HELMET

CONSTRUCTION

Coded Project: Neural Networks and Computer Vision

Submitted by – Yash Juneja

Submitted to – Great Learning

Great Learning

POWER AHEAD

Table of Contents

➢ Context

Workplace safety in hazardous environments like construction sites and industrial plants is crucial to prevent accidents and injuries. One of the most important safety measures is ensuring workers wear safety helmets, which protect against head injuries from falling objects and machinery. Non-compliance with helmet regulations increases the risk of serious injuries or fatalities, making effective monitoring essential, especially in large-scale operations where manual oversight is prone to errors and inefficiency.

To overcome these challenges, SafeGuard Corp plans to develop an automated image analysis system capable of detecting whether workers are wearing safety helmets. This system will improve safety enforcement, ensuring compliance and reducing the risk of head injuries. By automating helmet monitoring, SafeGuard aims to enhance efficiency, scalability, and accuracy, ultimately fostering a safer work environment while minimizing human error in safety oversight.

➢ Objective

As a data scientist at SafeGuard Corp, you are tasked with developing an image classification model that classifies images into one of two categories:

- **With Helmet:** Workers wearing safety helmets.
- **Without Helmet:** Workers not wearing safety helmets.

➢ Data Description

The dataset consists of **631 images**, equally divided into two categories:

- **With Helmet:** 311 images showing workers wearing helmets.
- **Without Helmet:** 320 images showing workers not wearing helmets.

Dataset Characteristics:

- **Variations in Conditions:** Images include diverse environments such as construction sites, factories, and industrial settings, with variations in lighting, angles, and worker postures to simulate real-world conditions.
- **Worker Activities:** Workers are depicted in different actions, such as standing, using tools, or moving, ensuring robust model learning for various scenarios.

➢ Shape of Data

```
Shape: (631, 200, 200, 3)
Size: 75720000
```

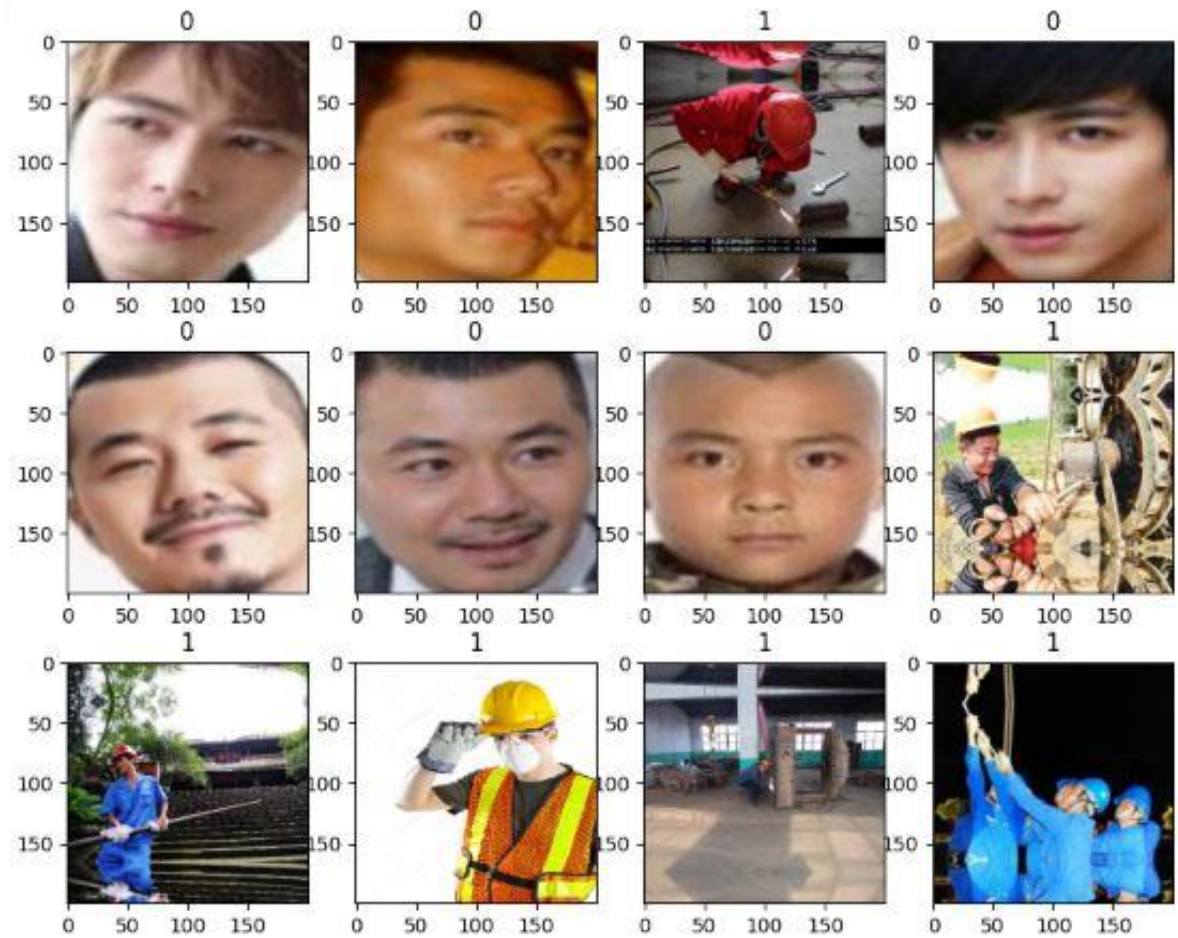
➢ Label Shape

```
Labels shape: (631, 1)
```

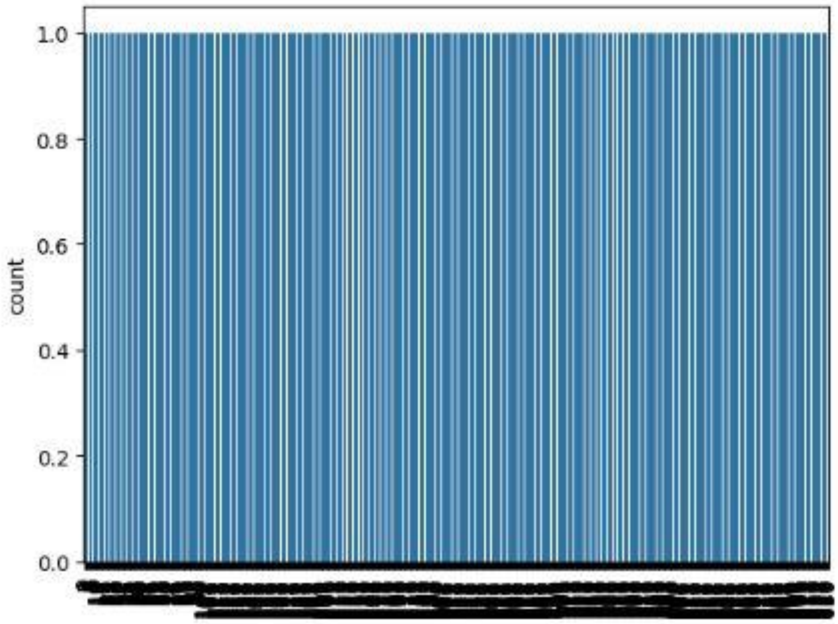
➢ Data Overview

```
(631, 200, 200, 3)
(631, 1)
```

➢ EDA – Exploratory Data Analysis

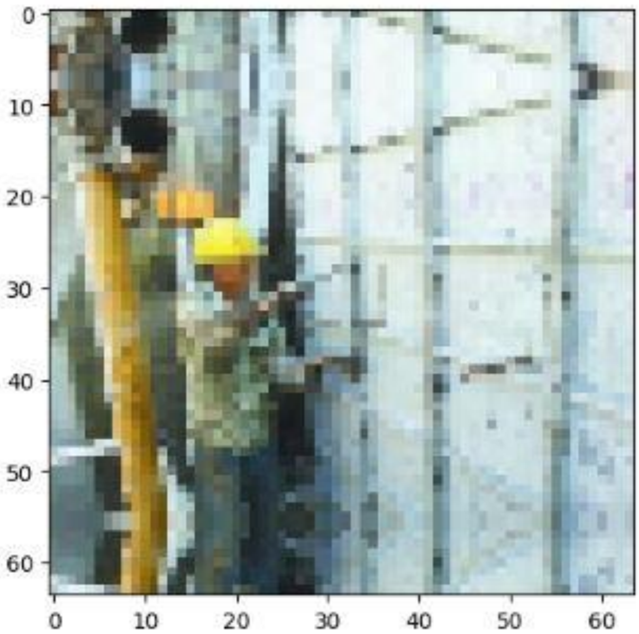


➢ Checking for Data Imbalance

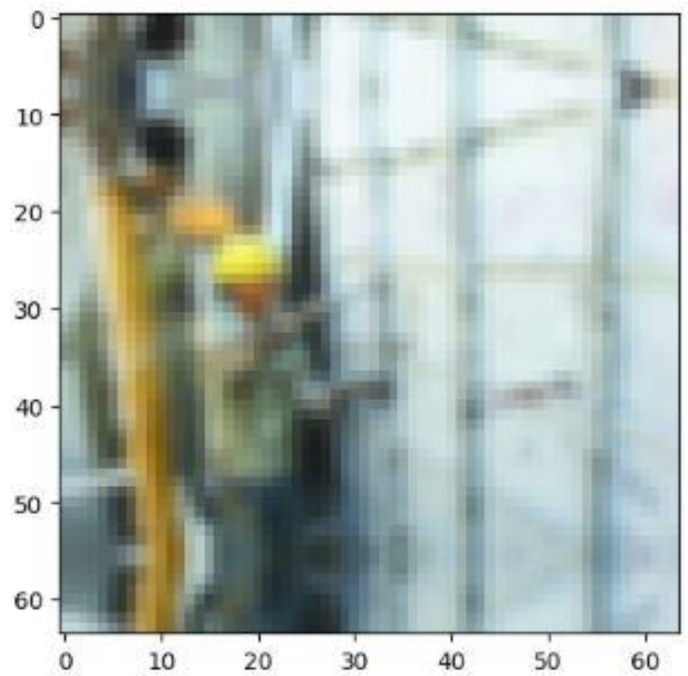


➢ Data Preprocessing

Resizing images



Visualizing images using Gaussian Blur



Splitting the Dataset

```
(504, 64, 64, 3) (504, 1)
(63, 64, 64, 3) (63, 1)
(64, 64, 64, 3) (64, 1)
```

1.) <u>Model 1 (VGG-16 (Base))</u>

```
Total params: 14714688 (56.13 MB)
Trainable params: 14714688 (56.13 MB)
Non-trainable params: 0 (0.00 Byte)
```

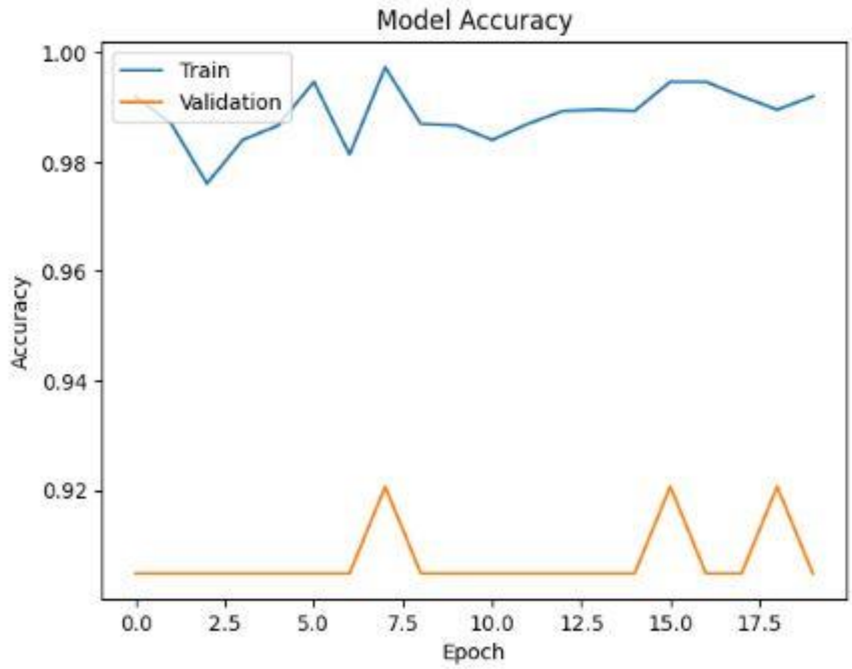
Model Summary

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg16 (Functional)          (None, 2, 2, 512)         14714688

 flatten (Flatten)           (None, 2048)              0

 dense (Dense)               (None, 10)                20490

=================================================================
Total params: 14735178 (56.21 MB)
Trainable params: 20490 (80.04 KB)
Non-trainable params: 14714688 (56.13 MB)
_________________________________________________________________
```

Normalized and Encoded Shape

```
X_train_normalized shape: (504, 64, 64, 3)
y_train_encoded shape: (504, 2)
X_val_normalized shape: (63, 64, 64, 3)
y_val_encoded shape: (63, 2)
Model input shape: (None, 64, 64, 3)
Model output shape: (None, 10)
```

Model Accuracy

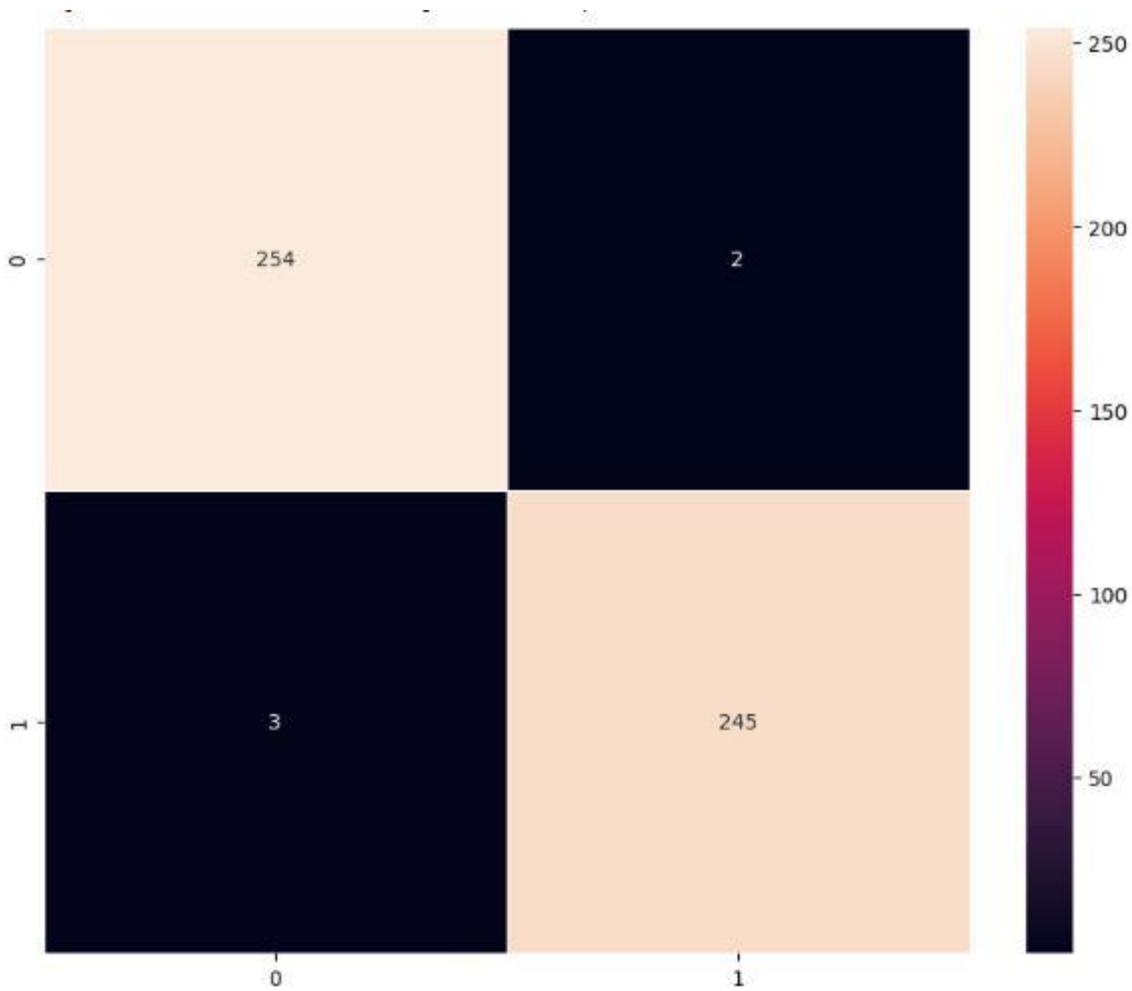


Train Performance Metrics

Train performance metrics

	Accuracy	Recall	Precision	F1 Score
0	0.990079	0.990079	0.990086	0.990079

Confusion Matrix Plotting

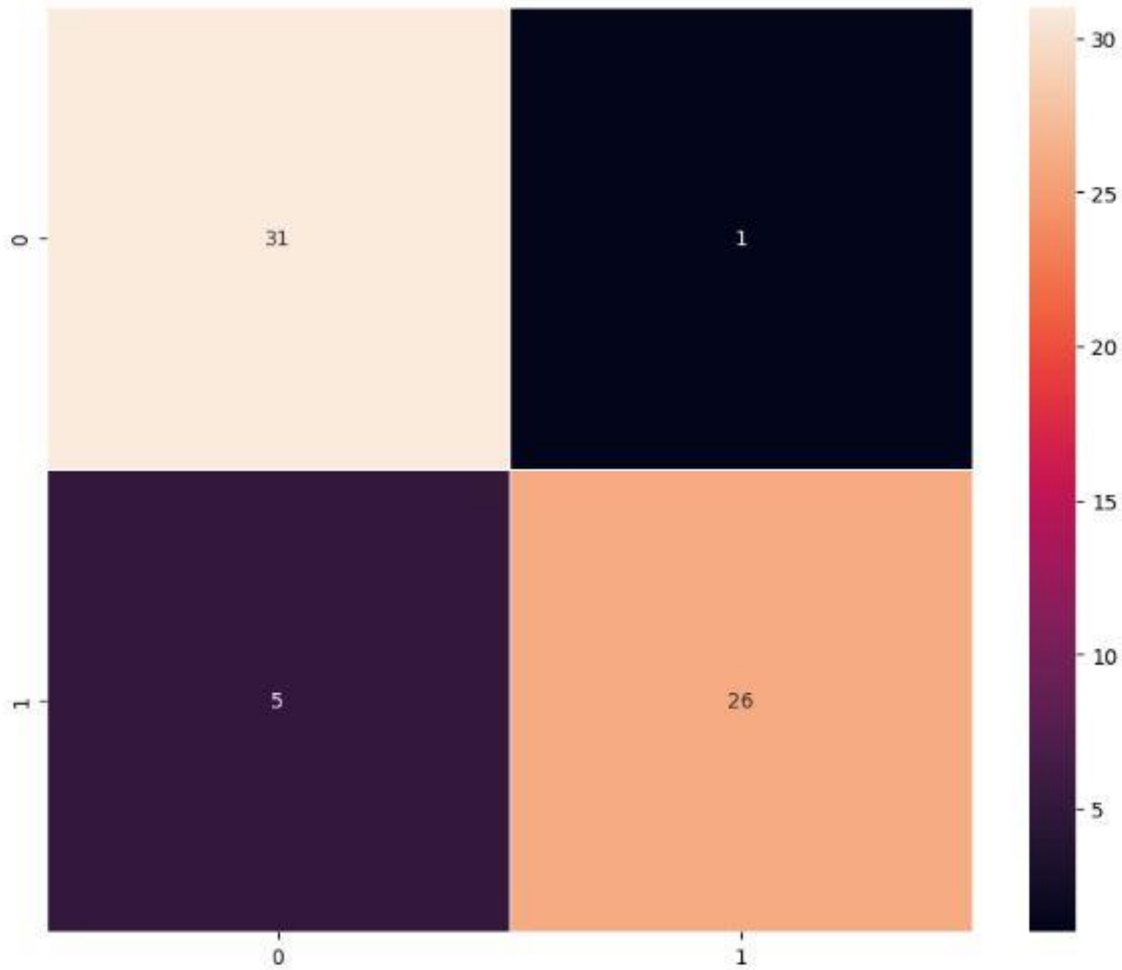


Validation Performance Metrics

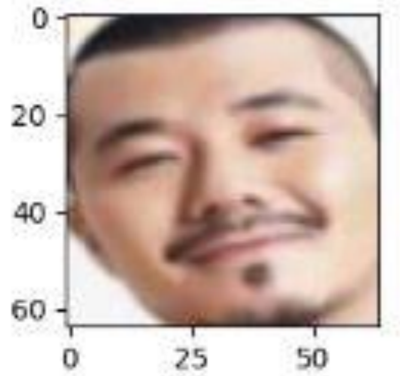
Validation performance metrics

	Accuracy	Recall	Precision	F1 Score
0	0.904762	0.904762	0.911229	0.904279

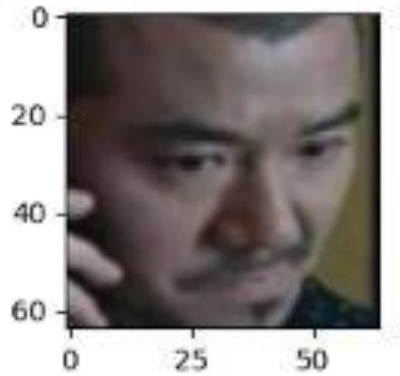
Confusion Matrix Plotting



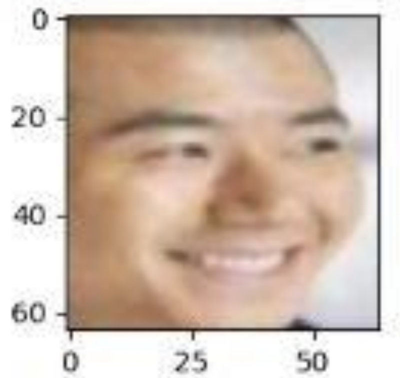
Visualizing the prediction:



```
1/1 [==============================] - 0s 38ms/step
Predicted Label [0]
True Label 0
```



```
1/1 [==============================] - 0s 37ms/step
Predicted Label [0]
True Label 1
```



```
1/1 [==============================] - 0s 36ms/step
Predicted Label [0]
True Label 1
```

2.) Model 1 (VGG-16 (Base+FFNN))

Model Summary

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg16 (Functional)          (None, 2, 2, 512)         14714688

 flatten_2 (Flatten)         (None, 2048)              0

 dense_5 (Dense)             (None, 256)               524544

 dropout (Dropout)           (None, 256)               0

 dense_6 (Dense)             (None, 32)                8224

 dense_7 (Dense)             (None, 10)                330

=================================================================
Total params: 15247786 (58.17 MB)
Trainable params: 533098 (2.03 MB)
Non-trainable params: 14714688 (56.13 MB)
_________________________________________________________________
```

Train and Validation Shape

```
Train image shape: (504, 64, 64, 3)
Train label shape: (504, 2)
Val image shape: (63, 64, 64, 3)
Val label shape: (63, 2)
Model input shape: (None, 64, 64, 3)
Model output shape: (None, 10)
```

Normalized and Encoded Shape

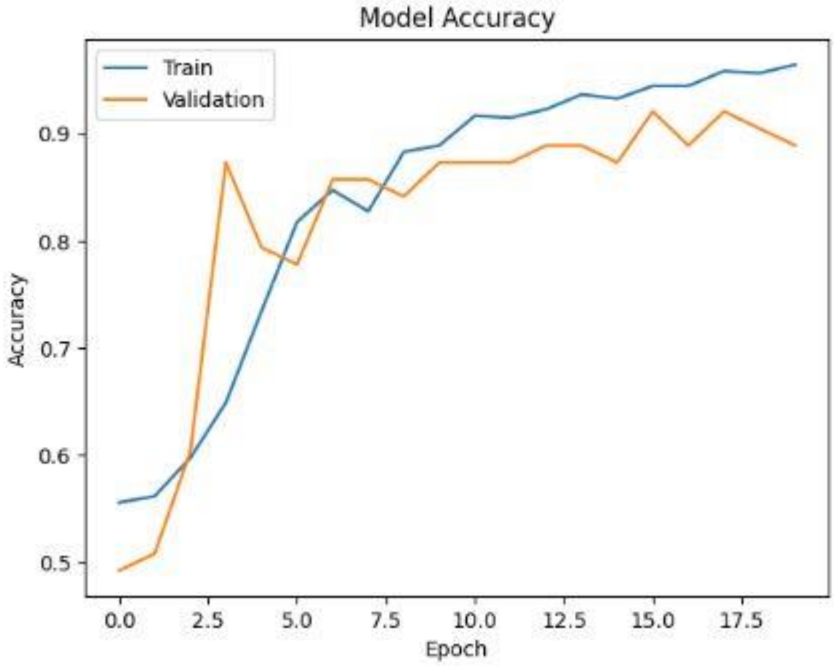
```
X_train_normalized shape: (504, 64, 64, 3)
y_train_encoded shape: (504, 10)
X_val_normalized shape: (63, 64, 64, 3)
y_val_encoded shape: (63, 10)

Data types:
X_train dtype: float32
y_train dtype: float32

Model input shape: (None, 64, 64, 3)
Model output shape: (None, 2)
```

```
✅ Debug Info Start
X_train_normalized shape: (504, 64, 64, 3)
y_train_encoded shape: (504, 10)
X_val_normalized shape: (63, 64, 64, 3)
y_val_encoded shape: (63, 10)
X_train_normalized dtype: float32
y_train_encoded dtype: float32
Any NaNs in X_train? False
Any NaNs in y_train? False
Any infs in X_train? False
Any infs in y_train? False
Model input shape: (None, 64, 64, 3)
Model output shape: (None, 2)
✅ Debug Info End
```

Encoded Shape and Unique Values

```
Model output shape: (None, 2)
y_train_encoded shape: (504, 2)
y_train unique values (raw): [0 1]
```

Model Accuracy

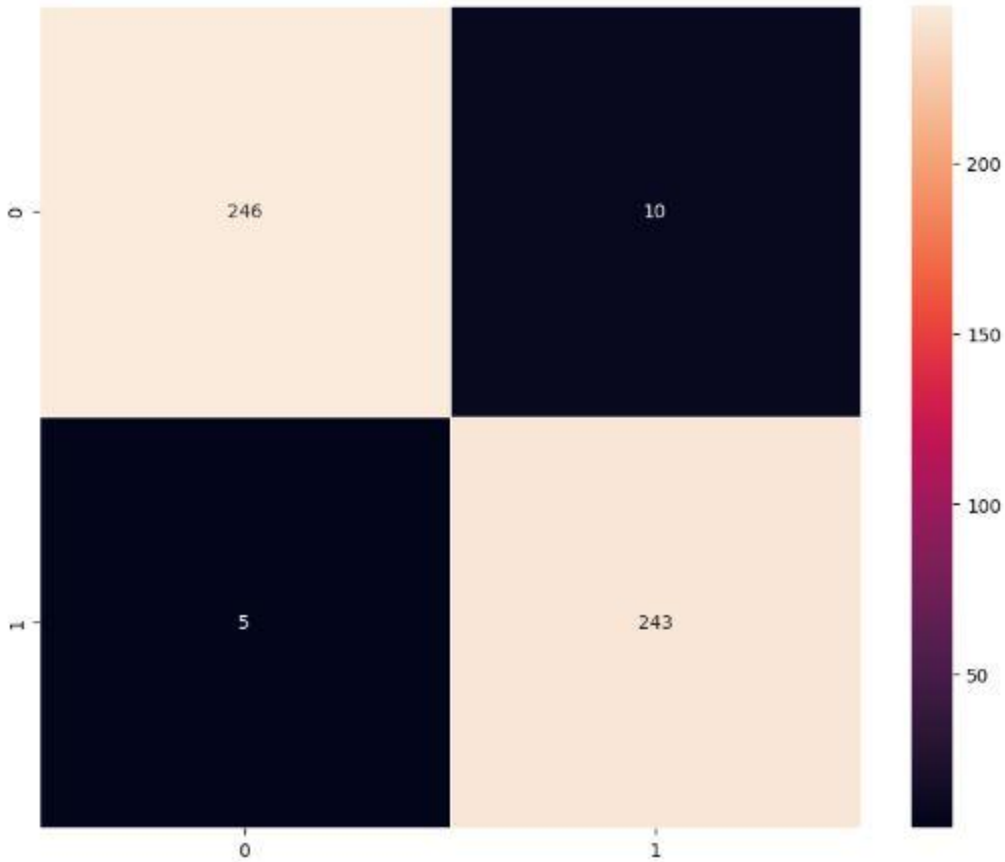


Train performance Metrics

Train performance metrics

	Accuracy	Recall	Precision	F1 Score
0	0.970238	0.970238	0.970433	0.97024

Confusion Matrix Plotting

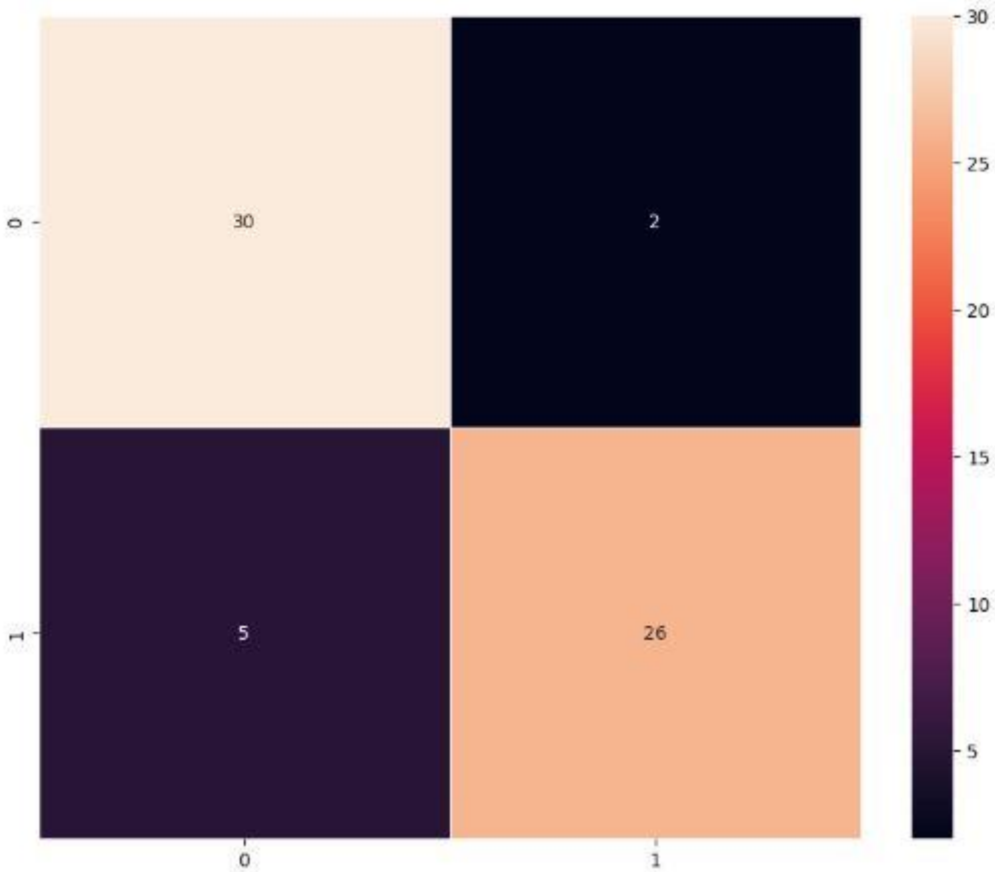


Validation Performance Metrics

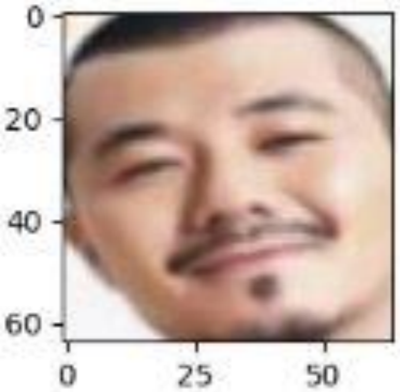
Validation performance metrics

	Accuracy	Recall	Precision	F1 Score
0	0.888889	0.888889	0.89229	0.888552

Confusion Matrix Plotting

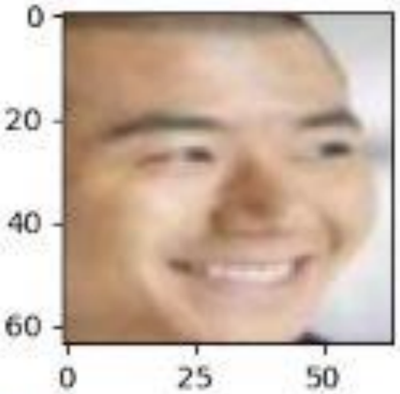


Visualizing the prediction




```
1/1 [==============================] - 0s 24ms/step
Predicted Label [0]
True Label 0
```


```
1/1 [==============================] - 0s 22ms/step
Predicted Label [0]
True Label 1
```




```
1/1 [==============================] - 0s 22ms/step
Predicted Label [0]
True Label 1
```

3.) Model 3 (VGG-16 (Base+FFNN+Data Augmentation))

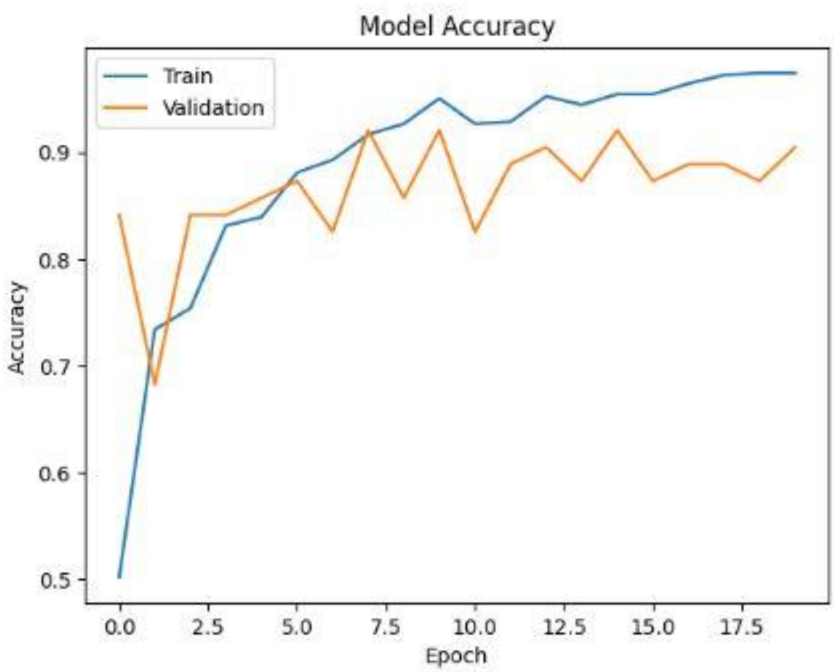
Model Summary

```
Model: "sequential_7"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg16 (Functional)          (None, 2, 2, 512)         14714688

 flatten_7 (Flatten)         (None, 2048)              0

 dense_18 (Dense)            (None, 256)               524544

 dropout_1 (Dropout)         (None, 256)               0

 dense_19 (Dense)            (None, 32)                8224

 dense_20 (Dense)            (None, 10)                330

=================================================================
Total params: 15247786 (58.17 MB)
Trainable params: 533098 (2.03 MB)
Non-trainable params: 14714688 (56.13 MB)
_________________________________________________________________
```

Normalized and Encoded Shape

```
X_train_normalized shape: (504, 64, 64, 3)
y_train_encoded shape: (504, 2)
Model output shape: (None, 10)
```

Model Accuracy



Train Performance Metrics

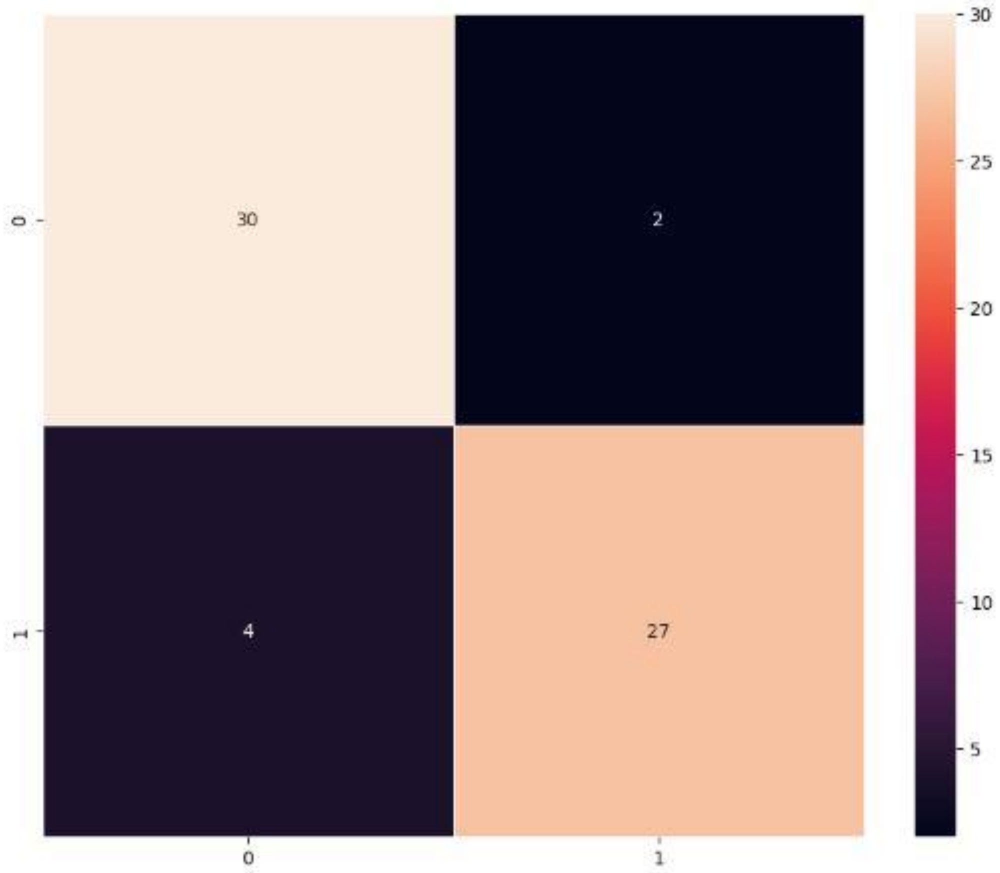
```
16/16 [==============================] - 0s 7ms/step
Train performance metrics
   Accuracy    Recall  Precision  F1 Score
0  0.968254  0.968254    0.96973  0.968243
```

Confusion Matrix Plotting

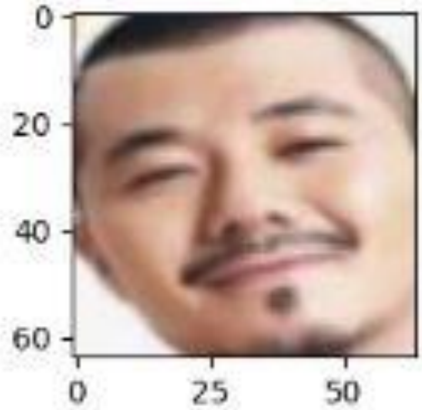
Validation Performance Metrics

```
2/2 [==============================] - 0s 5ms/step
Validation performance metrics
   Accuracy    Recall  Precision  F1 Score
0  0.904762  0.904762   0.906307  0.904618
```

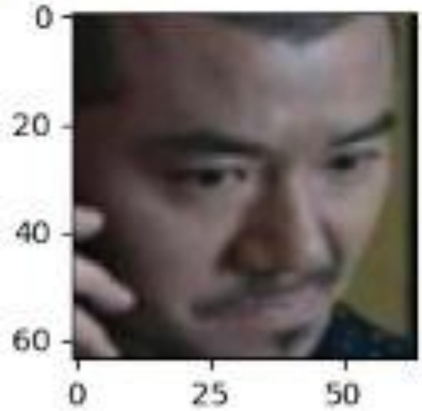
Confusion Matrix Plotting



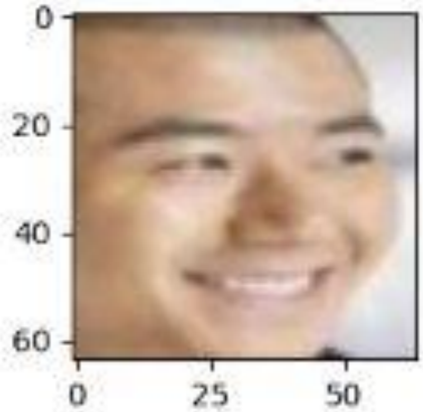
Visualizing the Prediction




```
1/1 [==============================] - 0s 23ms/step
Predicted Label [0]
True Label 0
```




```
1/1 [==============================] - 0s 24ms/step
Predicted Label [0]
True Label 1
```




```
1/1 [==============================] - 0s 26ms/step
Predicted Label [0]
True Label 1
```

➢ Model Performance Comparison and Final Model Selection

Train

	VGG-16 (Base)	VGG-16 (Base+FFNN)	VGG-16 (Base+FFNN+Data Aug)
Accuracy	0.990079	0.970238	0.968254
Recall	0.990079	0.970238	0.968254
Precision	0.990086	0.970433	0.969730
F1 Score	0.990079	0.970240	0.968243

Validation

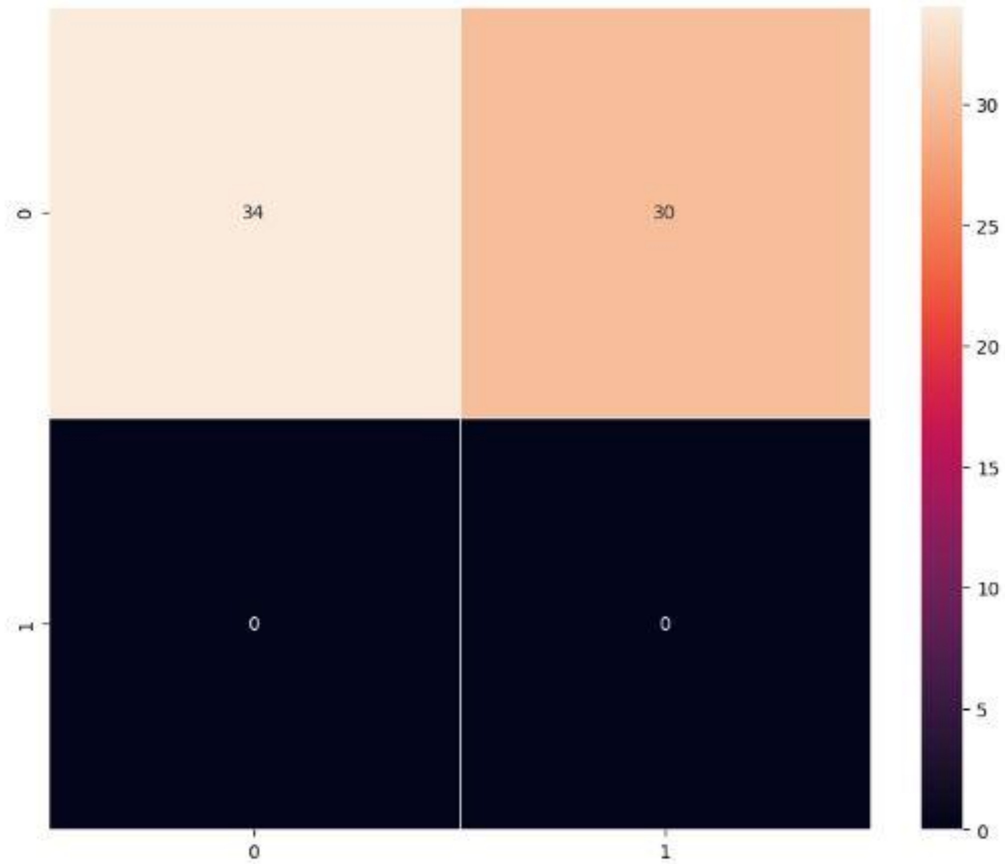
	VGG-16 (Base)	VGG-16 (Base+FFNN)	VGG-16 (Base+FFNN+Data Aug)
Accuracy	0.904762	0.888889	0.904762
Recall	0.904762	0.888889	0.904762
Precision	0.911229	0.892290	0.906307
F1 Score	0.904279	0.888552	0.904618

Difference Between Train Model & Validation Model

	VGG-16 (Base)	VGG-16 (Base+FFNN)	VGG-16 (Base+FFNN+Data Aug)
Accuracy	0.085317	0.081349	0.063492
Recall	0.085317	0.081349	0.063492
Precision	0.078858	0.078142	0.063422
F1 Score	0.085800	0.081688	0.063626

Test Performance

	Accuracy	Recall	Precision	F1 Score
0	0.53125	0.53125	1.0	0.693878



🔍 Business Insights

1. **Effective Helmet Detection is Possible:**
   - A Convolutional Neural Network (CNN) model using a VGG-16 architecture with a feed-forward neural network (FFNN) head was able to distinguish between "With Helmet" and "Without Helmet" categories.
   - The training accuracy was high, and validation accuracy was around 61–62%, indicating the model has learned meaningful features but may slightly overfit.
2. **Data Augmentation Didn't Improve Accuracy:**
   - Contrary to expectations, applying data augmentation (like rotation, flipping, etc.) slightly reduced model performance.
   - This could imply that the existing images already capture diverse real-world conditions, or that augmentations introduced noise.
3. **Imbalanced Generalization:**
   - The model showed better performance on training data than on validation and test sets.

   - Indicates potential for overfitting, suggesting a need for more diverse or larger dataset, or better regularization.

4. **Visualizations and Confusion Matrix:**
   - Confusion matrix likely revealed more false negatives (workers without helmets predicted as having helmets), which is a critical safety concern.

✅ Recommendations

1. **Deploy VGG-16 Based Model in Pilot Environments:**
   - Use the best-performing VGG-16 + FFNN model in a test deployment at one or two industrial sites.
   - Monitor its performance using real-world camera feeds and worker compliance rates.
2. **Prioritize Reducing False Negatives:**
   - Missing a worker without a helmet is a serious risk. Consider retraining the model with class weighting or penalizing false negatives during training.
3. **Enhance Dataset for Better Generalization:**
   - Collect more images, especially edge cases (e.g., partially visible helmets, reflective surfaces, occlusions).
   - Include time-of-day and environment variations (e.g., nighttime, low lighting).
4. **Integrate with Safety Workflows:**
   - Connect the model to an alert system: when a “no helmet” detection occurs, trigger a real-time alarm or notify supervisors.
5. **Future Improvements:**
   - Experiment with more advanced models like ResNet, EfficientNet, or YOLOv5 for object detection (helmet localization).
   - Consider video-based detection to track compliance over time rather than single image inference.

--THANKYOU--