Geo-distributed Surveillance System

1st Ido Jacob 2nd Victor Gwehenberger 3rd Changseok Woo

I. Introduction

This project showcases the design and implementation of a distributed cloud-edge-IoT surveillance system for detecting and responding to the presence of unknown individuals. The system is built on a three-layer structure compromising IoT, Edge and Cloud layers. The IoT layer consists of simulated devices that represent real-world cameras, which capture images at regular intervals. The edge layer which acts as a middleware, receives and preprocesses the data before forwarding it to the cloud layer. Finally, the cloud layer makes use of services such as AWS Lambda for scalable, event-driven computation and AWS Rekognition for advanced image analysis to efficiently identify known and unknown individuals. This approach effectively tackles key challenges such as scalability, resource optimization, and latency in a distributed system design and showcases the benefits of a tiered structure that leverages cloud computing for compute-intensive tasks.

II. System architecture

Describe your system in detail, including a figure for your architectural diagram (IoT, Edge, Cloud layers, components developed and services used). As seen in Figure 1, the surveillance system has four main components.

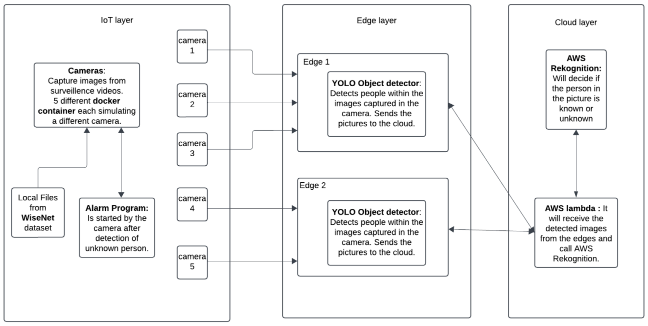
IoT layer
camera 1
camera 2
camera 3
camera 4
camera 5
Cameras: Capture images from surveillence videos. 5 different docker container each simulating a different camera.
Local Files from WiseNet dataset
Alarm Program: Is started by the camera after detection of unknown person.
Edge layer
Edge 1
YOLO Object detector: Detects people within the images captured in the camera. Sends the pictures to the cloud.
Edge 2
YOLO Object detector: Detects people within the images captured in the camera. Sends the pictures to the cloud.
Cloud layer
AWS Rekognition: Will decide if the person in the picture is known or unknown
AWS lambda : It will receive the detected images from the edges and call AWS Rekognition.

Fig. 1. Architectural diagram of the surveillance system.

A. IoT Layer

First is the IoT component which is responsible for mimicking a surveillance camera. This is done by locally storing a dataset containing multiple videos and passing the frames of the videos to the next components in intervals. The component that is in charge of using the frames from the IoT component is the edge component. The edge receives the frames then processes the image to detect a person. This component also contained the alarm system but is explained more in detail later in the paper.

B. Edge Layer

The edge will use an algorithm by the name of ”YOLO Object Detector” to search for a person within the frame. IoT will analyse through each given images using YOLO and if a person is detected it will then pass the images containing a person to the cloud. These two components can be run locally on a computer or on an EC2 instance thus are put together as one part of the architecture that deals with the dataset offline.

C. Cloud Layer

The third component of this surveillance system is the cloud layer. The cloud layer is responsible for processing the frames that include people and this is done by using two AWS services. AWS Lambda is the first service that allows us to set up a cloud layer which we can run the other AWS service called AWS Rekognition. A program is implemented within the AWS Lambda that will call AWS Rekognition through an API call to anaylse the person to see if the person is known or unknown. If the program detects an unknown person within the received frames, it triggers the alarm of the surveillance system. As mentioned earlier, alarm system is implemented in the IoT and is the last component of the surveillance system. The cloud layer will call on the alarm system to trigger if a known person is detected.

III. Implementation details

IOT devices – 5 camera devices. Each camera is implemented in our local machines as a python file running inside its own docker container. Each of the containers is emulating a camera using datasets from Wisenet and extracting a frame every 1 seconds. Thus the frames are being sent to the edge devices every 1 second. Cameras 1,2,3 are sending the frames to the first edge device through a nat server. Cameras 4,5 do the same to the second edge device. After sending the frames the cameras also wait for a response from the edge regarding the detection of an unknown person. If a respond is sent back to one of the camera with a confirmation of an unknown person the camera turns on the alarm.

Edge devices – 2 Edge devices. Each of the edge devices is implemented as a python file running inside its own individual docker container. The 2 edge devices are subscribed to the cameras frame channel. Every time a frame is being sent to one of the edge devices it immediately checks via its Yolo detecting algorithm, whether the frame contains a person or not. Every time the edge detects a person it will sent the frame with the person up to the cloud service. The edge device will get a respond from the cloud service whether the person is known to the system or not. If the person is known to the system the edge will send a negative respond to the alarm system. Else, the edge will send a positive response to the alarm system to turn on the alarm.

Cloud layer - One cloud layer. An AWS lambda is going to be running a python program. The python program will be calling the AWS Rekognition through an API call.

IV. Evaluation

Evaluation of the response time and scalability (number of devices and traffic) to prove the correctness of your implementation. The more detailed the better.