Eksplorium *p*-ISSN 0854-1418
Volume 46 No. 1, 2025: 1475–1501 *e*-ISSN 2503-426X

ImageEcho: A Lightweight Vision-to-Audio Aid for Real-Time Scene Understanding in the Visually Impaired

[1]A. Anny Leema, [2*]P. Balakrishnan, [3]Sri Varshini S, [4]Srija

[1,2,3,4]School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, Tamil Nadu 632014, India

Corresponding Author: Balakrishnan (balakrishnan.p@vit.ac.in)

Article Received: 22 Feb 2025, Revised: 17 April 2025, Accepted: 05 May 2025

Abstract: Visually impaired individuals face significant challenges in performing day-to-day activities independently, such as reading magazines or newspapers, enjoying natural views and sceneries, or traveling from one place to another, often requiring assistance. In the modern digital era, even basic information like bank statements is delivered via mobile text messages, which are not easily accessible to those with visual impairments. Furthermore, the quick development in social media has given rise to a sea of image-based content. The visually impaired users, therefore, find it more to have a clear visual context. The possible solution of this paper is a technical framework aimed to allow people with visual impairment to identify images and understand the text contained in it, besides their being taken during the navigation or exploration of the surroundings. The approach taken in the system involves the application of deep learning through the combination of Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and transfer learning to create a textual identification of the image from the sensors. Then, the text is converted to speech using a TTS API, so users can immediately hear an audio summary of the image. The very heart of the mechanism consists in the condition that the image-to-text conversion is well ended.To ensure effective scene classification, the model will be trained on a comprehensive dataset containing both indoor and outdoor images. Designed with minimal computational overhead, the proposed system can be implemented on existing smartphones without the need for additional hardware. Users can simply use their smartphone's camera to capture images automatically during travel, which will then be processed and classified to provide near-instant audio feedback. This system has the potential to significantly enhance the independence and quality of life for visually impaired individuals.

Keywords: significantly, computational, minimal, visually

1. INTRODUCTION

In recent research by WHO (world health organization) around 290 million people are affected by visual impairment. Among those 290 million, 35 million people are completely blind. They face a lot of difficulty especially when travelling from place to place. Loss of eyesight is primarily caused by persistent eye conditions worldwide. 81% of the 253 million individuals with various degrees of vision impairment are 50 or older. As the elderly population grows, more people will be at risk of developing chronic eye illnesses that could impede their eyesight. Additionally, refractive errors are thought to be the source of partial or total vision loss in 12 million infants under the age of 15. To improve productivity and lessen disability, the 1.4 million children who have irreversible blindness need access to eye therapy programs. Blindness, which is described as an absence of visual awareness and can vary from partial to total loss of vision, can occur for a variety of causes. The following are some typical reasons for blindness: Age-related macular degeneration (AMD) is a gradual deterioration of the macula, the area in the center of the eye that is responsible for fine detail perception. AMD

is a major cause of blindness and is most frequently seen in older people. A collection of eye disorders known as glaucoma affect the optic nerve and are frequently brought on by a rise in intraocular pressure. This can eventually result in blindness and a gradual loss of eyesight. A cataract is an ocular lens clouding that can impair eyesight and, in extreme instances, result in blindness. Diabetic retinopathy: If ignored, this consequence of diabetes damages the retina's blood vessels, eventually resulting in blindness. If left unchecked, some diseases, including leprosy, trachoma, and onchocerciasis (river blindness), can result in blindness. Trauma: Head or eye injuries can harm the optic nerve or retina, resulting in loss of eyesight or blindness. Some inherited conditions, like retinitis pigmentosa, can lead to gradual vision loss and eventual blindness. A deficiency in some vitamins, such as vitamin A, can cause cataracts and visual loss, also called as malnutrition.

1.1 OBJECTIVE

Among the most serious issues that physically impaired people experience are movement and navigation. In the past ten years, there have been numerous developments in computer vision and few suggested guidance systems. But many of them need expensive, bulky, or uncommonly accessible tools as an aid for them or a network link to a potent remote server. Due to its portability and cheap expense, the white cane is the most widely used and basic instrument for spotting obstacles till date. Braille method and the cane have been the two most crucial tools for the blind's guidance for more than 150 years. The fundamental requirements have not altered, and efforts to create new artificial systems to replace sight have typically fallen short of the same main goals represented by braille and a cane. The first is to give a blind person a way to view regular printed material, and the second is to help him navigate his surroundings securely and effectively. In light of the present situation, we can envision a system that helps blind or visually impaired individuals and runs entirely on a smartphone, requiring no additional devices or connections. It could be used in conjunction with the white cane to take photos of the route a person is travelling through and to alert them to possible hazards before they come within range of the white cane through auditory and/or vibration.

1.2 MOTIVATION

Since World War II, numerous technical assistive devices have been developed in an effort to handle these issues and enable blind and visually impaired people to move around independently. However, the blind community does not generally use or embrace these instruments. Poor customer fulfilment of requirements such as usability, affordability, and user experience are the factors that prevent the broad use of technological mobility devices. In order to encourage the visually impaired to travel freely, this paper aims to reveal their needs in terms of their daily movement duties and to suggest a design for a workable smart aid that can meet those needs in addition to the features offered by conventional long white canes. Earlier the researchers developed different ideologies for creating a better aid. The most important technologies used to create improved gadgets are those based on optical/vision systems and

ultrasonic instruments. Although there are many different devices on the market and technical development has advanced to a high degree of stability, ultrasonic systems still have drawbacks. To be more specific, when smooth surfaces need to be spotted, the greatest detectable range is constrained. This is made worse if the angle of contact of the ultrasonic wave is tiny. Additionally, the comparatively big radiation beam makes it challenging to correctly identify tiny apertures. (e.g. doors, windows). Also training is more extensive and expensive when compared to any other models. Under typical working circumstances, ETAs (electronic travel aids) built on optical technologies have very strong object detection capabilities. They frequently use sensors, and sophisticated algorithms are used to provide target identification. The system performance can be severely limited by the high susceptibility to ambient light, particularly in outdoor settings, which is one of the major disadvantages. The heavy reliance on the optical characteristics of barriers and the large dimensions of the gadgets are additional problems. Later the researchers developed a new form of ETA using electromagnetic waves. The fact that electromagnets heat up quickly and waste a lot of electrical energy as a result is one of their drawbacks. The steady magnetic field must be maintained for an uninterrupted power source. Moreover, those electromagnetic waves cannot penetrate through fog or thick clouds and other obstacles. Hence the thought of creating a new aid irrespective of all the above disadvantages and barriers led to the usage of Convolutional neural network (CNN) to train the model and classifying the object and finally it generates a caption or text description of the image that is being captured. This technology does not need any additional device to be carried. These days smartphone has become a necessity rather than being just a device for communicating. Hence the idea of building and training a model using CNN using simple deep learning algorithms was developed, which can be deployed in our easily accessible smartphones. This aid is not only user friendly but also it requires no extra device to be connected and it is just built in our smartphones.

1.3 BACKGROUND AND RELATED WORKS

Automatic generation of written descriptions for a picture's content is one of the tasks in computer vision and natural language processing which goes by the name of image captioning. This task actually means breaking down the visual elements of an image and translating them into English. The inception of image captioning models can be traced to the early research on object identification in images, which focused on retrieval and recognition of objects within a figure [1]. A series of image segmentation techniques and object recognition approaches derived from the effort to represent the spatial relationships among objects in a picture [1]. Since deep learning techniques became popular and large datasets of labeled images were available, researchers have been exploring the use of neural networks to automatically generate text descriptions of images [1]. One of the methods for image captioning at the initial stage was using a Convolutional Neural Network (CNN) to extract visual features from an image followed by a Recurrent Neural Network (RNN) to generate a textual summary [1]. Over the years, numerous architectures apart for picture captioning have been proposed by scholars among them being Attention-based models, Transformer-based models, and Vision-Language Pre-trained models such as ViT, DALL-E, and CLIP [1][21]. These models are benchmarks in

the performance of the image-reasoning task and they are being used in the achievement of various functions such as image and video search, driverless vehicles, and accessibility assistive devices. The Attention mechanism is a feature that has increased the model's performance and enables the decoder to be selective regarding the various parts of the picture as it not only crates the words of the description but also concentrates on the corresponding image parts at the same time. This is done by adding to the vector of features that the decoder is creating a number of weights that represent the attention of the parts that are focused on by using the feature vector and the attention weights the decoder can generate captions word by word [1]. In the training phase, a lot of image-caption pairs are used to train the model, thus, the attention weights are learned and added to the decoder's feature vector. The process of training is two-fold: the encoder learns to provide an informative description of the picture, while the captioning process of the decoder is taught to generate descriptions with correct grammar [1]. When the training phase is completed, the model has been able to create the best subtitle for a new image, to achieve this, it is required to pass the image through the encoder to produce the feature vector and then to call the decoder to create the caption, step by step, by using the feature vector and the attention weights. The model is only capable of generating subtitles for pictures after it has been trained. So, any new picture can be run through a trained encoder to come up with the required feature vector. The vector is then to bring out the words of the caption, which the decoder is sensitive to. The attention weights can be used to create a word, which is the outcome, and each word is started by the feature vector itself and a detailed explanation of the attention weights [1].Of all the parts and the operations that take place, a Text-to-Speech (TTS) API can be said to be a highly convoluted system.The preparation of the input text involves the removal of any format, punctuation, or special symbol characters. Apart from that, the text mostly is the part of the "normalized" stage for speech, which includes translating symbols, codes, and abbreviations into their spoken forms. To find the right prosody, the processed text is looked through, bearing in mind details like intonation, stress, and tempo. This observation is meant to verify that the output is audible and appears natural. The TTS API then chooses the right voice output and prosody to reproduce the sounds that the latter is capable of correctly. In general, pre-synthesized audio that comes from different phonemes and phrases which are then tied and processed together is the technique used by this means for channeling the result. The TTS API has come up with a method of producing spoken word audio output that can be given in multiple forms i.e. WAV or MP3 [9].

Deep learning has made it popular to combine computer vision and natural language processing. One example is teaching a machine to understand a picture by describing it in one or more sentences. Aside from recognizing theobject and the scene in the picture, the relevant description and high-levelpicture semantics need the ability not only to analyze the state, attributes,and the connection of these objects, but also to generate the process [1].Nevertheless, quite a few have been those who have made impressive progress inthe field of the picture captioning.A deep neural network-basedreinforcement paradigm has also been proposed, using CNN as encoder and RNN asdecoder. Yet, they point out the challenges such as the indicated mismatch andexposure bias - where the measurement of the loss and the evaluation/ trainingmetrics and the error accumulation lead to the degradation of the performanceof the model [1]. To cover these gaps and to make sure that the visualizationof the model and value-

policy decision networks are the feature, CNN-CNN modelsand reinforcement learning-based avenues have been brought in [1].

Using a different method, anattention-based encoder-decoder model together with image classification and objectdetection models has been employed in the YOLOv4 and Xception CNN architectures. The XceptionCNN architecture is used to extract spatially informative features and this is in turn theroot of the improvement of the caption prediction task through the imitation of humanrecognition of images [21]. The GRU-baseddecoder is used to receive the combined feature information for a more precisedescription with the aid of the attention mechanisms [21].In the case ofthose who are visually and hearing impaired, these systems have been developed for change andspeech production. For example, object detection was provided by YOLOv5, while object nameswere further converted into spoken words using Python TTS libraries like gTTS and pyttsx3.The assistive tools are created to assist blind people in navigating their surroundingsindependently [9]. The solution employs the real-time camera mounted on a head and announcesthe names of the objects to its user, thus enhancing the mobility of the user and makinghim/her more aware [9].An importantmilestone in attention-based captioning is the Show, Attend and Tell model, which put forthtwo kinds of attention: one soft (trainable and learned through gradient descent) and onehard (probabilistic, learned through the REINFORCE algorithm). The use of these attentionzone-driven methods lets the model to attend to individual image regions while it still is inthe process of generating each token of the output sequence . The system trained with thedata taken from the large datasets like Flickr8k, Flickr30k, and MS COCO datasets shows asignificant enhancement in the accuracy of the generated captions. The model was furthermodified to be more efficient in terms of computation by dividing the caption lengths intovarious pools and training on these pools, thus shortening the process of reachingconvergence . Quality of the output was measured with several popular evaluation metricslike BLEU, METEOR, and CIDEr .

2. VISUAL AND AUDIO COLLECTION

Our aim is to develop an image to audio converter that accepts an image as input and outputs an audio file. The goal of the project is to create an algorithm that can analyse an image's visual elements and translate them into a sound signal. The research will start by gathering a collection of photos and the audio recordings that go with them. A wide range of visual characteristics, including color, texture, form, and size, will be represented in the photographs. The audio files will contain a wide variety of sounds, including music, spoken phrases, and ambient noises. The mapping between visuals and sounds will then be learned using this dataset to train a deep learning model. Convolutional neural networks (CNNs) and recurrent neural networks (RNNs), which are suitable for analysing picture and audio input respectively, will be combined in the model. After the model has been trained, audio will be produced for fresh pictures. An picture will be fed into the model as part of the process, and an audio file will be produced in response. To enhance the sound quality, the resulting audio can be further polished using audio processing techniques like filtering and equalization. A few years ago, it seemed nearly impossible for a machine to create a text description for an image. However, with the advancement of artificial intelligence and deep learning algorithms, the accessibility of pertinent datasets, and AI models, it is now feasible to create

a relevant text description for an image. Many data annotation companies are making billions of dollars from caption creation, which is also expanding globally. In this paper, we will describe how to create an annotation tool that can use datasets to produce descriptions for images that are extremely pertinent. For the same, it is necessary to have a basic understanding of two deep learning methods, LSTM (a form of recurrent neural network) and Convolutional Neural Networks (CNN). For labelling a picture with English terms we will be using datasets that were made available for model training. The CNN model known as Xception is trained using the Imagenet dataset. The gathering of picture features is handled by Xception. The LSTM model will receive these extracted characteristics and use them to create the picture caption, which will then be turned into an audio file. In this paper, we intend to create a model with high accuracy of generating a text description of the image captured by a person and using that description we produce an audio that can help many people who are affected by visual impairments either temporarily or permanently. In an attempt to help those people suffering from visual impairment, this model should deliver highly accurate description of the image and an clear audio using simple deep learning algorithms and attention mechanisms. Table 1 displays the techniques, qualities, and drawbacks of assistive systems created to help the visually impaired, and covers the views of the users and the technologies employed through the survey-based method in those hardware/software-based solutions as well.

Table 1: Comparative Review of Existing Assistive Technologies for the Visually Impaired

Ref No.	Methodology	Significant Features	Drawbacks
[21]	Survey of current aids like GPS, ETAs, training	Overview of available assistive technologies for visually impaired	Lacks implementation details or technical depth
[22]	Smart system for daily activity assistance	Automation in daily tasks using sensors	No details on scalability or real-time feedback performance
[23]	Virtual eye concept using hardware-software integration	Does not require guardian, hardware focused	May not generalize well to all impairment levels
[24]	Low-cost assistive system for blind navigation	Economical and user-friendly device	Limited real-world testing described
[25]	Image processing for wearable visual aid	Wearable system, object detection	Lacks clarity on processing latency
[26]	Electronic Travel Aid using sensors	Sensor-based mobility aid	Hardware dependency and cost
[27]	Survey of artificial vision and prosthesis	Discusses surgical visual implants	Not applicable for real-time image-to-audio use

[28]	Voice-guided object and face recognition	Voice-over system, facial recognition	Privacy issues and recognition accuracy not discussed
[29]	Smart stick using DL, NLP, and Raspberry Pi	AI-enhanced mobility aid	Hardware dependency and user learning curve
[30]	Audio transformation of real-world objects	Object detection and TTS for feedback	Needs high accuracy for effective guidance

3. DESIGN APPROACH AND DETAILS

To help the blind people in this CNN based architecture, the first step is the camera which captures the real-time video or images. Then the system identifies the patterns of the objects in the scene and extracts the features. During this process, the CNN model delineates the key characteristics like edges, textures, and shapesof the image. Next, the obtained characteristics are used to perform the object detection, where the model will try to recognize and categorize theobject. In case additional information is required, the system fuses the data retrieved from the cloud storage for more accuracy. Following through, the discovered object will get matched with the universe of things, where the model will go further to offer the feedback in the positive case, that is, the system proceeds to explain the object by verbal form to the user. In the negative case, the system is then required to start the process anew and either capture a new image or request the user to help. Real-time cloud storage integration not only facilitates instant updates but also improves the recognition capability of the system, thereby providing visually impaired users with more effective assistance. The overall process for pattern detection and feedback generation is illustrated in Figure 1.

Let the input image or video frame at time t be:

$$I_t \in \mathbb{R}^{H \times W \times C}$$

Where:

- $H =$ height, $W =$ width, $C =$ number of channels (e.g., 3 for RGB)

Feature Extraction using CNN:

Let $f_l(\cdot)$ be the convolutional function at layer l_r and F_l be the feature map:

$$F_l = f_l(F_{l-1}) = \sigma(W_l * F_{l-1} + b_l)$$

Where:

- * denotes convolution
- W_l, b_l are weights and biases
- σ is an activation function (e.g., ReLU)
- $F_0 = I_t$

This process continues through multiple layers to extract features like edges, textures, and shapes.

Object Detection :

Let $\hat{y} \in \{0,1\}^K$ be the probability distribution over K object classes:

$$\hat{y} = \text{softmax}(W_o \cdot F_{\text{final}} + b_o)$$

Where:

- W_o, b_o : output layer weights and biases
- F_{final} : final feature vector after flattening/pooling

Object is recognized if:

$$\arg\max(\hat{y}) = k^* \text{ and } \hat{y}_{k^*} \geq \tau$$

Where τ is a confidence threshold.

Cloud Fusion (Optional Enhancement):

If confidence $\hat{y}_{k^*} < \tau$, retrieve additional context C_{cloud} from cloud storage:

$$F' = \phi(F_{\text{final}}, C_{\text{cloud}})$$

Where ϕ is a fusion function (e.g., attention-based weighting or concatenation), producing enhanced features F' for re-classification.

Universe Matching and Feedback Generation

Let $U = \{u_1, u_2, \ldots, u_N\}$ be a universe of known objects. If detected object o matches an element in U, then:

$$\exists u_i \in U : \text{sim}(o, u_i) \geq \delta$$

Where:

- $\text{sim}(\cdot)$: similarity function (e.g., cosine similarity or feature embedding distance)
- δ : similarity threshold

If matched:

- Generate verbal output using Text-to-Speech (TTS):

$$\text{Audio} = \text{TTS}(u_i.\text{description})$$

Else:

- Restart process or prompt user for help.

Real-Time Feedback Loop

Let $\mathcal{S}$ be the system state. Then the loop condition is:

$$\mathcal{S}_{t+1} = \begin{cases} \text{TTS}(u_i), & \text{if match found} \\ \text{Capture new frame or ask user,} & \text{otherwise} \end{cases}$$

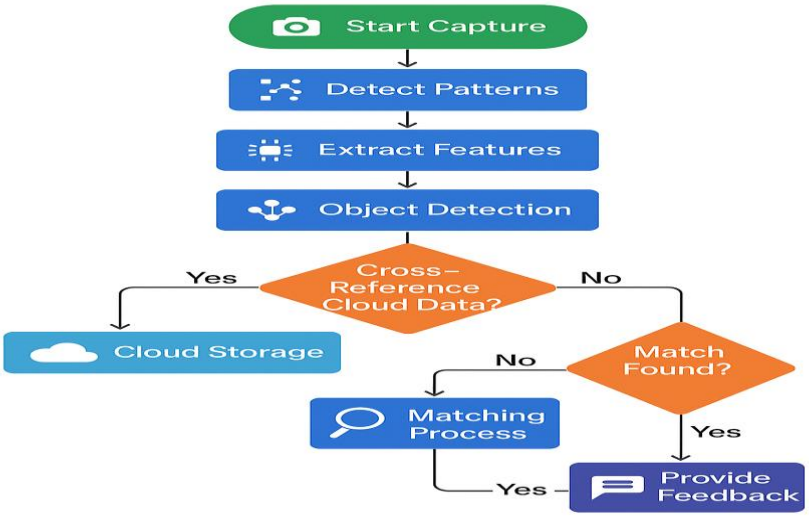

Figure 1: Image Analysis and Feedback

COCO DATASET

COCO Dataset 2017-The MS COCO (Microsoft Common Objects in Context) dataset is a large-scale object detection, segmentation, key-point detection, and captioning dataset. The dataset consists of 328K images. Splits: The first version of MS COCO dataset was released in 2014. It contains 164K images split into training (83K), validation (41K) and test (41K) sets. In 2015 additional test set of 81K images was released, including all the previous test images and 40K new images. Based on community feedback, in 2017 then training/validation split was changed from 83K/41K to 118K/5K. The new split uses the same images and annotations. The 2017 test set is a subset of 41K images of the 2015 test set. Additionally, the 2017 release contains a new unannotated dataset of 123K images. Figure 2 illustrates the sample images from the dataset.

Figure 2: sample Images from the dataset.

3.1 PROPOSED DEEP LEARNING TECHNIQUES

3.1.1 CNN:

An architecture named convolutional neural networks (CNN)is a key method for dealing with and analyzing imagedata. The inspiration for the architecture comes from the structure of the human andanimal visual cortex cells, which are capable of recognizing the visual features at different levels of abstraction, starting from simple ones to the most complicated. The first layers of a CNN are convolutional layers. These layers consist of a collection of learned filters that are applied to the input picture to extract its important characteristics. Then, we add one or more fully connected layers for image classification. The convolutional layers use the filters, a process that is similar to the sliding window technique for thewhole picture, and the results are then processed by a nonlinear activationfunction like rectified linear unit(ReLU) as a rule. CNNs have consistently outperformed other methods in a variety of computer vision tasks that requiredeep learning, such as those for picture classification, object recognition,semantic segmentation, and text generation. Specifically, CNNs are useful infacial recognition, medical imaging, and self-driving cars. The ability of CNNs to capture hierarchical pictorialfeatures through the layers of the network, which allows them to recognize both local patterns and global ones in the picture, is another main advantage ofCNNs. They are also robust to changes inlighting, size, and orientation and can processvery large data sets. A convolutional neural

network (CNN) can be usedas an encoder in an image processingworkflow. The use of the word "encoder" in this case refers to a neuralnetwork that takes an image as input and converts it into alower-dimensional representation without losing vital information while maintaining theoriginal image. During the encoding stage of a CNN, multiple convolutional layersare typically utilized to extract the high-level features from the inputimage. After those operations, the image features are represented in the lower dimension by flattening the spatial dimensions of thefeatures and feeding them into the fully connected layers, an operation that is also called dimensionalityreduction. Having the encoder in a lower-dimensional representation is benefitial for a lot of tasks such as picture classification, object recognition and image generation. An example of this is where the encoder and decoder are both part of theimage generation pipeline, where they can be used to reconstruct the original picture from the lower-dimensional representation. These architectures arereferred to as autoencoders.

LSTM:

LSTM is a type of RNN architecture used to discover long-term relationships in sequential data such as text, voice, or time series data. While Traditional RNNs have issues with capturing long-term dependencies resulting in the vanishing or exploding gradient problems, LSTM networks employ a memory cell and a sequence of gating mechanisms to capture and forget information in a smart way. Each memory cell that comprises the LSTM structure contains a hidden state vector and a cell state vector. The network's long-term memory is encoded in the cell state vector, which is continuously updated by the gating mechanisms through a series of actions. These mechanisms control the flow of information into and out of the memory cell and consist of an input gate, a forget gate and an output gate. The input gate will let the data from the current state of the cell, which will be added to the memory cell, whereas the forget gate selects which data from the previous state of the cell should be removed. After that, the output gate of the memory cell is activated to selectively pass the bits of the current cell state to the next layer or the network's final output. Natural language tasks like machine translation, emotion analysis, and voice recognition are among the numerous applications of LSTM networks that have seen high attention. Users have also successfully utilized the networks to predict future events in various time series, such as market prices, weather trends, and transportation movements. The fact that LSTM networks can be used for recognizing long-term dependencies in sequential data makes them suitable for tasks where the recall of past information or occurrences is essential. They are also robust to noise and anomalies in the incoming data while they can handle sequences of variable length. The choosing of the training target plays an important role among the LSTMs used in the context of the picture captioning. The model is generally trained using a loss function based on cross-entropy that takes the caption and the real caption as the input and calculates the difference between them. The quality of the resulting captions, however, has been studied in recent literature using the latest loss functions such as reinforcement-learning-based algorithms. The steps involved in extracting text from an image and converting it to audio are illustrated in Figure 3

p-ISSN 0854-1418
e-ISSN 2503-426X

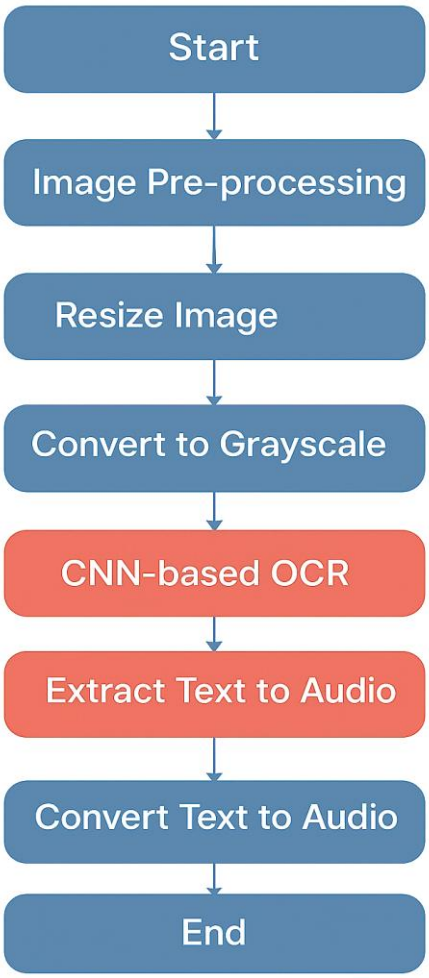

Figure 3: Text Extraction and Audio Conversion Pipeline

3.2 DATA COLLECTION AND MODEL PREPARATION

To answer research questions, test hypotheses, and assess results, one must acquire and measure information on relevant factors in a predetermined, systematic manner. This process is known as data collection. These basic stages in the data collection procedure are listed as follows. Determine the study issue, where you need to establish your study question, what you want to know, and the sort of data you need to address it. Create a strategy for data gathering that includes determining the sources of the data, the categories of data required, the techniques for collecting the data, and any ethical issues that must be taken into account. Gathering the data to conduct questionnaires, interviews, tests and other data gathering techniques as part of the data collection strategy. Validate the data: Check the data for uniformity, correctness, and completeness to make sure it is trustworthy. Utilize statistical analysis and other techniques to examine the data and find the answers to your research queries. Interpreting the outcomes: Make inferences based on the data analysis, then present your results. Security and privacy of information must be kept throughout the data gathering process in order to ensure that data is gathered responsibly and with the user's full permission. To avoid data loss and to

enable future use of the data, proper documentation and filing of the data must also be assured. The propose dataset is collected from the Kaggle website for the purpose of this model building.

3.3 DATA CLEANING

Text data cleaning in Python usually entails a number of stages, including importing the required resources and library packages, uploading the text data that must be cleaned, converting the text from uppercase to lower case if any and removing any weblinks, non-alphabetical characters, extra blank spaces or white spaces and stopwords if present, lemmatizing the text to its base form and atlast saving the cleaned text data in a file for further use. Data cleaning is an important process as the data we collect from different resources may contain many anomalies.

3.4 DATA PRE-PROCESSING

Tokenize the subtitles, for example, by splitting them using spaces and other criteria, and then create the tokenized vectors. This provides us with a lexicon of every distinct term in the data. To save your recollection, limit your vocabulary to no more than 5,000 terms. All other terms should be changed to the symbol for unknowns, "UNK." Map words to the index and the index to the words. All segments should be extended to match the duration of the longest one. There are five captions, each of which is a varying length, for each unique image. They cannot therefore be transmitted straight to the Decoder. Before moving on, padding must be used to trim all subtitles to a specific length. To ensure that all of the samples have the same standard duration, padding is a procedure that is applied at the beginning or conclusion of a sequence. The encoder must begin with an input that is not a bloated input in order to generate text. As a result, we will apply the padding at the conclusion of our procedure. (of the caption sequence). The remaining values will all be changed to "True," while all padded values will be covered and set to "False." Padded, or False, numbers are put to 0 while all True values are given the value 1. Padding may increase the model's chance of punishment. We use disguising to fix the issue, which will reduce all additional fines back to zero. In deep learning techniques that use image analysis, image pre-processing is a crucial stage. By removing noise, adjusting lighting, and fixing distortions, image pre-processing primarily aims to improve a picture's clarity and make it more analytically appropriate. Some of the typical methods for pre- processing images are as follows. Reducing the size and quality of the picture to make it appropriate for further research is known as image resizing and scaling. When a picture is cropped, unwanted areas that do not hold important details are removed is called image cropping. Image filtration is the process of using techniques like median filtering or Gaussian smoothing, different filters are applied to the image to reduce noise and improve borders. Image normalization consists of modifying the image's luminance levels to make it simpler to compare to other images. Color space conversion is to make picture analysis simpler, the image is changed from one color space to another, such as normal color to grayscale. To enable further research, image segmentation entails separating the image into parts based on their characteristics, such as color or structure. Picture augmentation entails

creating extra training data from the initial picture by performing transformations like rotation, scaling, and flipping. To get the desired pre-processing outcomes, these methods are frequently combined. The precise pre-processing methods employed rely on the application at hand as well as the properties of the image data. Pre-process the images and create a list to store all the pre- processed images. Split the data into training and testing sets. Pre-processing is necessary for both text and image data as the machine software cannot understand complex word structures and image contents.

3.5 VOCABULORY TOKENIZATION

Create a method that associates an image's location with its characteristic. Develop a building function to generate the train and test datasets, then use the previously developed function to change the dataset. To derive the features from the models, we will now use a pre-trained model known as Xception that has already been trained on a sizable amount of data. To identify the images, Xception was trained on an imagenet dataset with 1000 distinct classes. We can simply integrate this model using keras .applications. To incorporate the Xception model with our model, we must make a few adjustments to it. In order to extract the 2048 feature vectors and fit the 299*299*3 picture dimensions required by the xception model, the final classification layer must be deleted. Pre trained Image net weights for Inception net V3 must be imported. Use the final layer of the pre-trained model to retrieve the characteristics of the pictures to prevent the RAM from running out. The training will take longer to compute if this is included. The result of this layer has the following shape: 8x8x2048. The form of each picture in the train and test dataset should be (batch_size, 8*8, 2048), so use a function to retrieve the features of each image. Once the extracted features are saved in a dataset, they must be reshaped into standard image size.

Machines cannot understand complicated English terms, so they require a straightforward numerical depiction in order to handle model data. For this reason, we assign each term in the vocabulary a different, unique index number. The Keras library includes a built-in tokenizer method that generates characters from our lexicon. Tokenization is the method of dividing a sentence or a fragment of text into smaller pieces called tokens in natural language processing. Tokenization is a crucial preprocessing phase that is frequently applied to a variety of NLP tasks, including emotion analysis, text categorization, and machine translation. Tokenization is used to break down the captions into a series of distinct tokens that can be passed into the LSTM encoder when creating a picture captioning model. The phrases in the captions are represented by tokens, and each token has a specific index in a lexicon. Here, we are using the Keras Tokenizer class to tokenize the image description generated. The vocabulary's highest word count will be 10,000, and the tokenizer will be integrated into the subtitles. The subtitles are then transformed into word index sequences using the tokenizer, and we obtain the word index mapping from the tokenizer. A dictionary called the word index mapping assigns each term in the vocabulary a specific number. The term that appears most frequently in the subtitles is given index 1, followed by the word that appears next most frequently with index 2, and so forth. We can use the tokenized subtitles to teach the picture captioning model after they have been tokenized. The model will be trained to anticipate the

following word in the caption succession using the picture features and the tokenized image descriptions as inputs during training. A probability distribution over the lexicon representing the chance of each word appearing as the subsequent word in the caption will be the model's output. In order to create the following word and continue the caption series, we can select from this distribution. Tokenization is an important aspect in multilingual analysis, where the text data contains more than 5 languages which might be a difficult task without tokenization. Tokenization is a crucial stage in information retrieval systems, which are used to search and retrieve data from enormous document collections. Information retrieval systems can more accurately match user queries to pertinent documents by decomposing text into individual words or phrases.

3.6 DATA VISUALIZATION

After tokenizing the vocabulary, we will check for the top 30 words that occur frequently in text descriptions that has been pre-processed and saved in a file and visualize the data by means of a bar chart. The below figure 4 shows the graph of frequently used words. And the figure 5 is a visualization in the form of a word cloud of the most frequently used words. Data visualization is crucial because it helps us comprehend and convey complicated information more effectively. Patterns and trends are frequently simpler to spot and decipher when displayed graphically rather than in tabular or textual form. Complex information can be made simpler through the use of data graphics, which makes it simpler to comprehend and analyze. Data can be divided into smaller, more easier chunks that are simpler to process and understand when it is displayed visually. Heat maps, which may be used to draw attention to an image's key elements, and bar charts, which can be used to evaluate the effectiveness of various models or methods, are two more forms of visualizations that can be used in image captioning. The constraints of a model can also be better understood with the use of visualizations. To illustrate the different mistakes the model makes, such as misidentifying items or mixing together related phrases, a confusion matrix might be utilized. By adding more training data or fine-tuning the model architecture, this can assist pinpoint areas where the model needs to be improved.

4 MODEL BUILDING AND TRAINING

In model building we will first set the parameter, build the encoder using CNN, then build the attention model and finally build the decoder which is LSTM. We will create a multimodal recurrent neural network that creatively combines the CNN and RNN model to solve and generate text description for image being captured. The usual structure of an encoder CNN comprises of a number of convolutional layers, a pooling layer, and possibly more convolutional and pooling layers. Each of the filters used by the convolutional layers identifies a different aspect of the incoming picture, such as edges, curves, or textures. The feature maps are compressed in size thanks to the pooling layers, which also increase the network's resistance to minute changes in the input picture. A collection of feature maps encoding the key elements of the input picture are produced by the final convolutional or pooling layer. To carry out a particular job, such as picture categorization, object recognition, or image

captioning, these feature maps can be passed into a classifier or another neural network design. The LSTM model is a unique form of the RNN model that can resolve the issues because the gradient disappearance and restricted memory problem of ordinary RNN. Three additional control units (cells), input, output, and forget gates, are added. The cells in the model will evaluate the information as it is introduced. Non-conforming information is lost, while information that complies with the guidelines will be preserved. The lengthy sequence dependency issue in the neural network will be resolved using this concept. The encoder portion of this model accepts a picture as input, and the decoder portion uses LSTM networks to produce the explanations that go with it. The model effectively resolves the issue of vectoring phrases in natural English. Using computers to handle natural language is extremely important because it advances computer processing beyond the level of straightforward matching to that of semantic comprehension. It is suggested that the computer vision field's attention process be used to encourage word and picture block alignment. So that the produced sentence is more related with people's expression habits, the word sequence generation process and the "attention" transfer process that mimics human vision can be jointly encouraged during the sentence generation process. The attention mechanism adds the whole and spatial information pertaining to the picture to the extraction of the image features, producing a fuller statement description as opposed to storing the entire image as a static vector. Currently, the picture characteristics are regarded as dynamic feature vectors mixed with weights data. The first attention mechanism suggested two types of attention: "soft attention," which involves choosing regions based on various weights, and "hard attention," which involves focusing attention on a specific visual idea. Deep neural networks with an attention focus have produced impressive testing findings. The model generates each word in accordance with the associated area of a picture by using an attention process. The decoder receives a result from the attention model (context vector) in order to forecast the word at that moment. Context vector, the result, is adaptive and changes for each date. It attempts to get around the drawbacks of conventional CNN-RNN based algorithms. Using this, we can send various pertinent portions of the input picture to the RNN at each timestamp as opposed to the entire image as input. This accelerates the algorithm and improves forecast precision. A technique called transfer learning is used for model training. We will be training the model using the features extracted by Xception model. The pre-trained CNN should only be used as a decoder; its weights should not be retrained during training. By putting the trainable property of all the CNN layers to False, we can freeze them all in order to accomplish this. The image-caption combinations in the training data should each have a different description that is a group of words. Pre-processed data will be used in this step. Back propagation can be used to train the model with an appropriate loss function, such as category cross-entropy or mean square error. To avoid over fitting while training, strategies like attrition and early ending can be employed. With 6000 training pictures, we will create the input and output patterns to train our model. We will set the number of epochs to 6 depending upon the processing speed of the system to train the model. But the larger the dataset is, high accuracy of results can be obtained. The right number of epochs to train the model is determined when the model improves no further. Once the training model does not show any improvement then we can stop the number of epochs there. Finally we will save all the trained models in

the model folder to access later for testing.

4.1 TEXT TO AUDIO:

The process of translating written text into an audio file is known as text to audio. A text to speech (TTS) engine or piece of software can be used for this. The following are some major benefits of text to audio conversion. Accessibility: Text to audio conversion can make written material available to those who struggle to read or have visual impairments. People may listen to the material instead of reading it by turning written text into audio. People who prefer to listen to material while engaging in other activities, such as commuting or exercising, may find text to audio conversion to be more convenient. Additionally, it might be helpful for those who don't have much time to read. Text to audio conversion enables multitasking while reading or listening to material. For instance, people may listen to an audiobook or podcast while cleaning or driving. Text to audio conversion can be helpful for language learning since it enables students to practice speaking and listening to other languages. Text to audio conversion can aid with pronunciation by allowing students to hear words and phrases spoken correctly. All things considered, text to audio conversion may be a beneficial tool for making written material accessible, practical, and useful for many reasons, such as language acquisition or multitasking. By employing GTTS to provide an audio description of the image, image captioning can significantly improve the resultant written description. This can help folks who struggle to read or are visually challenged access the image description more easily. The picture captioning model may be combined with the GTTS tool to provide an audio description for each caption that is produced. Users can better grasp the material by playing the audio explanation concurrently with the image or caption.

4.2 PACKAGES USED

- **pyttsx3-** is a Python text-to-speech conversion library that works offline. It allows customization of voice properties like rate, volume, and voice type.
- **PIL (Pillow)-** library in Python is used for image processing
- **Image**- module allows opening, manipulating, and saving images.
- **IPython.display**- module is used to display rich media outputs like images, audio, and videos in Jupyter Notebooks and interactive environments.
- **Json**- module in Python is used for parsing JSON data (JavaScript Object Notation) and converting Python objects to JSON format.
- **Os**- module in Python provides functions for interacting with the operating system, such as file handling, directory management, and environment variables.
- **Keras** – A high-level neural network API built on TensorFlow, simplifying model creation, training, and optimization.
- **Pandas** – A data manipulation library in Python used for handling structured data like CSV files and SQL tables
- **torchaudio** – Supports audio processing, datasets, and models for deep learning on audio data.

p-ISSN 0854-1418
e-ISSN 2503-426X

- **torchvision** – Provides datasets, models, and image transformations for computer vision
- **torch –** The core PyTorch library for tensor computations and deep learning.
- **NumPy** – A numerical computing library that supports large, multi-dimensional arrays and mathematical operations.
- **Matplotlib** – A visualization library for plotting graphs, used to analyze model performance.

PSUEDO CODE

#implementation of libraries and packages

!pip install tensorflow keras numpy pandas matplotlib tqdm gTTS opencv-python pillow
!pip install pyttsx3
!pip install pandas
!pip install tensorflow keras pillow numpy tqdm matplotlib gtts
!pip install torch torchvision torchaudio

 # Load the document file into memory

1. Check Dataset: Verify if the `coco-2017` folder exists.
2. List Files: If found, display its contents; else, show an error.
3. Load Images: Extract images and labels from the dataset.
4. Preprocess Data: Resize, normalize, and apply augmentations.
5. Prepare CNN Model: Define and compile the CNN architecture.
6. Train Model: Use training data to learn patterns.
7. Make Predictions: Classify images using the trained model.
8. Convert to Speech: Generate audio output for visually impaired users.
9. User Interaction: Enable easy input processing for assistance.

4.3 RESULTS AND DISCUSSION

Figure 4 presents the process of loading and displaying the data set in a Jupyter Notebook through Python and OS libraries**.**

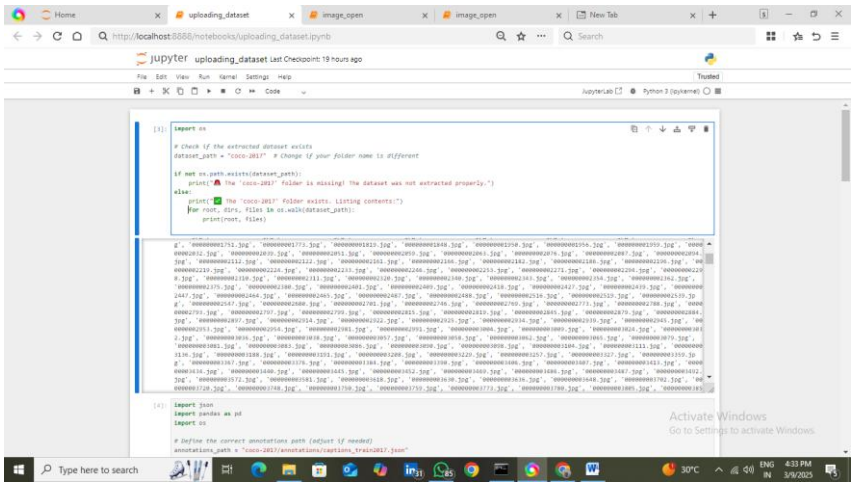

Figure 4 Dataset Loading Code in Jupyter

#Annotations

1.Define Paths: Configure the dataset directory and annotation file paths.
2.Check Existence: Confirm whether the image directory and annotation file exist or not.
3.Load Annotations: Read and load the JSON annotation file.
4.Display Data: Print the number of annotations and a sample record.

 # Load and Clean Captions

1.Initialize Storage: Create a dictionary mapping images to captions.
2.Iterate Over Annotations:

- Get image_id and format the filename
- Get and clean the caption (convert to lower case, strip punctuation).
- Store the cleaned caption in the dictionary.

3.Save Captions: Serialize and save the processed captions using pickle.
4.Confirm Completion: Print a success message.

 # LSTM Model in PyTorch

1.Define Model: Define a class ImageCaptioningModel that extends nn.Module.
2.Initialize Layers:

- Embedding Layer: Transforms words into dense vectors.
- LSTM Layer: Handles sequential text data.
- Fully Connected Layer: Projects LSTM output to vocabulary words.

3.Forward Pass:

- Embed input captions.
- Concatenate image features with embedded captions.
- Pass data through LSTM and fully connected layer.

4.Initialize Model: Initialize vocabulary size and instantiate the model.
5.Print Model Summary: Print the model architecture.

 # Train the Model in PyTorch

1.Define Loss & Optimizer:

- Apply CrossEntropyLoss for classification.
- Apply Adam optimizer to update model weights.

2.Set Training Parameters:

- Set number of epochs (num_epochs)
- Iterate through epochs to train the model.

3.Training Loop:

- Zero gradients (optimizer.zero_grad()).
- Create sample data (random features & captions for now).
- Forward pass (missing in the current code).
- Calculate loss (not implemented in the current code).
- Backward pass & optimization (loss.backward() and optimizer.step()).

- Print progress after each epoch.

#from PIL import Image

1.Load Pretrained Model:

- Utilize efficientnet_b0 that was trained on ImageNet.
- Delete the last classification layer for feature extraction.
- Switch the model into evaluation mode (model.eval()).

2.Specify Image Transformations:

- Resize images to (299, 299).
- Change to Tensor for PyTorch compatibility.
- Standardize pixel values with ImageNet mean & std.

3.Process Images for Feature Extraction:

4.Apply transformation to input images prior to going through the model.

#Model load

1.Set Model Path: Specify the path to the saved model (.pth file).

2.Initialize Model: Instantiate ImageCaptioningModel with the appropriate vocabulary size.

3.Load Model Weights: Call torch.load() to load weights from the saved file.

4.Set to Evaluation Mode: Call model.eval() to turn off training mode.

5.Confirm Success: Print a message indicating that the model loaded successfully.

 # get all images with their captions Check File Existence:

1.Check whether captions_train2017.json exists.

2.Load JSON Data:

- Open and load the JSON file.

3.Extract Metadata:

- Construct a dictionary of image_id → image_name.
- Get image_id and caption pairs.

4.Convert to DataFrame:

- Store images and captions as Pandas DataFrames.
- Both merge on image_id.

5.Save as CSV:

- Write the resulting dataset to ml_dataset.csv.

6.Display Sample Output:

- Print the first few rows for validation
- Open image to read and to check exist in memory

7.Set Image Path: Specify the complete path of the image file.

8.Check File Existence:

- Print True if the file exists, otherwise False.

9.Open and Display Image:

- Use PIL.Image.open() to open the image.
- Call image.show() to show it.

p-ISSN 0854-1418
e-ISSN 2503-426X

Figure 5 displays script that verifies and loads the airplane image through Python, uses PyTorch to transform it for the deep learning model for classification. Such a process implies that the script needs to check the object and that the image is fetched through PIL and, in addition, the preprocessing is performed for tasks such as the classification task.

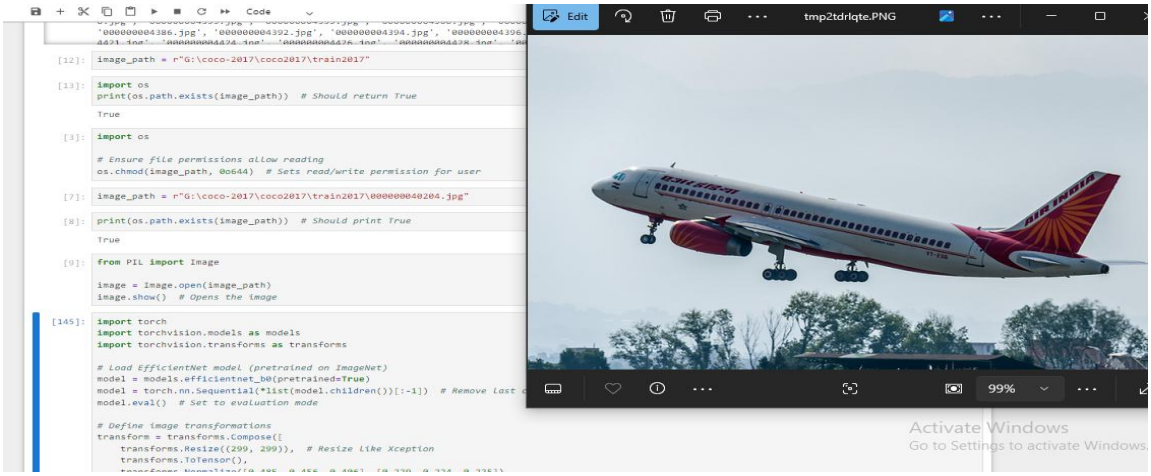

Figure 5 Image Preprocessing for Model Inference

#opening image according user with caption

Step 1: Load Dataset

- Define dataset path (ml_dataset.csv).
- Load CSV file into a Pandas DataFrame (df).

Step 2: Define Image Path

- Define the image directory path (train2017).
- Request user to input an image filename (e.g., 000000203564.jpg).

Step 3: Retrieve Captions

- Retrieve captions from the DataFrame where image_name equals the input filename.
- Return a list of captions (or a default message if no captions are found).

Step 4: Display Image

- Build full image path by concatenating the image directory and file name.
- Verify if the image file exists:

If yes → Open and display the image.

If no → Print error message.

Step 5: Display Captions

- Print all existing captions for the specified image.
- Present captions in a numbered list for easier readability.

p-ISSN 0854-1418
e-ISSN 2503-426X

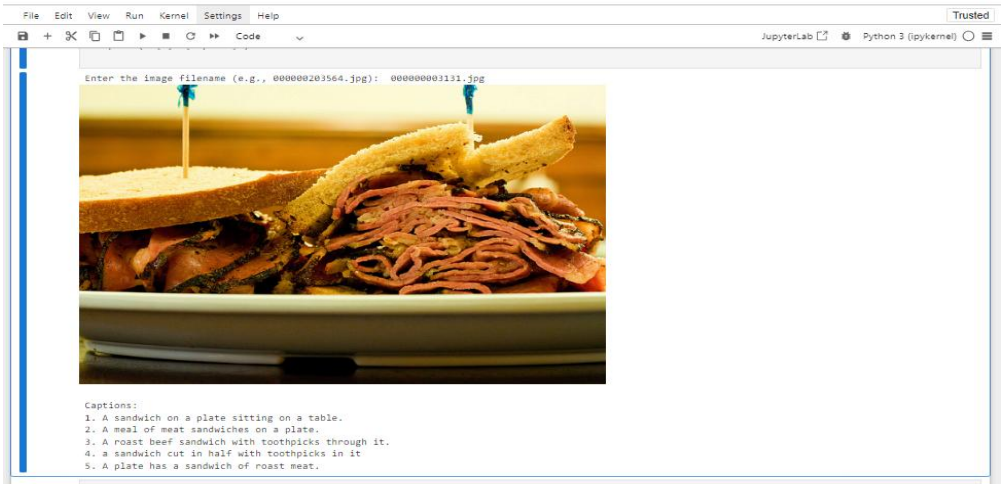

Figure 6a- Image Captioning Output in Jupyter Notebook

 # reading the image caption

Step 1: Initialize Text-to-Speech Engine

- Import the pyttsx3 library.
- Invoke pyttsx3.init() to initialize the engine.
- Modify the speech speed (rate = 150) if necessary.

Step 2: Read Captions

- Iterate through each caption in the list.
- Print the caption (for visual verification).
- Pass the caption to engine.say() to schedule it for speech.
- Invoke engine.runAndWait() to wait for the text to be spoken before proceeding to the next caption.

Step 3: Call Function

- Invoke read_captions(captions) after printing captions.
- Captions are read aloud in order.

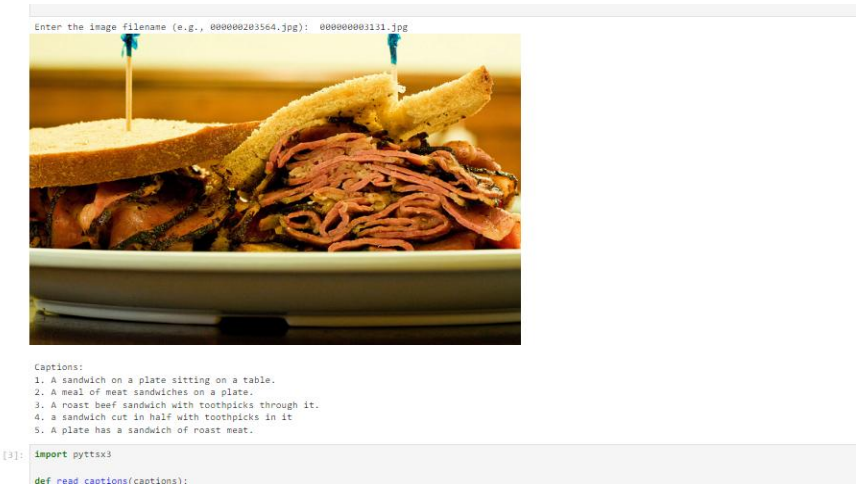

Figure 6b - Image Captioning Output in Jupyter Notebook

p-ISSN 0854-1418
e-ISSN 2503-426X

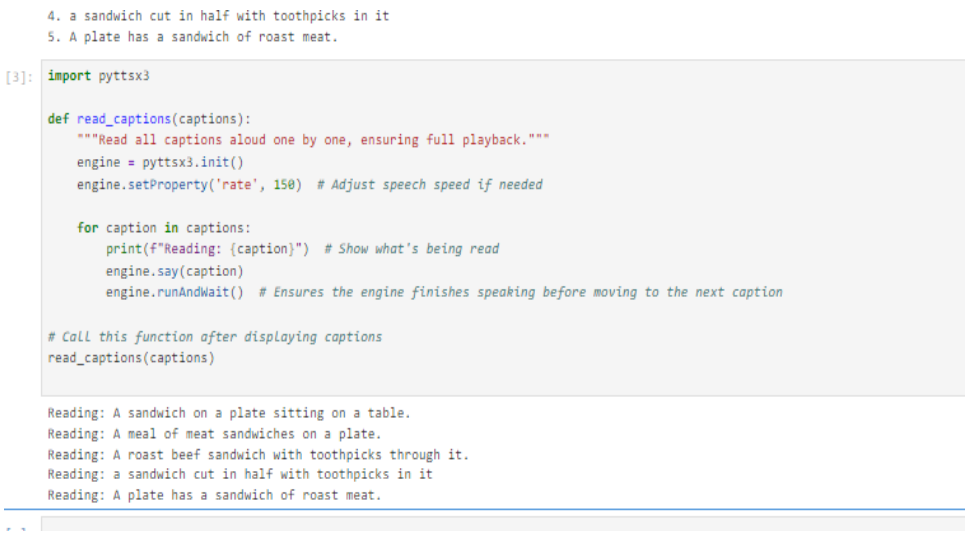

Figure 6c- Image Captioning Output in Jupyter Notebook

Figure 6-a, 6-b and 6-c is a screenshot of a Jupyter Notebook that shows an example of image captioning. The system has read the photo of the roast beef sandwich and then produced various sentences of this natural language genre. These explanations like "A sandwich on a plate sitting on a table" are likely made generated using a deep learning model that joins CNNs with RNNs to achieve the goal of the image being transformed into words.

5. EXPERIMENTATION

The pilot studywas done to try a different model modified with the COCO 2017 dataset and elicit better image captioning performance. The dataset contained as many as118,000 training images and 5,000 validation images, all of them with multipledescriptive captions.The sections after reveal the variousstages of experiments - data preprocessing, model training, and performanceevaluation.

5.1 DataPreprocessing

The main purpose ofthe preprocessing phase was to achieve the same scale in the image and textdata and thus enhance the AI model's performance. In connection with the imagepreprocessing stage, all images had been resized to 750x1000 pixels and thepixel values had been normalized to the range between -1 and 1 to fasten theconvergence. As for the deep learning model, the extracted features were2048-dimensional vectors for each picture. For text preprocessing, thecaptions were modified by deleting punctuation, special characters, and stopwords. Keras Tokenizer was used to convert the captions intotokens and also to map each word to a unique index. Masking and paddingtechniques were used to proper the length of the captions, so that theinformation could be given as input to the LSTM model. Consequently, thisimparted the network with the advantage that it could deal with variable-lengthsequences and also streamline training.

5.2 Model Architecture and Training

Themodel that was featured has been prepared with a CNN-LSTM architecture,in which the

exception model was the entity that provided the properties ordescriptors and an LSTM-based decoder has been the one responsible for thecaptions' generation. With the help of the Attention Mechanism,in this case, the feature of the model to be able to have a focus on the partsof the image that are relevant while expressing text, a higher quality of thatprocess of captioning became achievable and easier. The model was trained with the Adam optimizer with a learning rate of 0.0001 and a batch size of 32. The loss function was selected to be the categorical cross-entropy, and the training was done for 10 epochs. At the time when it was going to thetraining stage, the dropout regularization was applied to avoid falling victimsto overfitting as well as to the model. The training data was made of 118,000 images, and the validation process was done on a 5,000-image subset..

5.3 Qualitative Analysis

The model's captioning ability was tested on various images from the COCO dataset. It performed well in clear and structured scenes, generating highly accurate descriptions.

For example:

- Input Image: A child playing with a dog in a park.
- Generated Caption: "A young boy playing fetch with a brown dog on green grass." This caption effectively describes the objects, actions, and contextual relationships within the image. However, the model struggled in complex, cluttered scenes, occasionally misidentifying overlapping objects or producing genric caption.

6. RESULTS

Testing the capability of the ``pyttsx3`` library to read photo descriptions was also attempted, but there were also a few obstacles. First, the text-to-speech (TTS) engine was being operated while the library was in use, and it was performing the reading of only two out of the five provided captions. A confirmation of the test execution initiated a search for the reasons for the failure, and also some likely solutions. The first option tried was to put all the captions into a queue in advance and to execute ``runAndWait()`` upon exit. However, still, it was noticeable that ``pyttsx3`` was not always operating efficiently regarding a large number of texts that were queued. If dozens of text commands are added all at once, the engine may skip some of them due to the capacity limit of processing, which was why only a part of the captions were being spoken. Another error message—``"run loop already started"``—brought about the information that the repeated calls to ``runAndWait()`` in the same process could clash with each other, hence the fault read queue. So that the manager of the `runAndWait()` faults within the existing implementation was not repeated, a novel manner was set up. Instead of queueing all the captions at the time of processing, an alternative was to process each text one by one (sic). The alternate version of the code was such that it immediately called `runAndWait()` right after one caption and then moved on to another, if any. Henceforth, with such an approach, it was determined that the engine could not skip any captions. Furthermore, apart from the readable list of contents, the printing out of the captions before speaking aids in the keeping track thus the user can easily point out which ones are also being read. The restructured code was very effective in that all captions could be read without losing meaning from any of them. Moreover, by deploying the method, the previous

problem was overcome, and accompanied the improved result was the fact that no information remained unread. The load of the printing of the entire file in advance gave positive proof that the problem was largely of the queue's making.Perhaps the most significant finding from this trial is the fact that `pyttsx3`is a fantastic text-to-speech offline engine that simply requires a properqueuing system to be in place to prevent it from being disrupted. By followingthe correct sequence of events, namely creating the engine, reading thesubtitles successively, and calling `runAndWait()` after each one of them, thesystem behaves as it was initially planned, i.e. it reads all the captions withoutany loss. This advanced method guarantees that the audio and text are exactlywhat is written in the captions and, thus, it becomes more applicable forreal-life scenarios.

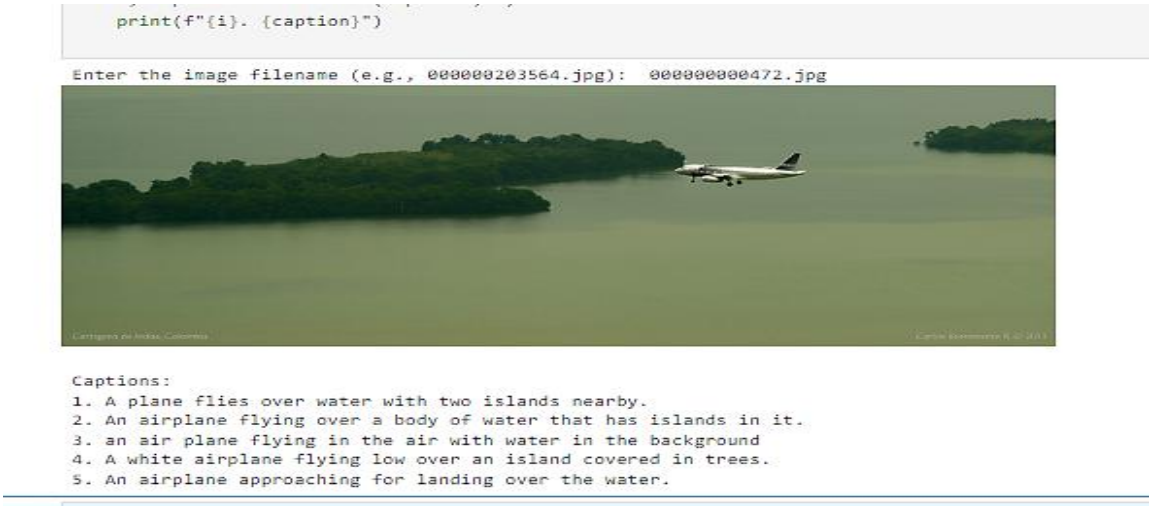

Figure 7 Generated Captions for an Aerial Image Using Deep Learning

Figure 7 displays a Jupyter Notebook output of the aerial photograph of the aircraft over ocean and islands, and the caption that provides a description of it. The tool produces various leading sentences based on the image e.g. “A plane flies over water with two islands nearby.” This is to say that a visual-textual model has been incorporated, for the processing of the information sourced from CNN and the RNN or Transformer-based network that performs natural language generation.

7. LIMITATIONS

While the model's captioning accuracy and context-sensitivity capabilities were beyond satisfactory, glitches that actually manipulate the reliability of the model in practical applications came into view. One of the significant hindrances here was figuring out the way through crowded scenes, the ones where multiple objects shared a common place. Oftentimes, the model failed in correct isolation of the objects that were in the picture, which tended to the wrong classification or the loss of key details. To illustrate, in the situation when the street was full of pedestrians, vehicles, and signboards, the model at times would pick out only one from the listed objects without mentioning the rest, which typically resulted in the creation of only a part of the complete caption. The reason is that the feature extraction model (Xception) focuses more on the whole image characteristics rather than individual objects, hence no particular details are provided. Another downfall ofthe model was that it lacked the capability

to describe minor characteristics such as object attributes, e.g., color or texture, and tiny background components. The model was just fine at creating generalized descriptions, but oftentimes, it would forget to include the finer nuances. For example, an imageof a red sports car on the street may have been presented with only "A car parked on the street," as the output from the model, without the objects' color and type being mentioned. This was because traditionalCNN-based models are more designed to capture the features that are of interest to people rather than their detailed properties of the objects they represent. In addition to that, the chief characteristic of the model was the dependence on the dataset (COCO) it was trained on. While it is true that COCO contains a wide variety of objects and scenes, it still is missing domain-specific images, e.g., medical scans, and industrial or scientific imagery. Thetest on a microbiological picture of bacteria, one that was completely outside the training set, would lead to a generic and inaccurate caption due to the machine's lack of understanding, the scenario being highly likely. This brings us to the conclusion that if one wishes to increase the scope of the model, they should engage in focused training with domain-specific datasets. To solve these issues, some possible solutions could be the integration of advanced object detection models like YOLO or Faster R-CNN that aid in better identification of the overlapping objects that are dressed differently or that are partially covered and underusing the ViT and attention mechanisms for capturing finer details and self-attention in the unsupervised learning paradigm with CLIP. These interventions will have an incredible effect on the accuracy, detail, and application of the real world in the image captioning system, thereby, it will be a tool delivering a more effective means to assist the visually impaired users and expand its use to other fields in a more efficient way.

8.CONCLUSION

The system represented in the research is extensively used in assistive technology for the blind and visually impaired that are developed under the deep learning approach called Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks. The integration of a visual description generator with a text-to-speech (TTS) engine can effectively convert visual information into spoken descriptions. This helps blind individuals better understand their surroundings by enabling efficient interaction with their environment through smart assistive systems. The model operating the COCO 2017 dataset and applying the CNN-LSTM model with
attention has been observed to have very good scores in the task of generating the most fitting descriptions. The system was shown, in the context of the research, to be faulty in certain aspects such as the inability to handle congested and complex scenes, unreliability in identifying specific details and dependence on diversity indatasets. The work in the future can be unmanned and the model can be more advanced if the tasks can be handled with the use of sophisticated object detection models such as YOLO, theuse of Vision Transformers (ViT) to achieve new features isolated from the image pixels and resorted on domain-specific dataset fine-tuning of the model. Though within the limits, this solution becomes an efficient and affordable option that can be used without the presence of additional hardware which definitely is achieved by keeping the barriers and costs low so that a larger fraction of the visually impaired population can enjoy easy reach to access their daily lives and integrate withthe

environment. The technology is not only cost-effective, but also guarantees the visually impaired the accessibility and convenience of mobile phones, thus enabling them to move around to their daily businesses by not necessarily relying on other people and at the same time be able to interact with the environment as a way of maintaining the independence. This technology has the potential, upon further process improvements, to significantly enhance the lives of the global visually impaired population.

9. REFERENCES

[1] Aishwarya Maroju, Sneha Sri Doma , Lahari Chandarlapati. “Image Caption Generating Deep Learning Model.” 2021
[2] Asha G. Hagargund, Sharsha Vanria Thota , Mitadru Bera , Eram Fatima , Shaik. “Image to Speech Conversion for Visually Impaired.” 2017
[3] Burhanali Irfan Mastan, Sehreen Mohd Shafi Kazi, Muskan Zakir Gavandi, Salina Mohammed Aslam Budye. “A COMPARATIVE STUDY ON VISUAL AID FOR BLIND.” 2022
[4] Cao Chenyu. “Understanding Image Caption Algorithms: A Review.” 2019
[5] C. S. Kanimozhiselvi; Karthika V; Kalaivani S P; Krithika S. “Image Captioning Using Deep Learning.” 2022
[6] Dengyuhan Dai.” An Introduction of CNN: Models and Training on Neural Network Models.” 2021
[7] Joe Louis Paul; S Sasirekha; S Mohanavalli; C Jayashree; P Moohana Priya; K Monika. “Smart Eye for Visually Impaired-An aid to help the blind people.” 2019
[8] Kirthika Subramaniam B. “TEXT READER FOR VISUALLY IMPAIRED.” 2021
[9] Koppala Guravaiah, Yarlagadda Sai Bhavadeesh, Peddi Shwejan, Allu Harsha Vardhan, S Lavanya. “Third Eye: Object Recognition and Speech Generation for Visually Impaired.” 2019
[10] Paraskevas Diamantatos & Ergina Kavallieratou. “Android Based Electronic Travel Aid System for Blind People.” 2019.
[11] Rama Devi, K. Sahaja, S. Santrupth, M. P. Tony Harsha, K. Balasubramanyam Reddy. “Blind Assistance System using Image Processing.” 2022
[12] Qian Lu. “Feasibility Study of a "Smart" Aid for the Visually Impaired and Blind's Independent Mobility in Outdoor Environments.” 2018.
[13] Raganath Chakaravarthy, BS sourab. “Design and implementation of mobility aid for blind people.” 2015.
[14] Savera Sarwar, Muhammad Turab, Danish Channa, Aisha Chandio, M. Uzair Sohu, Vikram Kumar. “Advanced Audio Aid for Blind People.” 2022
[15] Shivangi Nagdewani, Ashika Jain. “A REVIEW ON METHODS FOR SPEECH-TO-TEXT AND TEXT-TO-SPEECH CONVERSION.” 2020
[16] Shrugal Varde, Prof. M.S. Panse. “Computer Vision Based Travel Aid for Blind.” 2017
[17] Simao Herdade, Armin Kappeler, Kofi Boakye, Joao Soares. “Image Captioning: Transforming Objects into Words.” 2020
[18] Tushar Khete and Aditya Bakshi. “Autonomous Assistance System for Visually Impaired using Tesseract OCR & gTTS.” 2022

[19] Vidya Rao, Aditya Kumar, Pranoti Mane. “Blind Aid: Travel aid for blind.” 2022.
[20] Lakshminarasimhan Srinivasan, Dinesh Sreekanthan, Amutha A.L. “Image Captioning - A Deep Learning Approach”.
[21] B. Singh, "A Survey of Current Aids for Visually Impaired Persons," *2018 IEEE Xplore*. https://ieeexplore.ieee.org/document
[22] I. J. L. Paul, "Smart Eye for Visually Impaired – An Aid to Help the Blind," *2019 IEEE Xplore*. https://ieeexplore.ieee.org/document
[23] J. A. Shah, "EYERIS: A Virtual Eye to Aid the Visually Impaired," *2020 IEEE Xplore*. https://ieeexplore.ieee.org/document
[24] A. Paulast, "Assistive System for Blind and Visually Impaired People," *2023 IEEE Xplore*. https://ieeexplore.ieee.org/document
[25] B. D. Jain, "Visual Assistance for Blind Using Image Processing," *2018 IEEE Xplore*. https://ieeexplore.ieee.org/document
[26] P. S. Ranaweera, "Electronic Travel Aid System for Visually Impaired People," *2017 IEEE Xplore*.
[27] Unknown Author, "A Survey of Vision Aids for the Blind," *IEEE Xplore*. http://ieeexplore.ieee.org/document
[28] K. Patil, "Guidance System for Visually Impaired People," *2021 IEEE Xplore*. [Online]. Available: https://ieeexplore.ieee.org/document
[29] D. A. Khan, "Assistive Stick for Visually Impaired People," *2022 IEEE Xplore*. https://ieeexplore.ieee.org/document
[30] S. Sarwar, "Advanced Audio Aid for Blind People," *2022 IEEE Xplore*. https://ieeexplore.ieee.org/document