Transfer learning baced classification of poultry diseases for enhanced health management

Team ID: **LTVIP2025TMID20332**

Team Members:

Team Leader : Maddula Srilakshmi

Team member : M Abhilash

Team member : M Harshitha Sai Sri

Team member : M Rakesh

Transfer learning baced classification of poultry diseases for enhanced health management

🎯Objective:

To build a computer vision model using Transfer Learning to classify poultry diseases based on images of infected birds, supporting early detection and improved health management in poultry farms.

🎯**Architecture:**

1️⃣Input

- Images of poultry (sick or healthy)
- Stored in labeled folders (e.g., Fowlpox/, Healthy/, etc.)

2️⃣Preprocessing

- Resize images to 224x224 pixels
- Normalize pixel values
- Data Augmentation (flip, rotate, zoom)
-

3️⃣Transfer Learning Model

- Use MobileNetV2 (or ResNet) as a base model
- Freeze base layers (pre-trained on ImageNet)
- Add custom layers:

4️⃣Training

- Train the model on preprocessed images
- Monitor validation accuracy
- Save the best model as poultry_model.h5

5️⃣Prediction

- Load the saved model
- Input a new image
- Output: Predicted disease class with confidence %

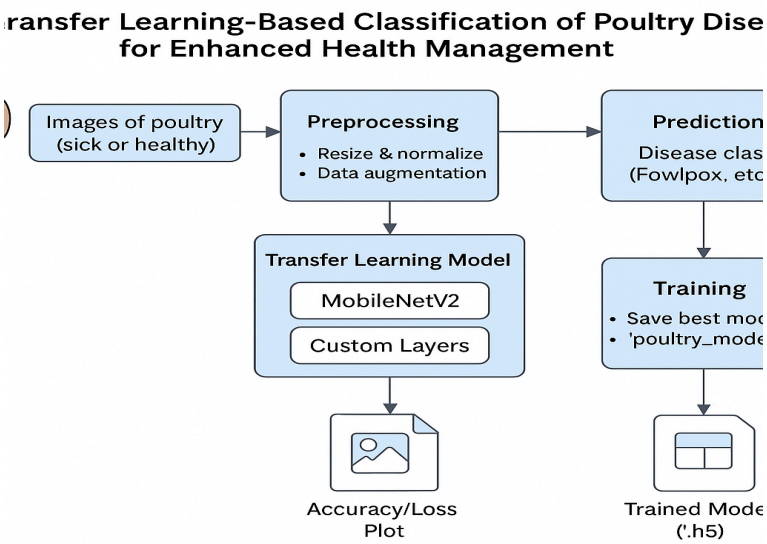
6️⃣Outputs

- Trained Model (.h5)
- Accuracy/Loss Plot (.png)
- Disease prediction from input images

🔧 **Tools Used:**

- Python
- TensorFlow/Keras
- Matplotlib
- VS Code

Structure:



Pre requisites:

Software requirements:

♥ Python 3.7+

🦋 Vs code

♥ Pip(python package manager)

Project structure:

poultry_disease_project/
├── dataset/
│ ├── Fowlpox/
│ ├── Coccidiosis/
│ ├── Newcastle/
│ ├── AvianInfluenza/
│ ├── InfectiousBronchitis/
│ └── Healthy/
├── model/

```
│   └── poultry_model.h5  ← Saved after training
├── src/
│   ├── train_model.py   ← Core model training code
│   └── predict_image.py ← Optional inference script
├── requirements.txt
└── README.md
```

Data collection and preparation:

📦 1. Data Collection

🔍 A. What You Need

Images of poultry showing visible symptoms of diseases and healthy birds.

Source:

Kaggle

Github repositories

Veterinary websites

Collaboration with poultry farms

ImageDataGenerator:

It's a TensorFlow/Keras tool that:

- Loads images from folders
- Applies real-time data augmentation
- Rescales and resizes images automatically
- Splits data into training and validation

Testing model and data prediction:

```
import numpy as np
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import os
# Load trained model
model_path = os.path.join('..', 'model', 'poultry_model.h5')
model = load_model(model_path)
# Define class names (should match your training classes)
class_names = ['AvianInfluenza', 'Coccidiosis', 'Fowlpox', 'Healthy',
'InfectiousBronchitis', 'Newcastle']
# Load test image
img_path = os.path.join('..', 'test_images', 'sample.jpg')  # Replace with your
image name
img = image.load_img(img_path, target_size=(224, 224))      # Resize same as
training
```

```
img_array = image.img_to_array(img) / 255.0              # Normalize
img_array = np.expand_dims(img_array, axis=0)            # Add batch dimension
# Predict
pred = model.predict(img_array)
class_idx = np.argmax(pred)
confidence = pred[0][class_idx]
print(f"\nPredicted class: {class_names[class_idx]}")
print(f"Confidence: {confidence * 100:.2f}%")
```

how to use:

save the above code
place a test image
run from terminal;

output:

opens a window showing the test image
title include predicted disease;

Application Building:

We will use **streamlit**

Stream lit:

Python:

```
import streamlit as st
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import numpy as np
from PIL import Image
# Load the model
model = load_model('../model/poultry_model.h5')
# Class names from your training labels
class_names = ['AvianInfluenza', 'Coccidiosis', 'Fowlpox', 'Healthy',
'InfectiousBronchitis', 'Newcastle']
# Streamlit app UI
st.set_page_config(page_title="Poultry Disease Classifier")
st.title("🐔 Poultry Disease Detection App")
st.write("Upload a poultry image to predict the disease class.")
uploaded_file = st.file_uploader("Choose an image...", type=["jpg", "jpeg",
"png"])
if uploaded_file is not None:
# Load and display image
```

```
    img = Image.open(uploaded_file)
    st.image(img, caption='Uploaded Image', use_column_width=True)
  # Preprocess image
    img = img.resize((224, 224))
    img_array = image.img_to_array(img) / 255.0
    img_array = np.expand_dims(img_array, axis=0)
  # Predict
    predictions = model.predict(img_array)
    pred_class = class_names[np.argmax(predictions)]
    confidence = np.max(predictions) * 100
   # Display prediction
   st.success(f"✅ Prediction: **{pred_class}**")
    st.info(f"Confidence: **{confidence:.2f}%**")
```

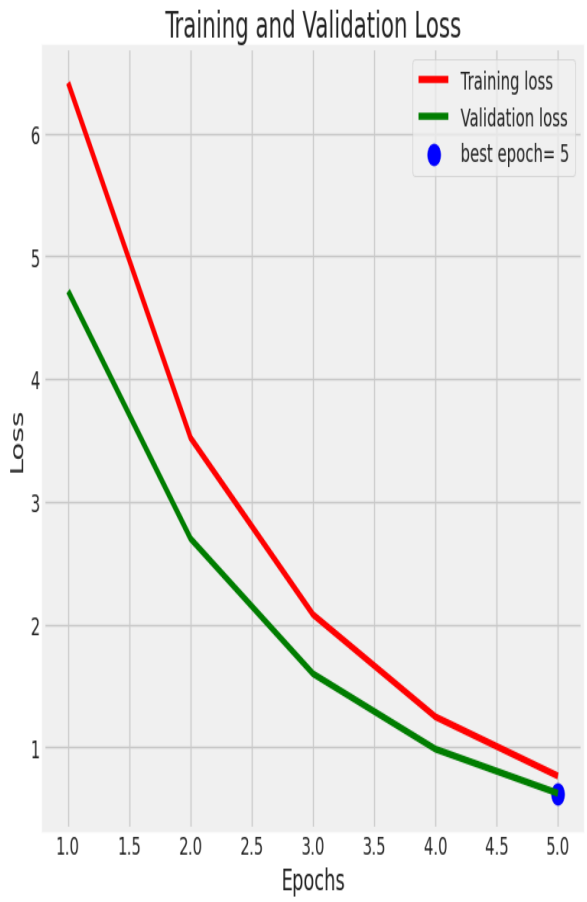
🖼️ Output:

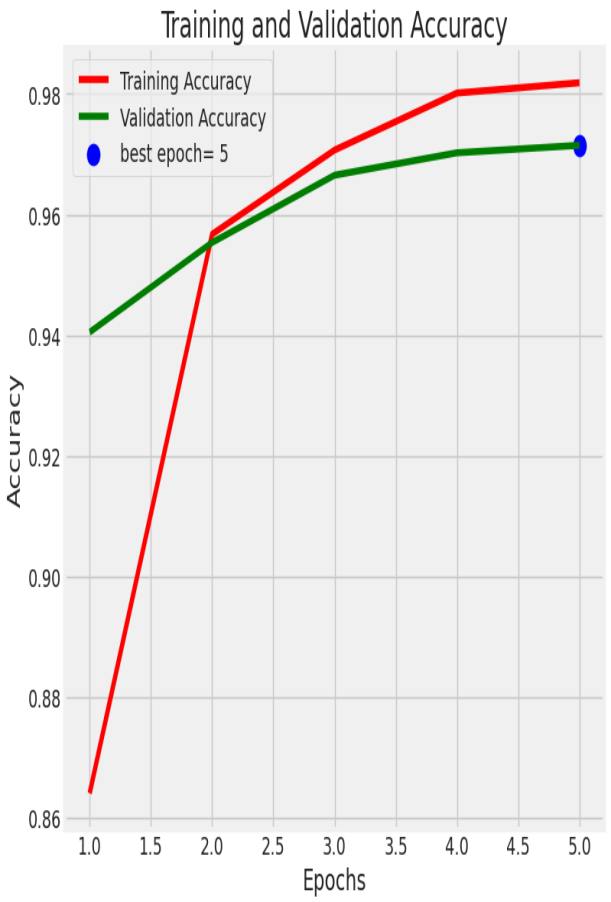
You'll see:

- Image preview
- Predicted disease class
- Confidence score

Example:

Plot training(history):





Sample prediction: