AIsight

Mihir Patel, Jainish Parikh, Nishit Doshi, Apoorva Banubakode
Department of Computer Engineering, San Jose State University
{mihirmaheshkumar.patel, jainishjitendrakumar.parikh, nishitajaykumar.doshi, apoorva.banubakode} @sjsu.edu

***Abstract*—AISight is a device which focuses on helping visually impaired people better understand their surroundings. It achieves this by providing real time scene description to the user with an audio output. AIsight is a Raspberry pi based device that captures images of surroundings and sends it to a pretrained machine learning model. This model uses CNN followed by RNN to caption individual images. These images are then converted to audio and presented to the user in real-time.**

***Index Terms*—Scene Description, Machine Learning, Visual Assistant, Raspberry Pi**

I. INTRODUCTION

THE field of Computer Vision has progressed immensely since its advent. Accuracy of the models in the field of object detection has skyrocketed in last decade. It includes a variety of concepts such as Object Detection, Visual Recognition, Image Captioning etc. The growth of Natural Language Processing is no different from that of Computer Vision. Plethora of applications in the real world makes them a topic of high interest and research.

Some pioneering approaches that address the challenge of generating scene descriptions have been developed. However, the solution they propose mainly focuses on Object Detection and generating an output in form of an audio. Although, very useful, object detection does not help users efficiently understand their surroundings. Moreover, most of the approaches are phone based i.e. requires an application on users's phone. Although, this may seem rudimentary but, for a visually impaired this can be a cumbersome task.

Leveraging Computer Vision and Natural Language Processing, we here aim to provide a solution that can help visually impaired efficiently comprehend their surroundings. Our approach includes using a completely separate device intended for only a special task. Unlike contemporaries, we do not use object detection in the solution. Instead, we use image captioning to describe a scene. It does not focus on detecting objects in an image but provides a general description of happenings in the image. A device (i.e. raspberrypi) captures images and sends it to an image captioning model running on the cloud. The model responds with a caption or text describing that image which is then converted and yielded as audio from the device. The main aim behind using a separate product is to have phone constantly available for users to do other tasks.

II. MOTIVATION

Currently the existing solutions in market like Google Lookout and Seeing AI, perform object recognition and dictate the objects in the frame to the user. These are good for the user to know and identify objects around them. The motivation behind AIsight is to make the users feel good by serving them with an all-time companion which would describe the happenings around them in real time. AIsight leverages this by describing the scenario in front of the user and not just the individual objects. The current solutions are all cellphone applications, which adds on a major drawback of continuously keeping the app open and hinders the user from utilizing the cell phone for elementary tasks. Moreover, it is often difficult for users with partial blindness to navigate through the app. We propose an independent, and portable device to remove this dependency on the cellphones.

III. PROPOSED SOLUTION

Our device AIsight, primarily comprises of a Raspberry pi, which is a portable single board computer. To this we have added a camera peripheral to support image capturing in the background. The battery bank ensures continuous power supply. It's a single plug power on device and no additional installations, settings are required from user end. Below is a picture of the device.

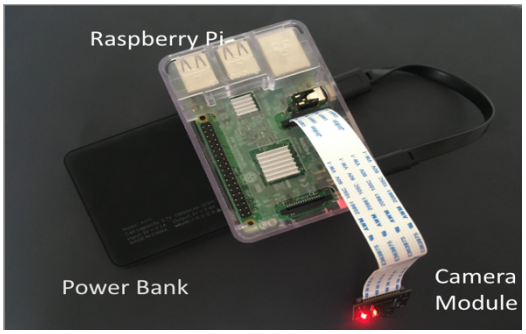


Fig. 1. AIsight device comprised of a Raspberry Pi, camera module and a power bank.

We experimented with three different approaches to come up with faster, most efficient and secure design namely, On edge computing, On cloud computing (using SCP) and on cloud computing (Client Server).

A. On Edge Computing

Our first approach includes a completely independent device (i.e. raspberrypi). The device itself is solely responsible for capturing images and generating captions using machine learning model.

Logical Steps are as follows:

1. Image is captured using camera module attached to the device and is sent to the model for caption generation.
2. Image Captioning model generates caption for the provided image.
3. Caption or text is then outputted in form of audio using text-to-speech module running on the device.

- **Advantages**:
  No, constant internet connection is mandatory.

- **Disadvantage**:
  Takes nearly forty seconds to generate captions which is not a viable option for realtime processing.

B. On Cloud (SCP)

Our prior approach lacked real-time processing which lead us to try and run machine learning model on the cloud. So, in this approach the device is responsible for capturing and sending images to the model running on the cloud. Instance on the cloud is responsible for generating captions for received images.

Logical Steps are as follows:

1. When the user scans the surroundings using the device (i.e. Raspberrypi), image frames are sent to Image Captioning Engine on the cloud using SCP.
2. Image Captioning Engine will then repeatedly generate a caption for every image frame it received and sends it back to the device via SCP.
3. The obtained caption is then converted to speech by Pico2wave on the device.
4. Eventually, it gives you a spoken description of surroundings.

Data transfer between end-points is done via SCP in this approach. Although this approach reduced processing time significantly, it added an issue of synchronization. SCP does not ensure any locking mechanism, therefore in case of multiple concurrent requests only one of sent images are processed and the generated caption is sent as response to all the simultaneous requests.

- **Advantages**:
  Processing time reduced drastically from nearly forty seconds to five seconds.

- **Disadvantage**:
  A constant internet connectivity is required for data transfer purposes. Use of SCP made data transfer asynchronous. On Cloud (Client Server)

C. On Cloud (Client Server)

Unlike, previous approach where SCP was used for data transfer, a client-server architecture is created in this approach using flask framework and data transfer is done using HTTP GET/POST methods. Along with the reducing latency in processing this approach overcame the drawback of asynchronization.

Logical Steps are as follows:

1. When the user scans the surroundings using the device (i.e. Raspberrypi), image frames are sent to Image Captioning Engine on the cloud via flask.
2. Image Captioning Engine will then repeatedly generate a caption for every image frame it received and sends it back to the device using flask server.
3. The obtained caption is then converted to speech by Pico2wave on the device.
4. Eventually, it gives you a spoken description of surroundings.

- **Advantage**:
  In addition to reducing processing time substantially to five to six seconds, it also ensures synchronization.

- **Disadvantage**:
  A constant internet connectivity is required for data transfer purposes.

- Client-side architecture:
  On client side the device is responsible for capturing as well as sending images and generating audio output of captions received from the server as well.

- Server-side architecture:
  The flask server with machine learning model running on the cloud is dockerized and runs on multiple instances to manage large number of requests simultaneously.

  Below is the architecture diagram of the third approach which was eventually used to design AIsight.

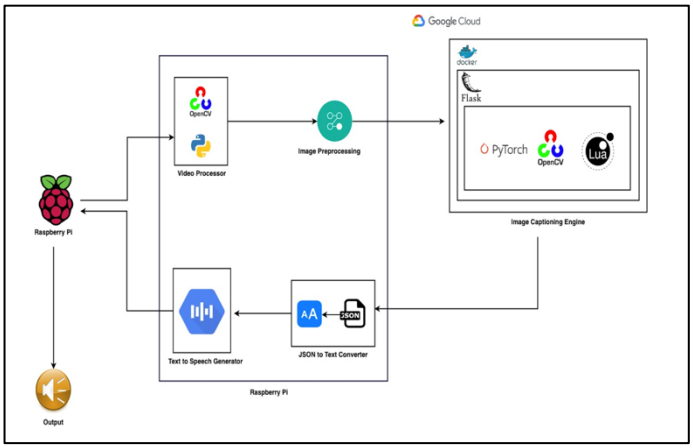
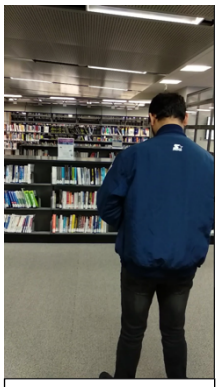


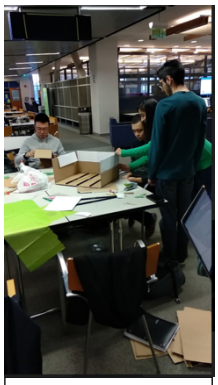
Fig. 2. Architecture Diagram

IV. RESULTS AND CONCLUSION

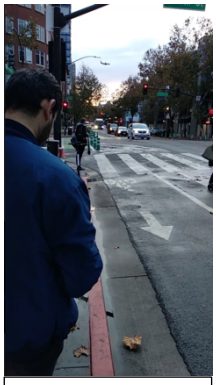
After experimenting with the different approaches, we found that the third solution using Cloud (Client Server) had the least latency along with synchronization of multiple requests. Below are some of the results showcasing the outcome in text format.



"A bookshelf filled with lots of books"



"A Group of people sitting around a table with laptops"



"A city street with traffic light on the side of it"

Fig. 3. Results of AIsight when tested at multiple locations using third approach

The latency comparisons among the three approaches can be interpreted from the below graph.

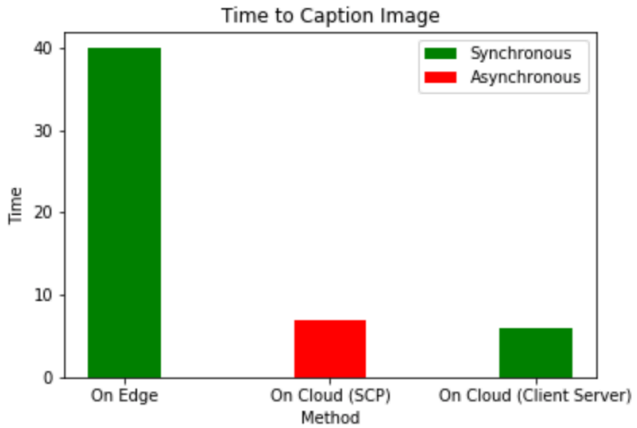


Fig. 4. Latency comparison

V. FUTURE SCOPE

While this product is a very useful, certain enhancements can make it more beneficial. Below are some features we plan to include in the later releases.

1. Tells the user the distance from the various objects and scenes.
2. Improvements and fine tune the model to continuously learn from the familiar surroundings.
3. Reduce time taken for computation on edge as this would remove the dependency of stable internet connection and can be used as an offline device.
4. Read out danger scenes and suggests caution steps.

VI. ACKNOWLEDGMENT

We are grateful to Professor Rakesh Ranjan for providing us with the opportunity of working on this project. With his constant support we were able to successfully implement the proposed solution.

VII. REFERENCES

[1] Andrej Karpathy, Li Fei-Fei; "Deep Visual-Semantic Alignments for Generating Image Descriptions," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 3128-3137

[2] "Docker- neural talk 2,"[Online].Available: https://github.com/SaMnCo/docker-neuraltalk2

[3] "Neuraltalk2,"[Online].Available: https://github.com/karpathy/neuraltalk2

[4] "Installing Operating System Images," [Online]. Available https://www.raspberrypi.org/documentation/installation/installing-images/README.md

[5] "Flask: User Guide"[Online]. Available http://flask.palletsprojects.com/en/1.1.x/#user-s-guide

[6] "Installing Open CV on Raspberry pi," [Online]. Available https://tutorials-raspberrypi.com/installing-opencv-on-the-raspberry-pi/

[7] "Raspberry pi Experiments,"[Online]. Available http://rpihome.blogspot.com/2015/02/installing-pico-tts.html