Security Personal Turnout Identification using Android App

- A Case Study

Objective

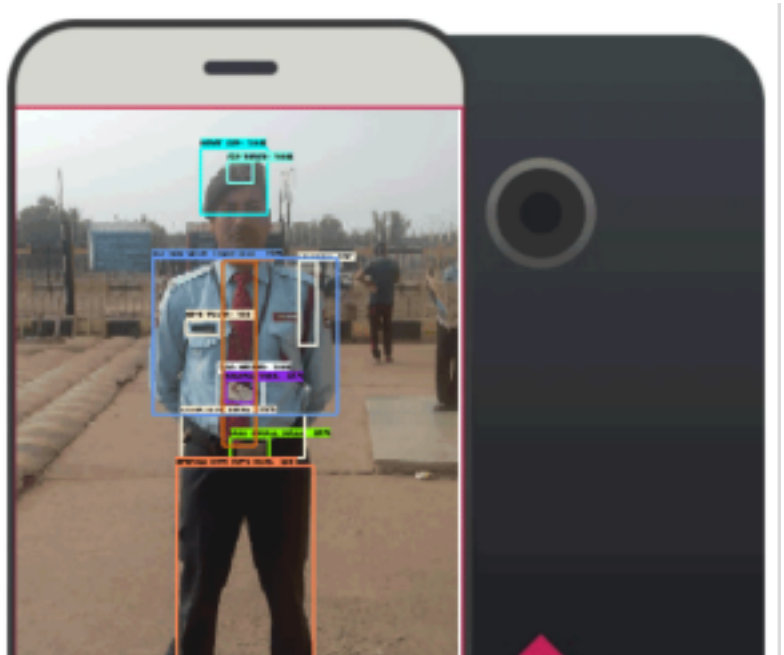
To build an AI-Powered Android app capable of identifying the appropriate or improper dress turnout from the image feed coming through the camera app and generating results based on the data locally.

Client

A leading private security firm in India that provides security personnel staffing in various institutions across India. The primary issue the client was facing due to the large network was manually auditing the information about the personnel staffing deployed.

The client was looking for a way to automate the process of manual auditing the enormous amount of resources deployed. Automated auditing would save both time and money for the client making the process faster and consuming less time as compared to manual procedures. The automation will also ensure that the data gets entered correctly and stays in the system for future reference.

Product Scope Overview



The model training can perform object detection in an android phone using the deep-learning object detection API. Then the trained and optimized model can be deployed to handheld devices like android.

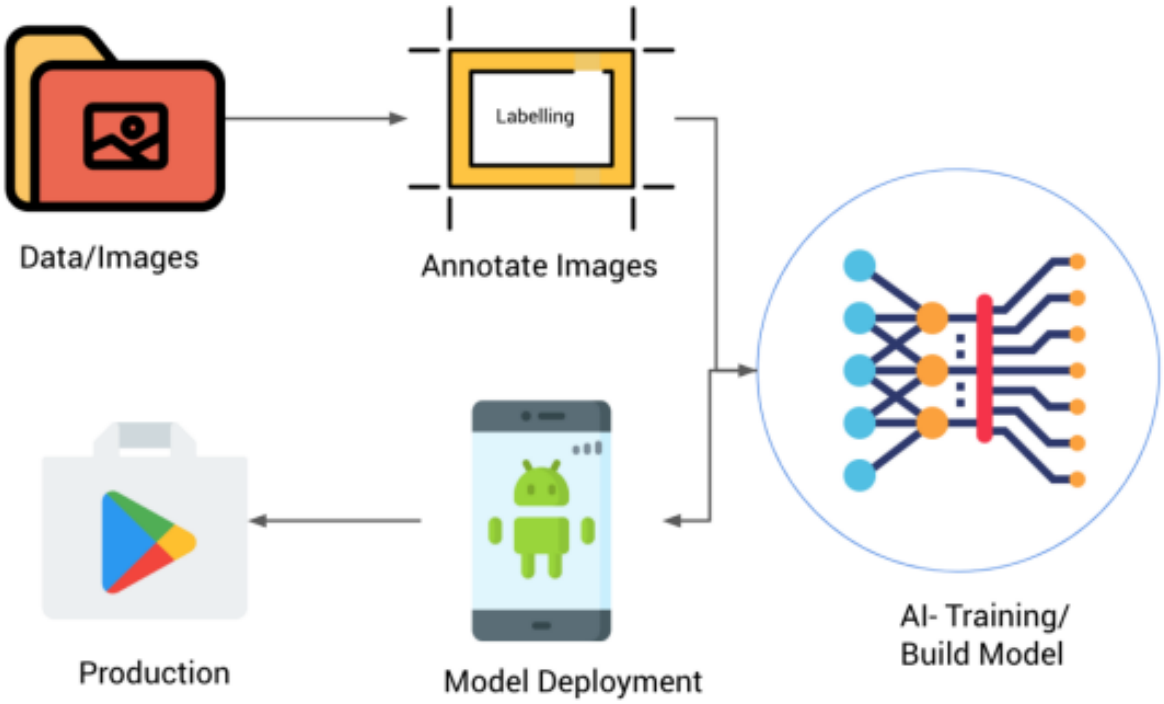
Given an image or a video stream, an object detection model can identify which of a known set of objects might be present and provide information about their positions within the image. An object detection model can identify various objects present in any photo or video and provide the necessary information.

For instance, if you take a picture of the security personnel, the object detection model will scan various components of the personnel's uniform and provide you with the necessary information. Below is an example of an object detection application using a mobile application. Multiple objects such as the shoes, cap, belt, the name tag, and even the shirt have been recognized, and their positions are marked and labeled

The pipeline consist of two major processes:

- Object detection training model
- Deployment to handheld devices

Overall Product Structure and Architecture Diagram



Product Flow Diagram

Below is the project's working flow diagram, which involves multiple steps to be followed, from data acquisition to deploying the files to the android device.

Dataset

Gather the dataset on which object detection is to be applied. Bring all the images to a uniform JPG format and resize them to a similar resolution. Store it in a folder for labeling it to the object present in it. Below is the reference for labeling the dataset.
https://github.com/tzutalin/labelImg

Labeling

Data should be labeled to the corresponding object present in the images. This is required for the deep learning object recognition pipeline for the learning process to learn it. After labeling, the dataset generates the XML annotated files and the images.

- Annotate
- XML generated
- Divide files to train-test folder(80:20 ratio)

Training

The data set is fed to the training pipeline for the learning process for that specific dataset. Then, a Deep Learning neural network maps the input to the output and generates the model.

- Model selection
- Hyperparameter setting (training step, batch size, etc.)
- Train the model

Optimize the model for accessible mobile devices

Model optimization reduces the size and inference speed of the model. The model optimization toolkit of TensorFlow is used to reduce the model's height and increase its efficiency with minimal impact on accuracy. Finally, the post-training quantization and optimization are done. There are two formats for optimizing the model.

- FLOAT
- QUANTIZEDINT8

Conversion and Deployment

After optimization, the model is converted to a flatbuffer file format compatible with the android device for inferencing. The file format generated is the tflite format. First, this exported file is deployed to the android device. Next, the trained object detection model is deployed to the android application asset folder of the project structure. Now the android application loads the model from the asset resource folder and runs the model to detect different objects from the device camera feed.

Detailed Product Structure