Object Detection On Construction Site

Rupali Saha, Snehal Motghare, Aachal Mate, Tanushri Sapate, Kirti Gokhale, Sakshi ChanneComputer Science and Engineering Department, Priyadarshini College of Engineering, Hingna Rd, Digdoh Hills, Nagpur-440019, Maharashtra, India.
E-mail: *snehalmotghare69@gmail.com, aachalmate143@gmail.com, sapatetanu@gmail.com,kirtigokhale9419@gmail.com,sakshichanne31@gmail.com.*

Abstract

Construction had the highest number of fatal work accidents of any industry, due to the high number of accidents each year. There are many solutions to ensure worker safety and reduce accidents, one of which is to ensure the proper use of appropriate personal protective equipment (PPE kit) as defined in safety regulations. However, monitoring the use of personal protective equipment, which is largely based on manual checks, is time-consuming and inefficient. The Several attempts were made. The resulting recognition accuracy on the 12 main armors is up to 98%, while the accuracy of face detection and recognition is 96%. The obtained results showed the ability to identify recognize faces and all the equipment very accurately and remember them in real time. The study uses convolutional neural network (CNN) models developed by applying transfer learning to the basic version Faster RCNN deep learning. Considering the presence the model predicts fulfillment of requirements in 5 categories, such as jacket, shoes, helmet, gloves, id card.

It is necessary for workers to wear helmets when working in large construction sites. The traditional way to supervise the workers whether wearing helmets or not for safety is artificial, which have brought out many problems such as many blind spots, labor and time costing. Since a large number of surveillance cameras are currently installed in these construction sites, the surveillance video can be developed in taking the place of human supervision in an intelligent way. This paper designs a elements detection network based on Faster R-CNN . With the development of advanced technologies, automation in construction has improved substantially
.This leads to an increasing number of different new technologies adopted in the construction industry to benefit from their introduction. Safety monitoring, an important task in construction, is usually undertaken and recorded manually

Keywords---*Machine learning, Artificial Intelligence ,neural network, safety management ,Construction Image classification, Object Detection, R-CNN , FasterRCNN,YOLOv7 ,Tensor flow .*

I. INTRODUCTION

The object detection technology has gradually become an active research topic, and used in many aspects of our daily life, and people are also exploring more directions that can be applied to object detection . The object detection mainly obtains the original image through the camera, and detects the object through the analysis of the original image, based on thetarget features. Object detection can be used in the field of intelligent transportation systems because it can detect and track targets timely and dynamically.

Construction sites are hazardous places, and safety measures are of utmost importance. It is essential to ensure that all workers on the construction site have the necessary safety gear to prevent any accidents. The safety gear includes helmets, safety glasses, gloves, boots, and other protective equipment. However, ensuring that all workers are wearing the required safety gear can be challenging, especially on large construction sites. This is where the Object Detection App forLabour Kit Detection on Construction Site comes in handy.

The Object Detection App for Labour Kit Detection on Construction Site uses computer vision and deep learning algorithms to detect workers' safety gear on a construction site. This section provides technical details of the app's underlying technology. The Object Detection App for Labour Kit Detection on Construction Site is a mobile application designed to detect and identify whether workers on a construction site are wearing the necessary safety gear. The app uses computer vision and deep learning algorithms to detect and identify workers' safety gear in real-time. The app can detect helmets, safety glasses, gloves, boots, and other protective equipment.

With the development of advanced technologies, automation in construction has improved substantially. This leads to an increasing number of different new technologies adopted in the construction industry to benefit from their introduction. Safety monitoring, an important task in construction, is usually undertaken and recorded manually. Monitoring construction progress accurately can help contractors control and manage costs and scheduling. This report also indicated that construction organizations are adopting AI to extract information from large masses of datato improve construction safety and progress efficiency.

II. LITERATURE REVIEW

The purpose of this project is to develop a to improve the efficiency and accuracy of monitoring safety and modular installation progress on construction sites, the modular objects should be detected .This leads to an increasing number of different new technologies adopted in the construction industry to benefit from their introduction. Safety monitoring, an important task in construction, is usually undertaken and recorded manually. Monitoring construction progress accurately can help contractors control and manage costs and scheduling.

First, image data were taken from different construction sites. In the second step, object detection models were built in the TensorFlow platform. Configuring the computer development environment, deep learning platform (TensorFlow) and object detection API were the three main tasks in this step. Third, the models based on chosen object detection algorithms were trained on the training data set and adjusted based on validation data sets to improve the generalizability of the object detection algorithm to unseen data. Two models based on faster RCNN and SSD were trained on the labelled images. The number of images, the chosen algorithm and training steps influenced detection accuracy. The last step was to analyze the performance trained models across the selected metrics

The Following describes the overall objective of this project:

- Improve safety on job sites: The primary objective of the app would be to improve safety for laborers by ensuring that they are wearing appropriate PPE while working. This would help to reduce the risk ofworkplace injuries and illnesses.

- Simplify PPE checks: The app would simplify the process of checking that laborers are wearing appropriate PPE. Rather than having to physically inspect each laborer, supervisors could simply scan a QR code on the laborer's PPE kit using the app, which would quickly confirm whether or not the laborer is properly equipped.

- Ensure compliance with regulations: By helping to ensure that laborers are wearing appropriate PPE, the app would help to ensure compliance with workplace safety regulations. This could help to avoid costly fines or legal issues for employers.

- Increase accountability: The app would increase accountability for both laborers and supervisors by providing a clear record of PPE checks. This could help to encourage compliance with safety protocolsand ensure that any lapses in safety are quickly identified and addressed.

III. METHODOLOGY

Step 1: Install Android studio

1. Go to the official Android Studio website:https://developer.android.com/studio.

2. Click on the "Download Android Studio" button.

3. Choose the version of Android Studio you want to download based on your operating system (Windows,macOS, or Linux).

4. Once the download is complete, open the downloaded fileand follow the installation wizard.

5. On the "Welcome to Android Studio" screen, click "Next".

6. Read and accept the terms of the license agreement, thenclick "Next".

7. Choose the installation location for Android Studio, or usethe default location, then click "Next".

8. Choose the components you want to install, then click"Next".

9. Choose the Start Menu folder name and click "Install".10.Once the installation is complete, click "Next".
11. Choose whether to import previous settings or not, thenclick "Finish".

12. Android Studio will launch automatically. You may needto install additional components, such as the Android SDK, if they are not already installed.

Step 2 : Create a project and set up on android studio

1. Open Android Studio on your computer.

2. On the Welcome screen, click on "Start a new AndroidStudio project" or go to File > New > New Project.

3. Choose the type of project you want to create, such as"Phone and Tablet" or "Wear OS".

4. Choose the minimum SDK you want to support.

5. Choose the template for your project, such as "EmptyActivity" or "Basic Activity".

6. Enter the name of your application in the "Applicationname" field.

7. Choose the package name for your application. This should be a unique identifier for your application, usually inreverse domain name format (e.g. com.example.myapp).

8. Choose the location where you want to save your project.9.Click "Finish" to create your project.
10.Once your project is created, you will see the project structure on the left-hand side of the screen, including folders for "app" (where you will write your application code), "res" (where you will store your resources such asimages and strings), and others.

Step 3 : Download and filter image of labor andconstruction workers for dataset

1. Choose a website to search for images, such as GoogleImages or Shutterstock.
2. In the search bar, enter keywords related to labor andconstruction workers, such as "construction worker," "factory worker," "laborer," or "warehouse worker."

3. Click on the "Images" tab to filter the search results toonly show images.

4. Use the "Tools" feature to further filter the images by size,color, and type. For example, you may want to filter the images to only show those with a minimum resolution of 1024x768 and in black and white.

5. Scroll through the images and select the ones that fit yourcriteria.

6. Download the selected images to a folder on yourcomputer.

7. Use an image editing software such as Adobe Photoshop or GIMP to filter out any irrelevant images or backgrounds,and crop the images to only show the worker in question.

8. Save the filtered images as separate files in a format that isappropriate for your dataset.

Step 4 : Create ZIP file of images and upload toRoboflow for annotation

1. Create a folder on your computer and add all the imagesthat you want to annotate to this folder.

2. Right-click on the folder and select "Compress" or "Createa ZIP file" from the menu. This will create a compressed ZIP file of the folder and its contents.

3. Go to the Roboflow website and sign in to your account.

4. Click on the "Datasets" tab at the top of the page and thenclick on "Create a dataset".

5. Enter a name for your dataset and select the "Upload"option.

6. Click "Select files" and choose the ZIP file of images thatyou created earlier.

7. Click "Upload" to start the upload process. The imageswill be uploaded and added to your dataset in Roboflow.

8. Once the upload is complete, you can use the Roboflow annotation tools to label and annotate the images as needed.

Step 5 : Selection TensorFlow ML mobile net algorithm

MobileNet is a popular convolutional neural network (CNN)architecture that is designed for mobile and embedded vision applications. It is known for its high accuracy and efficient performance, making it a popular choice for image classification tasks. Here are some steps to consider when selecting the MobileNet algorithm in TensorFlow:

Determine the requirements of your machine learning task: Consider the specific requirements of your task, such as accuracy, speed, and model size. MobileNet is optimized formobile devices and embedded systems, so it may be a good choice if you have limitations on computational power or memory. Choose the version of MobileNet: There are several versions of MobileNet available in TensorFlow, including MobileNet V1, MobileNet V2, and MobileNet V3. Each version has its own strengths and weaknesses, so consider which version would be the best fit for your task.

Select the pre-trained model or train from scratch: You caneither use a pre-trained MobileNet model from TensorFlowHub or train your own model from scratch. If you have a large dataset that is specific to your task, training your ownmodel from scratch may be the best option. Fine-tune the model: Depending on the specifics of your task, you may need to fine-tune the MobileNet model by adjusting hyperparameters or retraining certain layers. This can help optimize the model for your specific needs.

Evaluate the performance: Once you have selected the MobileNet algorithm and trained or fine-tuned the model, it is important to evaluate its performance on your test data. Use metrics such as accuracy, precision, recall, and F1 scoreto measure the performance of the model. Remember that selecting the right algorithm is just one part of the machine- learning process. Other factors such as data preprocessing, feature engineering, and model optimization can also have asignificant impact on the accuracy and performance of your model.

Step 5 : Implementation of Tensorflow mobile netalgorithm on Google Colab

1. Open Google Colab in your web browser and create a newnotebook.

2. In the first cell, import the TensorFlow library and checkthat you are using version 2.0 or later:
import tensorflow as tfprint(tf._version_)

3. Next, download the MobileNet model from TensorFlowHub using the following code:
import tensorflow_hub as hub
# Load MobileNet model from TensorFlow Hub model =
hub.KerasLayer('https://tfhub.dev/google/tf2-
preview/mobilenet_v2/feature_vector/4', input_shape=(224,224, 3))

4. Now you can create a new model that uses the MobileNetfeature extractor as its base:

```
# Create a new model that uses MobileNet as its basemobile_net_model = tf.keras.Sequential([
  model,
  tf.keras.layers.Dense(128, activation='relu'), tf.keras.layers.Dense(10, activation='softmax')
])
```

5. This creates a new model that adds a dense layer with 128units and a ReLU activation function, followed by another dense layer with 10 units and a softmax activation function.Compile the model using an appropriate loss function, optimizer, and metric:

```
# Compile the model mobile_net_model.compile(
    loss='categorical_crossentropy', optimizer=tf.keras.optimizers.Adam(lr=0.001), metrics=['accuracy']
)
```

6. Train the model on your dataset: # Train the model on your dataset history = mobile_net_model.fit(

```
    train_data, epochs=10,
    validation_data=val_data
```

1. In the Android Studio project, create a new folder named"assets" in the "main" folder if it doesn't already exist.

2. Copy the TensorFlow Lite model file (with .tflite extension) to the "assets" folder.In the "build.gradle" file ofyour app module, add the following code inside the "android" block:

```
android {
// ... aaptOptions {
        // ...
        // Include the .tflite file in the APK's assets directorynoCompress "tflite"
    }
}
```

This tells the build system not to compress the .tflite file,and to include it in the APK's assets directory.
)

Here, train_data and val_data are the training and validationdatasets, respectively.

7. Evaluate the model on the test set: # Evaluate the model on the test set test_loss, test_accuracy = mobile_net_model.evaluate(test_data) print('Test loss:', test_loss)
print('Test accuracy:', test_accuracy)

Step 6 : Converting .tf model to .tflite model

3. Sync the project with Gradle by clicking on the "SyncNow" button in the toolbar.

4. You can now access the TensorFlow Lite model file inyour Java code using the following code:

```
try {
// Load the TensorFlow Lite model from the asset folder Interpreter tflite = new Interpreter(loadModelFile(), newInterpreter.Options());
    // Run inference on the input data
    // ...
} catch (IOException e) {
// Handle the exception
```

1. First, make sure you have TensorFlow version 2.0 or laterinstalled.

2. Load your saved TensorFlow model:
```
import tensorflow as tf
model = tf.keras.models.load_model('saved_model_path')
```

3. Convert the model to a TensorFlow Lite model:# Convert the model to a TFLite model
```
converter = tf.lite.TFLiteConverter.from_keras_model(model) tflite_model = converter.convert()
```

4. Save the TensorFlow Lite model to a file:# Save the TFLite model to a file
```
with open('model.tflite', 'wb') as f:f.write(tflite_model)
```

5. You can also optimize the TensorFlow Lite model toreduce its size and increase its performance. Here's an example of how to do this:
```
# Optimize the TFLite model
converter.optimizations = [tf.lite.Optimize.DEFAULT]tflite_model = converter.convert()
# Save the optimized TFLite model to a file with open('model_optimized.tflite', 'wb') as f:
f.write(tflite_model)
```

Step 7 : Import tflite model file in the assets package toAndroid studio
```
}
private MappedByteBuffer loadModelFile() throwsIOException {
   // Open the model file as a stream AssetFileDescriptor fileDescriptor =
getAssets().openFd("model.tflite");
   // Get the file size
   long fileSize = fileDescriptor.getLength();
   // Map the file into memoryreturn new
FileInputStream(fileDescriptor.getFileDescriptor())
          .getChannel()
          .map(FileChannel.MapMode.READ_ONLY, fileDescriptor.getStartOffset(), fileSize);
}
```
This code loads the TensorFlow Lite model file from theassets folder, maps it into memory using a MappedByteBuffer, and creates a new Interpreter to run inference on the input data.

Step 8 : Create functionalities for object detection usingTensorFlow API classes

1. Open Android Studio and create a new project.

2. In the "Create New Project" window, enter the projectname and other details.
3. In the "Add an Activity" window, select the type of activity you want to create. For example, you can select"Empty Activity" to create a new blank activity.

4. Click on the "Finish" button to create the project andactivity.

5. In the Project Explorer panel on the left, expand the "res"folder and open the "layout" folder.

6. Open the activity_main.xml file. This file defines thelayout for the main activity.

7. In the "Design" view of the layout editor, you can drag and drop widgets from the Palette panel on the left to createyour UI. For example, you can add a TextView widget to display some text, and a Button widget to trigger some action.

8. Use the "Attributes" panel on the right to customize the properties of the widgets, such as the text, color, size, andlayout.

9. You can also switch to the "Text" view of the layout editorto edit the XML code directly.

10. Once you have created the UI, you can run the app in theemulator or on a physical device to see how it looks and behaves.

Step 9 : Create functionalities for object detection usingTensorFlow Api classes

1. In your Android Studio project, create a new Java class forthe object detection functionality. Let's call it "ObjectDetectionHelper".

2. Add the TensorFlow Lite Interpreter and Model classes tothe class.

```
import org.tensorflow.lite.Interpreter;
import org.tensorflow.lite.support.common.FileUtil;
```

3. Load the TensorFlow Lite model file from the assetsfolder.

```
private Interpreter tflite;
private void loadModelFile() throws IOException { ByteBuffer model = FileUtil.loadMappedFile(context,
"model.tflite");
   Interpreter.Options options = new Interpreter.Options();tflite = new Interpreter(model, options);
}
```

4. Define the input and output tensor shapes of the model.private final int NUM_DETECTIONS = 10;

```
private final int NUM_CLASSES = 3; private final int INPUT_SIZE = 224; private final
int IMAGE_MEAN = 128; private final float IMAGE_STD = 128.0f;

private final int[] intValues = new int[INPUT_SIZE *INPUT_SIZE];
private final float[][][][] inputBuffer = newfloat[1][INPUT_SIZE][INPUT_SIZE][3];
private final float[][] outputLocations = newfloat[1][NUM_DETECTIONS * 4];
private final float[][] outputClasses = new float[1][NUM_DETECTIONS * NUM_CLASSES];
private final float[][] outputScores = newfloat[1][NUM_DETECTIONS];
```

5. Define a method to preprocess the input image.private void preprocess(Bitmap bitmap) {

```
bitmap = Bitmap.createScaledBitmap(bitmap, INPUT_SIZE, INPUT_SIZE, false);
bitmap.getPixels(intValues, 0, bitmap.getWidth(), 0, 0,bitmap.getWidth(), bitmap.getHeight());
for (int i = 0; i < intValues.length; i++) {final int val = intValues[i];
inputBuffer[0][i / INPUT_SIZE][i % INPUT_SIZE][0] =(float) (((val >> 16) & 0xFF) - IMAGE_MEAN)
/ IMAGE_STD;
       inputBuffer[0][i / INPUT_SIZE][i % INPUT_SIZE][1]
= (float) (((val >> 8) & 0xFF) - IMAGE_MEAN) /IMAGE_STD;
                                                            inputBuffer[0][i / INPUT_SIZE][i % INPUT_SIZE][2]
                                                        = (float) ((val & 0xFF) - IMAGE_MEAN) / IMAGE_STD;
   }
}
```

6. Define a method to run object detection on the inputimage.

```
private void detectObjects(Bitmap bitmap) {preprocess(bitmap);
   tflite.run(inputBuffer, new Object[]{outputLocations,outputClasses, outputScores});
   // Process the output tensors to get the detection results
   // ...
}
```

7. Process the output tensors to get the detection results.private void processOutput() {

```
ArrayList<DetectedObject> detectedObjects = new ArrayList<>();
   for (int i = 0; i < NUM_DETECTIONS; i++) {float score = outputScores[0][i];
       if (score < 0.5) {continue;
       }
       float xPos = outputLocations[0][i * 4] * INPUT_SIZE;float yPos = outputLocations[0][i * 4 + 1] *
INPUT_SIZE;
       float width = (outputLocations[0][i * 4 + 2] -outputLocations[0][i * 4]) * INPUT_SIZE;
       float height = (outputLocations[0][i *
```

Step 10: Testing and Deployment of "APK".

1. Open your project in Android Studio.

2.Click on "Build" in the top menu and select "GenerateSigned Bundle / APK".

3.In the "Generate Signed Bundle / APK" dialog box, select"APK" and click on "Next".

4.Select the keystore you want to use to sign your APK andenter the necessary details, such as the keystore password, key alias, and key password. If you don't have a keystore, you can create one by clicking on "Create new".

5.Select the build type (Debug or Release) and the flavoryou want to build.

6.Click on "Finish" and wait for Android Studio to build theAPK. The APK file will be saved in the "app/build/outputs/apk" directory of your project.

7.You can install the APK on your Android device by copying it to your device and then opening the file.
Alternatively, you can use Android Studio to install the APK on your device by connecting your device to yourcomputer and clicking on "Run" in Android Studio.

That's it! You have successfully built and generated an APKfrom Android Studio.

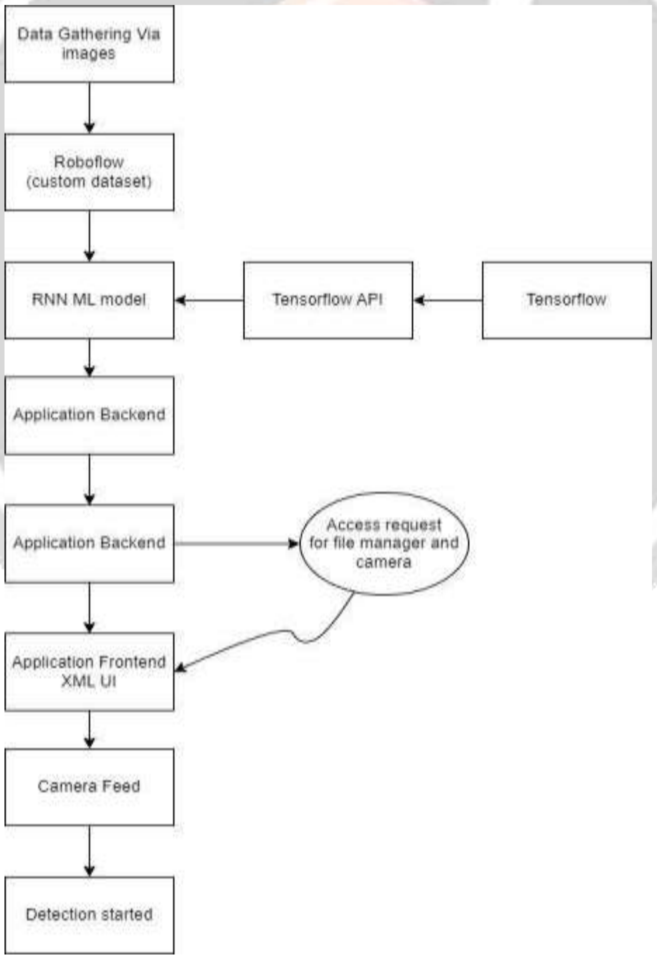
Data Gathering Via images
Roboflow (custom dataset)
RNN ML model
Tensorflow API
Tensorflow
Application Backend
Application Backend
Access request for file manager and camera
Application Frontend XML UI
Camera Feed
Detection started

Fig. 1. Flow Diagram for Proposed System Work

IV. DESIGN OF PROJECTS

System hardware was used to test the algorithm's real-time processing with field data. It consisted of Intel Core i7- 790@3.60GHz 8 CPU and Intel HD Graphics GPU with 8GB 1600MHz RAM. Ubuntu 16.0 base environment consists of python3, pip, OpenCV and Tensorflow. Image annotation is performed on the label using Image-master software, an open source software available on GitHub. The development was done with jupyter notebook and atomic text editor was used. The YOLOv7 model is used to convey learning in the spherical framework. The YOLOv7 trained weights are used as the initial set of CNN weights, and the convolutional and fully connected layers are opened for training using construction data. This is true for both Yolov7 and Faster RCNN models .

In the project firstly we have the environment setup then the algorithm are used i.e. YOLO (network modification for transfer learning) and TensorFlow. These are the library’s of python and use for object detection. The yolo is one type of machine learning in which the images are read in the form of matrix i.e. pixels in the form of (0,1). It is also called as annotation. The roboflow data set is use to detect the objects in the dataset many images are gather and than the images go to many phases that are training ,preparation, validation.

Roboflow is a computer vision platform that enables users to build computer vision models faster and more accurately by providing better data collection, pre-processing and model training techniques. Roboflow allows users to load custom datasets, draw annotations, edit image orientations, resize images, change image contrast, and perform data augmentations. It can also be used to train models. As I mentioned, Roboflow also has a generic conversion tool that allows users to load and convert annotations from one format to another without having to write conversion scripts for custom object detection datasets. First we need to login at https://roboflow.com/. Note that if you select the "Public Workspace" option when you sign up for a free tier account, you can upload image sets of up to 10,000 source images. After creating a project, the next step is to load a dataset containing both images and existing annotation files (can be in JSON, Txt or XML format) or design annotations from scratch. The last step after preparing the dataset is to export the data in ZIP format.

The Object Detection App for Labour Kit Detection on Construction Site has several features that make it ideal for use on construction sites. These features include:

Real-Time Detection: The app can detect workers' safety gear in real-time, allowing supervisors to monitor workers' safety gear usage continuously. This feature ensures that workers wear the necessary safety gear before entering the construction site, preventing accidents and injuries.

Alerts: Supervisors can set up alerts for when workers are not wearing the required safety gear. The app will send alerts to supervisors, notifying them of non-compliance with safety gear requirements.

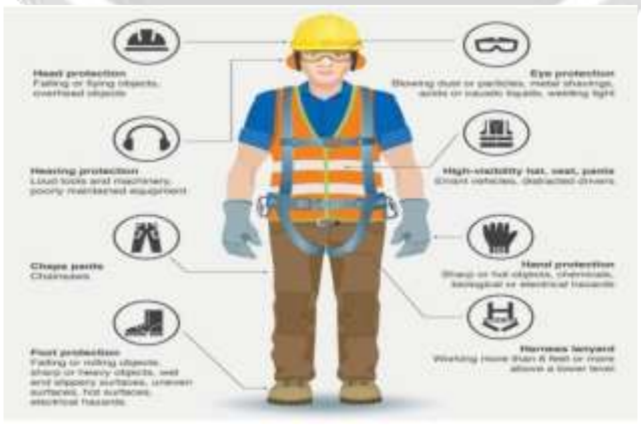

Fig. 2. Component Detection

Reporting: The app generates reports that supervisors can use to track compliance with safety gear requirements. The reports show which workers are complying with the safety gear requirements and which

workers are not.

Ease of Use: The app is user-friendly and easy to use. Supervisors can quickly learn how to use the app and start monitoring workers' safety gear usage.

The data gather from the many construction sources. Then the images go to dataset for the image detection (helmets, no helmets, jacket , no jacket).The models presents in databaseare trained then the output is show.

V. DISCUSSION

An Android app for labor PPE (Personal Protective Equipment) kit detection could be a useful tool to ensure the safety of workers in various industries, particularly those working in hazardous environments. The app could utilize the camera on a user's Android device to identify whether or not an individual is wearing the necessary PPE, such as a hard hat, gloves, goggles, or a respirator, before they enter ahazardous area.

One potential benefit of such an app is that it could improve compliance with PPE requirements, which could reduce the risk of workplace injuries and illnesses. For example, if an individual tries to enter a hazardous area without the proper PPE, the app could alert them and prevent them from accessing the area until they have the correct equipment. This could help to prevent accidents and injuries caused by individuals who forget or intentionally disregard PPE requirements.

Another potential benefit of the app is that it could simplify the process of tracking PPE usage and compliance. Employers could use the app to monitor PPE usage and compliance across their workforce, which could help them identify areas where additional training or enforcement is needed. Additionally, the app could be used to generate reports on PPE usage and compliance, which could be useful for regulatory compliance purposes.

However, there are also potential challenges and limitations to consider when developing an Android app for labor PPE kit detection. One key challenge is that the app would need to accurately detect the presence or absence of specific PPE items, which could be difficult if the items are not easily distinguishable from other objects or if the lighting conditions are poor. Additionally, the app would need to be able to distinguish between different types of PPE items, whichcould require complex image recognition algorithms.
Another challenge is that the app would need to be integrated with existing PPE policies and procedures, which could be complex depending on the industry and the specific PPE requirements. Employers would need to ensure that the app aligns with their current PPE policies and procedures, and that it does not create any additional administrative burden orcosts.

Overall, an Android app for labor PPE kit detection could be a useful tool to improve compliance with PPE requirements and simplify the process of tracking PPE usage and compliance. However, there are several challenges and limitations to consider, and the app would need to be carefully designed and implemented to ensure that it is effective and practical for the intended users.

The Object Detection App for Labour Kit Detection on Construction Site has several features that make it ideal for use on construction sites. These features include:

Real-Time Detection: The app can detect workers' safety gear in real-time, allowing supervisors to monitor workers' safety gear usage continuously. This feature ensures that workers wear the necessary safety gear before entering the construction site, preventing accidents and injuries.

Alerts: Supervisors can set up alerts for when workers are not wearing the required safety gear. The app will send alerts to supervisors, notifying them of non-compliance with safety gear requirements.

Reporting: The app generates reports that supervisors can use to track compliance with safety gear requirements. The reports show which workers are complying with the safety gear requirements and which workers are not.

Ease of Use: The app is user-friendly and easy to use. Supervisors can quickly learn how to use the app and start monitoring workers' safety gear usage.

VI. FUTURE SCOPE

Personal protective equipment (PPE) is critical for the safety of laborers working in hazardous environments. The proper use of PPE can reduce the risk of injury or illness in the workplace. In recent years, there has been a growing interestin developing technologies to help monitor PPE usage and ensure compliance.

An Android app for labor PPE kit detection has the potential to be a valuable tool in this regard. The app could use machine learning algorithms to detect whether workers are wearing the necessary PPE, such as helmets, gloves, safety shoes, and other protective gear. The app could also track usage data and provide feedback to both workers and supervisors, which canhelp improve compliance and reduce workplace injuries.

One potential use case for the app could be in the construction industry, where workers are often required to wear a variety of PPE to ensure their safety. The app could be integrated with wearable sensors, such as smart helmets or vests, to detect the presence of PPE and monitor workers in real-time. It could also provide alerts to supervisors if a worker is not wearing the required PPE or if there is an increased risk of injury based on environmental factors such as noise or temperature.

Another potential use case for the app could be in manufacturing plants, where workers are required to wear PPE to protect against hazards such as chemicals or machinery. The app could be integrated with existing safety systems to monitor compliance with PPE regulations and provide real-time feedback to workers and supervisors.

The development of an Android app for labor PPE kit detection would require collaboration between developers, machine learning experts, and safety professionals. The app would need to be trained on a large dataset of images and video footage to accurately detect the presence of PPE. It would also need to be tested in real-world environments to ensure that it can detect PPE under varying lighting and environmental conditions.

VII. CONCLUSION

An Android app for labor PPE kit detection has the potential to be a valuable tool for improving workplace safety and reducing the risk of injuries. With the increasing availability of machine learning technologies and wearable sensors, such an app could provide real-time feedback to workers and supervisors, helping to improve compliance with PPE regulations and reduce workplace injuries. However, the development of such an app would require collaboration between multiple stakeholders and rigorous testing in real- world environments to ensure its effectiveness.

The app performs object detection in real-time, allowing supervisors to monitor workers' safety gear usage continuously. The app's real-time detection feature uses the mobile device's camera to capture images continuously. The app's computer vision and object detection algorithms analyze these images in real-time to detect workers and identify their safety gear. The Object Detection App for Labour Kit Detection on Construction Site is an essential tool for construction site safety. The app's real-time detection, alerts, and reporting features make it easy for supervisors to monitor workers' safety gear usage and ensure compliance with safety gear requirements. The app is user-friendly and easy to use, making it an ideal tool for construction site supervisors.

REFFERNCE

P. F. Felzenszwalb et al., “Object detection with discriminatively trained part-based models,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 32, no. 9, pp. 1627–1645, Sep. 2010.

K.-K. Sung and T. Poggio, “Example-based learning for view-based human face detection,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 20, no. 1, pp. 39–51, Jan. 1998.

P. F. Felzenszwalb et al., “Object detection with discriminatively trained part-based models,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 32, no. 9, pp. 1627–1645, Sep. 2010. [2] K.-K. Sung and T. Poggio, “Example-based learning for view-based human face detection," IEEE Trans. Pattern Anal. Mach. Intell., vol. 20, no. 1, pp. 39–51, Jan. 1998.

D. Erhan, C. Szegedy, A. Toshev, et al, "Scalable object detection using deep neural networks," 2014 IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 2155-2162.

V Zeiler, Matthew D, and R. Fergus. "Visualizing and Understanding Convolutional Networks."European Conference on Computer Vision Springer (2014).