i-Sight

Minor Project Report

DECEMBER 2020

PRASOON MISH**MEMBERS** 18103077
SANNIDHYA SINGHAL 18103081
YASH AGARWAL 18103299

ABSTRACT

In this project, we systematically analyze a deep neural networks based image caption generation method. With an image as the input, the method can output an English sentence describing and speaking out the **scene** in the image.

We analyze three components of the method:

- Convolutional Neural Network (CNN)
- Recurrent Neural Network (RNN)
- Sentence generation and audio output

By replacing the CNN part with three state-of-the-art architectures, we find the **VGGNet** performs best according to the BLEU score. The simplified GRU achieves a comparable result when it is compared with the long short-term memory (LSTM) method. But it has few parameters which saves memory and is faster in training.

The experiments show that the modified method can generate captions comparable to the-state-of-the-art methods with less training memory.

CHAPTER 1
INTRODUCTION

1.1. Problem Introduction

1.1.1. Motivation

We want blind people to be more aware of their surroundings even when they are all by themselves. This software will be their guide at every place. Telling them the description of their surroundings.

1.1.2. Project Objective

i-Sight's scope is to provide an image to speech translation for visually impaired people.

1.1.3. Scope of the Project

The purpose of i-Sight is to ease access and to create a convenient and easy-to-use application for sightless people by capturing an image and reciting the scene description. The app is highly reliable as it's trained on more than 20,000 images. Above all, the user experience is specially designed for blind people, supporting many gestures.

1.2. Organization of the Report

CHAPTER 1: INTRODUCTION
CHAPTER 2: SOFTWARE REQUIREMENTS SPECIFICATION
CHAPTER 3: SYSTEM DESIGN AND METHODOLOGY
CHAPTER 4: IMPLEMENTATION AND RESULTS
CHAPTER 5: CONCLUSION

CHAPTER 2

SOFTWARE REQUIREMENT SPECIFICATION

This application takes a picture using a device camera and analyses the scene description and reads it out to the person.

2.1. Product Perspective

i-Sight is a new, self-contained product that increases the independence for the blind and the visually impaired.

It first takes a picture, then divides it into pixels. The cluster of pixels join together to identify the objects. The objects are then converted into an understandable sentence (based on their position) that identifies the theme. The sentence is then translated into speech. This all happens so quickly that the person gets the output immediately.
The major components are image identification and text to speech translation.

2.2. Interface

2.2.1 System Interface

Interface mainly depends on the hardware devices such as the camera and speaker. The camera is to be a part of the external device (Eg. White cane, smart glasses) to be connected with the help of Raspberry Pi.

2.2.2 User Interface

The interface is specially designed for blind people. Users can use multiple gestures to do various tasks.

1. The user is prompted.
2. The device's camera takes a picture (Giving haptic feedback).
3. The user hears the generated speech

However, settings would be available, which will be accessible by the user's helper.

2.2.3 Hardware Interface

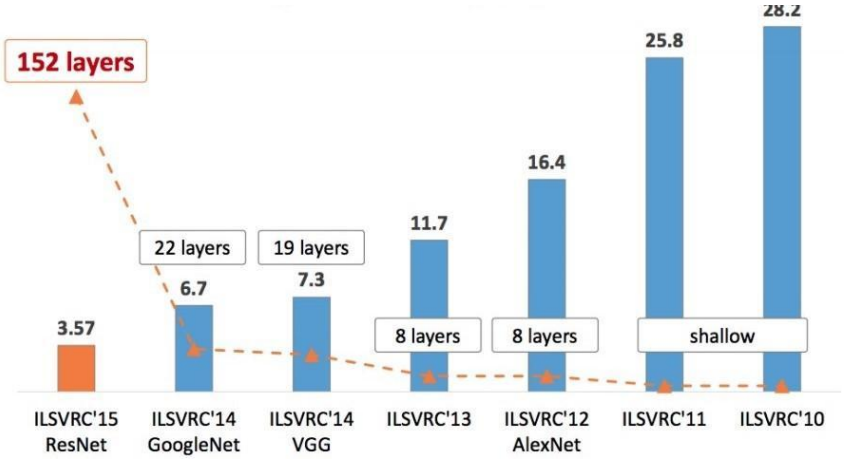
The hardware interface is met by most of the devices.
The hardware devices used:

1. **Camera** (To capture the pictures)
2. **Speaker** (To give the speech output)
3. **ERM motor** (To provide haptic feedback) **Optional**

2.2.4 Software Interface

This software uses:

1. **GloVe | CountVectorizer**
   NLP tools
2. **Python**
   Machine learning
3. **Flask**
   Backend
4. **HTML | CSS | Javascript**
   Frontend
5. **ResNet**
   Neural network
6. **CNN | LSTM**
   Recurrent neural network



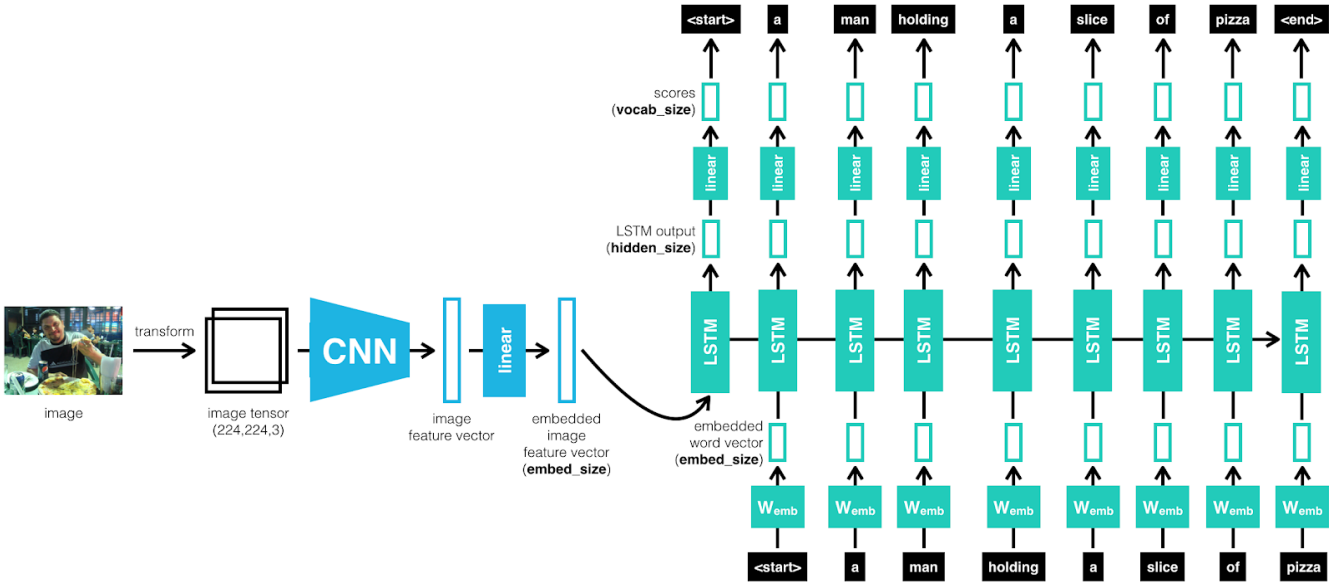


Fig 1) Working of our model

2.2.5 Communications Interface

This software uses the **AWS (Amazon Web Server)** which makes it accessible from anywhere in the world. Can be accessed from any web browser.

The communication standard used is HTTP.

The data transfer rates depend on the user's net connectivity speed.

The data is synchronized with every user activity.

2.2.6 Memory Constraints

There are no specific memory requirements. The web server can be accessed by not more than 3-4 users. Depends on the device's RAM. If all the customer's machines have only 128K of RAM, then your target design has to come in under 128K so there is an actual requirement. We have determined that our target has between 256-512M of RAM, therefore the design footprint should not exceed 256M.

2.2.7 Operations

Major operations of the product and a brief description of the product functions is as follows:

1. **Object detection:**
   Firstly the product identifies the objects which are there in the scene.
2. **Detection of the scene:**
   The product will then detect the things happening in the scene.
3. **Integrating the scene to text:**
   Then our product will integrate the object identified and things detected into a text.
4. **Converting text to speech:**
   The product will then convert the text into a speech.

2.3. Site Adaptation Requirements

1. **Portability:**
   The UI should be able to adapt to various screen sizes. Ensuring the gestures are working properly at the same time.
2. **Robustness:**
   The software is supposed to handle errors efficiently and jump to the home screen on critical errors.
3. **Reusability:**
   The software might be made reusable to support different operations for specially abled people.
4. **Availability:**
   If the internet connectivity gets disrupted while sending/receiving information to the server, the information should be sent again.

2.4. User Characteristics

Users of the system should be able to retrieve information of the scene.
The user should be able to do the following functions:

1. Capture image
2. Listen to speech output of the last description

2.4 Constraints

Since it is a web application, a constraint for it will be moderate loading speed, but, on the other hand this makes the web app to be accessible from anywhere. It might not support some unpopular web browsers. There will be no issues for memory management (data is stored on servers, not the device). Only English language is supported in the early builds. The web provider will be responsible for maintaining the delivered software.

2.5 Assumptions and Dependencies

The device is supposed to be used as an additional accessory to be attached with a white cane or a helmet. It is assumed that every user is having a smartphone and some accessory for personal use. This application depends on the camera and the speakers of the device.

2.6 Apportioning of Requirements

Identify requirements that may be delayed until future versions of the system. After you look at the project plan and hours available, you may realize that you just cannot get everything done. This section divides the requirements into different sections for development and delivery. Remember to check with the customer – they should prioritize the requirements and decide what does and does not get done. This can also be useful if you are using an iterative life cycle model to specify which requirements will map to which interaction.

CHAPTER 3
IMPLEMENTATION AND RESULTS

3.1 Image capturing

3.1.1 Description and Priority

This feature will capture the image taken by the camera.
(High priority)

3.1.2 Stimulus/Response Sequences

When the user swipes on the screen, this feature will open the camera and take a picture.

3.1.3 Functional Requirements

Requires the device's camera.

3.2 Image captioning

3.2.1 Description and Priority

This feature will caption the image taken by the camera.
(Medium priority)

3.2.2 Stimulus/Response Sequences

No user response required. The system response is generated when the image is captured by the user.

3.3 Text to speech translation

3.3.1 Description and Priority

This feature speaks out the generated caption.
(Low priority)

3.3.2 Stimulus/Response Sequences

When the user taps on the screen, this feature will speak out the caption.

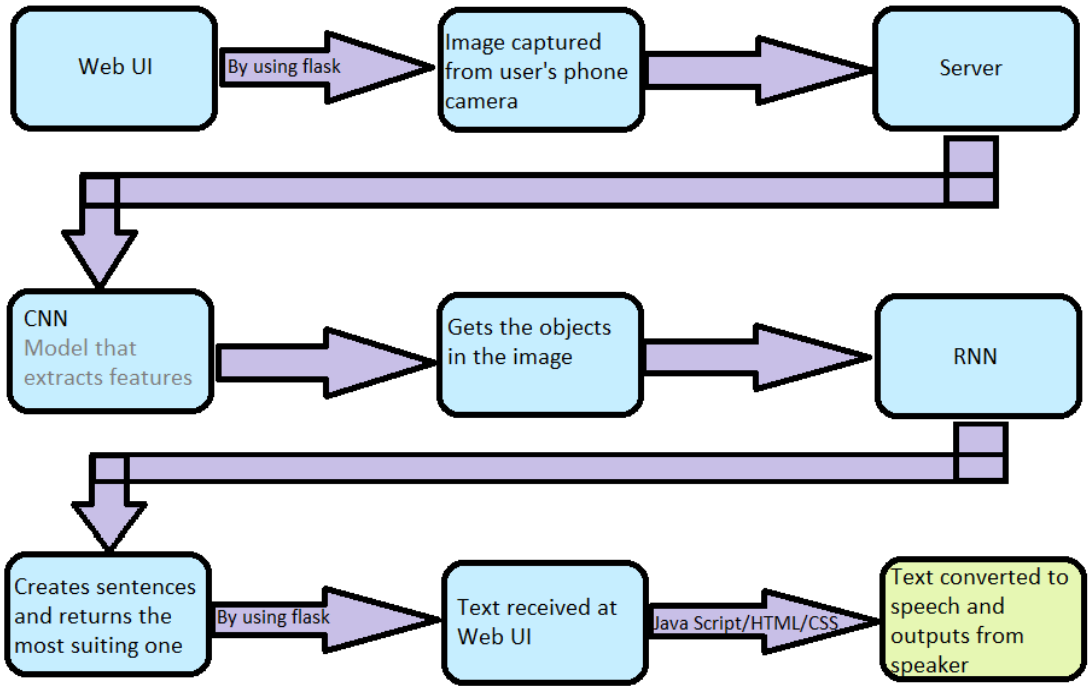


Fig 2) The Process

3.4 Performance Requirements

1. **Response time:**
   The response time is around 2 seconds after the image is uploaded to the server. The upload time is based on the user's net connectivity and the size of the image (200KB < Recommended size < 1 MB). Large sized files (>1 MB) often increase the image processing time by around 3 seconds.

2. **Performance intention:**
   The design choices are made to support the user by providing least buttons and maximum gesture control. The users need quick responses to save time.

3.5 Security Requirements

This product requires access to:

1. System files to store data
2. Device's camera to upload pictures to identify
3. Device's speakers to output the speech

The images taken will be made to be saved in our dataset to increase correctness of the caption based on the user's preferences.

3.6 Assumptions and dependencies

The device is supposed to be used as an additional accessory to be attached with a white cane or a helmet. It is assumed that every user is having a smartphone and some accessory for personal use. This application depends on the camera and the speakers of the device.

3.7 Constraints

Since it is a web application, a constraint for it will be limited users at a time, but, on the other hand this makes the web app to be accessible from anywhere. It might not support some unpopular web browsers. There will be no issues for memory management (data is stored on the amazon web servers, not the device). Only English language is supported in the early builds. The web provider will be responsible for maintaining the delivered software.

3.8 Implementation Details

RNN:

Recurrent Neural Networks suffer from short-term memory. If a sequence is long enough, they'll have a hard time carrying information from earlier time steps to later ones. So if we are trying to process a paragraph of text to do predictions, RNN's may leave out important information from the beginning. This limitation of RNN was one of the primary reasons why we opted for LSTM for making our models.

In [41]:
```
model = Model(inputs=[input_img_features,input_captions],outputs=outputs)
model.summary()
```

```
Model: "model_2"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_4 (InputLayer)            (None, 35)           0
__________________________________________________________________________________________________
input_3 (InputLayer)            (None, 2048)         0
__________________________________________________________________________________________________
embedding_2 (Embedding)         (None, 35, 300)      1536300     input_4[0][0]
__________________________________________________________________________________________________
dropout_3 (Dropout)             (None, 2048)         0           input_3[0][0]
__________________________________________________________________________________________________
dropout_4 (Dropout)             (None, 35, 300)      0           embedding_2[0][0]
__________________________________________________________________________________________________
dense_4 (Dense)                 (None, 256)          524544      dropout_3[0][0]
__________________________________________________________________________________________________
lstm_2 (LSTM)                   (None, 256)          570368      dropout_4[0][0]
__________________________________________________________________________________________________
add_2 (Add)                     (None, 256)          0           dense_4[0][0]
                                                                 lstm_2[0][0]
__________________________________________________________________________________________________
dense_5 (Dense)                 (None, 256)          65792       add_2[0][0]
__________________________________________________________________________________________________
dense_6 (Dense)                 (None, 5121)         1316097     dense_5[0][0]
==================================================================================================
Total params: 4,013,101
Trainable params: 4,013,101
Non-trainable params: 0
```

LSTM was created as the solution to short-term memory. They have internal mechanisms called gates that can regulate the flow of information. 8 These gates can learn which data in a sequence is important to keep or throw away. By doing that, it can pass relevant information down the long chain of sequences to make predictions. Almost all state of the art results based on recurrent neural networks are achieved with LSTM . The value of cryptocurrencies is a sequence of data and each value of each day is dependent on a history of previous values. This is the reason why we need to carry the information from previous days so that the model makes the correct pattern recognition to make a good model.

Web Server:

We had to deploy the project on the amazon web server so that other users can access the programs internally without having to know how to execute the scripts. For this we had to use the web server.

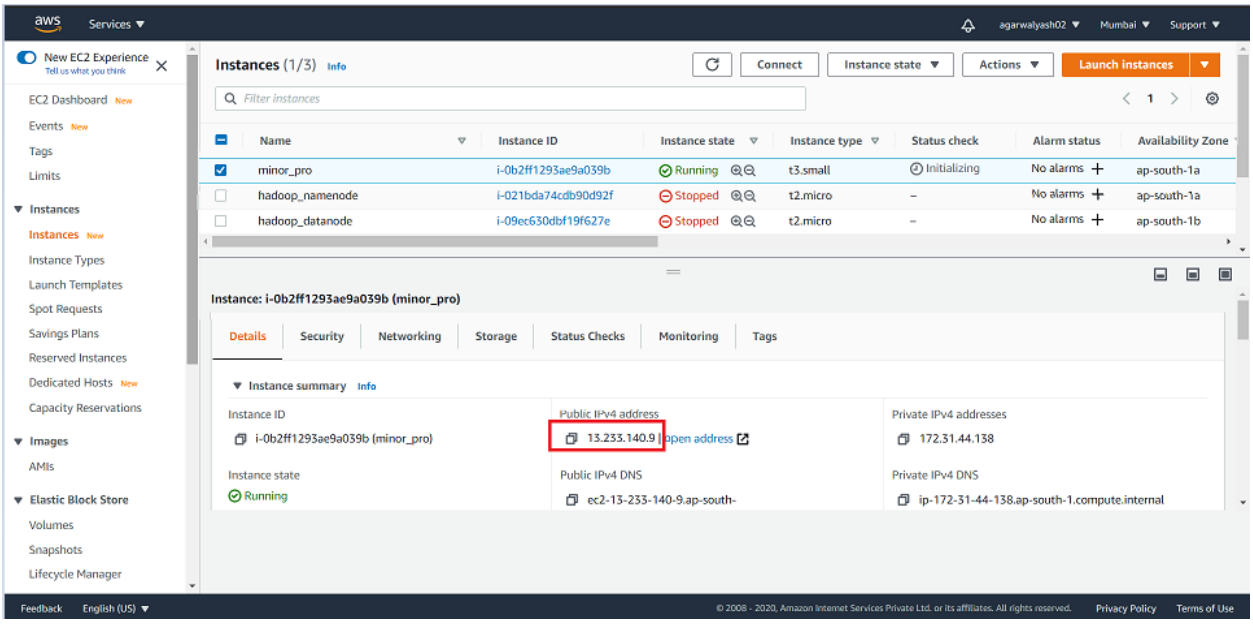
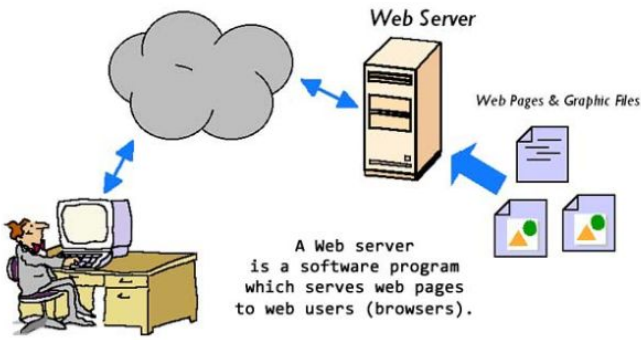


Fig 3) Snapshot of the Amazon Web Server used in our project

A web server is software and hardware that uses HTTP (Hypertext Transfer Protocol) and other protocols to respond to client requests made over the World Wide Web. The main job of a web server is to display website content through storing, processing and delivering web pages to users. For us the task was to



allow running of python scripts over HTTP protocol. We had to learn and understand the use of IP addresses and ports while using the web server. For any task to be performed over the network there are 2 primary things that are necessary. Firstly a protocol and an IP address. Both these things allow any system to connect to any other system on the network

Security:

While we are dealing with the local system we usually don't care about the security since no other user tries to login in the system. But when we set up the webserver we are allowing the clients or users to access the server file system we have to consider the security and the access controls too. We used the RHEL8 as the baseOS on AWS to deploy the web server so we had the control of the two levels of security. Firstly, AWS provides its own firewall which is called the security groups . We can specify the type of protocols we are going to allow on the system. We can enable the ICMP, HTTP, TCP/SSH protocol for the system. We also have the power to specify the IP address which would be allowed to access the server in AWS. 9 The web server listens at port number 80 so that ports need to be allowed too. IP address and port number The IP address and the port number together allows the clients to successfully run the scripts or perform any kind of action on the web server. The IP address allows us to reach a certain location on the network and the port number is attached to each program. So when a client comes to any certain IP with a port number. If it is allowed by the firewall it can execute a program specified to a certain port number. ICMP : The Internet Control Message Protocol is a supporting protocol in the Internet protocol suite. It is used by network devices, including routers, to send error messages and operational information. It allows the use of ping to the system this allows us to check the connectivity with the client. SSH: SSH is one of the most popular protocols in the UNIX systems.SSH or Secure Shell is a cryptographic network protocol for operating network services securely over an unsecured network. Enabling this allows us to remotely access the EC2 instance. Apart from these security the linux provides the SELinux (Security Enhanced Linux) on it's OS. Each file and folder has a certain group ID and user ID attached to it. When the user that wants to access a certain file or folder then it has to match are allowed to access them. This is called chroot jail. A chroot jail is a way to isolate a process and its children from the rest of the system. It should only be used for processes that don't run as root, as root users can break out of the jail very easily. Ownership of the files and folders also plays a vital role in security. There are only two types of user that have all the control over the files and folders. Firstly the root user which exercises all the power in the system and the other being the owner of the user. Common Gateway Interface The Common Gateway Interface (CGI) is described as a set of rules for exchanging information between a web server and a custom script. CGI is one of the most common ways for web servers to interact with users by sending the data. 10 We did not use any

pre created framework for our work as we wanted to understand the intricate details of the working of the web server. We were successful in doing so because most of the frameworks use the CGI in the background but coding everything from our end might have made the working a little more challenging but it helped us understand all the concepts. The web server when accessing a script first needs to understand how to execute the script. For this it confirms in two ways. Firstly it checks the extension of the file. For example all the python scripts end with .py extension , secondly we have to specify the header in the python script so that the web server knows how to interpret the files.

Preprocessing:

Because we want to keep the architecture of the CNN, the input images are randomly cropped to the size of 224 × 224. As a result, only part of the images are used in training at particular iteration. Because one image will be cropped multiple times in the training, the CNN can probably see the whole image in the training (once for part of the image). However, the method only sees part of the image in the testing except the dense cropping is also used (our project does not use dense crop). For the sentences, the method first creates a vocabulary only from the training captions and removes lower frequency words (less than 5). Then, words are represented by one-hot vectors.

In [37]:
```
from keras.models import Model, load_model
from keras.layers import Input,Dropout,Embedding,LSTM,Dense
from keras.layers.merge import add
```

In [38]:
```
input_img_features = Input(shape=(2048,))
inp_img1 = Dropout(0.2)(input_img_features)
inp_img2 = Dense(256,activation='relu')(inp_img1)
```

In [39]:
```
input_captions = Input(shape=(max_len,))
inp_cap1 = Embedding(input_dim=vocab_size,output_dim=emb_dim,mask_zero=True)(input_captions)
inp_cap2 = Dropout(0.2)(inp_cap1)
inp_cap3 = LSTM(256)(inp_cap2)
```

In [40]:
```
decoder1 = add([inp_img2,inp_cap3])
decoder2 = Dense(256,activation='relu')(decoder1)
outputs = Dense(vocab_size,activation='softmax')(decoder2)
```

In [24]:
```
def removing_waste(sentence):
    sentence = str(sentence) #i have to add this as at some point error is coming not able to lower float value so converting al
    sentence = sentence.lower()
    sentence = re.sub("[^a-z]+"," ",sentence)
    sentence = sentence.split()
    sentence = [s for s in sentence if len(s)>1]
    sentence = " ".join(sentence)
    return sentence
```

In [25]:
```
for key,cap in descriptions.items():
    for i in range(len(cap)):
        cap[i] = removing_waste(cap[i])
```

In [26]:
```
descriptions['1000092795.jpg']
```

Out[26]:
```
['two young guys with shaggy hair look at their hands while hanging out in the yard',
 'two young white males are outside near many bushes',
 'two men in green shirts are standing in yard',
 'man in blue shirt standing in garden',
 'two friends enjoy time spent together']
```

3.9 Snapshots Of Interface

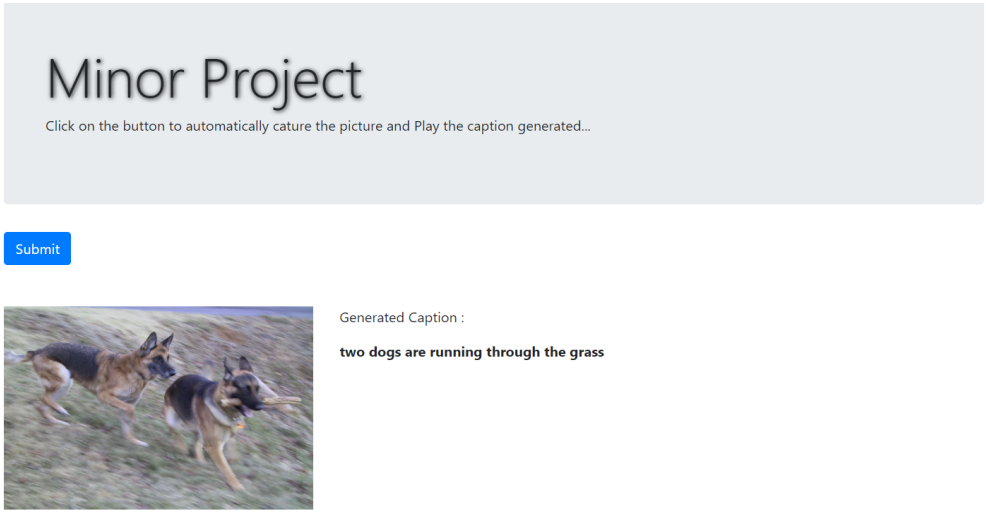


Fig 4) Image captured in neighbourhood park

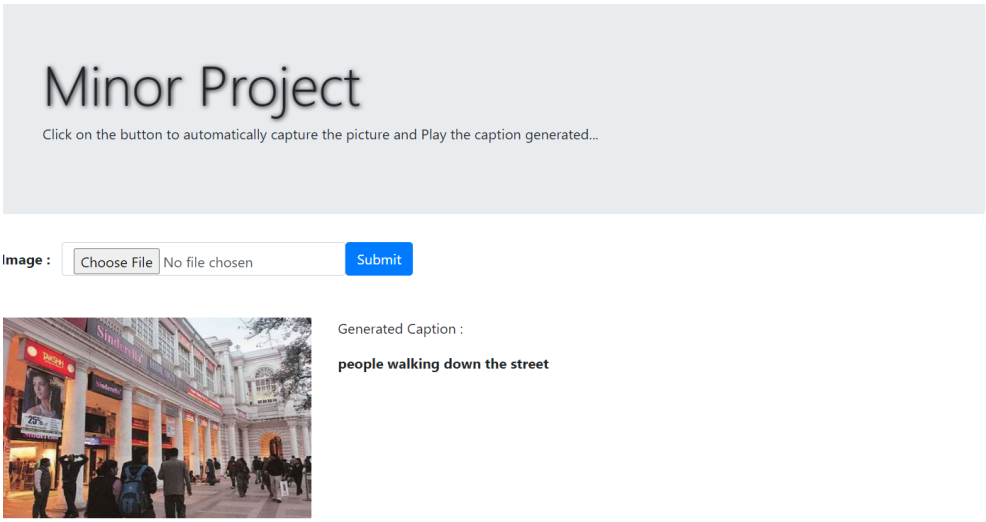


Fig 5) Image captured in Connaught place, Delhi

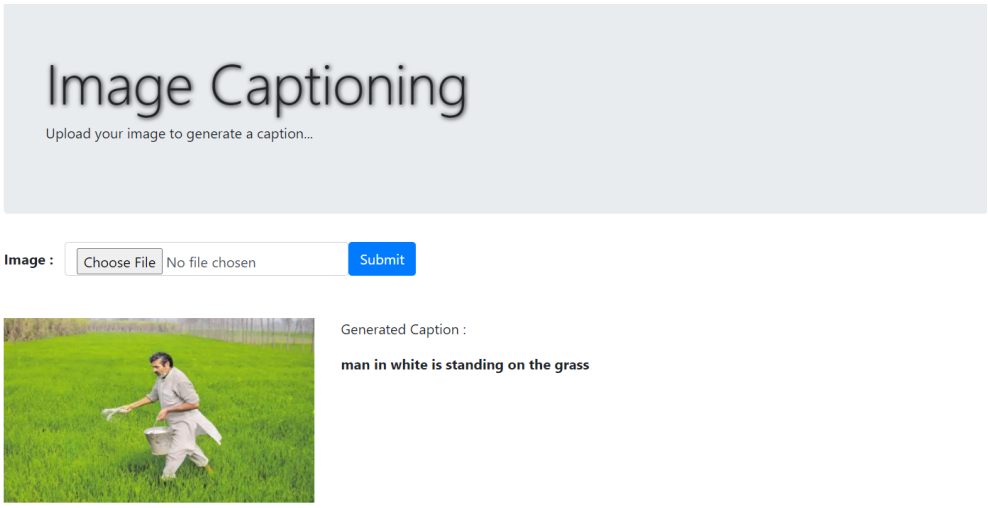


Fig 6) Image captured in a village farm

3.10 Test Cases

A number of datasets are used for training, testing, and evaluation of the image captioning methods. The datasets differ in various perspectives such as the number of images, the number of captions per image, format of the captions, and image size.
Two datasets: Flickr 30k and Flickr 8k are popularly used.

Flickr 30K: Dataset is a dataset for automatic image description and grounded language understanding. It contains 30k images collected from Flickr with 158k captions provided by human annotators. It does not provide any fixed split of images for training, testing, and validation. Researchers can choose their own choice of numbers for training, testing, and validation. The dataset also contains detectors for common objects, a color classifier, and a bias towards selecting larger objects.

Flickr 8K: Dataset is a dataset and has 8000 images collected from Flickr. The training data consists of 6000 images, the test and development data, each consists of 1,000 images. Each image in the dataset has 5 reference captions annotated by humans. A number of image captioning methods [21, 59, 61, 144, 150, 152] have performed experiments using the dataset. Two sample results by Jia et al. [59] on this dataset are shown in Figure 13.

3.11 Results

Since our model is data driven and trained end-to-end, and given the abundance of datasets, we wanted to answer questions such as “how dataset size affects generalization”, “what kinds of transfer learning it would be able to achieve”, and “how it would deal with weakly labeled examples”. As a result, we performed experiments on five different datasets, explained in Section 4.2, which enabled us to understand our model in depth.

```
    model.fit_generator(generator,epochs=1,steps_per_epoch=steps,verbose=1)
    model.save('/content/drive/MyDrive/Minor Project/save_model_weight4/model_'+str(i+12)+'.h5')
```

In [ ]:
```
train()
```

```
Epoch 1/1
9534/9534 [==============================] - 504s 53ms/step - loss: 3.1218
Epoch 1/1
9534/9534 [==============================] - 492s 52ms/step - loss: 3.1226
Epoch 1/1
9534/9534 [==============================] - 500s 52ms/step - loss: 3.1230
Epoch 1/1
9534/9534 [==============================] - 498s 52ms/step - loss: 3.1237
Epoch 1/1
9534/9534 [==============================] - 487s 51ms/step - loss: 3.1258
Epoch 1/1
9534/9534 [==============================] - 486s 51ms/step - loss: 3.1283
Epoch 1/1
9534/9534 [==============================] - 506s 53ms/step - loss: 3.1299
Epoch 1/1
9534/9534 [==============================] - 490s 51ms/step - loss: 3.1345
Epoch 1/1
8375/9534 [=========================>....] - ETA: 59s - loss: 3.1366Buffered data was truncated after reaching the output size
limit.
```

The model has been successfully trained to generate the captions as expected for the images. The caption generation has constantly been improved by fine tuning the model with different hyper parameters.
Higher BLEU score indicates that the generated captions are very similar to those of the actual caption present on the images.
BLEU is a popular machine translation metric that analyzes the co-occurrences of n-grams between the candidate and reference sentences.

```
    print('BLEU-1: %f' % corpus_bleu(actual, predicted, weights=(1.0, 0, 0, 0)))
    print('BLEU-2: %f' % corpus_bleu(actual, predicted, weights=(0.5, 0.5, 0, 0)))
    print('BLEU-3: %f' % corpus_bleu(actual, predicted, weights=(0.3, 0.3, 0.3, 0)))
    print('BLEU-4: %f' % corpus_bleu(actual, predicted, weights=(0.25, 0.25, 0.25, 0.25)))
```

```
In [16]: evaluate_model()
```

```
BLEU-1: 0.420399
BLEU-2: 0.222130
BLEU-3: 0.142831
BLEU-4: 0.060948
```

Fig 7) Bleu score results

The unigram scores (B-1) account for the adequacy of the translation, while longer n-gram scores (B-2, B-3, B-4) account for the fluency. Thus, having an accuracy of 42% which is considered to be high quality.

AutoML expresses **BLEU scores** as a percentage rather than a decimal between 0 and 1.
...
Interpretation.

BLEU Score	Interpretation
30 - 40	Understandable to good translations
40 - 50	High quality translations
50 - 60	Very high quality, adequate, and fluent translations
> 60	Quality often better than human

3 more rows

cloud.google.com › ... › AutoML › Documentation

Evaluating models - Google Cloud

According to Google, above than 40% accuracy is considered high quality in image detection

CHAPTER 4
CONCLUSION

4.1 Performance Evaluation

The project is successful. We have finished all the goals before the deadline. The system can generate sentences that are semantically correct according to the image. The strength of the method is on its end-to-end learning framework. The weakness is that it requires a large number of human labeled data which is very expensive in practice. Also, the current method still has considerable errors in both object detection and sentence generation.

4.2 Future Directions

In the future, we would like to explore methods to generate multiple sentences with different content. One possible way is to combine interesting region detection and image captioning.There can be potential improvements if given more time. First of all, we directly used a pre-trained CNN network as part of our pipeline without fine-tuning, so the network does not adapt to this specific training dataset. Thus, by experimenting with different CNN pre-trained networks and enabling fine-tuning, we expect to achieve a slightly higher BLEU score. Another potential improvement is by training on a combination of Flickr 8k, Flickr30k, and MSCOCO. In general, the more diverse training dataset the network has seen, the more accurate the output will be. We all agree this project ignites our interest in application of Machine Learning knowledge in Computer Vision and expects to explore more in the future.

CHAPTER 5
REFERENCES

1. P. Anderson, X. He, C. Buehler et al., "**Bottom-up and top down attention for image captioning**" in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, June 2018.

2. J. Aneja, A. Deshpande, and S. Alexander, "**Convolutional image captioning**," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, June 2018.

3. T. Yao, Y. Pan, Y. Li, Z. Qiu, and T. Mei, "**Boosting image captioning with attributes**," in Proceedings of the IEEE Conference on International Conference On Computer Vision, pp. 4904–4912, Las Vegas, NV, USA, June 2016.

4. H. R. Tavakoli, R. Shetty, B. Ali, and J. Laaksonen, "**Paying attention to descriptions generated by image captioning models**," in Proceedings of the IEEE Conference on International Conference on Computer Vision, pp. 2506–2515, Venice, Italy, October 2017.

5. C. C. Park, B. Kim, and G. Kim, "**Towards personalized image captioning via multimodal memory networks**," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 99, p. 1, 2018.

6. R. Zhou, X. Wang, N. Zhang, X. Lv, and L.-J. Li, "**Deep reinforcement learning-based image captioning with embedding reward**," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1151–1159, Honolulu, HI, USA, July 2017.

7. Y. Wu, M. Schuster, Z. Chen, and J. Dean, "**Google's neural machine translation system: bridging the gap between human and machine translation**," 2016, http://arxiv.org/abs/1609. 08144

8. M. Hodosh, P. Young, and J. Hockenmaier, "**Framing image description as a ranking task: data, models and evaluation metrics**," Journal of Artificial Intelligence Research, vol. 47, pp. 853–899, 2013.

9. L. Chen, H. Zhang, J. Xiao et al., "**SCA-CNN: spatial and channel-wise attention in convolutional networks for image captioning**," in Proceedings of the IEEE Conference

on Computer Vision and Pattern Recognition (CVPR), pp. 6298–6306, Las Vegas, NV, USA, June-July 2016.

10. J. Song, H. Zhang, X. Li, L. Gao, M. Wang, and R. Hong, "**Selfsupervised video hashing with hierarchical binary auto-encoder**," IEEE Transactions on Image Processing, vol. 27, no. 7, pp. 3210–3221, 2018.