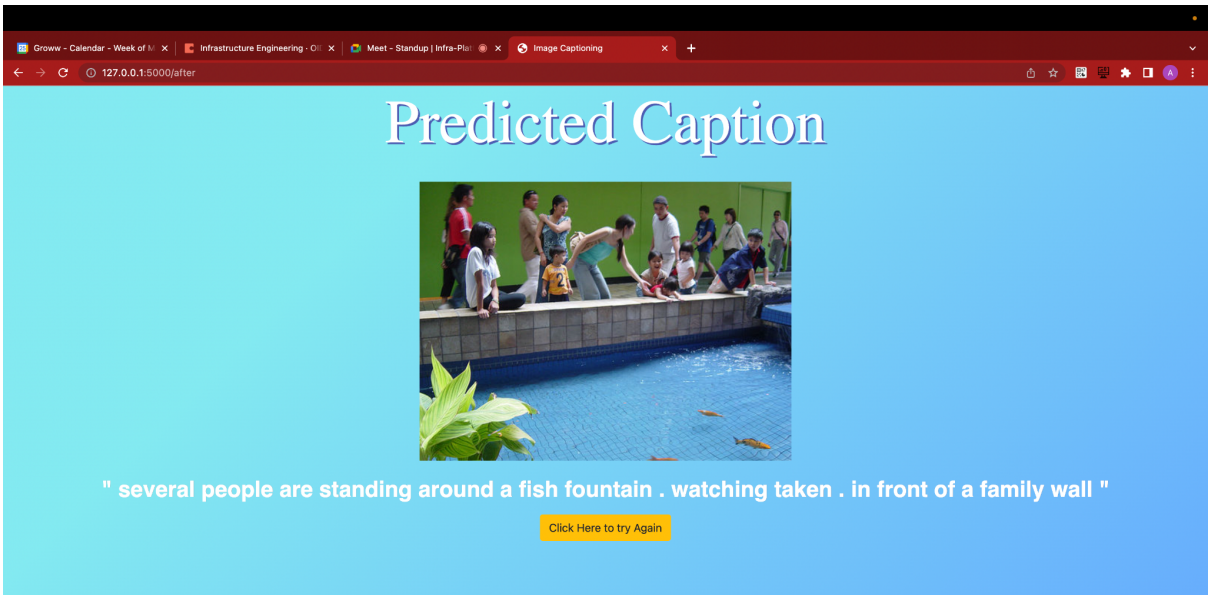


For this image, model generated caption “three people are standing around a table of a table with a lot of lights behind him . .”.
In this we can see that model captured the scenario of image accurately as it captures idea that 3 people are in the image and lighting is there with a table. Shortcoming is with the way, this sentence is arranged.

Medium Accuracy Captions:

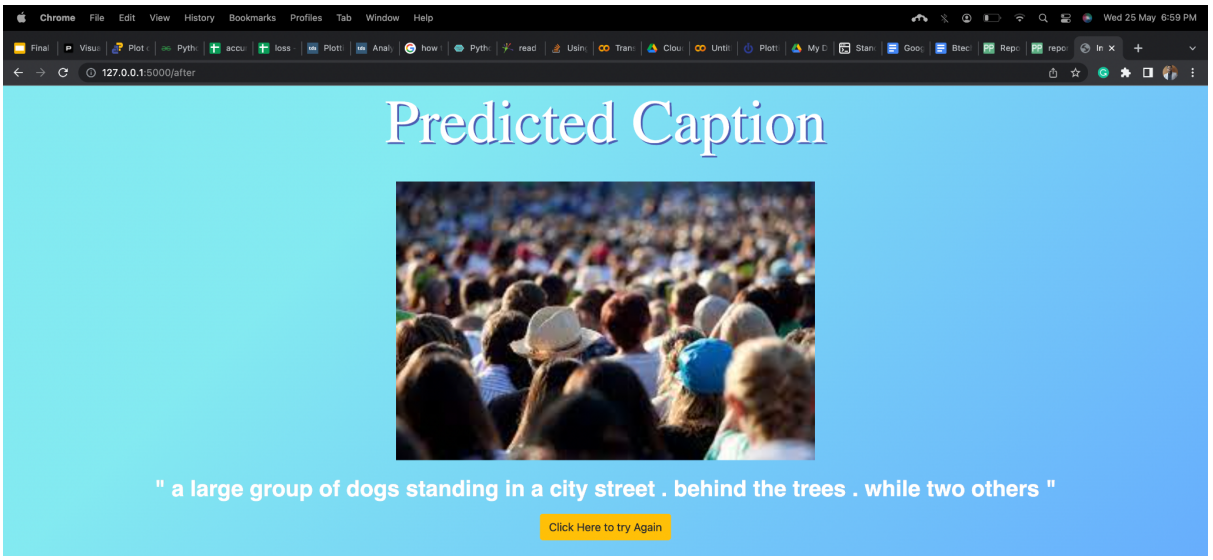


For this image, the model generated caption “a person is on a rope swing into the water . , people in the background . below”.
In this we see that model captured the sense of image correctly but added wrong captions in the sentence as well like “people in the background . below”. Also there are some syntactical mistakes as well.

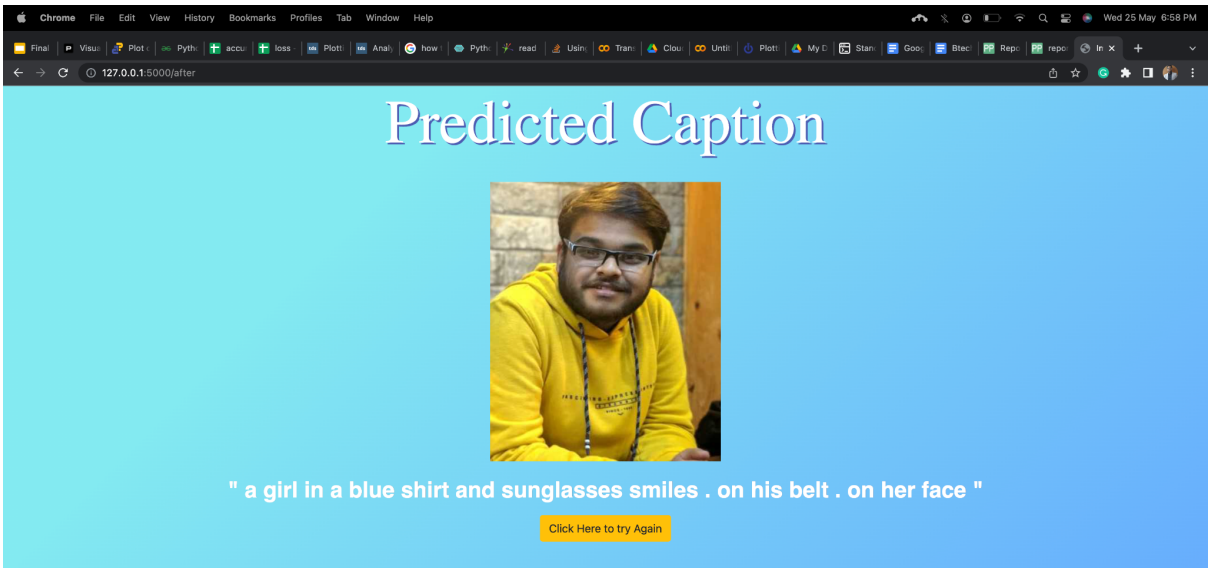


For this image, the model generated caption “several people are standing around a fish fountain . watching taken . in front of a family wall ”.
In this we see that model captured the sense of image correctly to some extent in manner that people are standing around a fish fountain but captured other parts of the caption incorrectly.

Low Accuracy Captions:



In this image, the model generated caption “a large group of dogs standing in a city street. behind the trees . while the others ”.
In this we see that the model is only able to sense some aspects of the image as keywords or phrases like “a large group” or “street” but has misjudged image to a large extent like judging people to be dogs and trees as well.

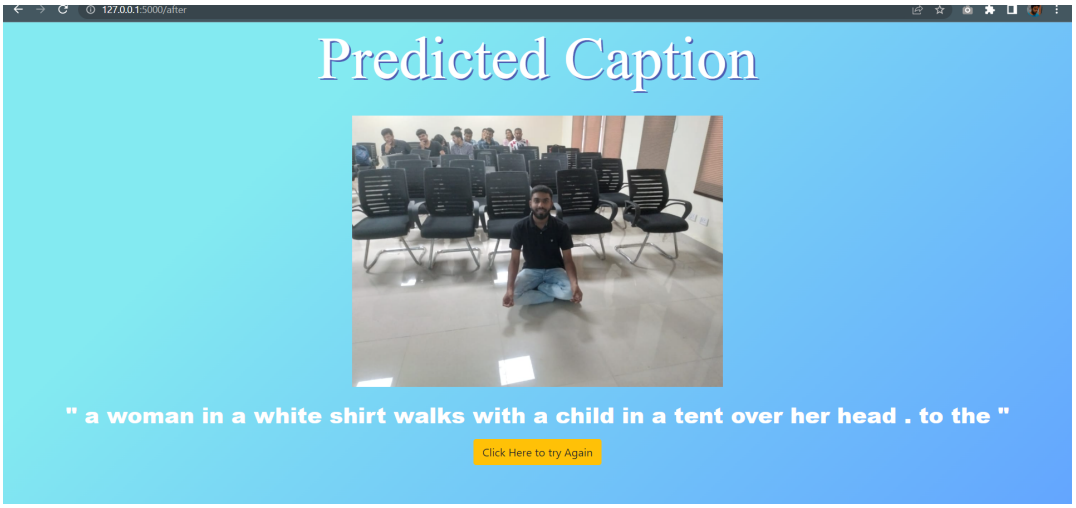


In this image, the model generated caption “a girl in a blue shirt and sunglasses smiles . on his belt . on her face ”.
In this we see that the model is hardly able to judge the image with only “sunglasses” and “face” close to what can be seen in the image. Rest everything highly differs from what is in the image.

Captioning on Real time Images **:-**

The above image shows a accurate Captioning to its extent.

here model shows an average accuracy as it guess the man and blue shirt right but other surroundings totally wrong.



In this case captioning is totally wrong as not a single word is related to image showing low accuracy.

6.3 Analysis

We trained our model on a dataset of around 8000 flickr images. Model training took about 19 hours. We were able to obtain an accuracy of about 65% for our model on training data.

Following figures and data show result of each epoch of training:

A	B	C
Epoch	Accuracy	Loss
1	0.2172	4.6687
2	0.3151	3.8704
3	0.3584	3.4754
4	0.3878	3.2353
5	0.4052	3.0925
6	0.4179	2.9907
7	0.4284	2.9172
8	0.4376	2.851
9	0.446	2.8005
10	0.4529	2.7534
11	0.4593	2.7121
12	0.4659	2.6738
13	0.4724	2.6301
14	0.4785	2.5883
15	0.4855	2.5469
16	0.4915	2.5093
17	0.4976	2.4693
18	0.5035	2.4274
19	0.5107	2.3876
20	0.5167	2.3436
21	0.5226	2.2999
22	0.5279	2.2559
23	0.5353	2.216
24	0.5408	2.1778
25	0.5463	2.1438
26	0.552	2.1091
27	0.5568	2.0793
28	0.5614	2.0449
29	0.566	2.0107
30	0.5705	1.9806
31	0.5758	1.9459
32	0.5799	1.9149
33	0.5846	1.882
34	0.5885	1.8583
35	0.5925	1.8273
36	0.596	1.8018
37	0.6017	1.7719
38	0.6054	1.744
39	0.61	1.7163
40	0.6136	1.6931
41	0.6176	1.6695
42	0.6212	1.6448
43	0.6252	1.6289
44	0.628	1.601
45	0.6317	1.5804
46	0.6343	1.5607
47	0.6384	1.5377
48	0.6422	1.5131
49	0.6458	1.492
50	0.6488	1.4742

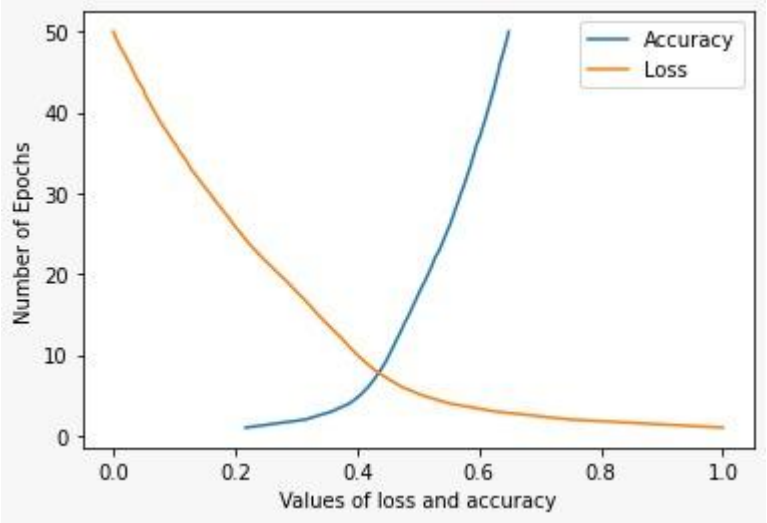
We obtained above figures from following model training output:

```
Epoch 1/50
938/938 [==============================] - 971s 1s/step - loss: 4.6687 - accuracy: 0.2172
Epoch 2/50
938/938 [==============================] - 964s 1s/step - loss: 3.8703 - accuracy: 0.3151
Epoch 3/50
938/938 [==============================] - 964s 1s/step - loss: 3.4754 - accuracy: 0.3584
Epoch 4/50
938/938 [==============================] - 958s 1s/step - loss: 3.2353 - accuracy: 0.3878
Epoch 5/50
938/938 [==============================] - 3696s 4s/step - loss: 3.0925 - accuracy: 0.4052
Epoch 6/50
938/938 [==============================] - 1487s 2s/step - loss: 2.9907 - accuracy: 0.4179
Epoch 7/50
938/938 [==============================] - 8225s 9s/step - loss: 2.9172 - accuracy: 0.4284
Epoch 8/50
938/938 [==============================] - 3503s 4s/step - loss: 2.8510 - accuracy: 0.4376
Epoch 9/50
938/938 [==============================] - 2597s 3s/step - loss: 2.8005 - accuracy: 0.4460
Epoch 10/50
938/938 [==============================] - 1705s 2s/step - loss: 2.7534 - accuracy: 0.4529
Epoch 11/50
938/938 [==============================] - 2530s 3s/step - loss: 2.7121 - accuracy: 0.4593
```

```
938/938 [==============================] - 2530s 3s/step - loss: 2.7121 - accuracy: 0.4593
Epoch 12/50
938/938 [==============================] - 2069s 2s/step - loss: 2.6738 - accuracy: 0.4659
Epoch 13/50
938/938 [==============================] - 1198s 1s/step - loss: 2.6301 - accuracy: 0.4724
Epoch 14/50
938/938 [==============================] - 2080s 2s/step - loss: 2.5883 - accuracy: 0.4785
Epoch 15/50
938/938 [==============================] - 970s 1s/step - loss: 2.5469 - accuracy: 0.4855
Epoch 16/50
938/938 [==============================] - 971s 1s/step - loss: 2.5093 - accuracy: 0.4915
Epoch 17/50
938/938 [==============================] - 970s 1s/step - loss: 2.4693 - accuracy: 0.4976
Epoch 18/50
938/938 [==============================] - 1783s 2s/step - loss: 2.4274 - accuracy: 0.5035
Epoch 19/50
938/938 [==============================] - 956s 1s/step - loss: 2.3876 - accuracy: 0.5107
Epoch 20/50
938/938 [==============================] - 953s 1s/step - loss: 2.3436 - accuracy: 0.5167
Epoch 21/50
938/938 [==============================] - 959s 1s/step - loss: 2.2999 - accuracy: 0.5226
Epoch 22/50
938/938 [==============================] - 968s 1s/step - loss: 2.2559 - accuracy: 0.5279
Epoch 23/50
938/938 [==============================] - 971s 1s/step - loss: 2.2160 - accuracy: 0.5353
Epoch 24/50
938/938 [==============================] - 1037s 1s/step - loss: 2.1778 - accuracy: 0.5408
Epoch 25/50
938/938 [==============================] - 957s 1s/step - loss: 2.1438 - accuracy: 0.5463
Epoch 26/50
938/938 [==============================] - 983s 1s/step - loss: 2.1091 - accuracy: 0.5520
Epoch 27/50
938/938 [==============================] - 977s 1s/step - loss: 2.0793 - accuracy: 0.5568
Epoch 28/50
938/938 [==============================] - 970s 1s/step - loss: 2.0449 - accuracy: 0.5614
Epoch 29/50
938/938 [==============================] - 991s 1s/step - loss: 2.0107 - accuracy: 0.5660
Epoch 30/50
938/938 [==============================] - 992s 1s/step - loss: 1.9806 - accuracy: 0.5705
Epoch 31/50
938/938 [==============================] - 987s 1s/step - loss: 1.9459 - accuracy: 0.5758
```

```
image-captioning-keras-resnet — python3 < python3 app.py — 132×41
938/938 [==============================] - 987s 1s/step - loss: 1.9459 - accuracy: 0.5758
Epoch 32/50
938/938 [==============================] - 1000s 1s/step - loss: 1.9149 - accuracy: 0.5799
Epoch 33/50
938/938 [==============================] - 990s 1s/step - loss: 1.8820 - accuracy: 0.5846
Epoch 34/50
938/938 [==============================] - 974s 1s/step - loss: 1.8583 - accuracy: 0.5885
Epoch 35/50
938/938 [==============================] - 974s 1s/step - loss: 1.8273 - accuracy: 0.5925
Epoch 36/50
938/938 [==============================] - 977s 1s/step - loss: 1.8018 - accuracy: 0.5960
Epoch 37/50
938/938 [==============================] - 979s 1s/step - loss: 1.7719 - accuracy: 0.6017
Epoch 38/50
938/938 [==============================] - 973s 1s/step - loss: 1.7440 - accuracy: 0.6054
Epoch 39/50
938/938 [==============================] - 982s 1s/step - loss: 1.7163 - accuracy: 0.6100
Epoch 40/50
938/938 [==============================] - 975s 1s/step - loss: 1.6931 - accuracy: 0.6136
Epoch 41/50
938/938 [==============================] - 968s 1s/step - loss: 1.6695 - accuracy: 0.6176
Epoch 42/50
938/938 [==============================] - 961s 1s/step - loss: 1.6488 - accuracy: 0.6212
Epoch 43/50
938/938 [==============================] - 958s 1s/step - loss: 1.6289 - accuracy: 0.6252
Epoch 44/50
938/938 [==============================] - 959s 1s/step - loss: 1.6010 - accuracy: 0.6280
Epoch 45/50
938/938 [==============================] - 962s 1s/step - loss: 1.5804 - accuracy: 0.6317
Epoch 46/50
938/938 [==============================] - 986s 1s/step - loss: 1.5607 - accuracy: 0.6343
Epoch 47/50
938/938 [==============================] - 979s 1s/step - loss: 1.5377 - accuracy: 0.6384
Epoch 48/50
938/938 [==============================] - 972s 1s/step - loss: 1.5131 - accuracy: 0.6422
Epoch 49/50
938/938 [==============================] - 967s 1s/step - loss: 1.4920 - accuracy: 0.6458
Epoch 50/50
938/938 [==============================] - 974s 1s/step - loss: 1.4742 - accuracy: 0.6488
(ansh) (base) ansh.lehri@BGVMC414 data % ls
```

Graphical conclusion of accuracy and loss:



Along Y-axis we have Number of epochs.
Along X-axis we have values for loss and accuracy. As we can see range of loss values is from [1,5) and of accuracy is of [0,1), we have standardised the value points for loss in range to [0,1] using Normalization.

From the graph, we can see that initially accuracy increased exponentially but as epochs increased rate of increase in accuracy decreased and ended at about 64.8%. Whereas loss valued decreased with increase in number of epochs.

6.4 Limitations:

We divide our shortcomings and limitations into 3 categories:

- Hardware Limitations
- Dataset Limitations
- Implementation Limitations

Hardware Limitations:

- Machine used has limited GPUs.
- Machine used to train the model has 16 GB RAM. To train bigger and better datasets, a machine with higher RAM is required with hisg computing capacity.

Dataset Limitations:

- The only dataset available was of Flickr.
- Flickr dataset had less variety in terms of nature of images. Most of images were of animals, scenic beauty, sports etc.
- Images present are only high resolution.

Implementation Limitations:

- Due to machine constraints, we could not try hyper-parameter tuning to a large extent.
- Different architectures could not run on the machine.

- BLEU to gauge exactness couldn't be carried out.
- Because of the limited dataset, vocabulary generated for the model was very constrained. As it had not many words to choose from, this resulted in many bad captions.

These limitations can be overcome by following measures:

- Utilizing strong machines with high RAM and registering limit.
- Using variety of datasets which can have images with high to low resolution, nature of images can be different as well.
- Applying hyper-parameter tuning and using different architectures.
- Using word embeddings to increase our dictionary.
- Using large datasets to generate bigger vocabulary dictionary.

Conclusion And Future Scopes

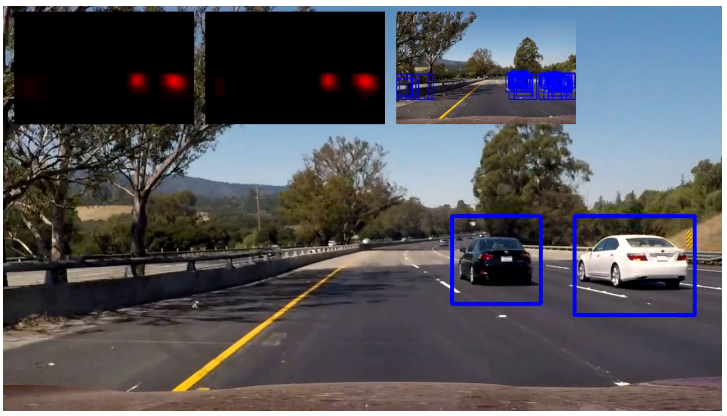
Image captioning is nonetheless a creating subject and many researches are nonetheless in progress. Recent work primarily based on deep studying methods has resulted in a leap forward in the accuracy of photograph captioning as it have breakdown the complicated fashions to easy structure. The textual content description of the picture can enhance the content-based photograph retrieval efficiency, the increasing utility scope of visible grasp in the fields of medicine, security, navy and different fields, which has a vast utility prospect. At the equal time, the hypothetical system and query procedures of photo inscribing can advance the improvement of the thought and utility of photograph comment and apparent question addressing (VQA), go media recovery, video subtitling and video exchange, which has vital instructive and reasonable programming esteem.

7.1 Use case and future Scopes

- Medical Application
  This model can be incorporated with a couple of sunglasses,cameras and listening devices, to assist the outwardly debilitated individual with getting the information on their environmental elements. One of the instances of this application is Horus Technology which in association with NVIDIA are chipping away at a similar task which is still in the improvement stage at the present time.

This image is an example of Horus Technology[20]

- Intelligent monitoring permits the laptop to perceive and decide the behaviour of humans or automobiles in the captured scene and generate alarms beneath fantastic prerequisites to immediate the person to react to emergencies and forestall useless accidents.



One of the suc technologies are developing phase where a palm sized device is connected to camera , giving a view of road alerting and helping driver to reduce the accidents . This technology is being developed by Intel IIIT Hyderabad and CPRI[21] .

- Campus Level Implementation
  This model can be integrated with cameras and alert systems such as messaging,sirens etc. to detect any absurd activities .

  In the above, image cameras are detecting on-going robbery.
- Social Media, Platforms like facebook can surmise straightforwardly from the picture, where you are (ocean side, bistro and so forth), what you wear (variety) and all the more significantly the thing you're doing likewise (as it were) . This permits them to elevate promotions to the specific client of their advantage .

References

[1] Shuang Liu, Liang Bai,a, Yanli Hu and Haoran Wang. I*mage Captioning Based on Deep Neural Networks*. EITCE (2018)
Available : Link

[2] Lakshminarasimhan Srinivasan, Dinesh Sreekanthan,Amutha A.L. I2T: *Image Captioning - A Deep Learning Approach* (2018).
Available : Link

[3] Simao Herdade, Armin Kappeler, Kofi Boakye, Joao Soares.*Image Captioning: Transforming Objects into Words* . San Francisco, CA (2019).
Available : Link

[4] Aishwarya Maroju ,Sneha Sri Doma ,Lahari Chandarlapati , *Image Caption Generating Deep Learning Model ,J.N.T.U, Hyderabad , Sreenidhi Institute of Science And Technology* (2021).
Available : Link

[5] Zhongliang Yang, Yu-Jin Zhang, Sadaqat ur Rehman, Yongfeng Huang: *Image Captioning with Object Detection and Localization, Department of Electronic Engineering, Tsinghua University*, *Beijing 100084, China*
Available : Link

[6] Oriol Vinyals , Alexander Toshev , Samy Bengio Dumitu Erhan(2014). *Show and Tell : A Neural Image Caption Generator . Google*
Available : Link

[7] Andrej Karpathy Li Fei Fei(2015). *Deep Visual-Semantic Alignments for Generating Image Descriptions. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 3128-3137.*
Available : Link

[8] Elliott, D., & Keller, F. (2013). I*mage Description using Visual Dependency Representations. EMNLP.*

[9] Farhadi, A., Hejrati, M., Sadeghi, M.A., Young, P., Rashtchian, C., Hockenmaier, J., & Forsyth, D.A. (2010). *Every Picture Tells a Story: Generating Sentences from Images. ECCV.*

[10] Fang, H., Gupta, S., Iandola, F.N., Srivastava, R.K., Deng, L., Dollár, P., Gao, J., He, X., Mitchell, M., Platt, J.C., Zitnick, C.L., & Zweig, G. (2015). *From captions to visual concepts and back. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1473-1482.*

[11] Kuznetsova, P., Ordonez, V., Berg, A.C., Berg, T.L., & Choi, Y. (2012). *Collective Generation of Natural Image Descriptions. ACL.*

[12] Lebret, Rémi et al. *"Simple Image Description Generator via a Linear PhraseBased Approach." CoRR abs/1412.8419* (2014): n. pag.

[13] Li, S., Kulkarni, G., Berg, T.L., Berg, A.C., & Choi, Y. (2011). *Composing Simple Image Descriptions using Web-scale N-grams. CoNLL.*

[14] Chen, X., & Zitnick, C.L. (2015). Mind's eye: *A recurrent visual representation 49 for image caption generation. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2422-2431.*

[15] Donahue, Jeff et al. *"Long-Term Recurrent Convolutional Networks for Visual Recognition and Description." 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015): 2625-2634.*

[16] Flask : Link

[17] Kaggle : Link

[18] Spyder : Link

[19] VsCode : Link

[20] Horus Technology Link

[21]AI Powered Road System : Link