Multiclass Fish Image Classification Project Report

Prepared by Malathi Sathish

Email: malathisathish2228@gmail.com
GitHub: https://github.com/Malathisathish
LinkedIn: https://linkedin.com/in/Malathisathis

Date: September 27, 2025

Contents

1 Introduction

This project develops a deep learning-based system for classifying fish images into multiple species, leveraging convolutional neural networks (CNNs) and transfer learning. The primary objectives are to achieve high classification accuracy, deploy a user-friendly web application for real-time predictions, and provide a comprehensive evaluation of model performance. The project is implemented using TensorFlow, with models trained in Google Colab, saved in '.h5' format, and deployed via a Streamlit application in Visual Studio Code.

1.1 Business Use Cases

- Enhanced Accuracy: Identify the optimal model architecture for accurate fish species classification.
- Deployment Ready: Develop a Streamlit web application for real-time fish image classification.
- Model Comparison: Evaluate and compare performance metrics across models to select the best approach.

1.2 Approach

- Data Preprocessing and Augmentation: Normalize images to [0, 1], resize to 224x224 pixels, and apply augmentation (rotation, zoom, flipping) to enhance training robustness.
- Model Training: Train a custom CNN and fine-tune five pre-trained models (VGG16, ResNet50, MobileNet, InceptionV3, EfficientNetB0). Save the best model in '.h5' format.
- Model Evaluation: Compare accuracy, precision, recall, and F1-score across models using validation and test datasets. Visualize training history and confusion matrices.
- Deployment: Build a Streamlit application with pages for Home, Prediction, Model Comparison, and Developer information.
- Documentation: Provide detailed documentation, including code, evaluation results, and a GitHub repository with a comprehensive README.

2 Dataset

The dataset, stored as Copy of Dataset.zip, is organized into train, validation, and test folders, each containing subfolders for different fish species. The dataset is loaded using TensorFlow's ImageDataGenerator for efficient processing.

- Preprocessing: Images are rescaled to 224x224 pixels and normalized to [0, 1].

- Augmentation: Applied to training data with random rotation (20 degrees), zoom (20%), horizontal flipping, and shear (20%).

3 Models and Training

3.1 Model Architectures

- CNN from Scratch: Three convolutional layers (32, 64, 128 filters with 3x3 kernels), ReLU activation, max pooling (2x2), followed by a dense layer (512 units, ReLU) and dropout (0.5).
- Transfer Learning Models: Pre-trained models (VGG16, ResNet50, MobileNet, InceptionV3, EfficientNetB0) from ImageNet, with frozen base layers initially and fine-tuning of upper layers.

3.2 Training Configuration

- Epochs: 50, with early stopping (patience of 10 epochs based on validation loss).
- Optimizer: Adam (learning rate 0.001 for initial training, 0.0001 for fine-tuning).
- Loss Function: Categorical crossentropy.
- Model Saving: Best models saved as .h5 files (e.g., best_fish_model.h5).

4 Evaluation Results

The models were evaluated on validation and test datasets using accuracy, precision, recall, and F1-score (weighted averages). The metrics are sourced from df_comparison.csv, generated during training in Google Colab.

4.1 Model Performance Comparison

Table 1: Model Performance Metrics

Model	Val. Acc.	Val. Prec.	Val. Rec.	Val. F1	Test Acc.	Test Prec.	Test Rec.	Test F
CNN_Scratch	98.44%	97.68%	98.44%	98.04%	98.18%	98.14%	98.18%	98.05%
VGG16	98.72%	98.79%	98.72%	98.75%	99.31%	99.36%	99.31%	99.33%
ResNet50	84.43%	86.29%	84.43%	84.13%	88.33%	89.46%	88.33%	88.17%
MobileNet	99.82%	99.82%	99.82%	99.81%	99.87%	99.88%	99.87%	99.87%
InceptionV3	99.82%	99.82%	99.82%	99.81%	99.81%	99.81%	99.81%	99.81%
EfficientNetB0	90.57%	90.72%	90.57%	90.17%	89.24%	89.83%	89.24%	89.07%

4.2 Key Insights

- MobileNet and InceptionV3: Achieved the highest validation and test accuracies (99.82% and 99.87%/99.81%, respectively), with MobileNet being the most efficient due to its smaller size (13.4 MB).
- VGG16: Strong performance (99.31% test accuracy), but larger model size (57.2 MB).
- CNN from Scratch: High validation accuracy (98.44%) but slight overfitting (98.18% test accuracy), suggesting potential for regularization improvements.
- ResNet50: Lowest performance, likely due to dataset-specific challenges or insufficient fine-tuning.
- EfficientNetB0: Balanced performance but underperformed compared to MobileNet and InceptionV3.

4.3 Training History

- CNN from Scratch: Training stopped at epoch 46 with validation accuracy of 98.44% and loss of approximately 0.07.
- Transfer Learning Models: Training history plots (accuracy and loss curves) were generated in Colab, showing stable convergence for MobileNet and InceptionV3.

4.4 Confusion Matrices

Confusion matrices for each model were generated to analyze per-class performance, highlighting strong classification for dominant species and minor misclassifications for visually similar species.

5 Deployment

The project includes a Streamlit application deployed locally in Visual Studio Code, featuring a sea-themed interface with the following pages:

- Home: Overview of the project and model architectures.
- Prediction: Allows users to upload fish images, displays predicted species, confidence scores, top-3 predictions, and a probability plot.
- Model Comparison: Presents a table of performance metrics and a 2x2 grid of bar charts for test accuracy, precision, recall, and F1-score.
- Developer: Includes developer details (Malathi Sathish, malathisathish2228@gmail.com, GitHub, LinkedIn).

Local Deployment Command:

streamlit run app.py

The app uses best_fish_model.h5 (MobileNet, based on test accuracy) and class_names.pkl for predictions, with metrics from df_comparison.csv.

6 Conclusion

The Multiclass Fish Image Classification project successfully demonstrates high-accuracy fish species classification using deep learning. MobileNet and InceptionV3 outperformed other models, with MobileNet being the preferred choice due to its efficiency. The Streamlit application provides a user-friendly interface for real-time predictions. Future improvements include:

- Addressing overfitting in the CNN from Scratch model via increased dropout or L2 regularization.
- Expanding the dataset to include more diverse fish species.
- Deploying the Streamlit app to a cloud platform (e.g., Streamlit Cloud, Heroku).

7 About the Developer

- Name: Malathi Sathish
- Profile: Data scientist specializing in computer vision and deep learning.
- Contact: malathisathish2228@gmail.com
- GitHub: https://github.com/Malathisathish
- LinkedIn: https://linkedin.com/in/Malathisathis

Thank you for exploring the Multiclass Fish Image Classification project!