SoDistant: Social distancing detection for surveillance cameras

Saurabh Rajeshkumar Panchal, Harsh Sheth, Vagdevi Junnuri, and Anjali Kumari
Department of Software Engineering, San Jose State University, San Jose, CA
saurabhrajeshkumar.panchal@sjsu.edu, harshashlesh.sheth@sjsu.edu, vagdevi.junnuri@sjsu.edu, anjali.kumari@sjsu.edu

***Abstract*— Since the outbreak of covid-19 it has been very much understood the importance Of Social distancing in order to restrain the spread of the virus across the globe. CDC has defined Social distancing as having a minimum of 6 feet (about 2 arm's length) between two persons in order to maintain social Distancing. But can people across the globe always follow this new atypical method of maintaining social distancing without any help? It is hard and in order to solve this complex problem comes the idea of 'SoDistant' which helps in Social distancing surveillance through cameras.**

I. INTRODUCTION

SoDistant is a tool that helps outdoors, malls, Hospitals, schools, office spaces, airports, banks and places where people tend to gather and detect if people are following social distancing or not. It helps in surveillance of social distancing using cameras with real time violation detection and real time graphs. SoDistant helps any institution which thrives to maintain social distancing.

II. ARCHITECTURE

There are 4 major components: Object Detection, Machine Learning model (calculating distance), Services and Client.

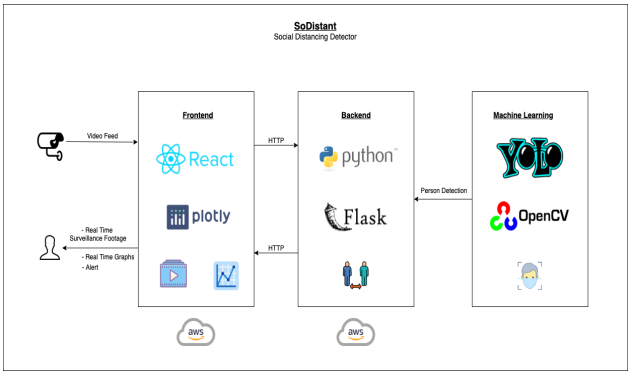

Fig. 1. Project Architecture

III. OBJECT DETECTION

We have used YOLOv4.tiny model for object detection. In general, single-stage detectors like YOLO tend to be less accurate but are significantly faster. YOLO treats object detection as a regression problem, taking a given input image and simultaneously learning bounding box coordinates and corresponding class label probabilities. It is used to return the person prediction probability, bounding box coordinates for the detection, and the centroid of the person.

IV. DISTANCE CALCULATION

NMS (Non-maxima suppression) is also used to reduce overlapping bounding boxes to only a single bounding box, thus representing the true detection of the object. Having overlapping boxes is not exactly practical and ideal, especially if we need to count the number of objects in an image. Euclidean distance is then computed between all pairs of the returned centroids. Simply, a centroid is the center of a bounding box. Based on these pairwise distances, we check to see if any two people are less than/close to 'N' pixels apart.

V. SERVICE

The backend service was developed using python and framework flask. It contains several handlers/APIs such as.

1. Get video feed
2. Upload video
3. Update config values

The service is deployed on Amazon EC2 instances with a single availability zone (US-EAST-1A). The link to the instance is **http://54.160.156.176:3000/**

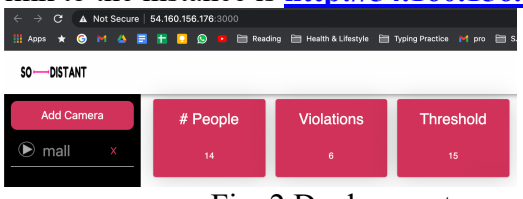

Fig. 2 Deployment

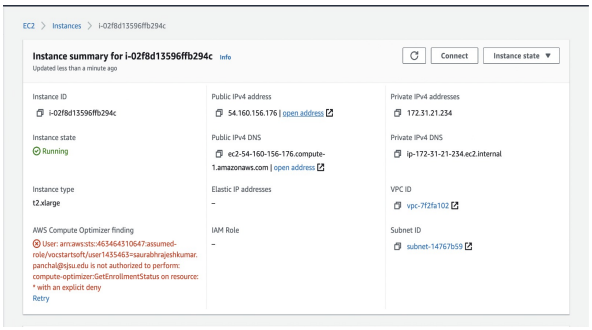

Fig. 3 AWS EC2 deployment

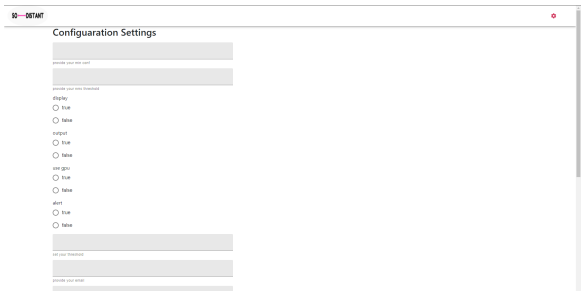

Fig. 4 Configuration Form

The services abstract away the complex processing of generating video frames on the fly in real time which involves calculating the number of people in the frame, number of violations people are making by crossing the threshold which can be set by the clients who have installed the SoDistant tools.

The configuration settings can be changed which include changing the minimum distance, threshold value, alert flag, output flag, display flag, use GPU, email etc. In this way SoDistant can be highly customized according to the client requirements.

The video rendering from the services is real time which helps in detecting social distancing between people at the exact time the events are happening without any lag.

The official deployment of the service is automated via the AWS Elastic Container Service (ECS). The service exposes many endpoints; each is accessed via GET/POST.

List of GET/POST endpoints:

A. config (GET)

This endpoint returns the list of configurations in the form of JSON data.

B. config (POST)

This endpoint sets the configuration into the portal as per the user requirements. The data is sent in JSON format, user must fill up the form in order to change configuration values.

C. upload_file

This endpoint fetches the file which user wants to set as the feed and sends it to the backend.

D. video_feed

This endpoint returns the video feed onto the dashboard which user had already set to view the social distancing measures.

VI. CLIENT

The frontend was developed using ReactJS which helped in rendering the backend services in a seamless way like a single page application.

Fig. 5 Dashboard

The main page is the dashboard page which has the feature to add a video by inputting the cameral URL or uploading a video file from local system. After submitting that the video gets added in the page which can be played using the play button.

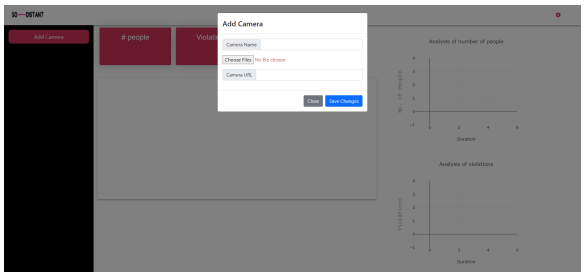

Fig. 6 Add Camera

As soon as the video gets played real time rendering from the services starts to begin which calculating the number of people and violations occurring in the video. These calculations happen based on the social distancing threshold that was set in the config values in real time.

Fig. 7 Video Feed on Dashboard

Simultaneously the two charts display the number of people and number of violations by the side. Charts were developed using Plotly.

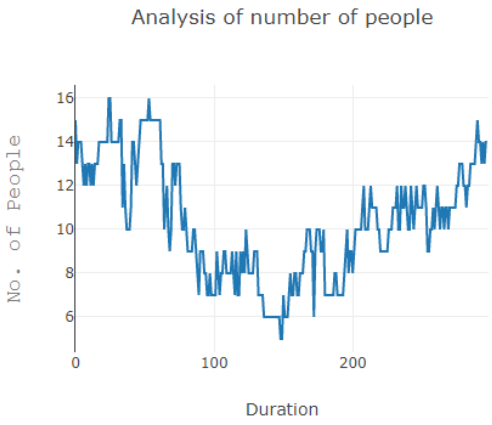

Fig. 8 Analysis for number of people

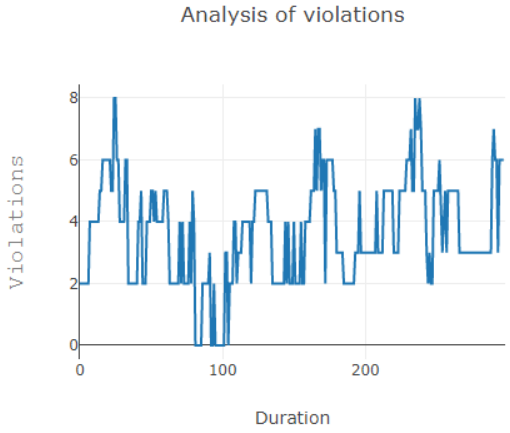

Fig. 9 Analysis for violations

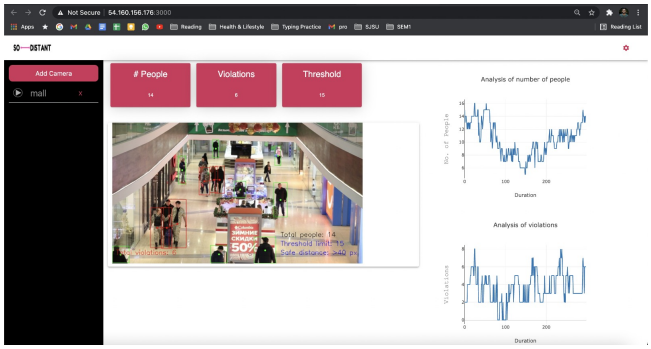

Fig. 10 EC2 Deployed Project

A. React/React-bootstrap

React-Bootstrap replaces the Bootstrap JavaScript. Each component has been built from scratch as a true React component, without unneeded dependencies like jQuery.

As one of the oldest React libraries, React-Bootstrap has evolved and grown alongside React, making it an excellent choice as your UI foundation.

B. Axios

Axios is a popular, promise-based HTTP client that sports an easy-to-use API and can be used in both the browser and Node.js.

C. React Plotly

React Plotly is used embed D3 charts in your React-powered web application. This React component takes the chart type, data, and styling as Plotly JSON in its data and layout props, then draws the chart using Plotly.js.

VII. CONCLUSION

As businesses gradually reopen post Covid pandemic, it has become essential to ensure that social distancing is being followed. Currently most businesses have dedicated employees who manually monitor customer capacity and ensure that customers follow social distancing norms set by government guidelines. In that scenario, SoDistant – a computer vision tool comes handy where all the manual monitoring is done in real time with the help of preinstalled cameras.

VIII. FUTURE IMPROVEMENTS

As the project grows, our future improvements include:

A. Birds Eye View
B. Face Mask Detection
C. Alerts for visitors
D. Integration with government guidelines to include dynamic updates to social distancing norms

IX. DELIVERABLES

A. GitHub Repository
https://github.com/SJSUSpring21/SoDistant

B. Project Presentation
https://github.com/SJSUSpring21/SoDistant/blob/main/SoDistant%20Presentation.pptx

C. Hosted Application
http://54.160.156.176:3000/

X. REFERENCES

A. Landing AI Creates an AI Tool to Help Customers Monitor Social Distancing in the Workplace
https://landing.ai/landing-ai-creates-an-ai-tool-to-help-customers-monitor-social-distancing-in-the-workplace/

B. OpenCV Social Distancing Detector
https://www.pyimagesearch.com/2020/06/01/opencv-social-distancing-detector/