DAYANANDA SAGAR UNIVERSITY

KUDLU GATE, BANGALORE – 560068

Bachelor of Technology
in
COMPUTER SCIENCE AND ENGINEERING

Major Project Phase - II Report

on

" IMAGE CAPTION GENERATOR FOR VISUALLY IMPAIRED "

By
Anusha J – ENG18CS0041
Karthik R – ENG18CS0128
Yashaswini N L – ENG18CS0327
Yashwanth G – ENG18CS0328
Yakshitha K C – ENG19CS1023

Under the supervision of

Dr. Revathi V
Associate Professor and Computer Science Engineering

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING,
SCHOOL OF ENGINEERING
DAYANANDA SAGAR UNIVERSITY,
BANGALORE

(2021-2022)

DAYANANDA SAGAR UNIVERSITY

School of Engineering

Department of Computer Science & Engineering

Kudlu Gate, Bangalore –560068
Karnataka, India

CERTIFICATE

This is to certify that the Phase-II project work titled **“IMAGE CAPTION GENERATOR FOR VISUALLY IMPAIRED”** is carried out by **ANUSHA J (ENG18CS0041), KARTHIK R (ENG18CS0128), YASHASWINI N L (ENG18CS0327), YASHWANTH G (ENG18CS0328), YAKSHITHA K C (ENG19CS1023),** bonafide students of Bachelor of Technology in Computer Science and Engineering at the School of Engineering, Dayananda Sagar University, Bangalore in partial fulfillment for the award of degree in Bachelor of Technology in Computer Science and Engineering, during the year **2021-2022**.

Dr. Revathi V	**Dr. Girisha G S**	**Dr. A Srinivas**
Associate Professor Dept. of CS&E, School of Engineering Dayananda Sagar University	Chairman CSE School of Engineering Dayananda Sagar University	Dean School of Engineering Dayananda Sagar University
Date:	Date:	Date:

Name of the Examiner	**Signature of Examiner**
1.	
2.	

DECLARATION

We, **Anusha J (ENG18CS0041),Karthik R (ENG18CS0128),Yashaswini N L (ENG18CS0327), Yashwanth G (ENG18CS0328),Yakshitha K C (ENG19CS1023),** are students of the seventh semester B.Tech in **Computer Science and Engineering**, at School of Engineering, **Dayananda Sagar University**, hereby declare that the phase-II project titled **"Image Caption Generator for Visually Impaired"** has been carried out by us and submitted in partial fulfillment for the award of degree in **Bachelor of Technology in Computer Science and Engineering** during the academic year **2021-2022**.

Student	Signature
Name1: Anusha J	
USN: ENG18CS0041	
Name2: Karthik R	
USN: ENG18CS0128	
Name3: Yashaswini N L	
USN: ENG18CS0327	
Name4: Yashwanth G	
USN: ENG18CS0328	
Name5: Yakshitha K C	
USN: ENG19CS1023	

Place: Bangalore
Date:

ACKNOWLEDGEMENT

It is a great pleasure for us to acknowledge the assistance and support of many individuals who have been responsible for the successful completion of this project work.
First, we take this opportunity to express our sincere gratitude to School of Engineering & Technology, Dayananda Sagar University for providing us with a great opportunity to pursue our Bachelor's degree in this institution.
We would like to thank **Dr. A Srinivas.Dean**, **School of Engineering & Technology**, **Dayananda Sagar University** for his constant encouragement and expert advice. It is a matter of immense pleasure to express our sincere thanks to **Dr. Girisha G S,Department Chairman**, **Computer Science, and Engineering**, **Dayananda Sagar University,** for providing the right academic guidance that made our task possible.
We would like to thank our guide **Dr. Revathi V**, **Associate Professor**, **Dept. of Computer Science and Engineering**, **Dayananda Sagar University**, for sparing her valuable time to extend help in every step of our project work, which paved the way for smooth progress and the fruitful culmination of the project.
We would like to thank our Project**Coordinators Dr. Meenakshi Malhotra** and **Dr.Bharanidharan N**, and all the staff members of Computer Science and Engineering for their support.
We are also grateful to our family and friends who provided us with every requirement throughout the course. We would like to thank one and all who directly or indirectly helped us in the Project work.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long Term Short Memory
API	Application Programming Interface
RAM	Random Access Memory
NLTK	Natural Language Toolkit

LIST OF FIGURES

Abstract

Being able to automatically describe the content of an image using properly formed English sentences is a challenging task, but it could have great impact by helping visually impaired people better understand their surroundings. Most modern mobile phones are able to capture photographs, making it possible for the visually impaired to make images of their environments. These images can then be used to generate captions that can be read out loud to the visually impaired, so that they can get a better sense of what is happening around them.

Our proposed approach aims to build an encoder-decoder architecture where the encoder is a pre-trained model like VGG16, ResNet50, InceptionV3 and MobileNet. It may be a single model or the combination of vectors consisting of high-level features from two or more models works exceptional in most of the cases. Whereas the decoder part is a slightly modified LSTM network and contains the Time Distributed Layer which are helpful for time series data as we post the entire sentence or sequence as a time series problem. The captions generated by this model is translated into speech which helps visually impaired to know their surroundings easily.

CHAPTER 1
INTRODUCTION

CHAPTER 1 INTRODUCTION

The problem of generating natural language descriptions of an image to describe the visual content has received much interest in the fields of computer vision and natural language processing, driven by applications such as image indexing or retrieval, virtual assistants, image understanding and support of the visually impaired people. The goal of this project is generating captions for images to help visually impaired people know about their surroundings. The model is a form of encoder-decoder architecture that is trained on Flickr8k dataset.

1.1. PURPOSE AND PROCESS:

The main purpose of this project is to provide a technology oriented, low computational power, easily scalable, and a robust model.

Being able to automatically describe the content of an image using properly formed English sentences is a very challenging task, but it could have great impact, for instance by helping visually impaired people better understand the image.

The secondary thing is to set a color composition to images and represent the high-level features in an effective way using several pre-trained models. Treating the input sentence as a time series data and maintaining the sequence information using time distributed LSTM layers.

1.2. SCOPE

This project is meant for generating captions for the given image. It minimizes the human effort for generating captions manually that can be used for future use or for an application development based on the elements in the image. CCTV cameras are meant for monitoring, but if useful captions are provided in addition to watching the world, we can trigger warnings as soon as dangerous conduct is detected. This is likely to help minimize crime or accidents. It is also helpful in making an app which gives captions for videos.

Image caption generation can also make the Web more accessible to visually impaired people. The last decade has seen the triumph of the rich Graphical desktop, replete with colorful icons, controls, buttons, and images. Automated caption Generation of online images can make the web a more inviting place for visually impaired surfers.

CHAPTER 2

PROBLEM DEFINITION

CHAPTER 2 PROBLEM DEFINITION

Every day, we are exposed to a significant number of images from a variety of sources, including the internet, news articles, schematics in documents, and advertisements. These resources provide visuals that visitors must interpret for themselves. Majority of photos do not contain a description, yet humans can make sense of them without captions. However, if people with visible impairment want automated image captions from the machine, so the system must be able to understand and interpret them.

In this Project, an encoder-decoder architecture model is implemented where the encoder is a pre-trained model like VGG16, ResNet50, InceptionV3 and MobileNet. It may be a single model or the combination of vectors consisting of high-level features from two or more models works exceptional in most of the cases. The decoder part is a slightly modified LSTM network and contains the Time Distributed Layer which is helpful for time series data as the entire sentence or sequence is posted as a time series problem. The system generated captions are translated to speech that helps blind people to know the image and also the outside world.

CHAPTER 3
LITERATURE REVIEW

CHAPTER 3 LITERATURE REVIEW

According to the research [1], the Inception architecture's complexity makes it more difficult to make network adjustments, and if the design is scaled up too quickly, huge portions of the computational advantages might be lost. Inception is compared to other models in this research, and it produces less error and has a low computational cost. The conclusion reached from this article is that Inception is a complicated network in which making modifications is challenging.

Many of the difficulties they encountered when training the models were due to overfitting, according to this research [2]. True, purely supervised algorithms need a big quantity of data, but high-quality datasets contain less than 100,000 photos. The process of providing a description is far more difficult than classifying an object. Different datasets were examined, and it was discovered that as the size of the available datasets for picture description grows larger, so does the performance, but giving descriptions to each image gets more challenging. The Flickr8k dataset is more efficient, according to the findings.

This model [3] is provided with LRCN, a family of models that is both spatially and temporally deep, as well as versatile enough to be used for a wide range of visual problems requiring sequential inputs and outputs. LSTM outperforms STM, according to the results.

The model was presented as a family of mobile-first computer vision models for Tensor flow, according to the research [4], and was developed to successfully optimize accuracy while being conscious of the limited resources for on-device or embedded applications. Mobile Nets are low-latency, low-power models that have been parameterized to satisfy the resource restrictions of various use cases. With a percentage of 70.6 per cent, Mobile Net outperforms ImageNet when compared to other models.

They employed the ResNet50 network as an encoder to transform the pictures into a vector representation, and the LSTM network as a decoder to decode the vector and create captions in this article [5]. Soft Attention is a method that allows the model to focus on a specific region of the image to better anticipate phrases. The captions are most likely based on the greatest probability. As a consequence, appropriate captions are created automatically with an image classification accuracy of 71%.

According to the study [6], LSTM is used to increase the precision of image captioning by mapping variable sequences of words in NLP to a distributed vector using a decoder. The neural model creates a description for the picture in this method. With CNN as the image encoder, the RNN decoder uses the categorized image to create captions using the hidden layer.

CHAPTER 4
PROJECT DESCRIPTION

CHAPTER 4 PROJECT DESCRIPTION

4.1. PROJECT DESIGN

4.1.1. CLASS DIAGRAM

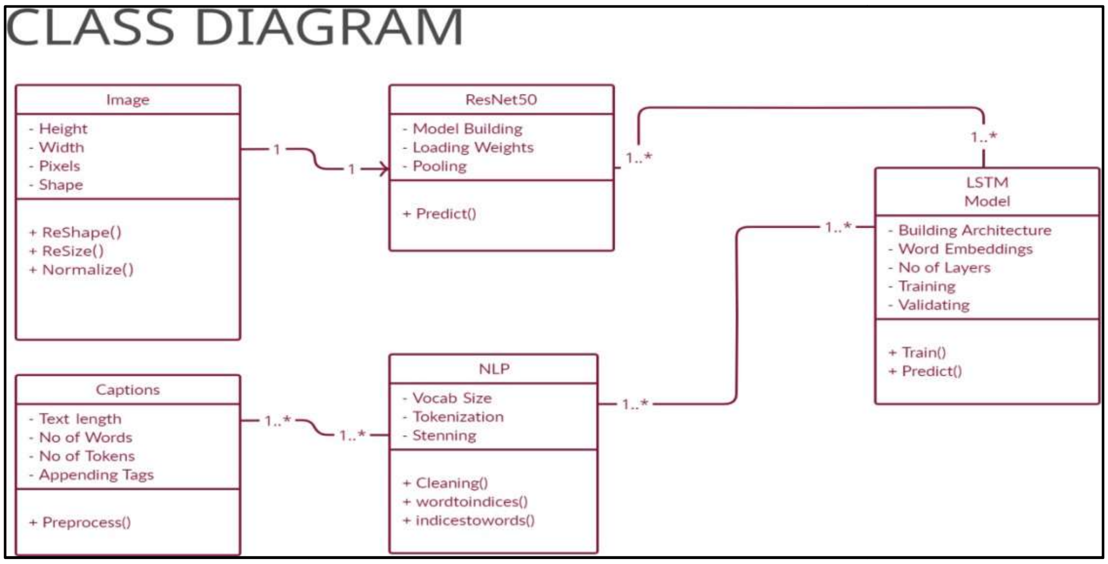


Figure 4.1 Class Diagram

The static view of an application is represented by a class diagram, which is used for visualizing, describing, and documenting various parts of a system as well as constructing executable code for the software application. It is frequently used in the modelling of object-oriented systems since it is the only UML diagram that can be mapped directly with object-oriented languages. It describes the attributes and operations of a class as well as the constraints imposed on the system. Fig 4.1 that illustrates about class diagram of our project.

4.1.2. SEQUENCE DIAGRAM

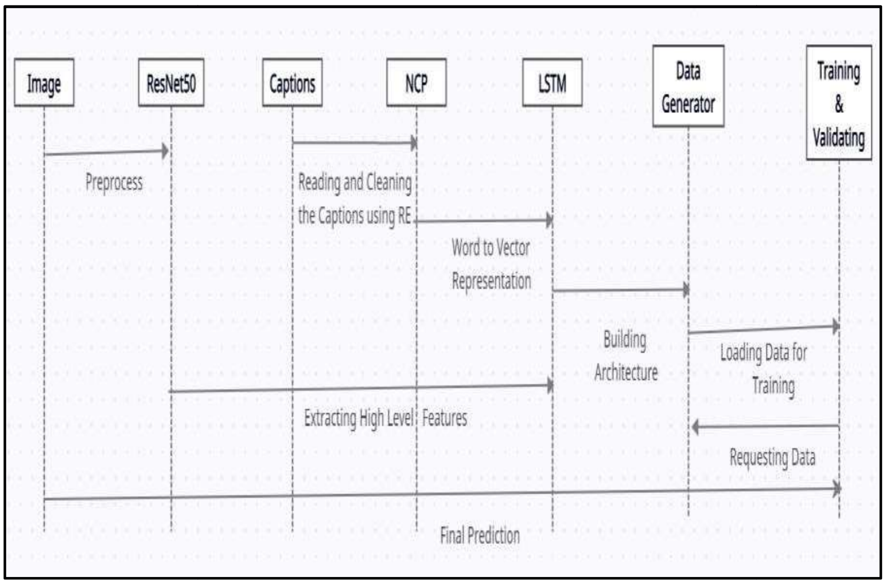


Image
ResNet50
Captions
NCP
LSTM
Data Generator
Training & Validating
Preprocess
Reading and Cleaning the Captions using RE
Word to Vector Representation
Building Architecture
Loading Data for Training
Extracting High Level Features
Requesting Data
Final Prediction

Figure 4.2 Sequence Diagram

A sequence diagram simply shows how items interact in a sequential order as shown in the Fig 4.2. A sequence diagram can also be referred to as an event diagram or an event scenario. Sequence diagrams show how and in what order the components of a system function. Business people and software developers often use these diagrams to document and understand requirements for new and existing systems.

4.1.3. USE CASE DIAGRAM

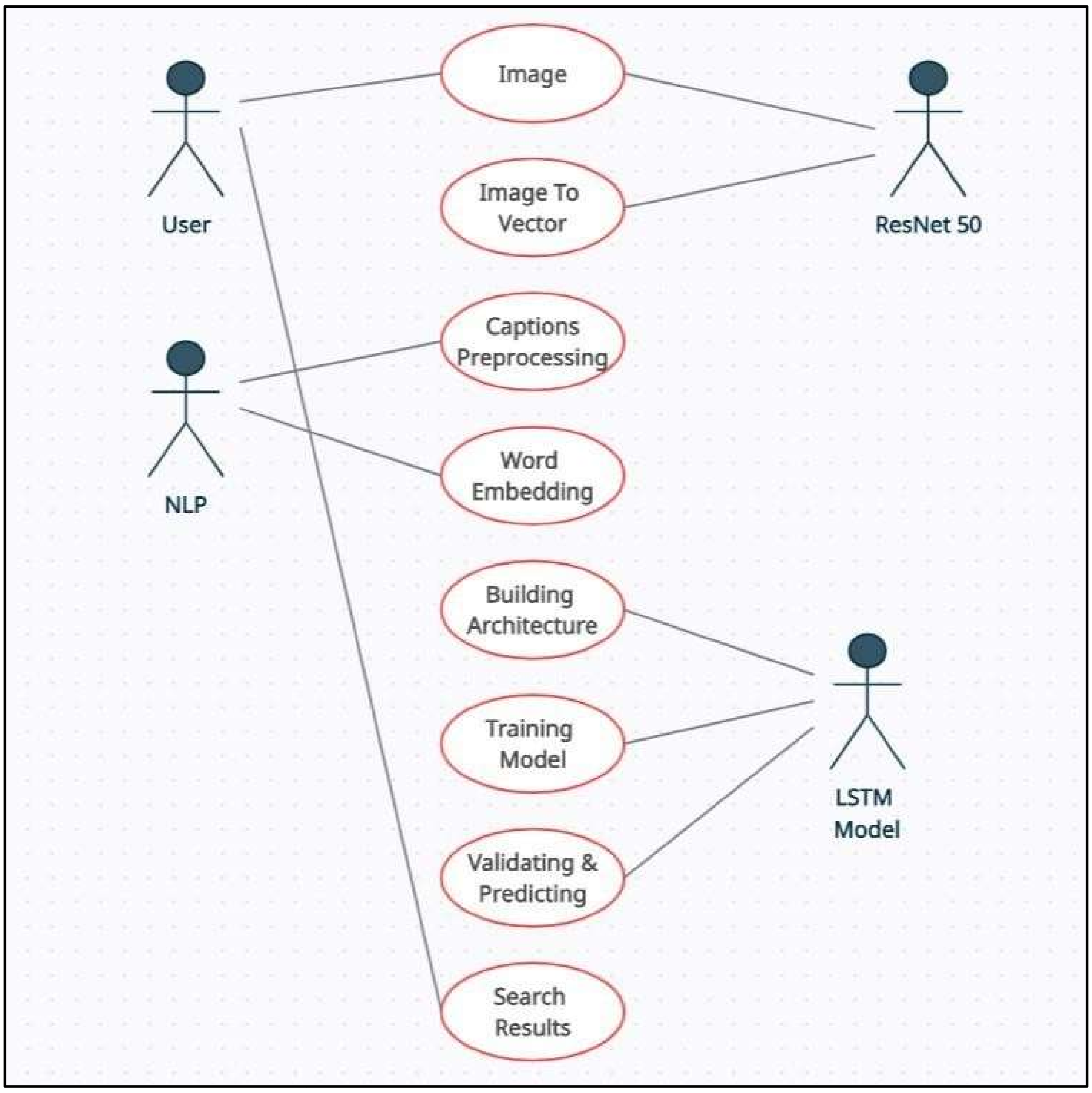


Figure 4.3 Use Case Diagram

The purpose of a use case diagram is to capture the dynamic aspect of a system. Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. As a result, use cases are generated and actors are identified when a system is studied to gather its functionality as shown in the Fig 4.3.

4.1.4. STATE CHART DIAGRAM

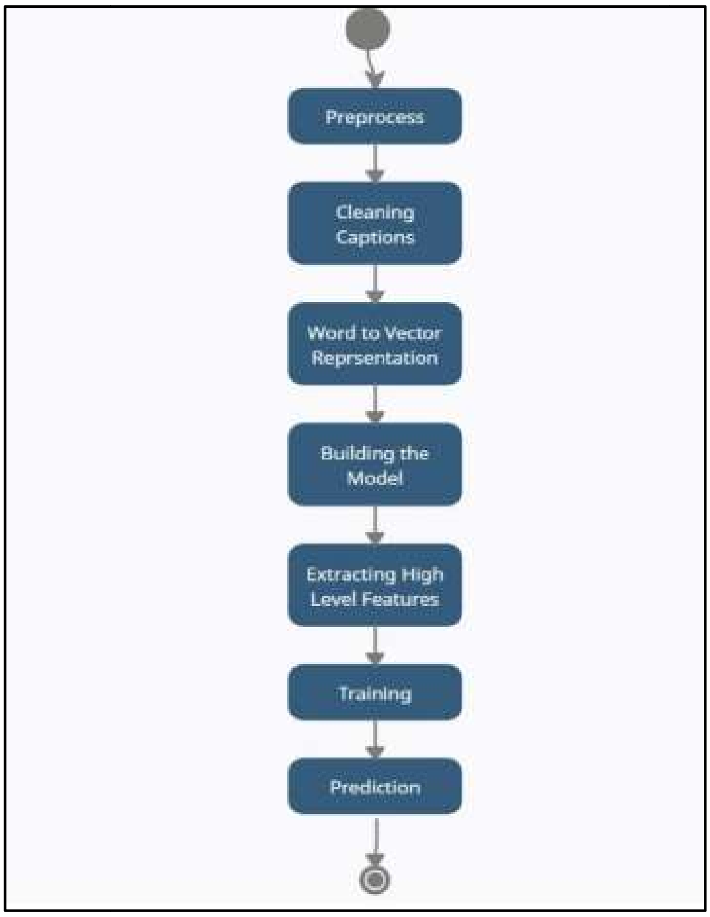


Figure 4.4 State Chart Diagram

State chart diagram itself clarifies the purpose of the diagram and other details. It describes different states of a component in a system. The states are specific to a component/object of a system. A State chart diagram describes a state machine. A state machine is a machine that defines multiple states of an entity and controls these states through external or internal events as shown in the figure 4.4.

4.1.5. FLOW CHART

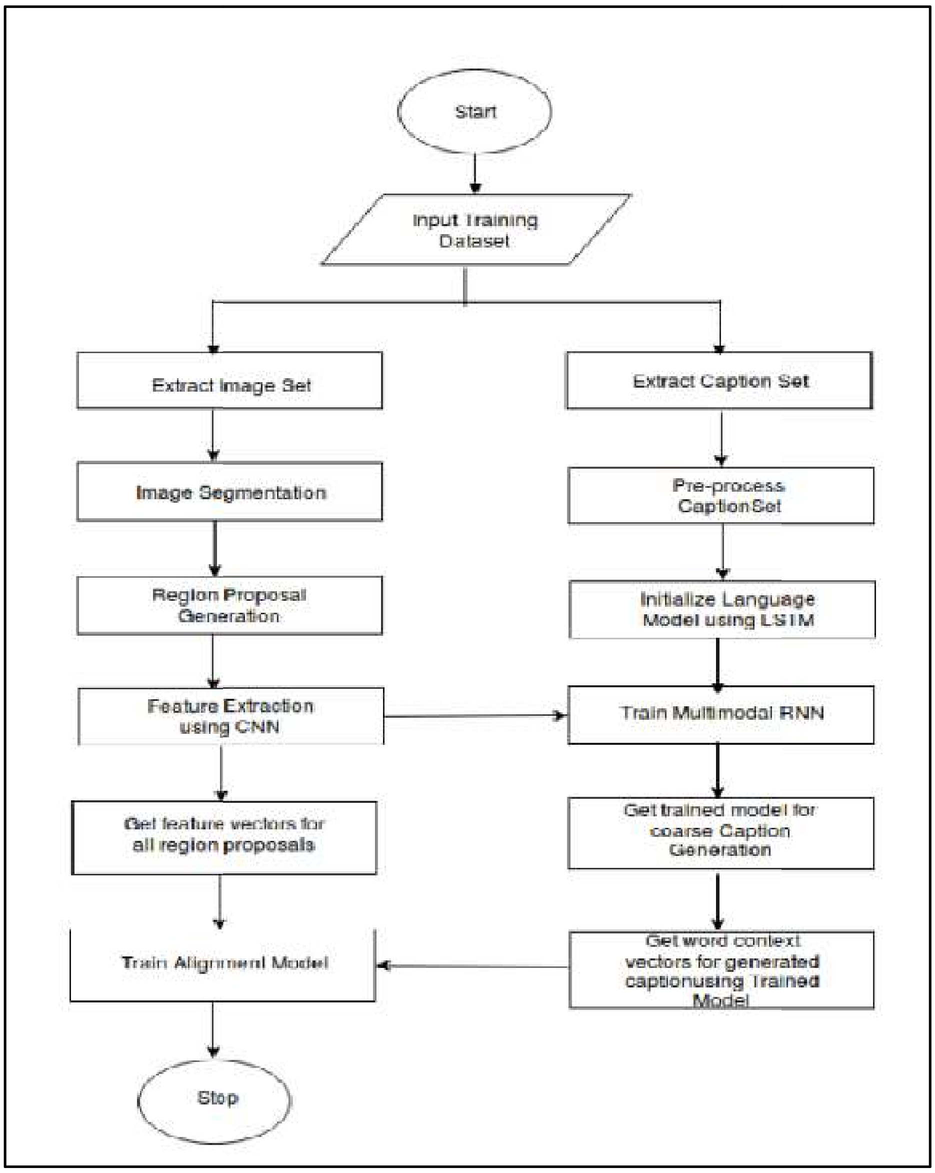

Start
Input Training Dataset
Extract Image Set
Extract Caption Set
Image Segmentation
Pre-process CaptionSet
Region Proposal Generation
Initialize Language Model using LSTM
Feature Extraction using CNN
Train Multimodal RNN
Get feature vectors for all region proposals
Get trained model for coarse Caption Generation
Train Alignment Model
Get word context vectors for generated captionusing Trained Model
Stop

Figure 4.5 Flow Chart

A flowchart illustrates the individual steps of a process in a sequence order. It is a generic tool that may be used for a wide range of purposes and can be used to describe a number of processes, including manufacturing, administrative and service processes, and project plans as shown in the Fig 4.5.

4.2. ASSUMPTIONS AND DEPENDENCIES

- Being a robust model, we are assuming that our model requires Low Computational Power and do not require special training.
- Our model is trained Flickr8k Dataset provided by University of Illinois at Urbana Chamcaign selected to reflect a diversity of scenario and circumstances.
- There is no repetition of the Images.

.

CHAPTER 5
REQUIREMENTS

CHAPTER 5 REQUIREMENTS

5.1. FUNCTIONALREQUIREMENTS:

- Dataset: Flickr8k dataset (which contains 8k photos) provided by the University of Illinois at Urbana-Champaign. This is obtained from Kaggle website.
- Pre-Processing: The dataset obtained consists of unwanted, redundant and incomplete data which must be cleaned before being used by neural network. It includes tokenization and stemming.
- Python Distribution: Anaconda Enterprise is a business-ready, safe, and scalable data science platform that enables teams to manage, collaborate, and implement data science projects.
- Modules to be installed: TensorFlow, Keras, NumPy, NLTK, Pandas, OpenCV.

5.2. NON-FUNCTIONALREQUIREMENTS:

- Technical Feasibility: The technical issue usually raised during the feasibility stage of the investigation includes the following:
    - Does the necessary technology exist to do what is suggested?
    - Do the proposed equipment 's have the technical capacity to hold the data required to use the new system?
    - Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
    - Are there technical guarantees of accuracy, reliability, ease of access and data security?
- Operation Feasibility: The operational feasibility includes User friendly, reliability, security, portability, availability, and maintainability of the software used in the project.
- Economic Feasibility: Analysis of a project costs and revenue in an effort to determine whether or not it is logical and possible to complete.

5.3. SOFTWAREREQUIREMENTS:

- Operating System: Windows7/8/10 or Ubuntu
- Programming Language: Python
- Front-end Design: Html, CSS, JavaScript
- Tool: Anaconda
- Modules to be Installed: TensorFlow, Keras, NumPy, NLTK, Pandas, OpenCV
- Designing UML Diagrams: Star UML

5.4. HARDWAREREQUIREMENTS

- 8 GB RAM and above recommended.
- 10 GB internal storage and above recommended.
- Intel core i3 processor and above.
- Ryzen 3 processor and above recommended.

CHAPTER 6
METHODOLOGY

CHAPTER 6 METHODOLOGY

6.1. ENCODER DECODER ARCHITECTURE

6.1.1. ENCODER

The encoder is used to encode the data to obtain insights and is used to extract high-level properties from images. In this situation, ResNet50 is used as the encoder. ResNet50, also known as Residual Networks, is a widely used neural network in computer vision. Using ResNet, 150-layer deep neural networks can be effectively trained. The problem of vanishing gradients made training very deep neural networks difficult before ResNet. ResNet is a computer vision backbone model that is utilized in a range of applications. ResNet uses skip connections to fix the problem of disappearing gradients.

Each step of the ResNet50 model has its convolution and identity block. There are three convolution layers in each convolution block, and three convolution layers in each identity block. There are around 23 million trainable parameters in the ResNet-50. A minor change was made for ResNet50 and higher: shortcut connections previously skipped two layers; now, they skip three levels, and 1 x 1 convolution layers have been introduced.

6.1.2. DECODER

Long-term memory networks are a type of Recurrent Neural Network that can learn long-term dependencies. Long-term memory networks are also called LSTMs. LSTMs were created to overcome the problem of long-term reliance. They don't have to exert much effort to recall information for long periods; it comes naturally to them. Recurrent neural networks are made up of a series of modules that repeat. In classic RNNs, this repeating module will have a basic structure, such as a single tanh layer.

The cell state, which is represented by the horizontal line running along the top of the image, is the key to LSTMs. The cell is in a condition similar to that of a conveyor belt. With only a few modest linear exchanges, it flows along the whole chain. It's really simple for data to pass through it unaltered. The LSTM's capacity to remove or add information to the cell state is closely controlled by the structure.

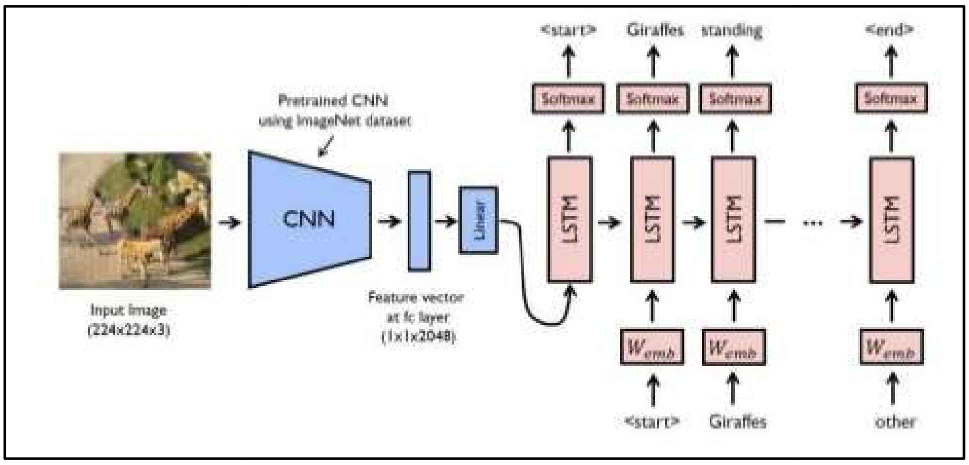


Figure 6.1. Encoder Decoder Model

6.2. METHODOLOGY

- **Image and Caption Dataset Access and Reading:** We use the flicker 8k dataset which contains 8000 images provided by the University of Illinois at Urbana-Champaign. This dataset is available on the Kaggle website. Each image is provided with 5 captions which are later combined by the model to output a single sentence that describes the image the best.
- **Image Data Preparation and Processing:** We need to convert the image into an encoding so that the machine can understand the patterns in it. InceptionV3, a model that is trained on large datasets is used to extract the features from images. Every image should be converted to a fixed-sized vector which can be fed as an input to the neural network. the last SoftMax layer is removed from the model and a 2048 length vector is extracted for every image.
- **Creating a Resnet50 encoder model and extracting features from images**: An image from any source is given as input to the encoder-decoder model, which processes it into a pre-trained Resnet50 model and extracts the 2048-dimensional vector from the model. This contains high-level image features and then it embeds the vector into the dimensions of the sentence vector, which is initially a tag and whose two vectors are concatenated and passed over a Time Distributed Layer followed by LSTM.
- **Cleaning and conducting word embeddings on the text data:** The model must predict the captions based on the image. Hence, during the training period captions will be the target variables that the model is learning to predict. It then predicts the caption word by word. Therefore; each word should

be encoded into a fixed-sized vector. Every unique word in the vocabulary is represented by an integer which is later used for pre-processing.

- **LSTM Decoder Model development:** In long-short term memory, the output of each LSTM cell is transmitted as an input to the next LSTM cell. The final output is created by concatenating the outputs of all LSTM cells. The last output sentence provides a concise description of the input image. Finally, the text is converted to the audio format for the output.
- **Using Data Generators and Progressive loading to train the model: The** amount of memory to store data blocks in the data matrix consumes a lot of space in the main memory. Even if it is managed to store in the RAM, it slows down the computer. The ImageDataGenerator class provided by the Keras API is nothing but an implementation of the generator function in Python. This makes sure that the entire dataset need not be stored in the main memory at a time, reducing storage consumption.
- **Use the test dataset to evaluate the trained model:** The Flicker 8k dataset is divided into train and test datasets to check how efficient is its prediction for unseen data i.e., test data. The model is evaluated with the test dataset to check the prediction accuracy. The dataset is split into a 75:25 ratio where 6000 images are used for training and the rest 2000 images are used for testing.
- **Text to speech conversion:** After the caption is predicted for the image, it is converted into audio using the flutter_tts plugin.
- **Developing a user-friendly interface**: An application using flutter is developed where the user can get a description of the image in the form of speech.

CHAPTER 7
EXPERIMENTATION

CHAPTER 7 EXPERIMENTATION

ENCODER

```
def preprocessing(img_path):
im = image.load_img(img_path, target_size=(224,224,3))
im = image.img_to_array(im)
im = np.expand_dims(im, axis=0)
   return im
imgs={}
for i in tqdm(x_train[:6000]):
   if i in imgs.keys():
      continue
   path = images_path + i
img = preprocessing(path)
   pred = model.predict(img).reshape(2048)
imgs[i]=pred

img = []
for j in tqdm(range(df.shape[0])):
   if df.iloc[j, 0] in imgs.keys():
img.append(imgs[df.iloc[j, 0]])

img = np.asarray(img)
print(img.shape)
```

DECODER

```
conca = Concatenate()([image_model.output, language_model.output])
x = LSTM(128, return_sequences=True)(conca)
x = LSTM(512, return_sequences=False)(x)
x = Dense(vocab_size)(x)
out = Activation('softmax')(x)
```

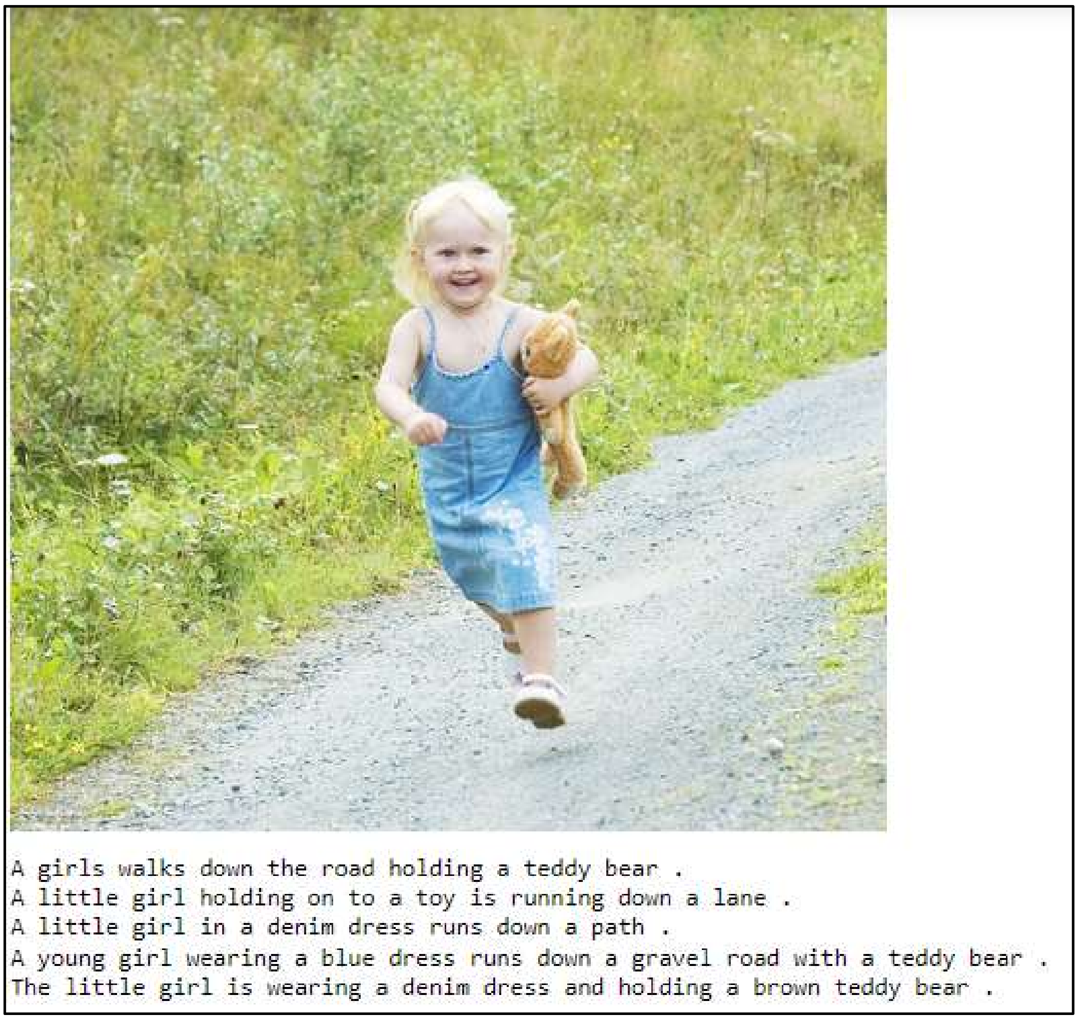
```
model = Model(inputs=[image_model.input, language_model.input], outputs = out)
model.compile(loss='categorical_crossentropy', optimizer='RMSprop', metrics=['accuracy'])
model.summary()
def tr_gen(inp_imgs,captions,batch_photos=256):
  x1, x2, y = list(), list(), list()
  n=0
  while True:
    for i in range(df.shape[0]):
      n+=1
      photo=inp_imgs[i]
      text = captions[i].split()
      text = [word_2_indices[i] for i in text]
      for j in range(1, len(text)):
partial_seqs=text[:j]
next_words=text[j]
padded_partial_seqs = sequence.pad_sequences([partial_seqs], 40, padding='post')[0]
next_words = to_categorical([next_words], num_classes=vocab_size)[0]
        x1.append(photo)
        x2.append(padded_partial_seqs)
y.append(next_words)
      if n==batch_size:
        yield [np.array(x1), np.array(x2)], np.array(y)
        x1, x2, y = list(), list(), list()
        n=0
def preprocessing(img_path):
im = image.load_img(img_path, target_size=(224,224,3))
im = image.img_to_array(im)
im = np.expand_dims(im, axis=0)
  return im
im = 'C:/Users/SRI/Desktop/Project/archive/Flickr_Data/Flickr_Data/Images/'+x_test[121]
test_img = get_encoding(resnet, im)
def predict_captions(image):
start_word = ["<start>"]
```

```
   while True:
par_caps = [word_2_indices[i] for i in start_word]
par_caps = sequence.pad_sequences([par_caps], maxlen=max_len, padding='post')
      preds = model.predict([np.array([image]), np.array(par_caps)])
word_pred = indices_2_word[np.argmax(preds[0])]
start_word.append(word_pred)

      if word_pred == "<end>" or len(start_word) >max_len:
         break

   return ' '.join(start_word[1:-1])

Argmax_Search = predict_captions(test_img)
z = Image(filename=im)
display(z)

print(Argmax_Search)
```

CHAPTER 8
TESTING AND RESULTS

CHAPTER 8 TESTING AND RESULTS

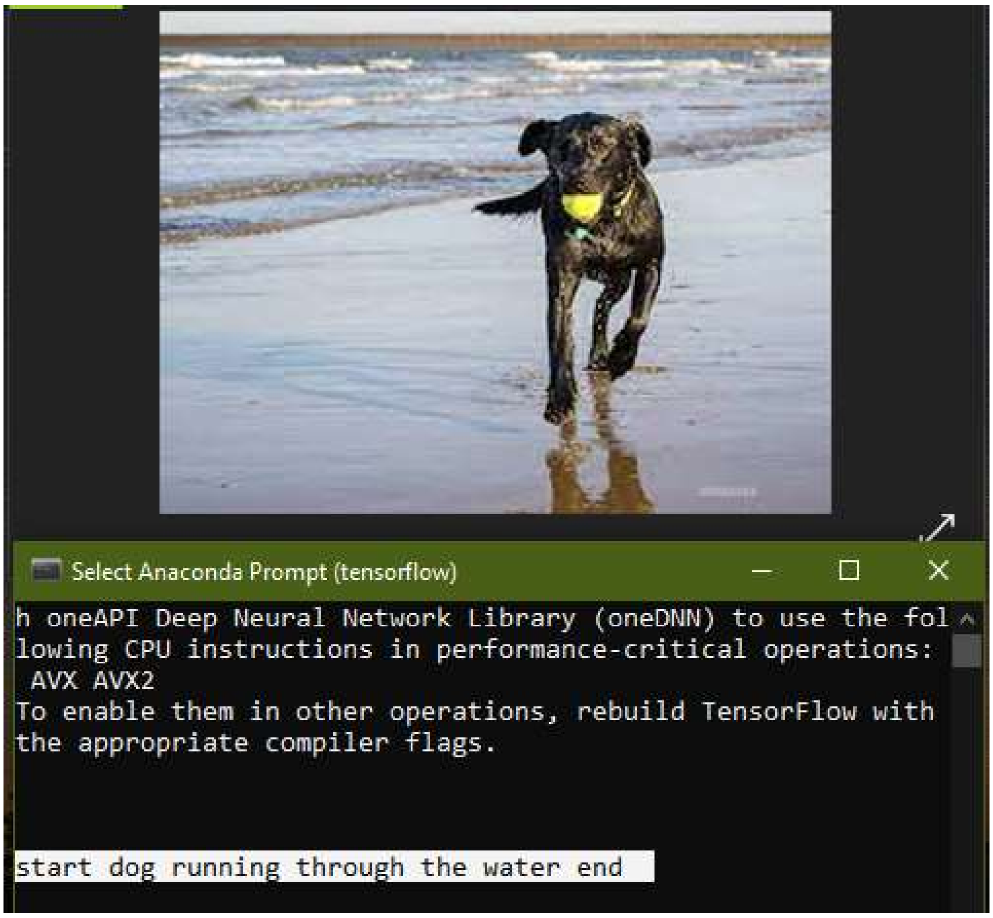
8.1. TEST CASES



```
A girls walks down the road holding a teddy bear .
A little girl holding on to a toy is running down a lane .
A little girl in a denim dress runs down a path .
A young girl wearing a blue dress runs down a gravel road with a teddy bear .
The little girl is wearing a denim dress and holding a brown teddy bear .
```

Figure 8.1 Image with five captions from dataset

8.2. RESULTS



```
h oneAPI Deep Neural Network Library (oneDNN) to use the fol
lowing CPU instructions in performance-critical operations:
 AVX AVX2
To enable them in other operations, rebuild TensorFlow with
the appropriate compiler flags.


start dog running through the water end
```

Figure 8.2. Output of an image

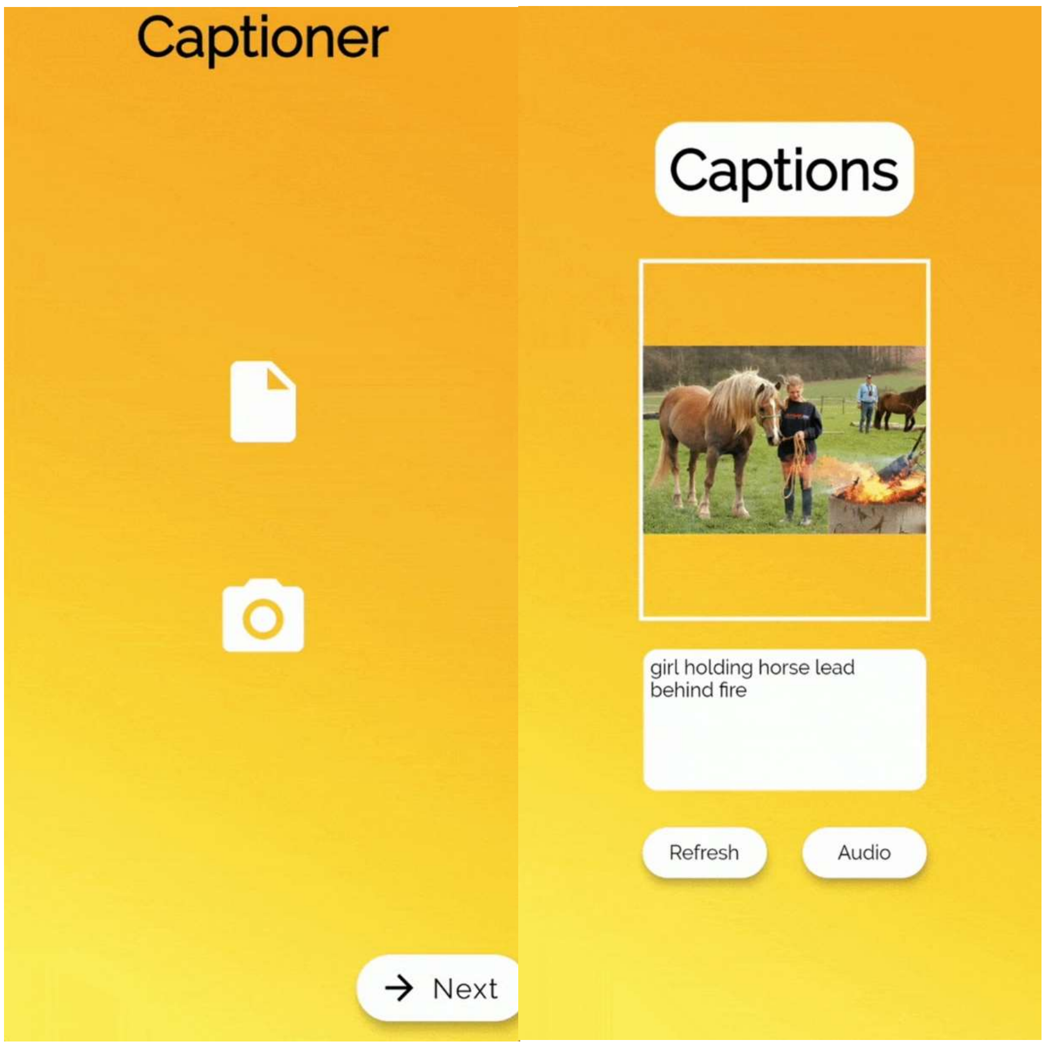


Figure 8.3. frontend of project

CHAPTER 9
CONCLUSION

CHAPTER 9 CONCLUSION

The suggested model is resilient, requires little processing resources, and requires no specific training since its accuracy on training data is over 92.4 per cent with a loss of 0.32, and its validation accuracy is 90 per cent with a loss of 0.12. This architecture incorporates previous pre-trained model advances as well as a variety of deep learning and natural language methodologies. The scalability of big applications with this paradigm is significant since the whole system is streamlined. Because we employed Time distributed layers followed by LSTM cells, the sentence's sequence information will be saved, and because video frames are time based, these time-distributed layers will perform very well on such data for creating captions for movies. The generated captions while validating the model with test data are accurate and have well-defined semantic meaning.

CHAPTER 10
FUTURE WORKS

CHAPTER 10 FUTURE SCOPE

The model's accuracy can be improved even further by training on a larger dataset. It can be built into a smart glass or a portable gadget that is easy to use for visually impaired people. With facial recognition and optical character recognition, this model may be updated to provide a caption for a live video feed. Depending on the user's preferences, it can create captions in a variety of languages.

REFERENCES

[1] Christian Szegedy1, Vincent Vanhoucke2, Sergey Ioffe3, Jonathon Shlens4, Zbigniew Wojna5. "Rethinking the Inception Architecture for Computer Vision," in 2016 IEEE Conference on Computer Vision and Pattern Recognition; pages 2818-2826, 2016

[2] Oriol Vinyals Google1, Alexander Toshev Google2, SamyBengio Google4, Dumitru Erhan Google6. "A Neural Image Caption Generator," in 2015 IEEE Conference on Computer Vision and Pattern Recognition; pages 3156-3164, June 2015

[3] Jeff Donahue1, Lisa Anne Hendricks2, Marcus Rohrbach3, Subhashini Venugopalan4, Sergio Guadarrama5, Kate Saenko6, Trevor Darrel7. "Long-term Recurrent Convolutional Networks for Visual Recognition and Description," in 2015 IEEE Conference on Computer Vision and Pattern Recognition; June 2015.

[4] Andrew G. Howard1, Menglong Zhu2, Bo Chen3, Dmitry Kalenichenko4, Weijun Wang5, Tobias Weyand6, Marco Andreetto7, Hartwig Adam8. "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," April 2017.

[5] Yan Chu1, Xiao Yue2, Lei Yu3, Mikhailov Sergei4, Zhengkui Wang5. "Automatic Image Captioning Based on ResNet50 and LSTM with Soft Attention," Volume 2020, October 2020.

[6] Chetan Amritkar1, Vaishali Jabade2. "Image Caption Generation using Deep Learning Technique," in 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA); 2018.

APPENDIX A

```
def localProperties = new Properties()
def localPropertiesFile = rootProject.file('local.properties')
if (localPropertiesFile.exists()) {
localPropertiesFile.withReader('UTF-8') { reader ->
localProperties.load(reader)
   }
}

def flutterRoot = localProperties.getProperty('flutter.sdk')
if (flutterRoot == null) {
   throw new GradleException("Flutter SDK not found. Define location with flutter.sdk in the
local.properties file.")
}

def flutterVersionCode = localProperties.getProperty('flutter.versionCode')
if (flutterVersionCode == null) {
flutterVersionCode = '1'
}

def flutterVersionName = localProperties.getProperty('flutter.versionName')
if (flutterVersionName == null) {
flutterVersionName = '1.0'
}

apply plugin: 'com.android.application'
apply plugin: 'kotlin-android'
apply from: "$flutterRoot/packages/flutter_tools/gradle/flutter.gradle"

android {
compileSdkVersion 28

sourceSets {
main.java.srcDirs += 'src/main/kotlin'
   }

lintOptions {
      disable 'InvalidPackage'
   }

defaultConfig {
      // TODO: Specify your own unique Application ID
(https://developer.android.com/studio/build/application-id.html).
applicationId "com.example.captioner"
minSdkVersion 21
targetSdkVersion 28
versionCodeflutterVersionCode.toInteger()
versionNameflutterVersionName
testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

buildTypes {
        release {
            // TODO: Add your own signing config for the release build.
            // Signing with the debug keys for now, so `flutter run --release` works.
signingConfigsigningConfigs.debug
        }
    }
}

flutter {
    source '../..'
}

dependencies {
    implementation "org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version"
testImplementation 'junit:junit:4.12'
androidTestImplementation 'androidx.test:runner:1.1.1'
androidTestImplementation 'androidx.test.espresso:espresso-core:3.1.1'
}
```

Github Link:

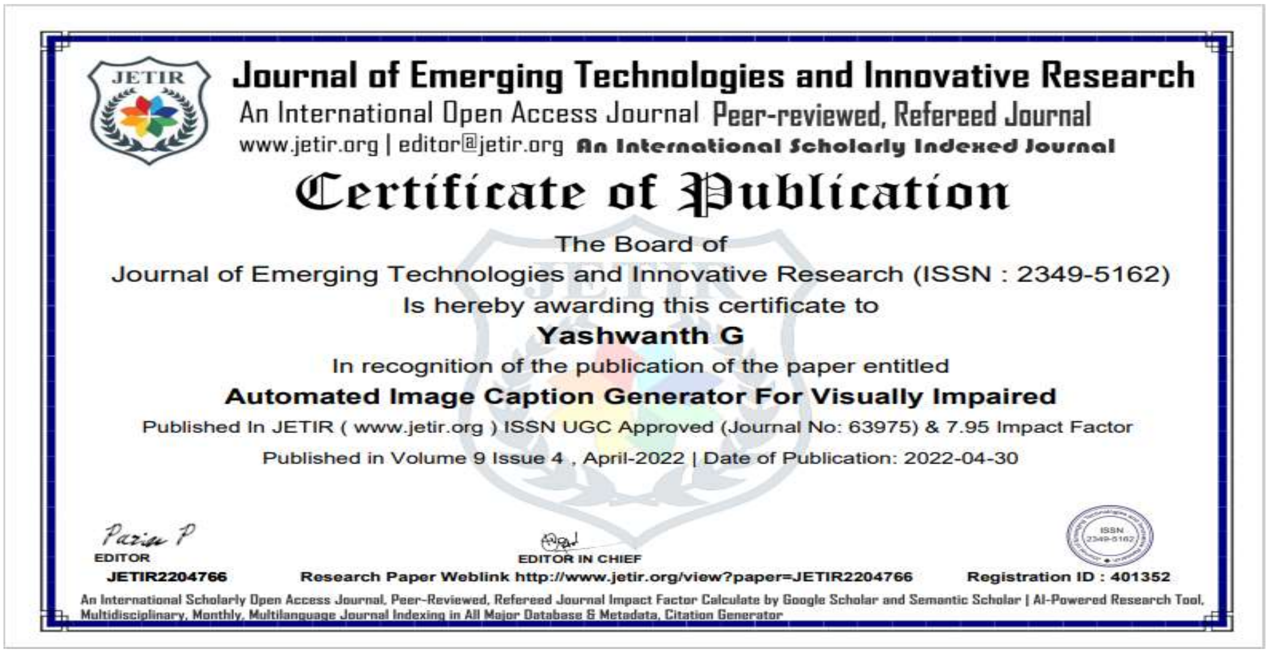
https://github.com/ImageCaptionGenerator/Image-Caption-Generator-for-visually-impaired

PUBLISHED PAPER DETAILS

PAPER DETAILS

We have successfully published a paper entitled "Automated Image Caption Generator For Visually Impaired" in Journal of Emerging Technologies and Innovative Research, in vol 9 issue 4, April 2022

CERTIFICATES



Journal of Emerging Technologies and Innovative Research
An International Open Access Journal Peer-reviewed, Refereed Journal
www.jetir.org | editor@jetir.org An International Scholarly Indexed Journal

Certificate of Publication

The Board of
Journal of Emerging Technologies and Innovative Research (ISSN : 2349-5162)
Is hereby awarding this certificate to
Yashwanth G
In recognition of the publication of the paper entitled
Automated Image Caption Generator For Visually Impaired
Published In JETIR (www.jetir.org) ISSN UGC Approved (Journal No: 63975) & 7.95 Impact Factor
Published in Volume 9 Issue 4 , April-2022 | Date of Publication: 2022-04-30

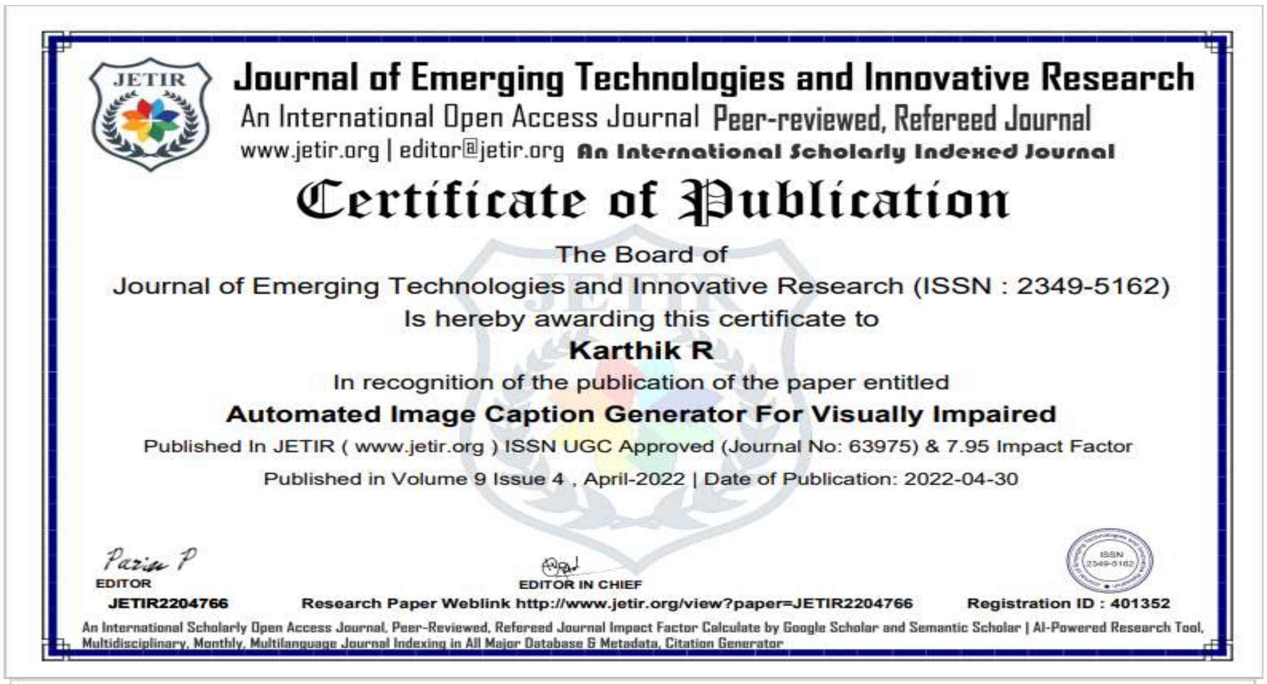
[illegible]
EDITOR

[illegible]
EDITOR IN CHIEF

ISSN 2349-5162

JETIR2204766 **Research Paper Weblink http://www.jetir.org/view?paper=JETIR2204766** **Registration ID : 401352**

An International Scholarly Open Access Journal, Peer-Reviewed, Refereed Journal Impact Factor Calculate by Google Scholar and Semantic Scholar | AI-Powered Research Tool, Multidisciplinary, Monthly, Multilanguage Journal Indexing in All Major Database & Metadata, Citation Generator



Journal of Emerging Technologies and Innovative Research

An International Open Access Journal Peer-reviewed, Refereed Journal

www.jetir.org | editor@jetir.org An International Scholarly Indexed Journal

Certificate of Publication

The Board of

Journal of Emerging Technologies and Innovative Research (ISSN : 2349-5162)

Is hereby awarding this certificate to

Karthik R

In recognition of the publication of the paper entitled

Automated Image Caption Generator For Visually Impaired

Published In JETIR (www.jetir.org) ISSN UGC Approved (Journal No: 63975) & 7.95 Impact Factor

Published in Volume 9 Issue 4 , April-2022 | Date of Publication: 2022-04-30

Parisa P

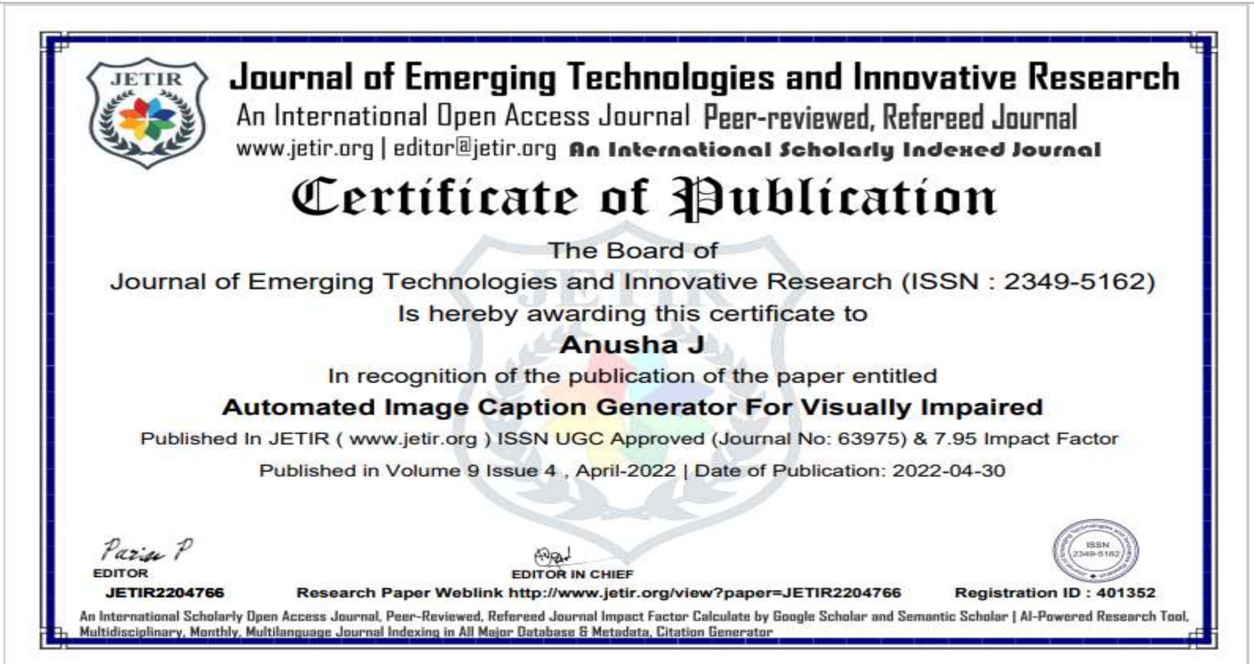
EDITOR

EDITOR IN CHIEF

ISSN 2349-5162

JETIR2204766 Research Paper Weblink http://www.jetir.org/view?paper=JETIR2204766 Registration ID : 401352

An International Scholarly Open Access Journal, Peer-Reviewed, Refereed Journal Impact Factor Calculate by Google Scholar and Semantic Scholar | AI-Powered Research Tool, Multidisciplinary, Monthly, Multilanguage Journal Indexing in All Major Database & Metadata, Citation Generator



Journal of Emerging Technologies and Innovative Research

An International Open Access Journal Peer-reviewed, Refereed Journal

www.jetir.org | editor@jetir.org An International Scholarly Indexed Journal

Certificate of Publication

The Board of

Journal of Emerging Technologies and Innovative Research (ISSN : 2349-5162)

Is hereby awarding this certificate to

Anusha J

In recognition of the publication of the paper entitled

Automated Image Caption Generator For Visually Impaired

Published In JETIR (www.jetir.org) ISSN UGC Approved (Journal No: 63975) & 7.95 Impact Factor

Published in Volume 9 Issue 4 , April-2022 | Date of Publication: 2022-04-30

Parisa P

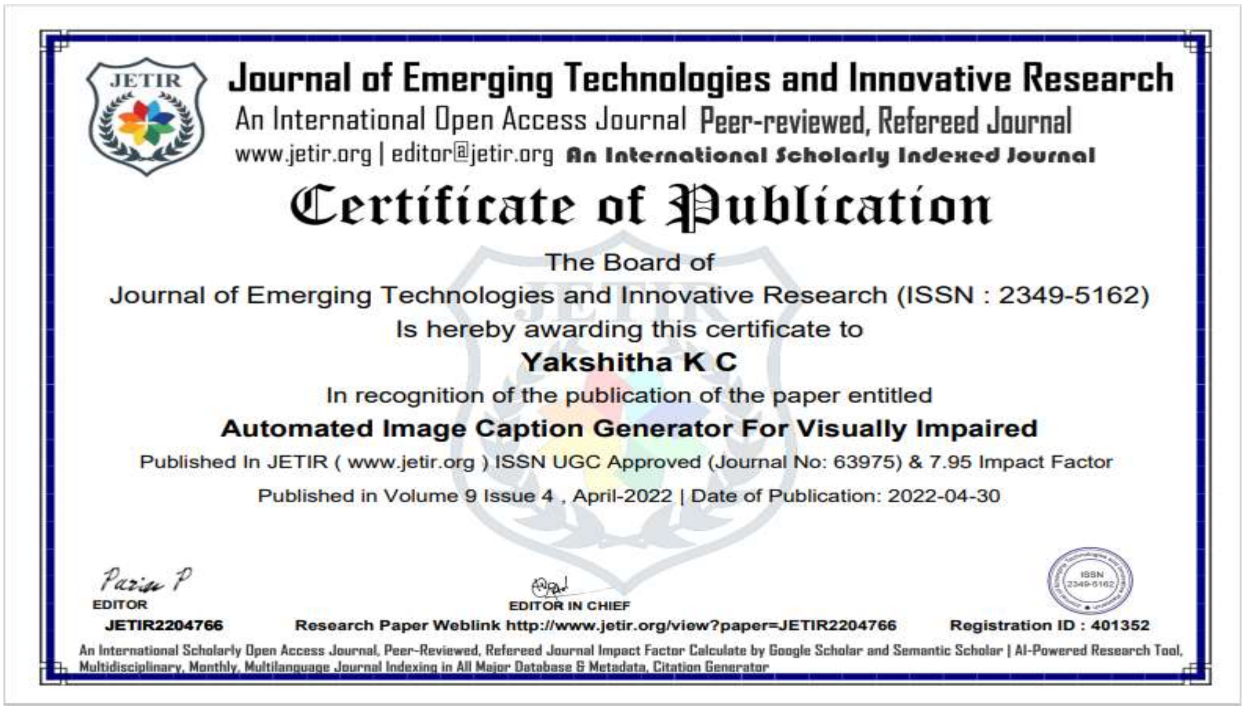
EDITOR

EDITOR IN CHIEF

ISSN 2349-5162

JETIR2204766 Research Paper Weblink http://www.jetir.org/view?paper=JETIR2204766 Registration ID : 401352

An International Scholarly Open Access Journal, Peer-Reviewed, Refereed Journal Impact Factor Calculate by Google Scholar and Semantic Scholar | AI-Powered Research Tool, Multidisciplinary, Monthly, Multilanguage Journal Indexing in All Major Database & Metadata, Citation Generator



Journal of Emerging Technologies and Innovative Research

An International Open Access Journal Peer-reviewed, Refereed Journal

www.jetir.org | editor@jetir.org An International Scholarly Indexed Journal

Certificate of Publication

The Board of

Journal of Emerging Technologies and Innovative Research (ISSN : 2349-5162)

Is hereby awarding this certificate to

Yakshitha K C

In recognition of the publication of the paper entitled

Automated Image Caption Generator For Visually Impaired

Published In JETIR (www.jetir.org) ISSN UGC Approved (Journal No: 63975) & 7.95 Impact Factor

Published in Volume 9 Issue 4 , April-2022 | Date of Publication: 2022-04-30

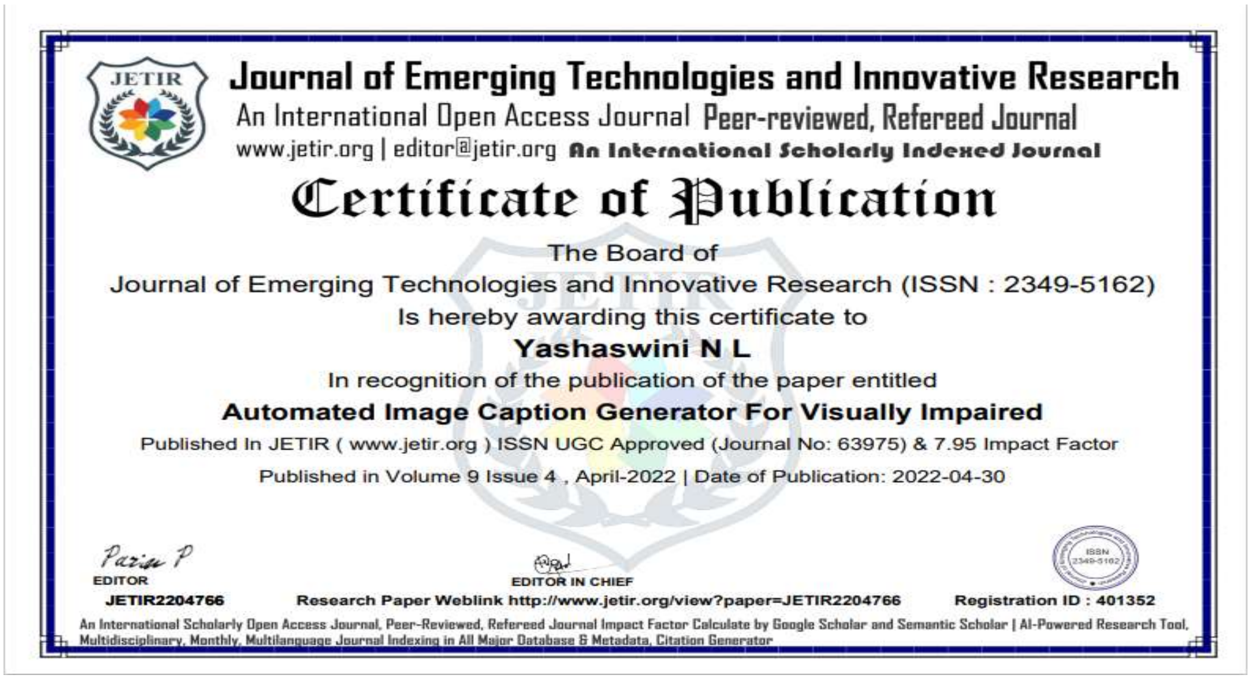
Parisa P
EDITOR

EDITOR IN CHIEF

ISSN 2349-5162

JETIR2204766 Research Paper Weblink http://www.jetir.org/view?paper=JETIR2204766 Registration ID : 401352

An International Scholarly Open Access Journal, Peer-Reviewed, Refereed Journal Impact Factor Calculate by Google Scholar and Semantic Scholar | AI-Powered Research Tool, Multidisciplinary, Monthly, Multilanguage Journal Indexing in All Major Database & Metadata, Citation Generator



Journal of Emerging Technologies and Innovative Research

An International Open Access Journal Peer-reviewed, Refereed Journal

www.jetir.org | editor@jetir.org An International Scholarly Indexed Journal

Certificate of Publication

The Board of

Journal of Emerging Technologies and Innovative Research (ISSN : 2349-5162)

Is hereby awarding this certificate to

Yashaswini N L

In recognition of the publication of the paper entitled

Automated Image Caption Generator For Visually Impaired

Published In JETIR (www.jetir.org) ISSN UGC Approved (Journal No: 63975) & 7.95 Impact Factor

Published in Volume 9 Issue 4 , April-2022 | Date of Publication: 2022-04-30

Parisa P
EDITOR

EDITOR IN CHIEF

ISSN 2349-5162

JETIR2204766 Research Paper Weblink http://www.jetir.org/view?paper=JETIR2204766 Registration ID : 401352

An International Scholarly Open Access Journal, Peer-Reviewed, Refereed Journal Impact Factor Calculate by Google Scholar and Semantic Scholar | AI-Powered Research Tool, Multidisciplinary, Monthly, Multilanguage Journal Indexing in All Major Database & Metadata, Citation Generator