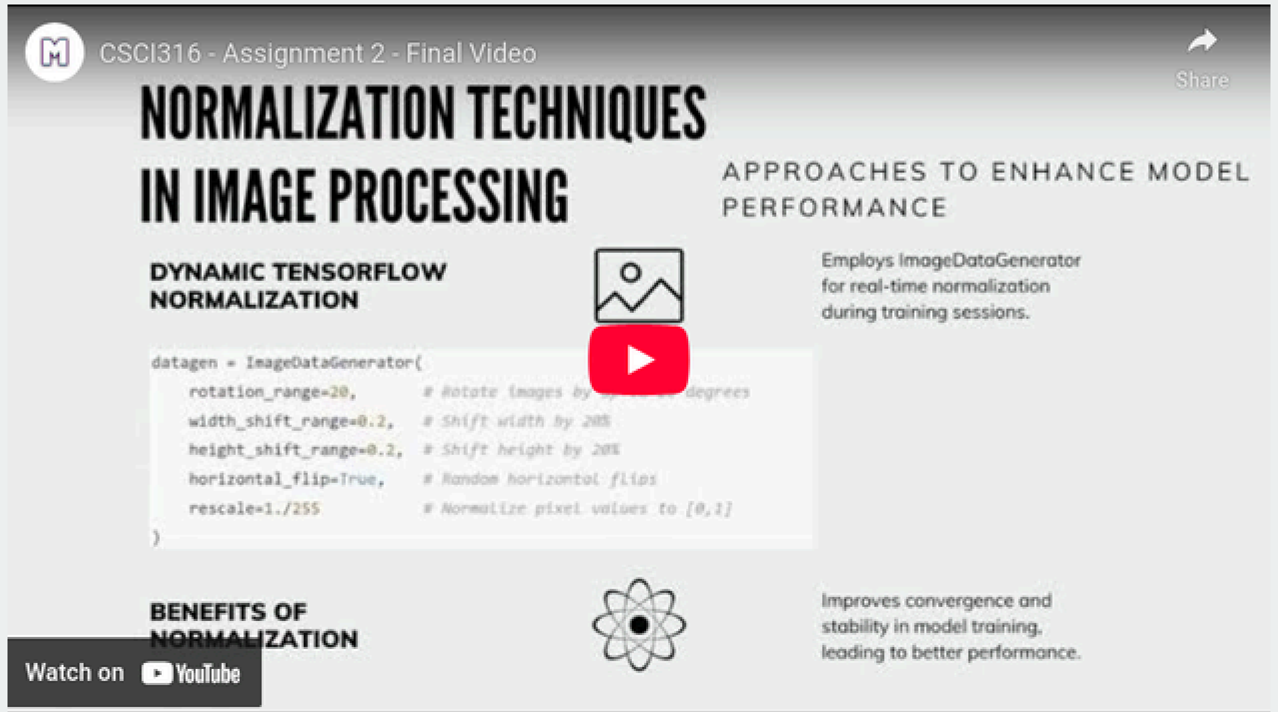
MOHAMMED SHADI - 7736356
HAIFA SHKHEDEM- 7820197
EMAN YAHYA- 7773225
MIKAEEL FARAZ SAFDAR - 8074689
MUHAMMAD BISHAM ADIL PARACHA - 7935407
MUHAMMAD SHAHEER KASHIF - 7877146
RABAIL LAL - 7778144
HIBA NASIR- 7788745
SAAD MUSADDIQ - 7883766
YASIN SHEIKH - 8140352

ENHANCING IMAGE CLASSIFICATION WITH TRANSFER LEARNING

A COMPREHENSIVE EXPLORATION OF HOW TRANSFER LEARNING, PARTICULARLY INCEPTIONV3, CAN SIGNIFICANTLY IMPROVE IMAGE CLASSIFICATION TASKS IN PRACTICAL APPLICATIONS.



https://youtu.be/qn3jjj9sXQY?si=uZL2eC2zFUVpdEUW

TRANSFER LEARNING IN IMAGE CLASSIFICATION

LEVERAGING TRANSFER LEARNING FOR ENHANCED CLASSIFICATION

1

APPLICATION IN IMAGE CLASSIFICATION

It enhances image classification tasks by applying pre-trained models to new datasets.

2

DEFINITION OF TRANSFER LEARNING

Transfer Learning is a technique where a model is reused for a second task, improving efficiency.

3

USE OF PRE-TRAINED MODELS

Models like InceptionV3, pre-trained on large datasets, serve as effective starting points.

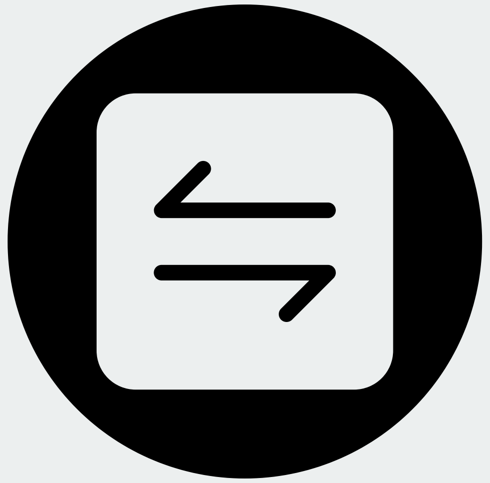
4

FOCUS ON PRACTICAL IMPLEMENTATION

We will explore practical steps to implement Transfer Learning using InceptionV3.

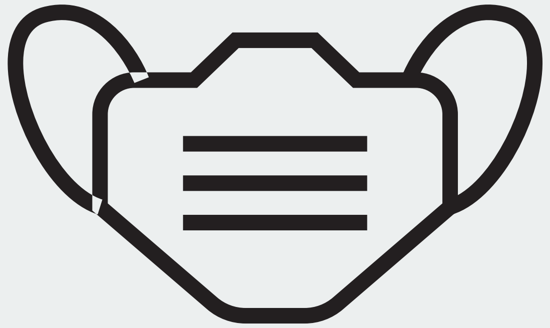
TRANSFER LEARNING PROJECT OVERVIEW

IMAGE CLASSIFICATION CHALLENGE



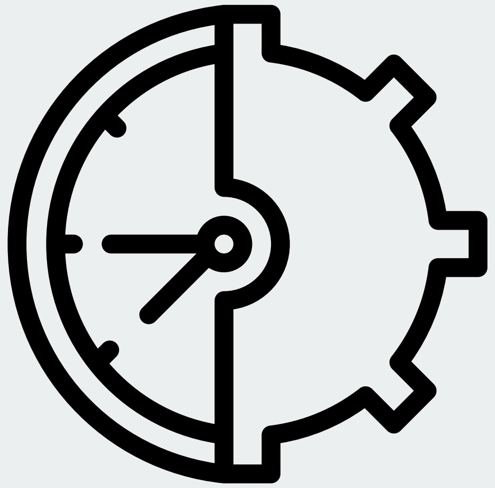
PRIMARY GOAL

Provide a comprehensive understanding of Transfer Learning for image classification.



FACE MASK DETECTION DATASET

Images are classified into three categories: mask, no mask, and incorrect mask.



MODEL FINE-TUNING

Evaluate the effectiveness of fine-tuning a pre-trained model (InceptionV3) for image classification.

BENEFITS OF TRANSFER LEARNING

KEY ADVANTAGES

1. **EFFICIENCY**
   Utilising pre-trained models like InceptionV3 saves significant time and resources in training.

2. **OVERFITTING MITIGATION**
   Transfer Learning minimises overfitting risks by leveraging generalised features from pre-trained models.

3. **HIGH ACCURACY**
   Models like InceptionV3 achieve high accuracy even with limited data due to prior training on extensive datasets.

4. **ADAPTABILITY**
   InceptionV3 can be easily tailored by replacing its classification layers for specific tasks.

5. **PROVEN EFFECTIVENESS**
   Transfer Learning has demonstrated success in various fields such as medical imaging and object detection.

CATEGORIZING IMAGES FROM XML ANNOTATIONS

CATEGORISATION OF IMAGES

Images were organised based on XML annotations into respective categories.

STEPS TO EXTRACT LABELS FROM XML FILE

- Read each XML file.
- Extract the image filename.
- Extract the object field, which contained the category (with_mask, without_mask, mask_weared_incorrect).
- Map these labels to our dataset categories (mask, no_mask, incorrect_mask).

```
# Ensure category folders exist
categories = ["mask", "no_mask", "incorrect_mask"]
for category in categories:
    os.makedirs(os.path.join(train_dir, category), exist_ok=True)

# Function to move images safely
def move_image_safe(src_path, dest_path):
    if os.path.exists(src_path):
        shutil.move(src_path, dest_path)  # Change copy → move
        print(f" Moved {src_path} → {dest_path}")
    else:
        print(f" Image not found: {src_path}")

# Process all XML files
for xml_file in os.listdir(annotation_dir):
    if xml_file.endswith(".xml"):
        xml_path = os.path.join(annotation_dir, xml_file)

        # Parse XML file
        tree = ET.parse(xml_path)
        root = tree.getroot()
```

📷 Sample images from mask category:

📷 Sample images from no_mask category:

📷 Sample images from incorrect_mask category:

PREPROCESSING FOR EFFECTIVE TRAINING

DATA COLLECTION

Selected and ensured usability of datasets for Transfer Learning.

DATA CLEANING

Addressed missing or irrelevant data to ensure a high-quality dataset for model training.

IMPORTANCE OF CLEAN DATA

Maintaining a clean dataset enhances model performance and generalization capabilities.

PROCEDURES OF CLEANING MASK DETECTION DATA SET

LOADING THE DATASET

- Images and XML annotation files (bounding boxes) were loaded.
- Annotation files were checked for accuracy.

DELETING CORRUPT/UNREADABLE IMAGES

- Used Python's PIL (Pillow) library to check image integrity.
- Removed corrupt images to avoid processing issues.

CONVERTING IMAGES TO STANDARD FORMATS

- Converted images to WebP format (JPEG/PNG for compatibility).
- Improved storage efficiency and loading speed.

RESIZING IMAGES

- Scaled all images to 512x512 pixels using OpenCV.
- Ensured consistent input dimensions for the model.

VERIFYING ANNOTATION FILES

- Checked XML files to ensure bounding boxes were within image dimensions.
- Removed annotations for missing or corrupt images.

STRUCTURING AND EXPORTING CLEAN DATA

- Moved cleaned images to a new directory (final_clean_data).
- Zipped the dataset for easy access and sharing.

NORMALIZATION TECHNIQUES IN IMAGE PROCESSING

APPROACHES TO ENHANCE MODEL PERFORMANCE

OPENCV NORMALIZATION TECHNIQUE

Utilises float32 conversion, scaling values from 0-255 to 0,1 for enhanced processing.

- **Rotation**

```
# Function to apply OpenCV-based augmentations
def augment_image(image):
    # Random rotation
    angle = random.choice([0, 90, 180, 270])
    rotated = cv2.rotate(image, {0: cv2.ROTATE_90_CLOCKWISE,
                                 90: cv2.ROTATE_180,
                                 180: cv2.ROTATE_90_COUNTERCLOCKWISE,
                                 270: cv2.ROTATE_180}[angle])
    return rotated
```

- **Flipping**

```
def flip_image(image):
    flipped_h = cv2.flip(image, 1)  # Horizontal flip
    flipped_v = cv2.flip(image, 0)  # Vertical flip
    return flipped_h, flipped_v
```

- **Resizing**

```
def resize_image(image, target_size=(128, 128)):
    return cv2.resize(image, target_size)
```

NORMALIZATION TECHNIQUES IN IMAGE PROCESSING

APPROACHES TO ENHANCE MODEL PERFORMANCE

DYNAMIC TENSORFLOW NORMALIZATION

Employs ImageDataGenerator for real-time normalization during training sessions.

```
datagen = ImageDataGenerator(
    rotation_range=20,        # Rotate images by up to 20 degrees
    width_shift_range=0.2,    # Shift width by 20%
    height_shift_range=0.2,   # Shift height by 20%
    horizontal_flip=True,     # Random horizontal flips
    rescale=1./255            # Normalize pixel values to [0,1]
)
```

BENEFITS OF NORMALIZATION

Improves convergence and stability in model training, leading to better performance.

DATA PREPARATION FOR MODEL TRAINING

STEPS TO SPLIT THE DATASET EFFECTIVELY

TRAINING SET COMPOSITION

70% of the data is allocated for training the model.

DATA SPLITTING STRATEGY

The dataset was split into three subsets for effective model training.

TEST SET IMPORTANCE

10% of the data is reserved for evaluating the model's performance.

VALIDATION SET PURPOSE

20% is used for hyperparameter tuning to improve model performance.

CLASS DISTRIBUTION ANALYSIS

Class distribution was plotted to verify dataset balance.

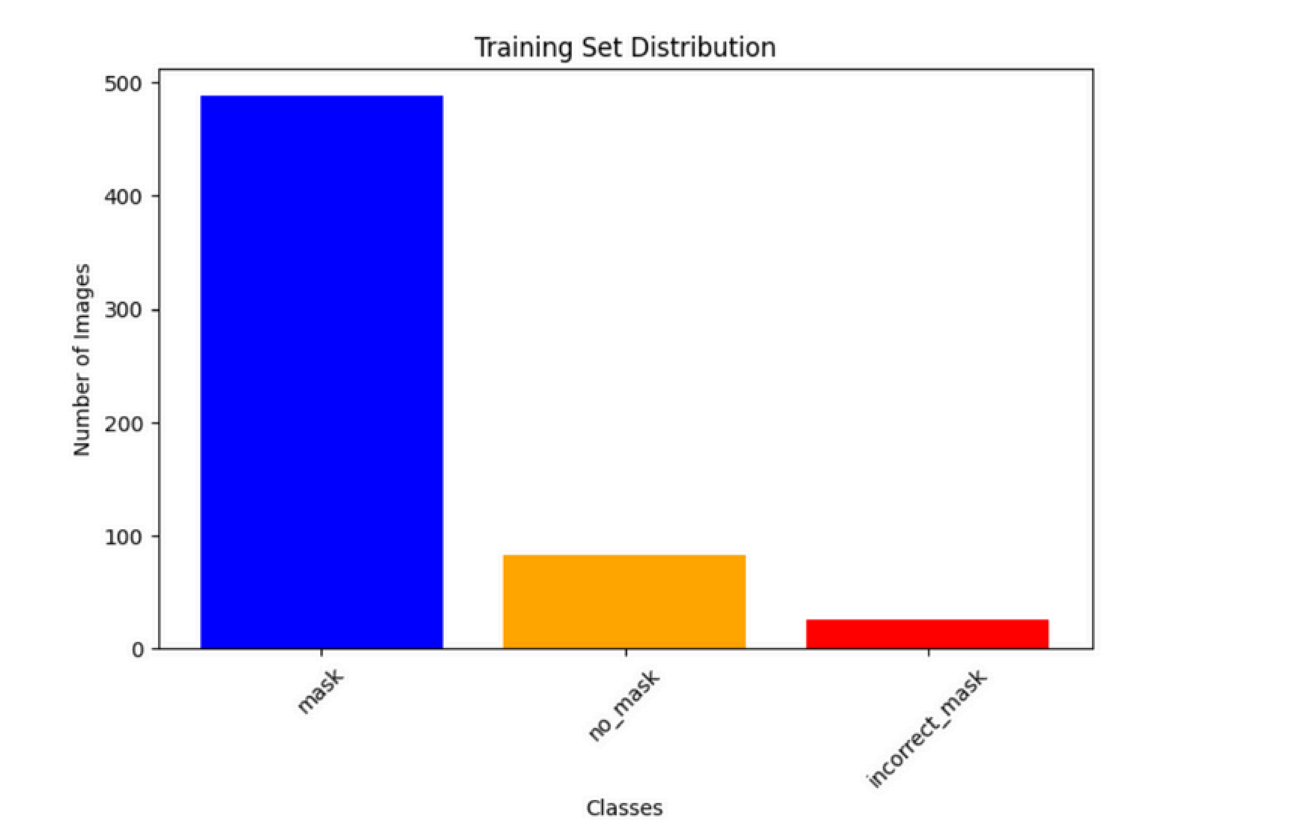
SYSTEMATIC APPROACH BENEFITS

This structured method ensures the model is well-prepared for training and evaluation.

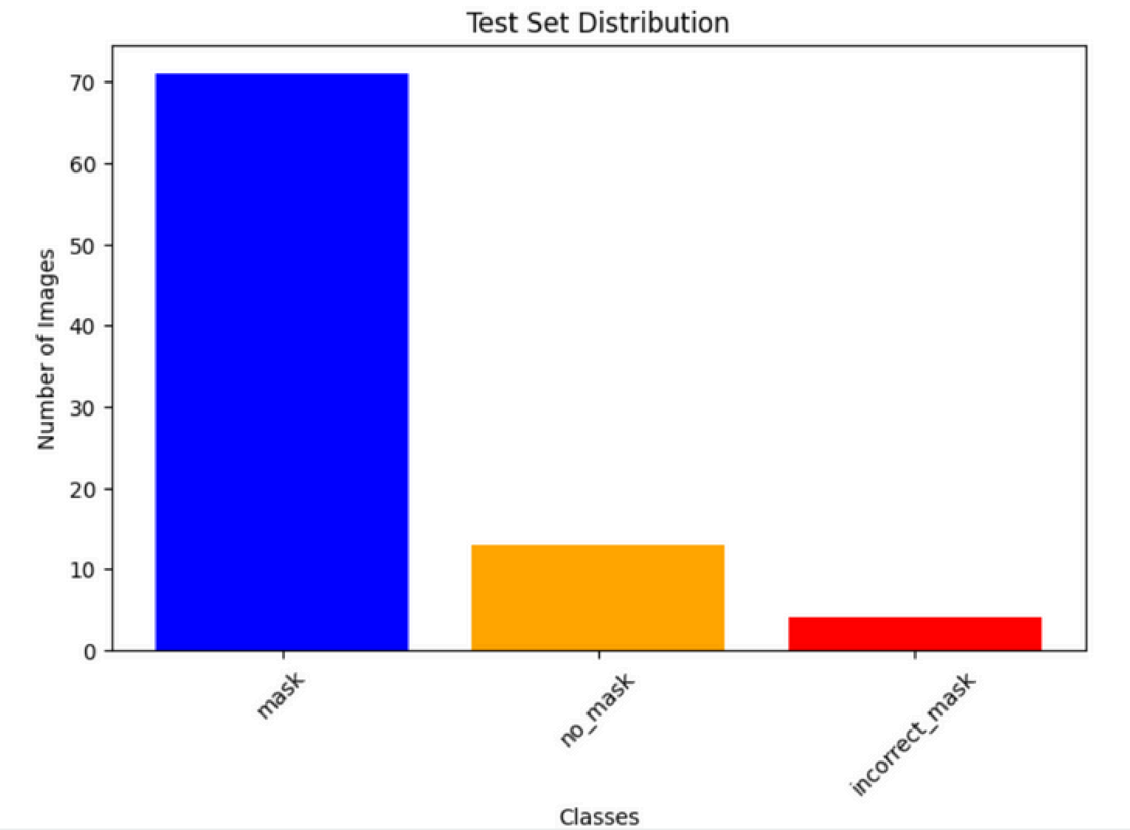
DATA PREPARATION FOR MODEL TRAINING

STEPS TO SPLIT THE DATASET EFFECTIVELY

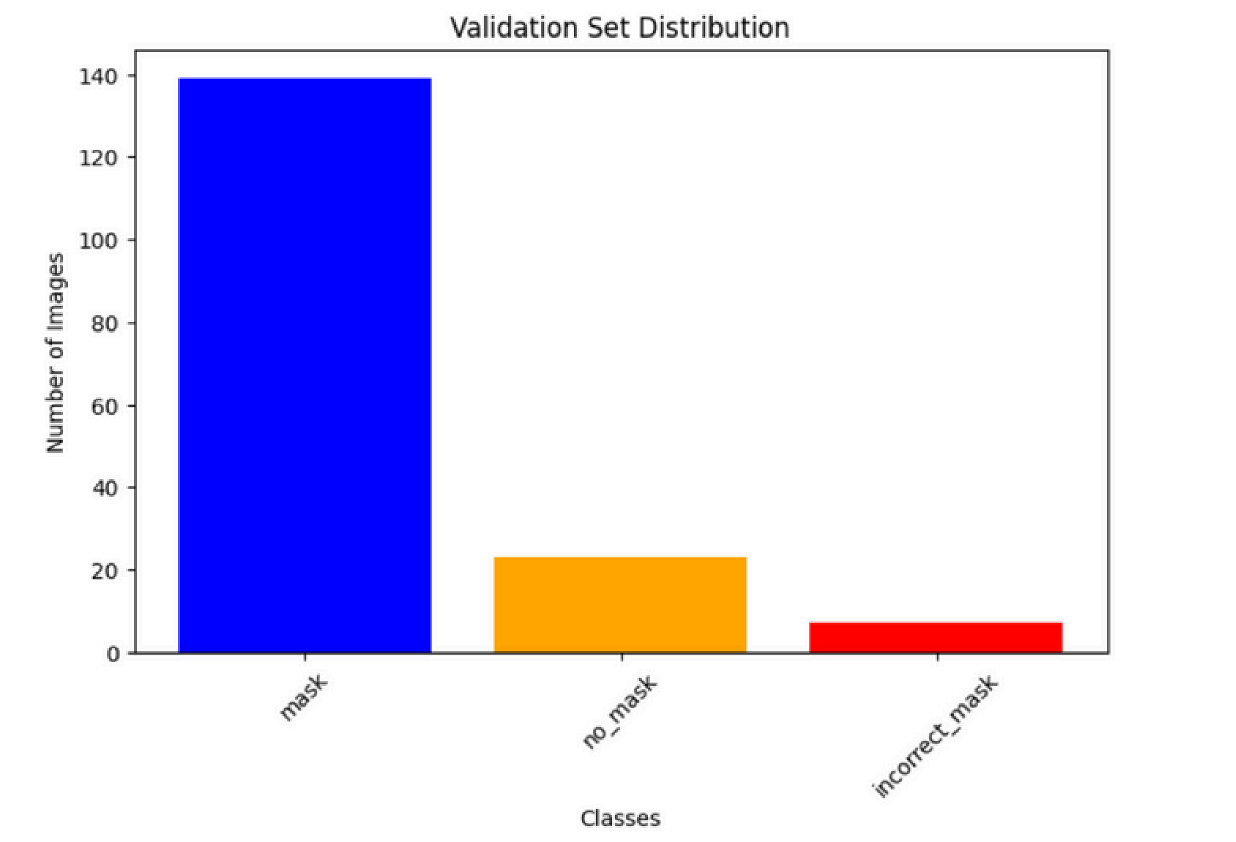
TRAINING SET COMPOSITION



TEST SET IMPORTANCE



VALIDATION SET PURPOSE



IMPLEMENTING INCEPTIONV3 FOR IMAGE CLASSIFICATION

STEPS AND EXPECTED OUTCOMES

MODEL FINE-TUNING

Adapt the pre-trained InceptionV3 model by replacing top layers for specific tasks.

```
[ ] base_model = InceptionV3(weights='imagenet', include_top=False, input_shape=(128, 128, 3))
    base_model.trainable = True
    for layer in base_model.layers[:100]:
        layer.trainable = False
```

```
[ ] x = base_model.output
    x = layers.GlobalAveragePooling2D()(x)
    x = layers.Dense(256, activation='relu')(x)
    x = layers.BatchNormalization()(x)
    x = layers.Dropout(0.3)(x)
    outputs = layers.Dense(3, activation='softmax')(x)
```

IMPLEMENTING INCEPTIONV3 FOR IMAGE CLASSIFICATION

STEPS AND EXPECTED OUTCOMES

TRAINING PROCESS

Train the model on augmented and normalized datasets using CPU and GPU resources.

```
]   train_datagen = ImageDataGenerator(
        rescale=1./255,
        rotation_range=20,
        width_shift_range=0.2,
        height_shift_range=0.2,
        shear_range=0.1,
        zoom_range=0.2,
        horizontal_flip=True,
        vertical_flip=True,
        brightness_range=[0.8, 1.2]
    )

    val_datagen = ImageDataGenerator(rescale=1./255)
    test_datagen = ImageDataGenerator(rescale=1./255)
```

```
[ ] train_generator = train_datagen.flow_from_directory(train_dir, target_size=(128, 128), batch_size=32, class_mode='categorical')
    val_generator = val_datagen.flow_from_directory(val_dir, target_size=(128, 128), batch_size=32, class_mode='categorical')
    test_generator = test_datagen.flow_from_directory(test_dir, target_size=(128, 128), batch_size=32, class_mode='categorical', shuffle=False)
```

IMPLEMENTING INCEPTIONV3 FOR IMAGE CLASSIFICATION

STEPS AND EXPECTED OUTCOMES

EXPECTED PERFORMANCE

Anticipate improved classification in detecting and distinguishing face masks.

Evaluate Model

```
[ ] test_loss, test_acc = model.evaluate(test_generator)
    print(f"✅ Test Accuracy: {test_acc * 100:.2f}%")
```

```
6/6 ━━━━━━━━━━━━━━━━━━━━ 5s 720ms/step - accuracy: 0.8769 - loss: 0.4297
✅ Test Accuracy: 86.55%
```

Classification report and confusion matrix

```
[ ] predictions = model.predict(test_generator)
    y_pred = np.argmax(predictions, axis=1)
    y_true = test_generator.classes

    print("Classification Report:")
    print(classification_report(y_true, y_pred, target_names=["mask", "no_mask", "incorrect_mask"]))

    cm = confusion_matrix(y_true, y_pred)
    plt.figure(figsize=(8, 6))
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=["mask", "no_mask", "incorrect_mask"], yticklabels=["mask", "no_mask", "incorrect_mask"])
    plt.xlabel('Predicted')
    plt.ylabel('True')
    plt.title('Confusion Matrix')
    plt.show()
```

CONCLUSION

This study demonstrated the effectiveness of transfer learning in image classification by fine-tuning the InceptionV3 model. Leveraging pre-trained CNNs improved classification accuracy, reduced overfitting, and optimized computational efficiency.

Key findings include:

- **Improved Model Performance:** Leveraging pre-trained CNNs enhanced classification accuracy and reduced overfitting.
- **Optimized Computational Efficiency:** Transfer learning minimized training time and resource requirements.
- **Robust Training Pipeline:** Data preprocessing, augmentation, and hyperparameter tuning contributed to model stability.
- **Accelerated Convergence:** Pre-trained feature extraction enabled effective learning from limited datasets.

Future work can explore advanced augmentation, adaptive learning rates, and ensemble models to further improve performance. This research reinforces transfer learning as a scalable and efficient solution for real-world image classification tasks.