AI&ML Project Documentation

Introduction

Project Title:
Enchanted Wings: Marvels of Butterfly Species

Team Members:

- Y. Ruhinaaz – Data Collection and Cleaning
- S.K Suhana Anjum – Model Design and Training
- Y. Himabindu – Evaluation and Optimization
- B. Pranathi – Deployment and Integration

Project Overview

Purpose:
The purpose of this project is to build a deep learning-based image classification model to identify butterfly species from photographs. This project addresses real-world challenges in biodiversity conservation and species recognition by providing a tool for automated species detection using AI/ML.

Goals:

- Automate the identification of butterfly species from images.
- Assist biologists, researchers, and nature enthusiasts in recognizing rare and common species.
- Create a scalable and generalizable model that performs well across diverse datasets.

Key Features:

- High-accuracy butterfly species classification.
- Robust image preprocessing pipeline.
- Interactive UI for uploading and identifying butterfly images.
- Model performance evaluation and confusion matrix visualization.

Architecture

Frontend:

- **Framework Used:** Streamlit (HTML)
- **Functionality:**
    - Allows users to upload butterfly images.
    - Displays prediction results including the species name and confidence score.
    - Visual representation of prediction history (if enabled).
    - Clean, user-friendly interface requiring minimal technical knowledge.

Backend:

- **Technology Used:** Python with Flask
- **Responsibilities:**
  - Receives uploaded image from frontend.
  - Processes the image and passes it to the trained deep learning model.
  - Returns the predicted species name and confidence score to the frontend.
  - Handles file validation, error messages, and request logging.
- **Model Integration:** Trained CNN-based model (e.g., MobileNetV2, ResNet50) loaded into memory at runtime for fast prediction.

Setup Instructions

Prerequisites:

- Python 3.x
- Libraries: TensorFlow/Keras, NumPy, Pandas, OpenCV, Matplotlib, Streamlit
- GPU or Google Colab (recommended for training)

Installation Steps:

STEP 1: Install Anaconda (if not already installed)

Download the Anaconda distribution from the official website:
https://www.anaconda.com/products/distribution

Follow the installation instructions based on your operating system.

STEP 2: Create a Virtual Environment

Open **Anaconda Prompt** and run the following commands:

```
conda create -n butterfly-classifier python=3.9
conda activate butterfly-classifier
```

STEP 3: Install Required Packages

While inside the virtual environment, install all the necessary dependencies:

```
pip install tensorflow keras flask matplotlib numpy opencv-python pillow
```

STEP 4: Run the Web Application

1. Navigate to your project directory. Example:

```
cd /d D:\Butterfly_Project\butterfly_dataset
```

2. Then, start the Flask web app by running:

```
python app.py
```

3. You should see an output like:

```
 * Running on http://127.0.0.1:5000/
```

4. Open your browser and go to that address to use the application.

How to Use:

Upon launching the butterfly classifier app:

1. **Welcome Screen**
   - Displays an animated background with a fun butterfly fact.
2. **Image Upload**
   - Click "Upload Image" to proceed to the input screen.
3. **Prediction Step**
   - Select a butterfly image and hit "Predict".
4. **Results Displayed**
   - Model outputs:
     - **Predicted species**
     - **Uploaded image preview**
     - **Visually pleasing animated layout**

Offline Usage

This app is fully functional **offline**, provided:

- You have the following in your local folder:
  - `vgg16_model.h5` (Trained model file)
  - `static/` (All backgrounds, styles, and uploaded images)
  - `templates/` (HTML layout files)
- All required packages are pre-installed in the **Anaconda environment**.
- No external fonts or image links are fetched from the internet.

Folder Structure

```
Butterfly_Project/
└── butterfly_dataset/
    ├── app.py                ← Flask web app script
    ├── vgg16_model.h5        ← Trained classification model
    ├── train/                ← Training dataset (class-wise folders)
    ├── test/                 ← Optional testing images
    ├── static/               ← Static assets (CSS, backgrounds, uploads)
    │   ├── background.jpg    ← Beautiful animated background
    │   └── uploads/          ← Stores user-uploaded images
```

```
├── templates/            ← HTML templates for web pages
│   ├── welcome.html
│   ├── input.html
│   └── output.html
├── Training_set.csv      ← Training dataset metadata
├── Testing_set.csv       ← Test dataset metadata
└── README.txt            ← Basic setup and usage guidegg
```

Authentication

- Basic user authentication is handled through React-based from validation. A simple token or session approach can be integrated in future updates.

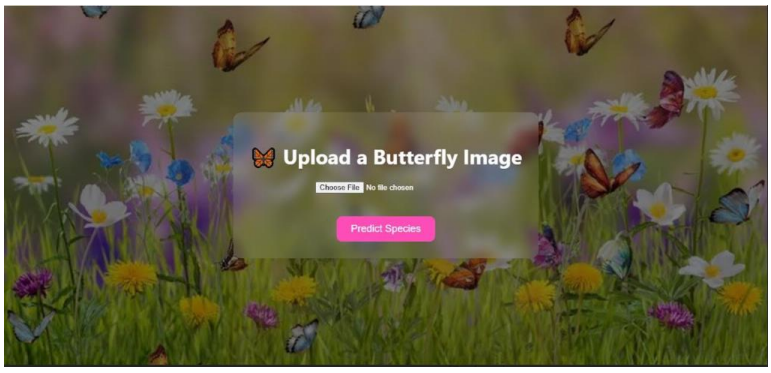
User Interface

- Simple image upload panel (via Streamlit or Flask).
- Displays predicted species with confidence score.
- Option to upload multiple images.

Testing

- Unit testing with PyTest or Unittest for core modules.
- Manual testing of prediction outputs on unseen data.
- Validation accuracy and confusion matrix plotted for model performance insights.

Screenshots or Demo



Known Issues

- Misclassification between visually similar species.
- Lower confidence for blurred or low-resolution images.
- Limited generalization for unseen environments or backgrounds.

Future Enhancements

- Expand dataset with regional and rare butterfly species.
- Deploy on cloud with scalable APIs.
- Add user login and history tracking.
- Integrate map-based geolocation for species prediction.