**Multiclass Fish Image Classification**

**Introduction:**

This project aims to classify fish images into multiple categories using advanced deep learning techniques. By leveraging convolutional neural networks (CNNs) and transfer learning with pre-trained models, we aim to build an efficient and accurate image classification system. Additionally, a user-friendly Streamlit web application is developed for real-time fish species prediction based on image input.

**Project Objectives:**

* Train a CNN model from scratch for multiclass fish image classification.
* Apply transfer learning using pre-trained models: VGG16, ResNet50, MobileNetV2, InceptionV3, and EfficientNetB0.
* Address class imbalance using targeted data augmentation techniques.
* Evaluate the performance of all models using standard classification metrics.
* Deploy the best-performing model through a web application using Streamlit.
* Create a GitHub repository with complete documentation and codebase for reproducibility.

**Business Use Cases:**

* **Enhanced Accuracy:** Identifying the most accurate model architecture for fish classification.
* **Deployment Ready:** Providing a real-time, interactive solution to classify fish images.
* **Model Comparison:** Offering detailed insights into model performance to determine the most suitable approach.

**Dataset Description:**

* The dataset comprises images of different fish species, organized into separate directories by class**.**
* It is split into three subsets: training, validation, and testing
* TensorFlow’s ImageDataGenerator is used for efficient image loading and augmentation.
* Special attention was given to underrepresented classes. For instance, *‘animal fish bass’* images were augmented 10x per original image using targeted augmentation to address class imbalance.
* Image sizes used:
* For most models: 224x224 pixels.
* For InceptionV3: 299x299 pixels.

**Methodology:**

**Data Preprocessing and Augmentation**

* Images are rescaled to a pixel range of [0, 1].
* Extensive augmentation techniques were applied to the training set to enhance model robustness and generalization:
  + **Rotation:** Random rotations up to 20 degrees.
  + **Zoom:** Random zoom in and out up to 20%.
  + **Horizontal Flip:** Random flipping of images horizontally.
  + **Vertical Flip:** Random vertical flipping to simulate different orientations.
  + **Width & Height Shift:** Random shifts in width and height by 10%.
  + **Shear Transformation:** Random shearing up to 15%.
  + **Brightness Adjustment:** Varying brightness randomly in the range [0.8, 1.2].
  + **Channel Shift:** Randomly altering the color channels by a factor of 0.2.
  + **Fill Mode:** Used 'nearest' strategy to fill in newly created pixels during transformations.
* This helps improve generalization and prevent overfitting.
* Targeted augmentation was specifically applied to the “animal fish bass” class to boost the number of training samples and improve the model’s ability to learn underrepresented features.

**Model Training**

* A CNN model was built from scratch using a multi-block architecture with Batch Normalization and Dropout regularization. The final layers used GlobalAveragePooling2D() followed by Dense layers.
* Five pre-trained models were used for transfer learning:
* VGG16
* ResNet50
* MobileNetV2
* InceptionV3
* EfficientNetB0
* All models were fine-tuned for the fish dataset.
* For InceptionV3, images were resized to 299x299
* The base layers of pre-trained models were frozen initially to retain learned features from ImageNet.
* Early stopping was implemented to prevent overfitting.
* Class imbalance was handled using computed class weights based on training class distribution.
* Batch size and step sizes were dynamically calculated for consistent training across all datasets.

**Model Evaluation**

* Models were evaluated using:
* Accuracy
* Precision
* Recall
* F1-Score
* Confusion Matrix
* Training histories were visualized using accuracy and loss plots for each model.
* Performance metrics were collected and compared across models.
* Evaluation included model predictions on unseen test data and detailed classification reports.

**Model Saving**

* All trained models were saved in .h5 format for deployment and reuse.
* The CNN from scratch was saved as cnn\_fish\_model.h5.
* Each pre-trained model was saved using its respective name.
* The best-performing model was selected based on F1-score and saved the model’s name in path “**Best model name saved to:** **best\_model\best\_model\_name.csv**”.

**Deployment Using Streamlit:**

A Streamlit web application was developed to:

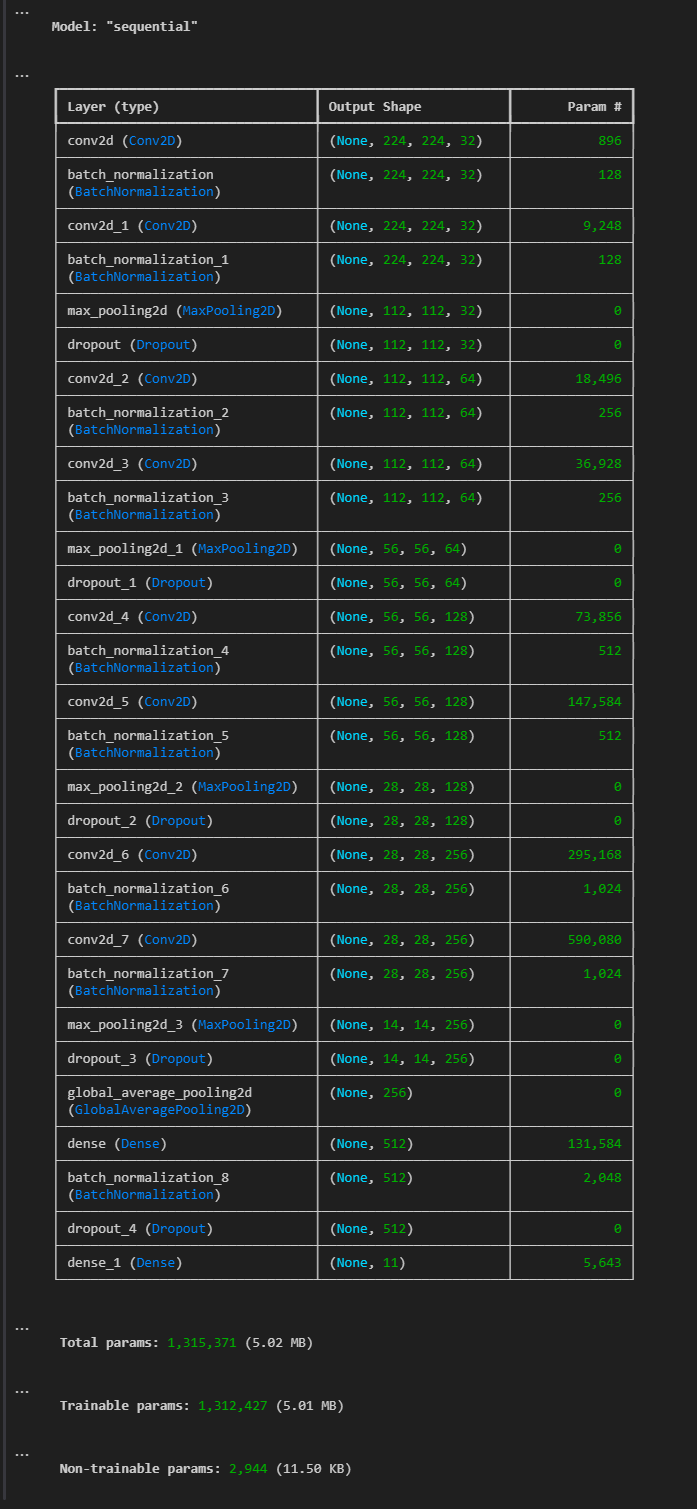
* Allow users to upload fish images.
* Automatically predict the fish category using the trained model.
* Display the prediction label along with the confidence score.
* Provide an intuitive and easy-to-use interface for end users.

**Results and Model Comparison:**

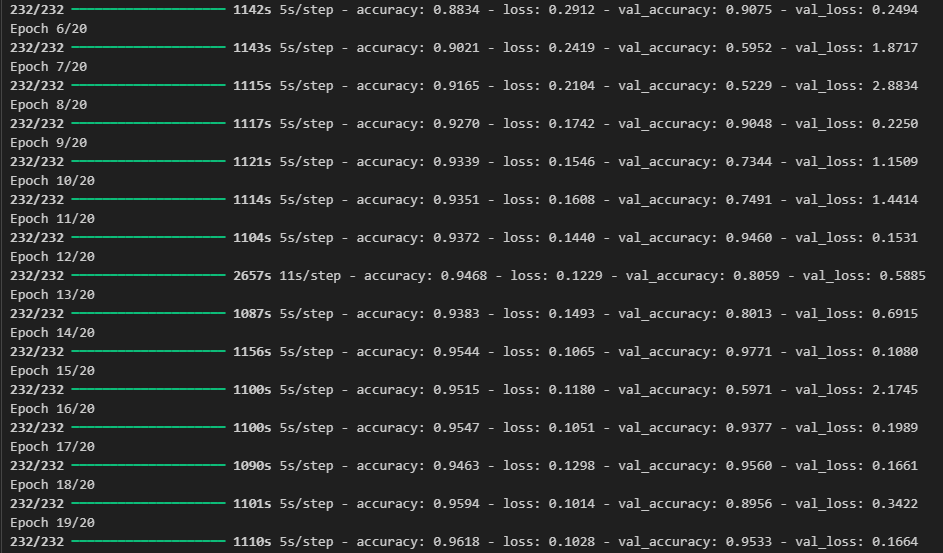
* Each model’s performance was recorded and tabulated.
* Metrics such as accuracy, precision, recall, and F1-score were compared.
* The best-performing model was identified based on the highest F1-score.
* Visual plots comparing training and validation accuracy/loss over epochs were created.
* Confusion matrices and classification reports provided detailed insights into per-class performance

**Experimental Results and Observations:**

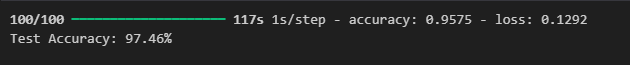
**CNN from Scratch – Training Insights**



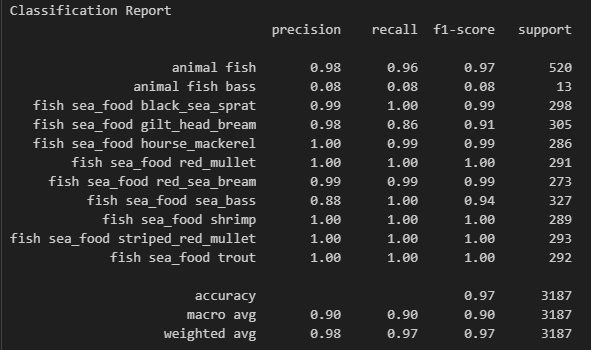
CNN from Scratch - Architecture

****

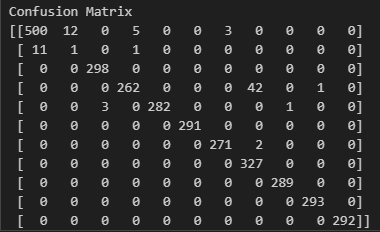
CNN from Scratch - Epoch



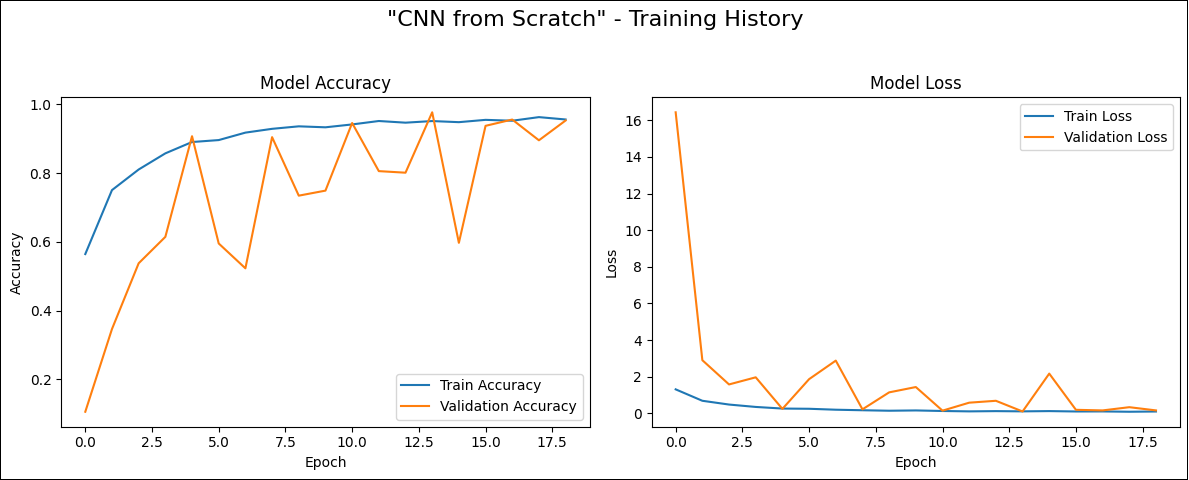
CNN from Scratch - Test Accuracy



CNN from Scratch – Classification Report

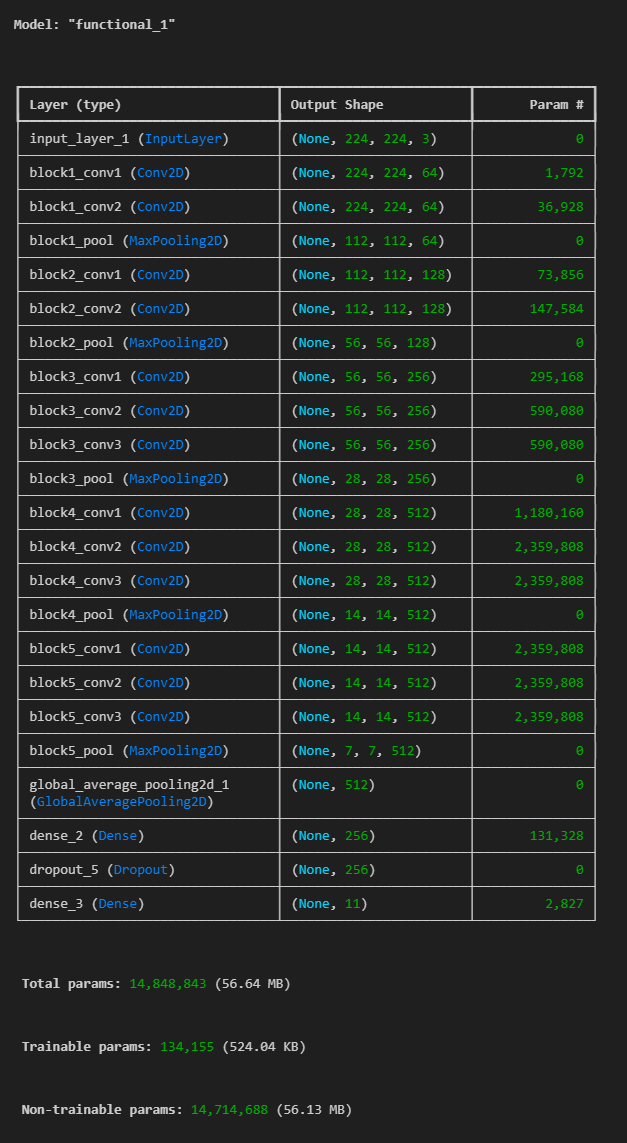
****

CNN from Scratch – Confusion Matrix

****

CNN from Scratch – Training History Plot

**VGG16 – Training Insights**

****

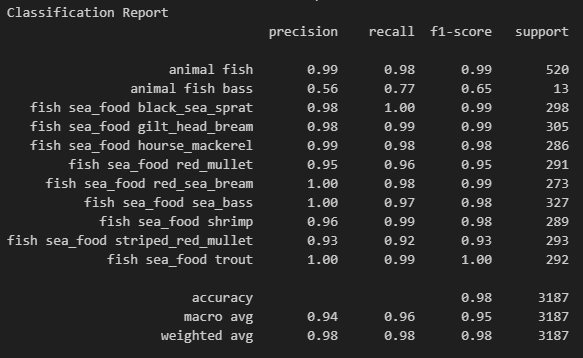
VGG16 – Architecture



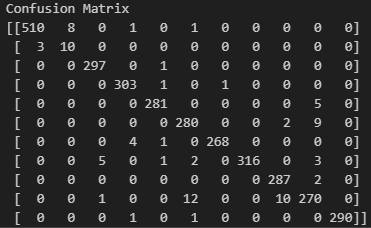
VGG16 – Epoch

****

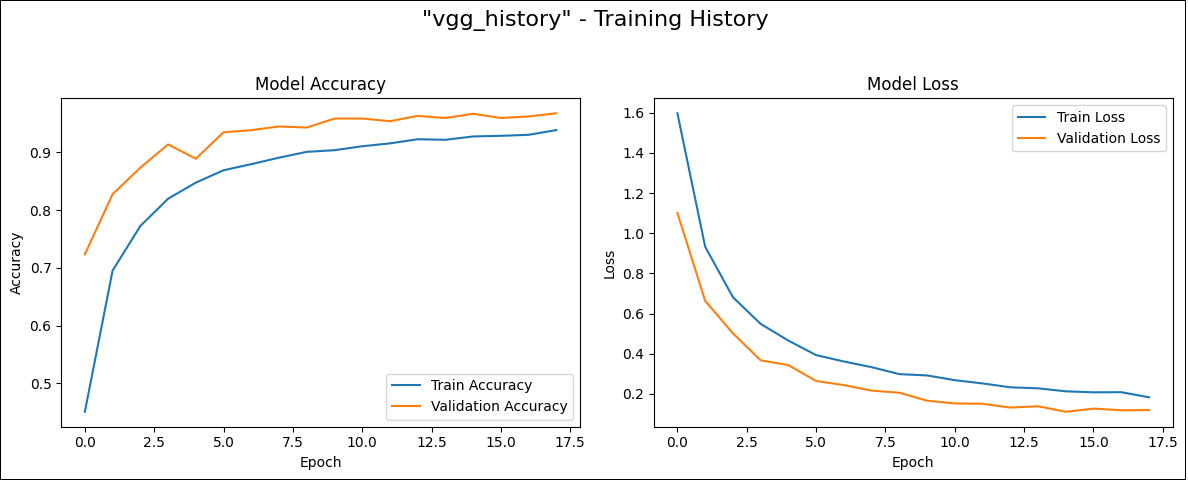
VGG16 – Test Accuracy

****

VGG16 – Classification Report

****

VGG16 – Confusion Matrix



VGG16 – Training History Plot

**ResNet50 – Training Insights**



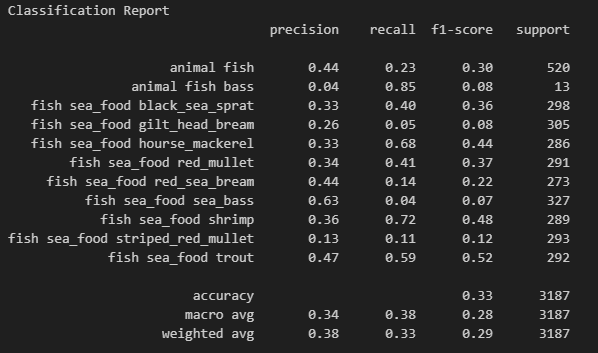
ResNet50 – Architecture

****

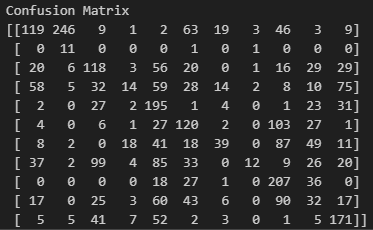
ResNet50 – Epoch

****

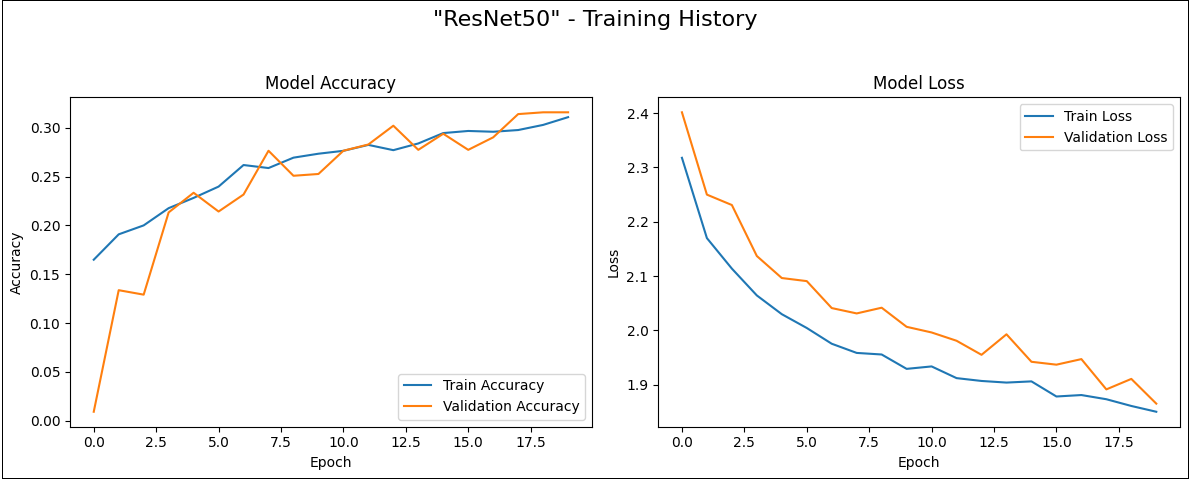
ResNet50 – Test Accuracy



ResNet50 – Classification Report

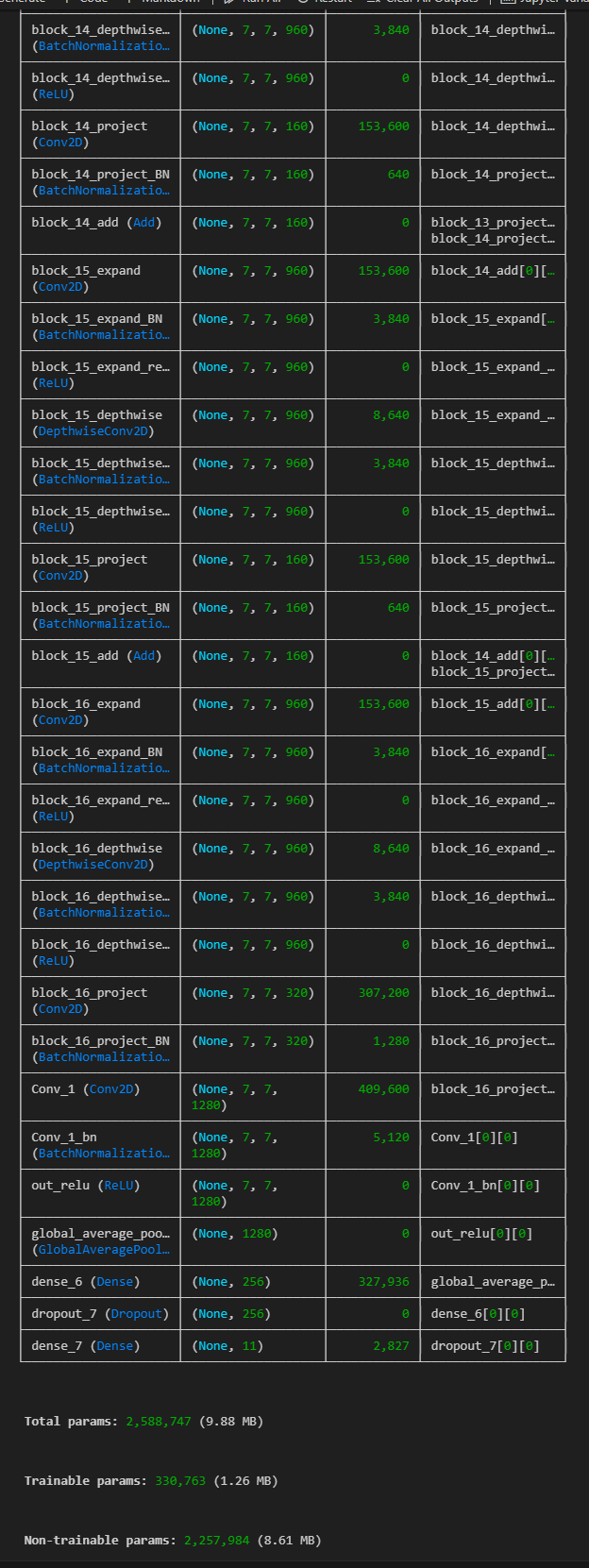


ResNet50 – Confusion Matrix

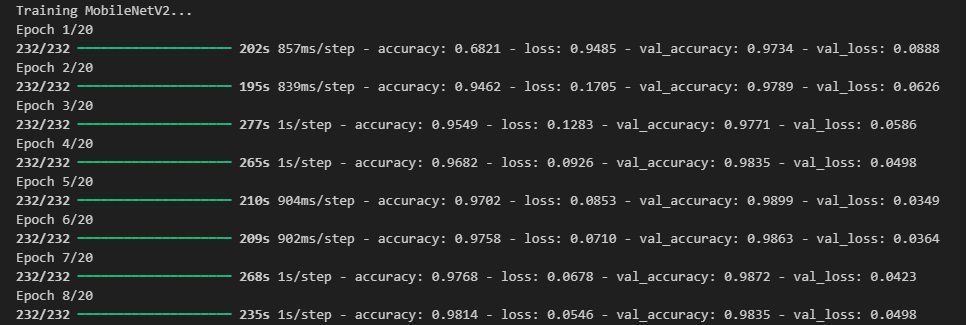


ResNet50 – Training History Plot

**MobileNetV2 – Training Insights**



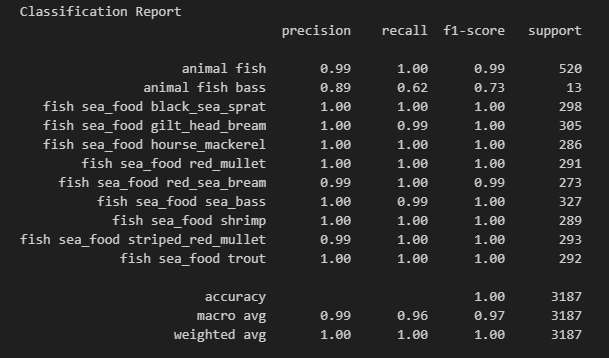
MobileNetV2 – Architecture



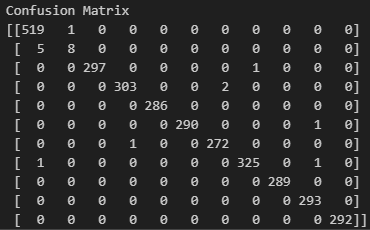
MobileNetV2 – Epoch



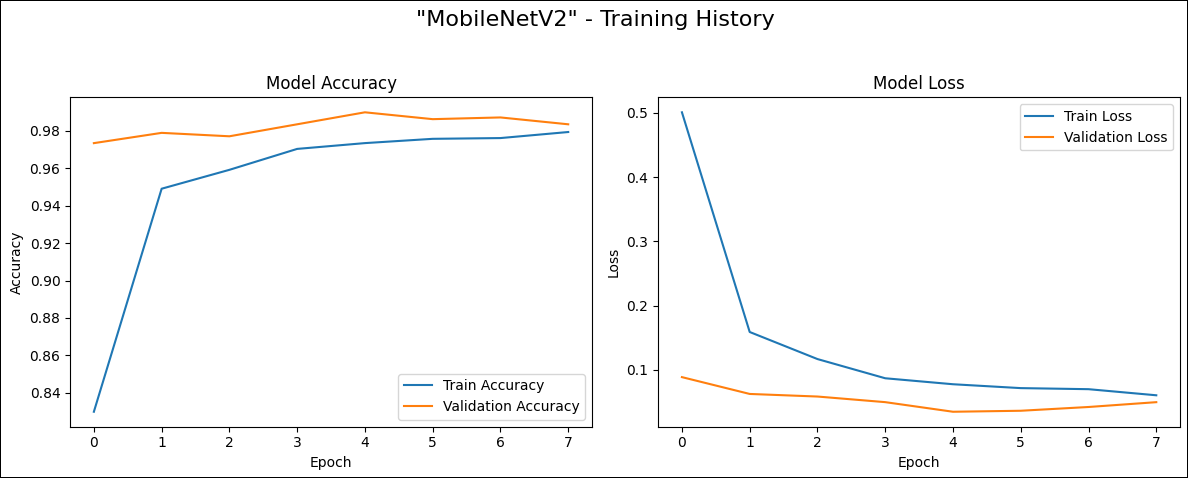
MobileNetV2 – Test Accuracy



MobileNetV2 – Classification Report



MobileNetV2 – Confusion Matrix



MobileNetV2 – Training History Plot

**InceptionV3 – Training Insights**



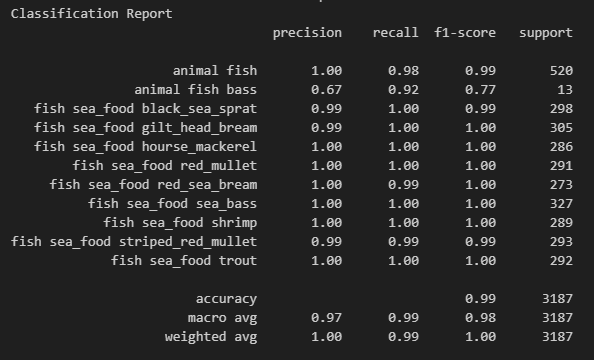
InceptionV3 – Architecture



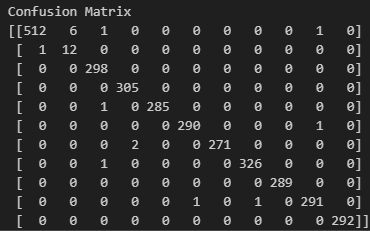
InceptionV3 – Epoch



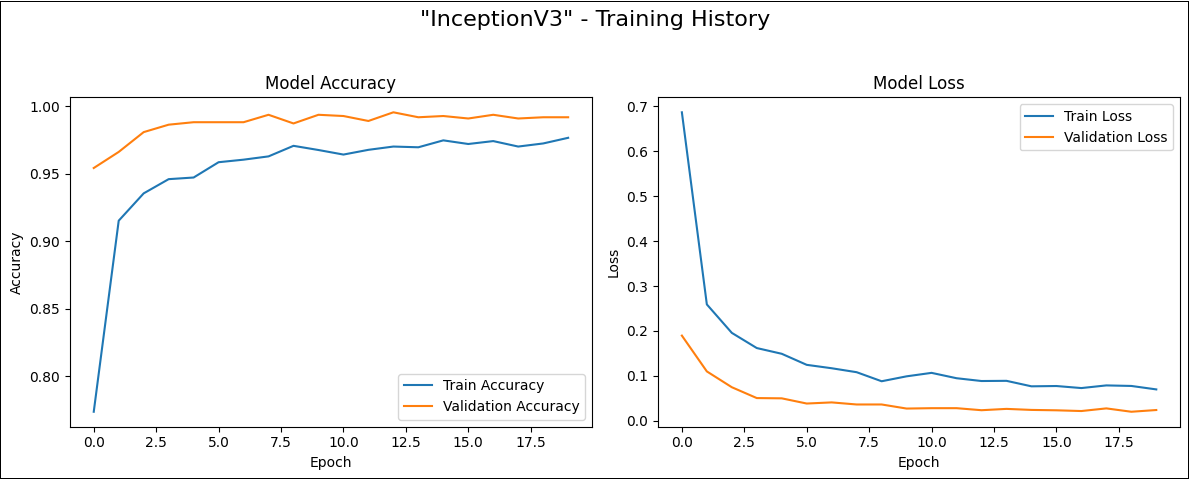
InceptionV3 – Test Accuracy



InceptionV3 – Classification Report



InceptionV3 – Confusion Matrix

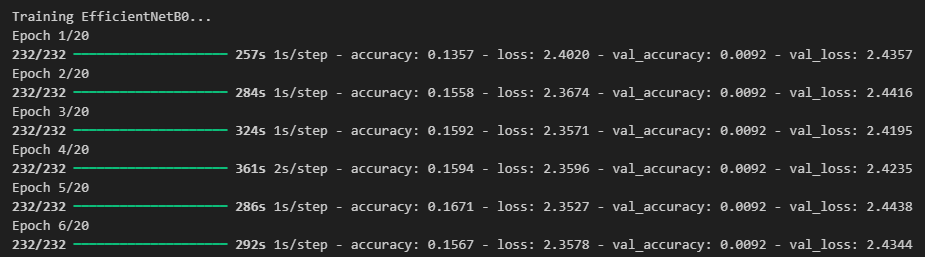


InceptionV3 – Training History Plot

**EfficientNetB0 – Training Insights**



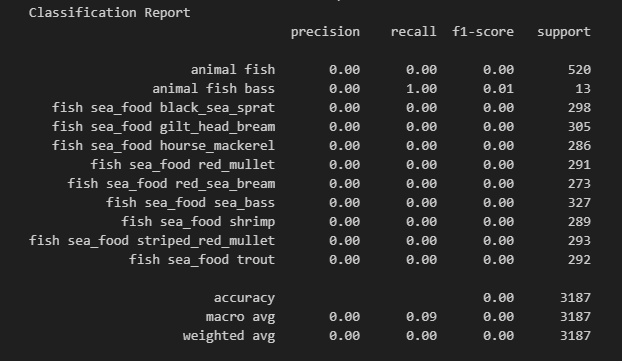
EfficientNetB0 – Architecture



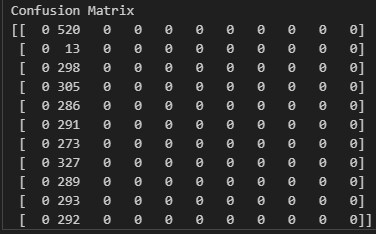
EfficientNetB0 – Epoch



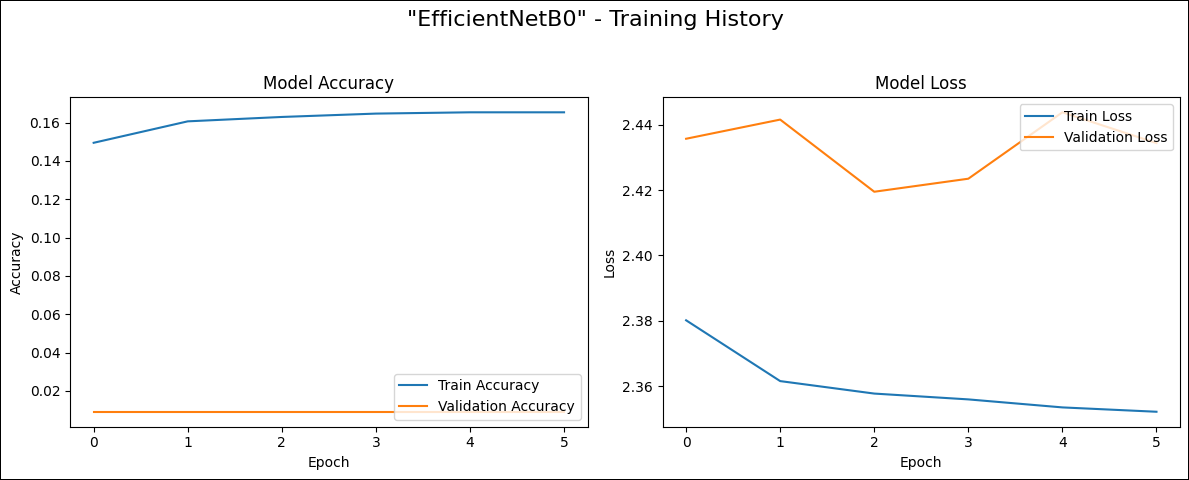
EfficientNetB0 – Test Accuracy



EfficientNetB0 – Classification Report

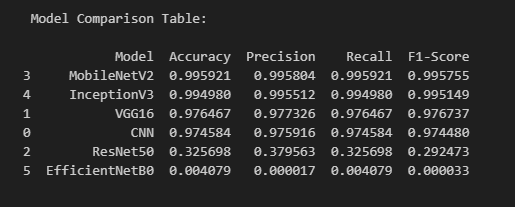


EfficientNetB0 – Confusion Matrix



EfficientNetB0 – Training History Plot

**Model Comparison Table:**

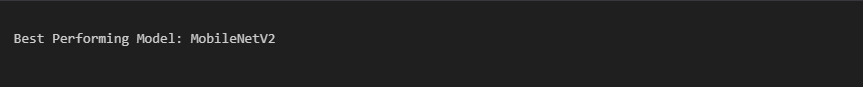


Model Comparison Table

**Insights from Model Comparison**

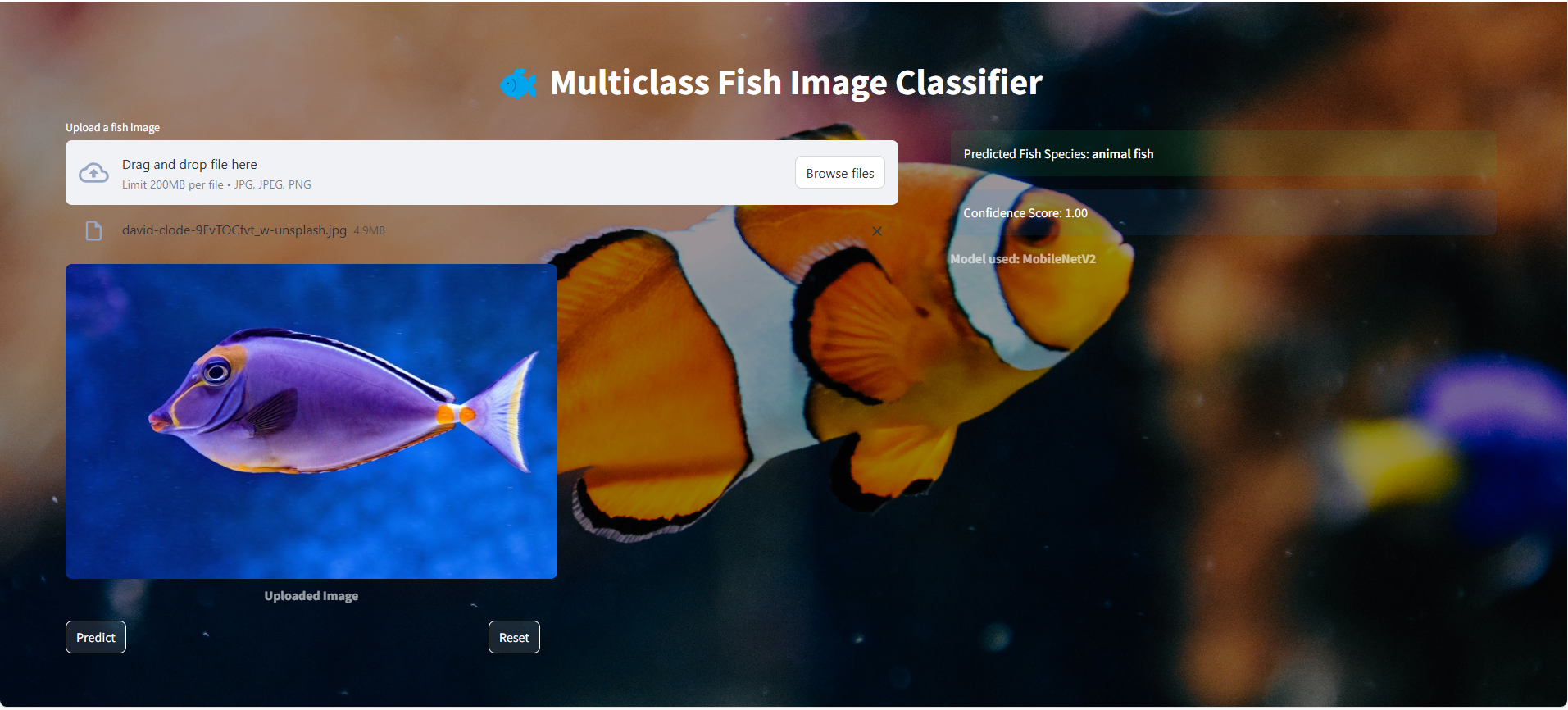
* **MobileNetV2** achieved the **highest overall performance**, with an **F1-Score of 0.9958**, making it the best model for deployment.
* **InceptionV3** closely followed, with a strong F1-Score of **0.9951**, showing excellent generalization.
* **VGG16** and the **CNN from scratch** also performed well, with F1-scores of **0.9767** and **0.9744**, respectively.
* **ResNet50** had a significantly lower F1-Score (**0.2924**), indicating poor learning or possible training issues.
* **EfficientNetB0** performed the worst (**F1-Score: 0.00003**), suggesting a mismatch with the dataset or a training failure.

**Best Model:**



Best Performing Model

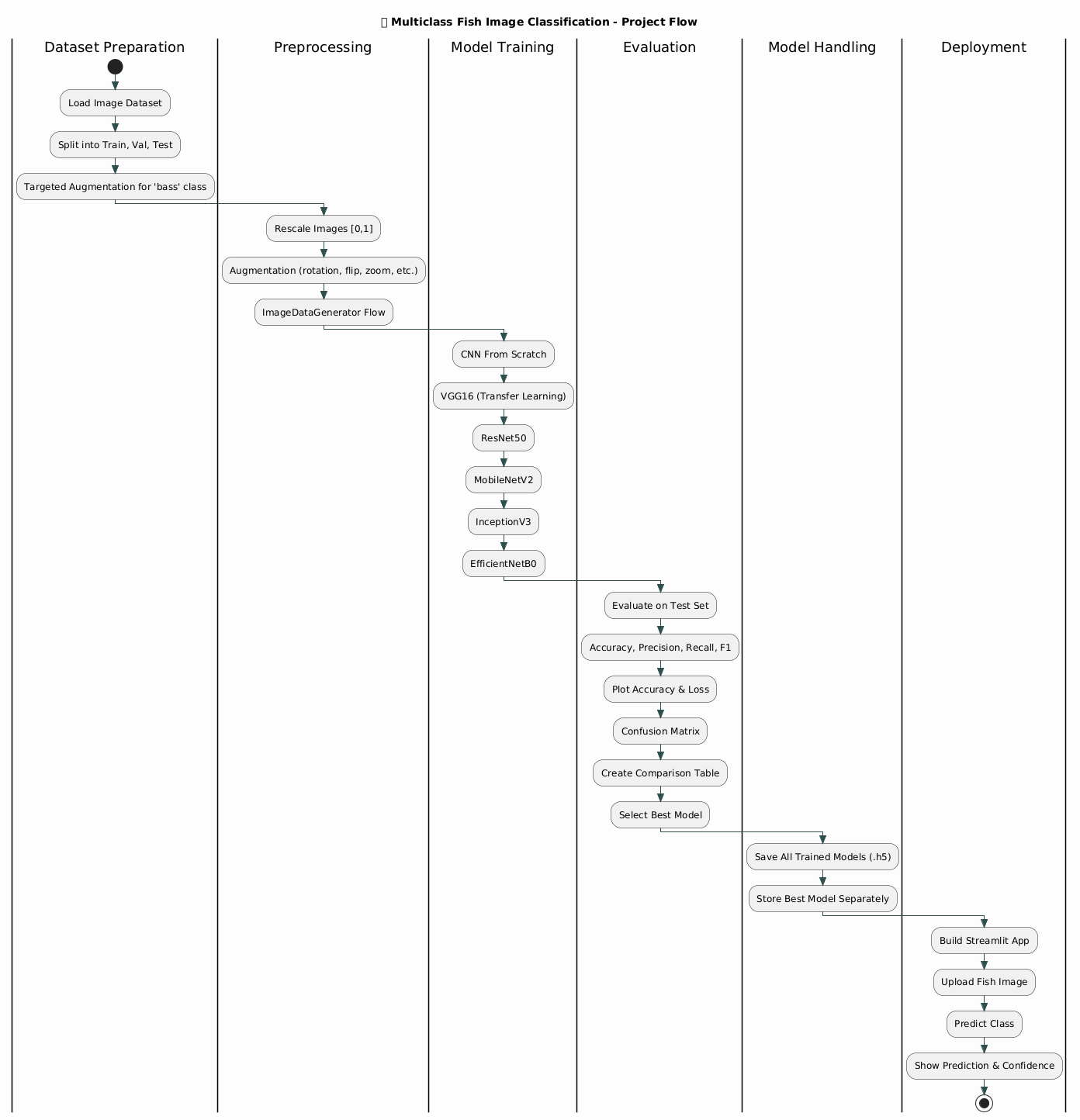
**Streamlit UI:**



Streamlit Application

* **Click "Browse files"** and upload a fish image (JPG, JPEG, PNG).
* **Wait for the image preview** to appear below the upload box.
* **Click "Predict"** to get the fish species along with the confidence score and model used.
* **Click "Reset"** to clear the image and results, and upload a new one.

**Project Flow:**



**Conclusion:**

This project demonstrated a complete pipeline for multiclass image classification using both custom CNNs and transfer learning. Through targeted augmentation and metric-based evaluation, the model's generalization was improved significantly. The best model was deployed using Streamlit, enabling real-time predictions. This solution has practical implications in fisheries, marine research, and ecological monitoring.