Fish Species Recognition Using Convolutional Neural Networks

Student Names & IDs:

. Hazem Mostafa Ahmed - 2227328

. Taha Eid Gomaa - 2227338

. Mohamed Ayman - 2227027

. Magdy Abdelfadeel - 2227157

. Mostafa Ahmed - 2227463

Submission Date: December 14, 2025

Course Name: Neural Networks

Instructor Names: Omar Mourad, Mahmoud Esmat

1. Project Overview

This project addresses **fish species recognition** as an image classification task. Given an input image of a fish, the goal is to predict its species label from a fixed set of classes. Fish recognition is a fine-grained visual problem because multiple species can share similar shapes, textures, and color patterns. The project implements and compares two approaches:

1. A **Baseline CNN** trained from scratch.
2. A **MobileNet transfer learning model** (pretrained on ImageNet) with a custom classification head.

Both models are trained and evaluated using the same dataset split, preprocessing pipeline, and evaluation metrics to ensure a fair comparison.

---

2. Significance and Applications

Automated fish recognition can support multiple real-world use cases:

- **Marine biodiversity monitoring**: helping researchers track species distribution.
- **Fisheries management**: supporting sustainability and stock assessment.
- **Fish markets and supply chain**: enabling automated sorting and quality control.
- **Education**: assisting students and practitioners with species identification.
- **Conservation**: tracking vulnerable or overfished species over time.

---

3. Dataset Description

The dataset consists of labeled fish images organized by species, where each class represents a distinct fish species. All images share consistent visual properties, such as similar image dimensions, which simplifies preprocessing and model training.

The dataset used in this project is publicly available and can be accessed at:
https://www.kaggle.com/datasets/crowww/a-large-scale-fish-dataset

Classes Used

The dataset includes the following nine fish species:

- Black Sea Sprat
- Gilt-Head Bream
- Hourse Mackerel
- Red Mullet
- Red Sea Bream
- Sea Bass
- Shrimp
- Striped Red Mullet
- Trout

Original Dataset Size

The original dataset provided two separate folders:

- One folder containing **augmented images**.
- Another folder containing **non-augmented images** with a very limited number of samples.

In the non-augmented folder:

- Most classes contained **50 images per class**.
- The **Trout** class contained only **30 images**, making it significantly underrepresented.

Due to the small size of the non-augmented dataset, it was excluded from training.

Augmented Dataset Selection

To ensure sufficient data for effective model training, only the **augmented dataset folder** was used. After augmentation, each class contained approximately:

- **1000 images per class**

This resulted in a balanced dataset across all classes and improved the robustness of the trained models.

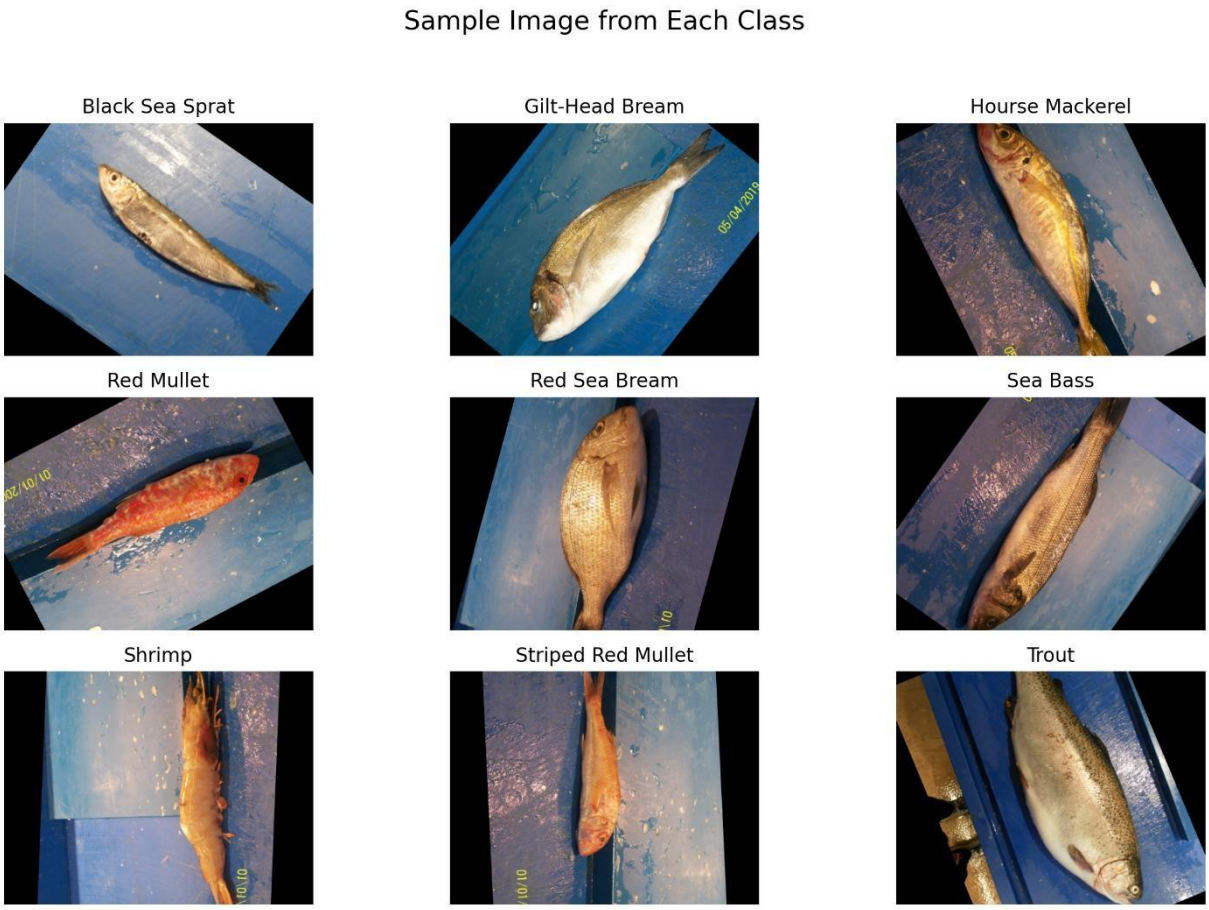
Dataset Split

The final augmented dataset was split into training, validation, and test sets using the following ratios:

- **Training set:** 70%
- **Validation set:** 15%
- **Test set:** 15%

Data Leakage Handling

During initial experiments, an overlap between training and test sets was detected (data leakage), caused by duplicate image files appearing across different splits. To resolve this issue, the dataset was re-split carefully, and **unique file naming** was enforced to ensure that no identical images appeared in more than one split. This step was essential to obtain a valid and realistic evaluation of model performance.



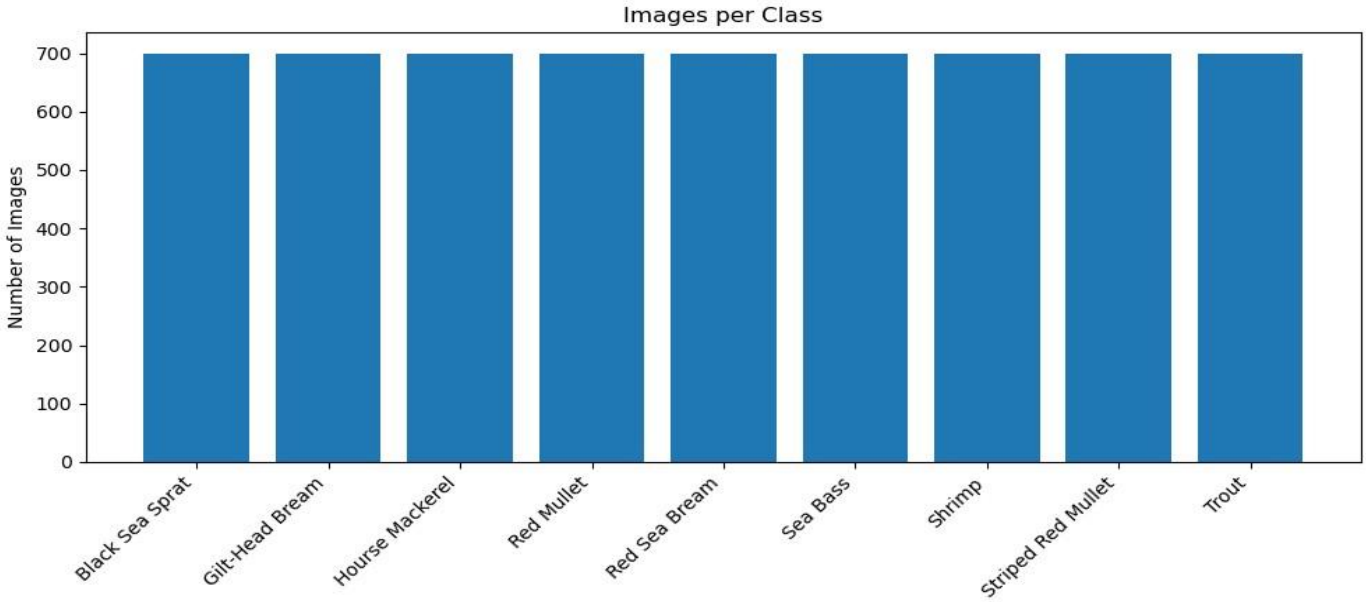
---

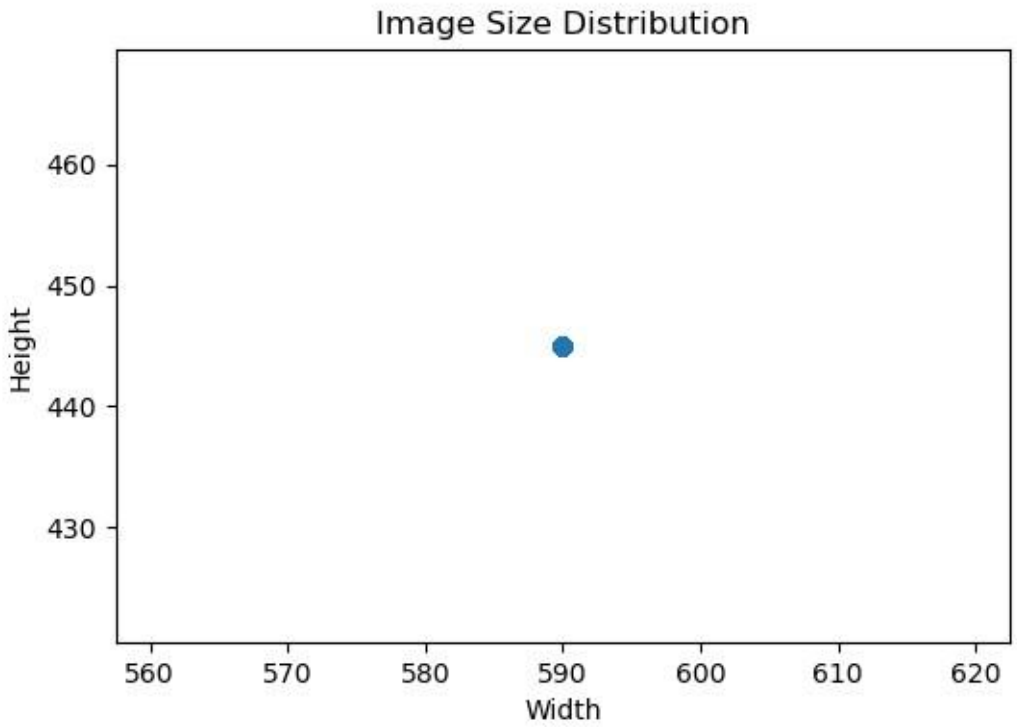
4. Exploratory Data Analysis and Data Cleaning

4.1 Exploratory Data Analysis (EDA)

EDA was performed to understand dataset structure and quality before training. The main checks included:

- **Number of classes** and images per class.
- **Class balance**: the dataset is reasonably balanced across classes, so no resampling or class weighting was required.
- **Image size distribution**: images showed consistent dimensions, reducing risk of irregular resizing issues.
- **Corrupted images detection**: images that failed to load correctly were identified.





4.2 Data Cleaning

A cleaning step removed corrupted or unreadable images. This prevents runtime errors and improves training stability. After cleaning, the dataset was re-verified to ensure all images could be loaded successfully.

---

5. Preprocessing and Data Augmentation

5.1 Preprocessing

All images are processed consistently before training:

- **Resizing** images to a fixed input size (e.g., 128×128).
- **Normalization**: pixel values are scaled to the range **[0, 1]** by dividing by 255.

This ensures stable gradients and faster convergence.

5.2 Data Augmentation

To improve generalization and reduce overfitting, data augmentation is applied **only to the training set**.
Augmentation operations include:

- Random rotations
- Horizontal flipping
- Shifts (width/height)
- Zoom
- Brightness adjustments

Validation and test sets are not augmented; they are only normalized to measure real performance.

---

6. Methodology

6.1 Baseline CNN (From Scratch)

The baseline model is a simple convolutional neural network trained from scratch. Its purpose is to:

- Validate the full training/evaluation pipeline.
- Provide a reference point for comparison against transfer learning.

Typical structure:

- Convolution + ReLU + MaxPooling blocks
- Fully connected layer(s)
- Dropout for regularization
- Softmax output for multi-class classification

6.2 MobileNet (Transfer Learning)

MobileNet is used as a second model to improve performance and training efficiency:

- The MobileNet backbone is loaded with **ImageNet pretrained weights**.
- The top classification layer is removed (include_top=False).
- The backbone is frozen initially (feature extractor mode).
- A custom head is added:
    - Global Average Pooling ○ Dense layer ○ Dropout
    - Softmax output layer with number of classes

This approach benefits from strong pretrained visual features and typically converges faster than training from scratch.

---

7. Training Setup

7.1 Training Configuration

- Optimizer: **Adam**
- Loss: **Categorical Cross-Entropy**
- Metrics: **Accuracy**
- Batch size: **32**
- Epochs: **30**
- EarlyStopping: used to stop training when validation performance stops improving
- ModelCheckpoint: saves the best model weights

Note: Training was performed on CPU due to local hardware limitations. Model training speed can be significantly improved using cloud GPU platforms if needed.

7.2 Saved Artifacts

The workflow saves:

- Trained model weights (.h5)
- Classification report text file
- Confusion matrix images (raw counts + normalized)
- accuracy/loss curves

---

8. Evaluation

The models were evaluated using multiple metrics to capture both overall and class-level behavior.

8.1 Accuracy

Accuracy measures the percentage of correct predictions across the test set. While accuracy is a useful global measure, it does not fully explain which classes are confused with others.

- **Baseline test accuracy: 0.9726**
- **MobileNet test accuracy: 0.9948**

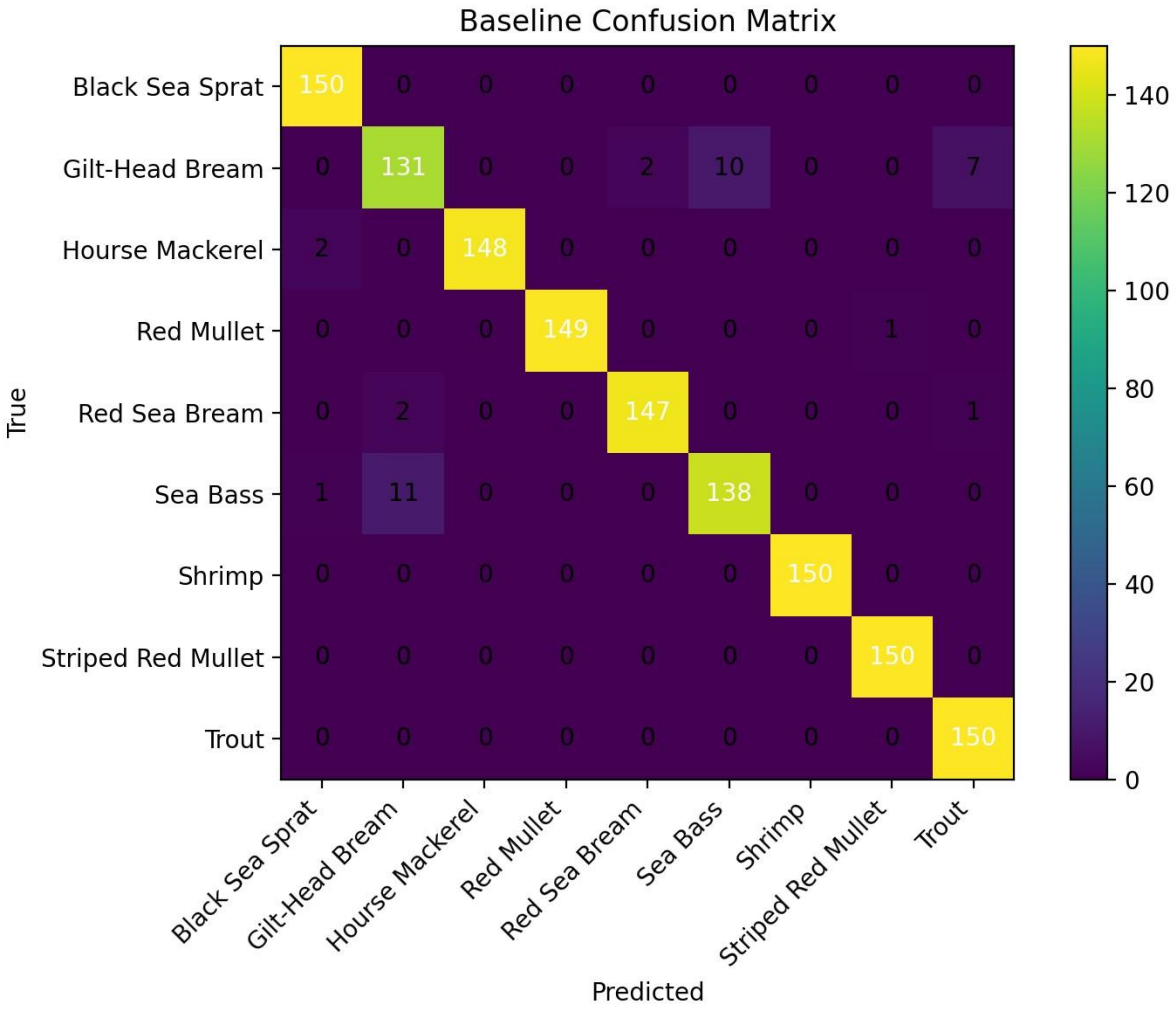
8.2 Classification Report (Precision, Recall, F1-score)

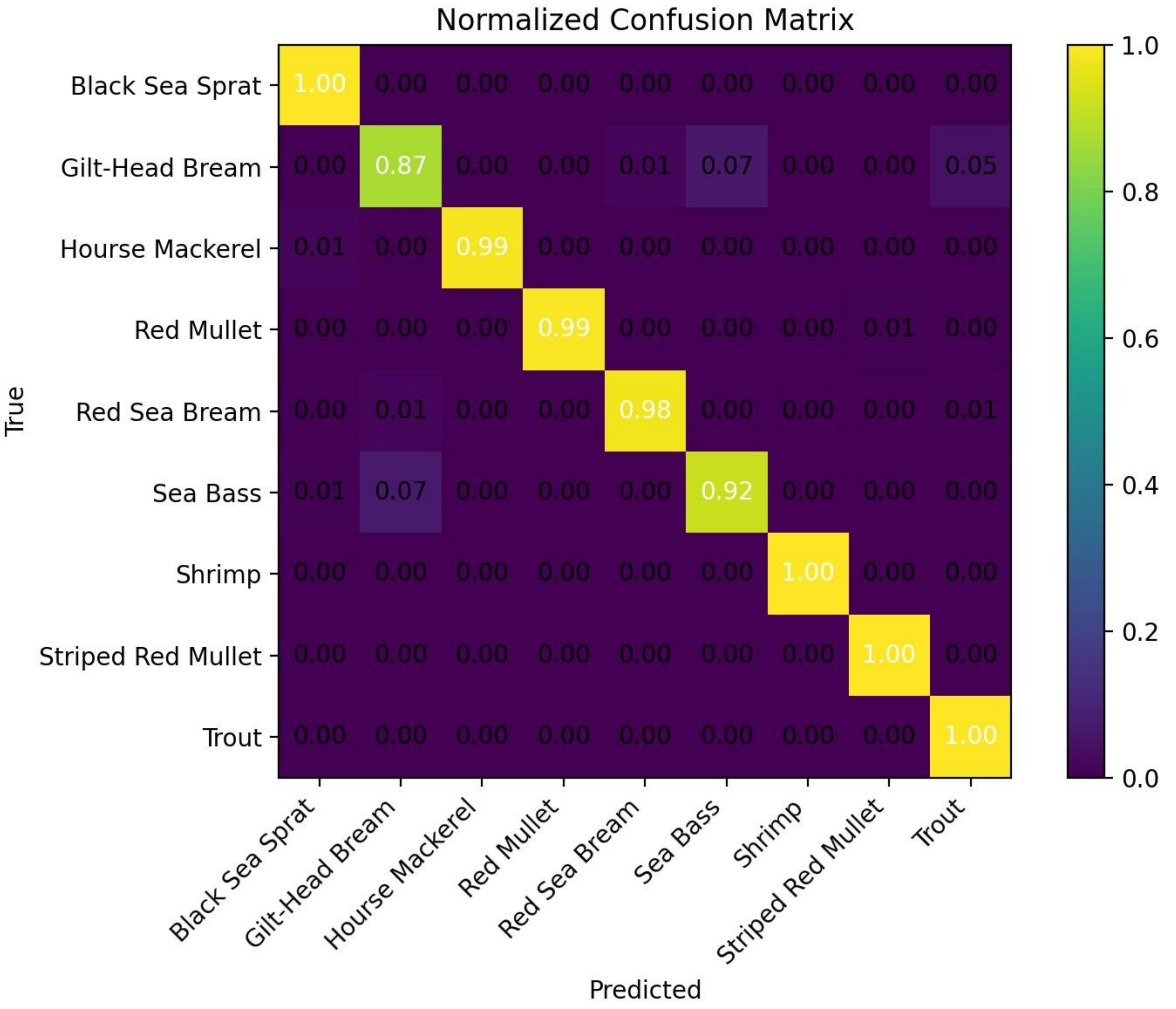
The classification report provides per-class metrics:

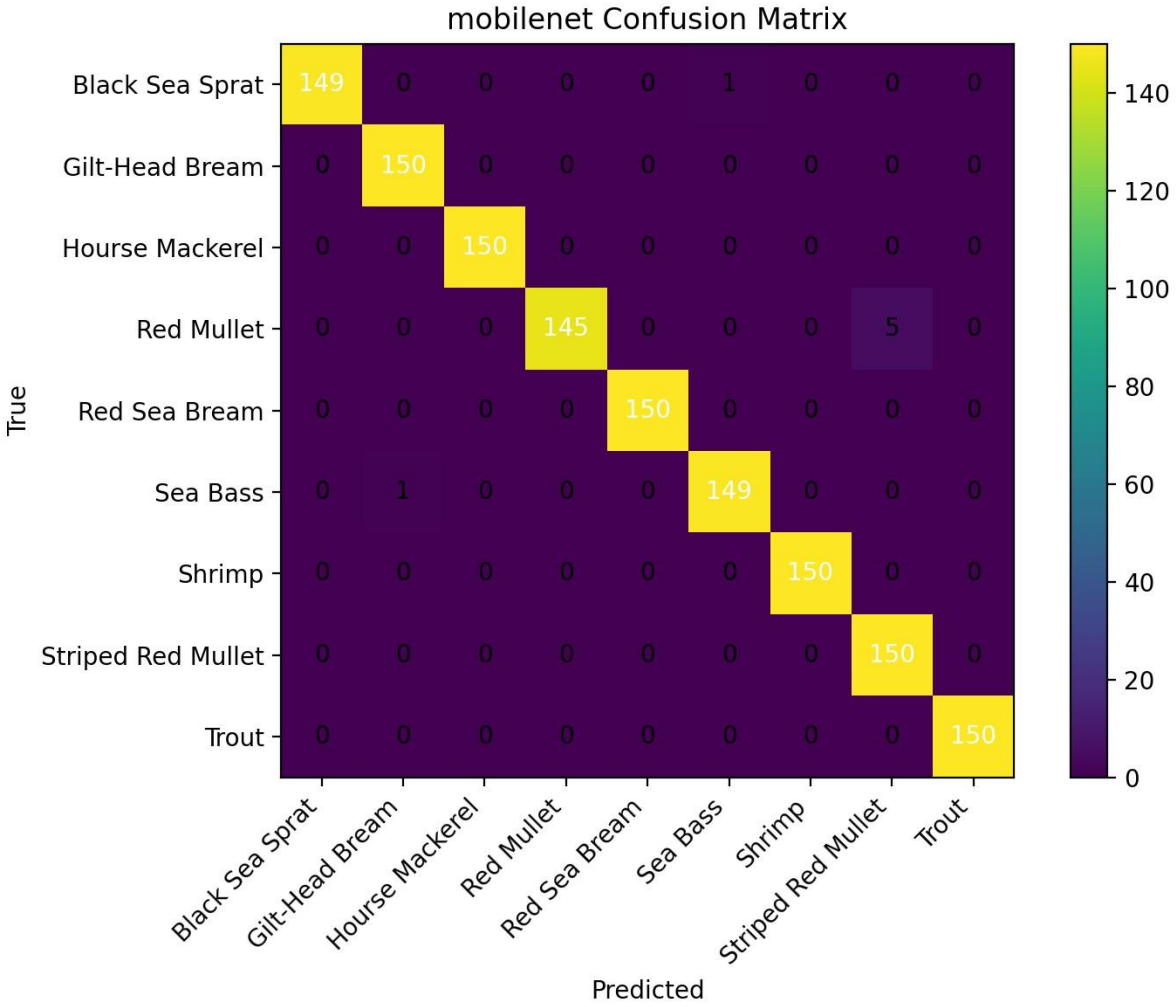
- **Precision:** how many predicted samples for a class are correct
- **Recall:** how many true samples of a class were correctly detected
- **F1-score:** harmonic mean of precision and recall (most informative overall)
- **Baseline macro/weighted F1: 0.9724**
- **MobileNet macro/weighted F1: 0.9948**

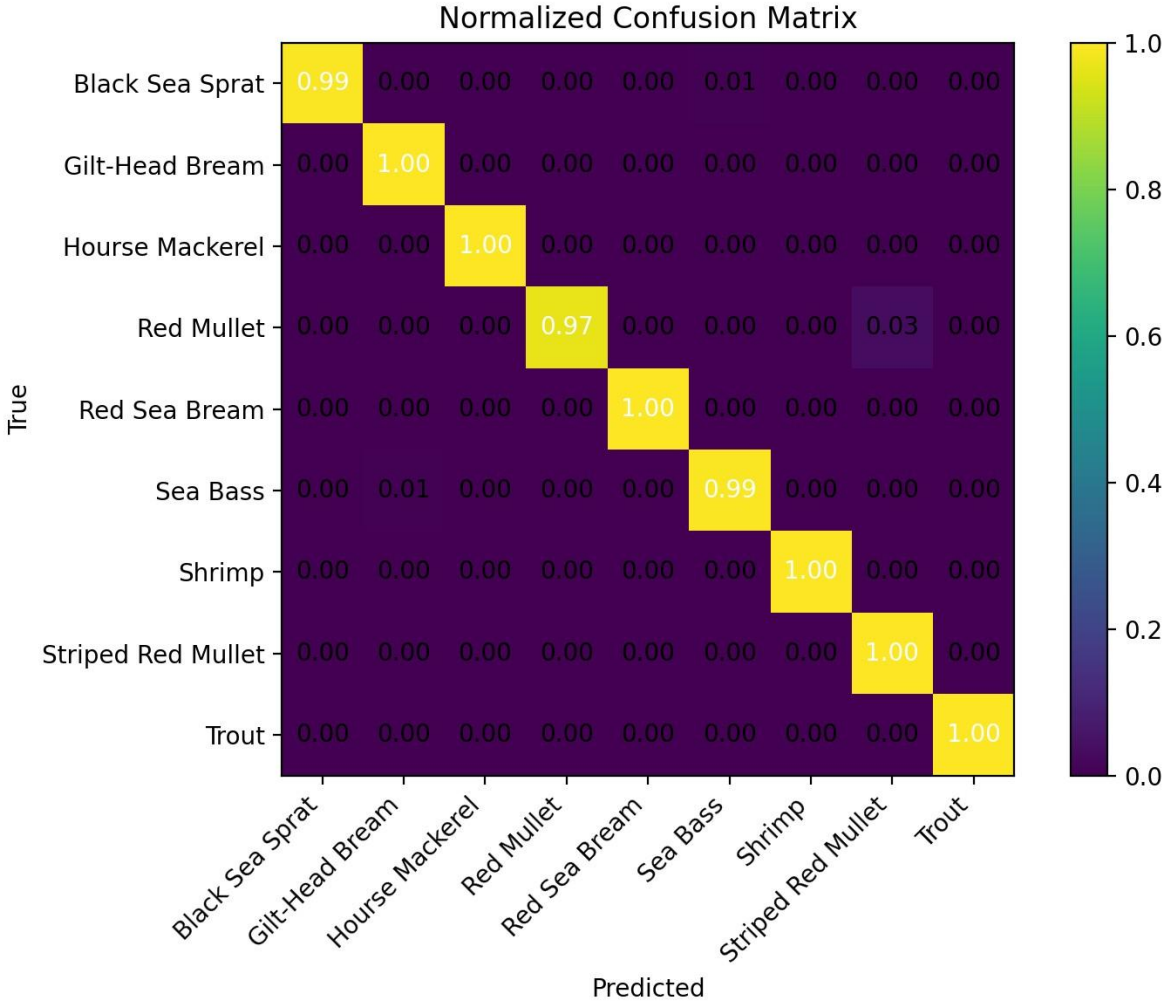
8.3 Confusion Matrix

The confusion matrix shows which classes the model confuses. A strong model shows a clear diagonal structure with minimal off-diagonal errors. In this project, most misclassifications occur between visually similar fish species, which is expected in fine-grained classification tasks.


Baseline Confusion Matrix
Black Sea Sprat
Gilt-Head Bream
Hourse Mackerel
Red Mullet
Red Sea Bream
Sea Bass
Shrimp
Striped Red Mullet
Trout
True
Predicted
150 0 0 0 0 0 0 0 0
0 131 0 0 2 10 0 0 7
2 0 148 0 0 0 0 0 0
0 0 0 149 0 0 0 1 0
0 2 0 0 147 0 0 0 1
1 11 0 0 0 138 0 0 0
0 0 0 0 0 0 150 0 0
0 0 0 0 0 0 0 150 0
0 0 0 0 0 0 0 0 150
140
120
100
80
60
40
20
0


Normalized Confusion Matrix
Black Sea Sprat
Gilt-Head Bream
Hourse Mackerel
Red Mullet
Red Sea Bream
Sea Bass
Shrimp
Striped Red Mullet
Trout
True
Predicted
1.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.87 0.00 0.00 0.01 0.07 0.00 0.00 0.05
0.01 0.00 0.99 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.99 0.00 0.00 0.00 0.01 0.00
0.00 0.01 0.00 0.00 0.98 0.00 0.00 0.00 0.01
0.01 0.07 0.00 0.00 0.00 0.92 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 1.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 1.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 1.00
1.0
0.8
0.6
0.4
0.2
0.0


mobilenet Confusion Matrix
Black Sea Sprat
Gilt-Head Bream
Hourse Mackerel
Red Mullet
Red Sea Bream
Sea Bass
Shrimp
Striped Red Mullet
Trout
True
Predicted
149 0 0 0 0 1 0 0 0
0 150 0 0 0 0 0 0 0
0 0 150 0 0 0 0 0 0
0 0 0 145 0 0 0 5 0
0 0 0 0 150 0 0 0 0
0 1 0 0 0 149 0 0 0
0 0 0 0 0 0 150 0 0
0 0 0 0 0 0 0 150 0
0 0 0 0 0 0 0 0 150
140
120
100
80
60
40
20
0

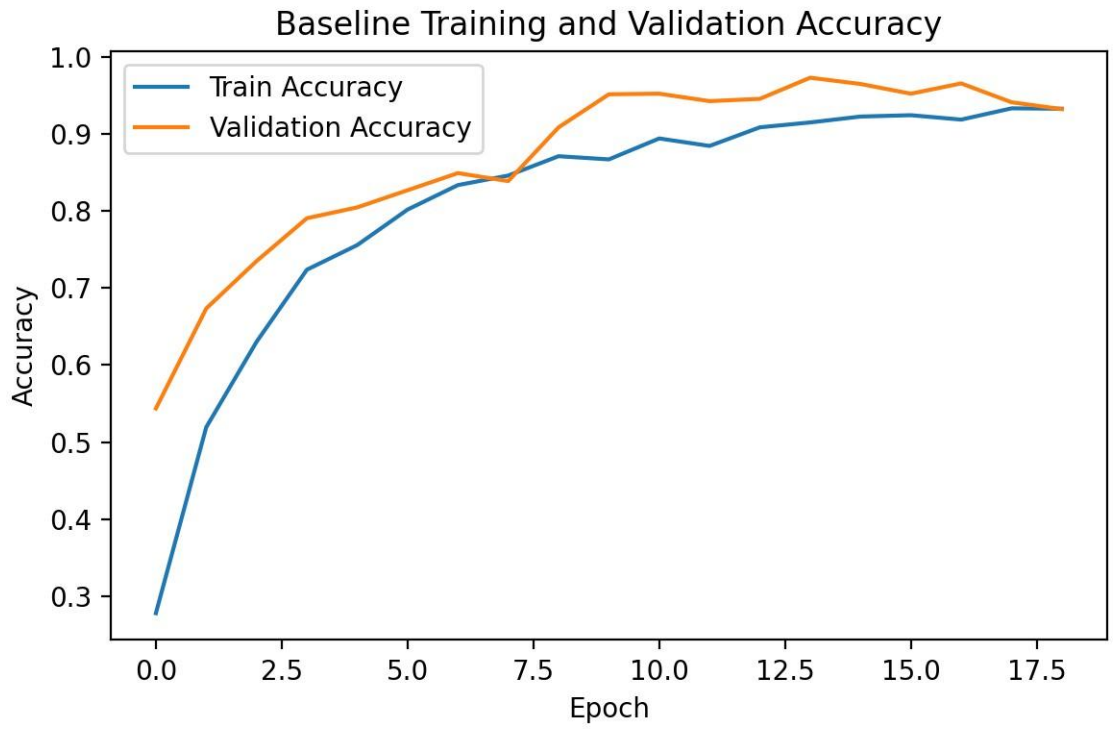

Normalized Confusion Matrix
Black Sea Sprat
Gilt-Head Bream
Hourse Mackerel
Red Mullet
Red Sea Bream
Sea Bass
Shrimp
Striped Red Mullet
Trout
True
Predicted
0.99 0.00 0.00 0.00 0.00 0.01 0.00 0.00 0.00
0.00 1.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 1.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.97 0.00 0.00 0.00 0.03 0.00
0.00 0.00 0.00 0.00 1.00 0.00 0.00 0.00 0.00
0.00 0.01 0.00 0.00 0.00 0.99 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 1.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 1.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 1.00
1.0
0.8
0.6
0.4
0.2
0.0

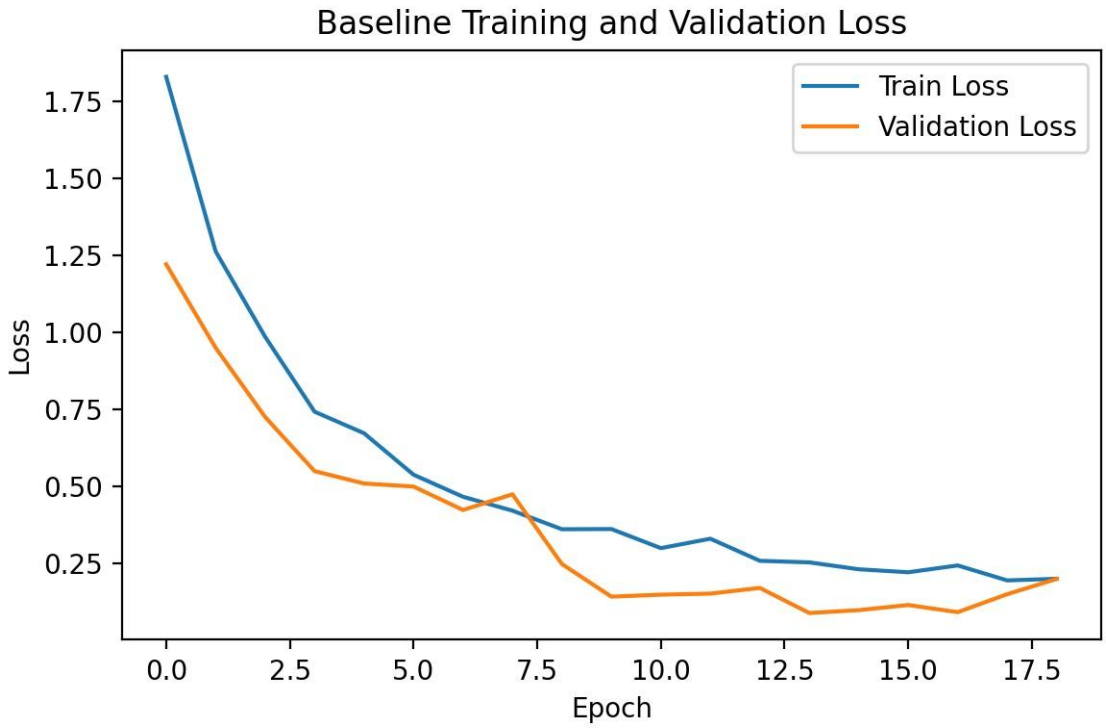
9. Results and Comparison

This section compares Baseline CNN and MobileNet using the same test split.

9.1 Baseline Summary

- Strength: Simple, interpretable, validates pipeline.
- Performance:

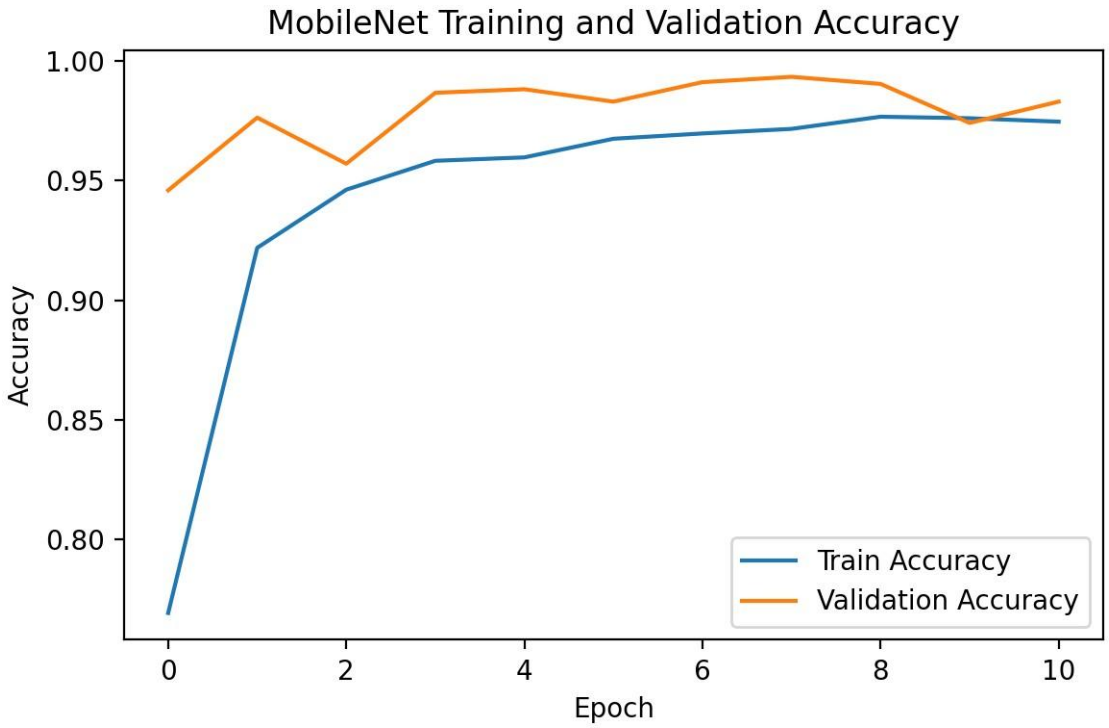


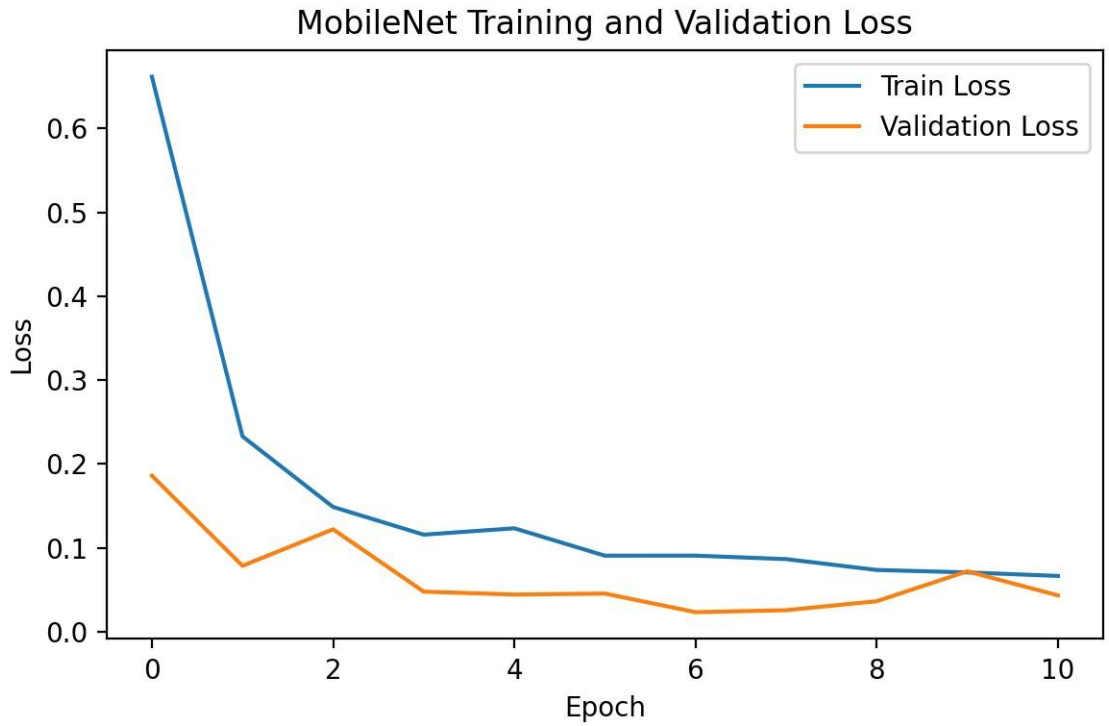


- Errors: Mostly between visually similar species (as shown in confusion matrix).

9.2 MobileNet Summary

- Strength: Uses pretrained features, faster convergence, typically better generalization.
- Performance:





- Errors: Reduced confusion in difficult class pairs compared to baseline.

9.3 Discussion of Differences

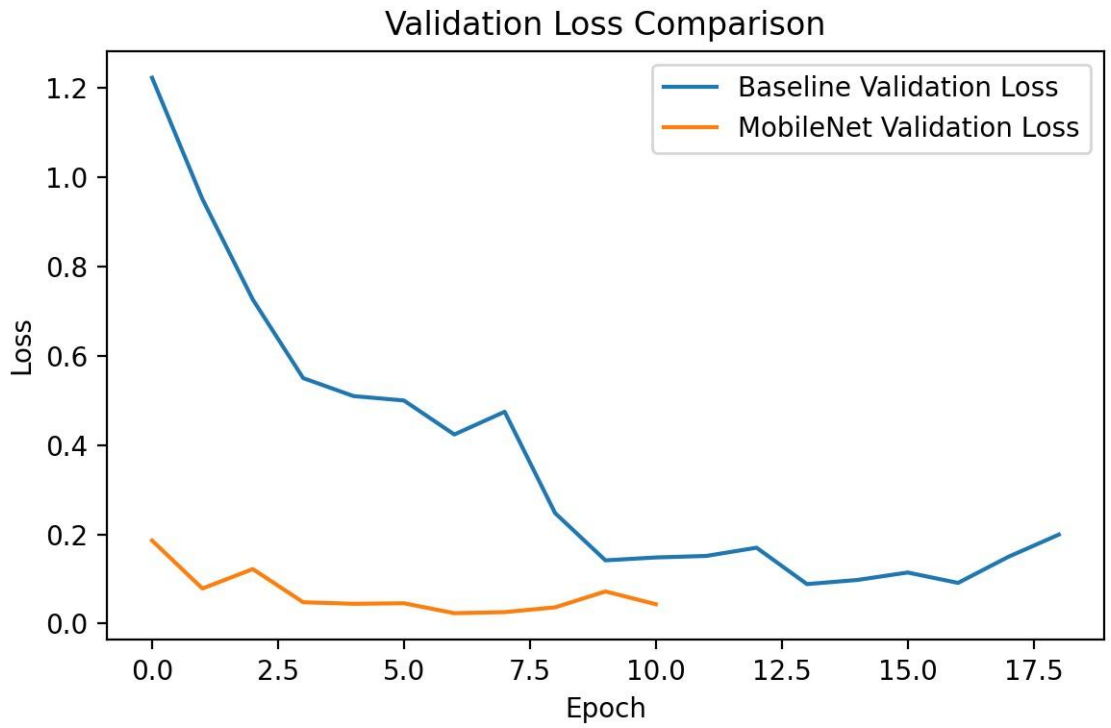
Possible reasons MobileNet performs better (if it does):

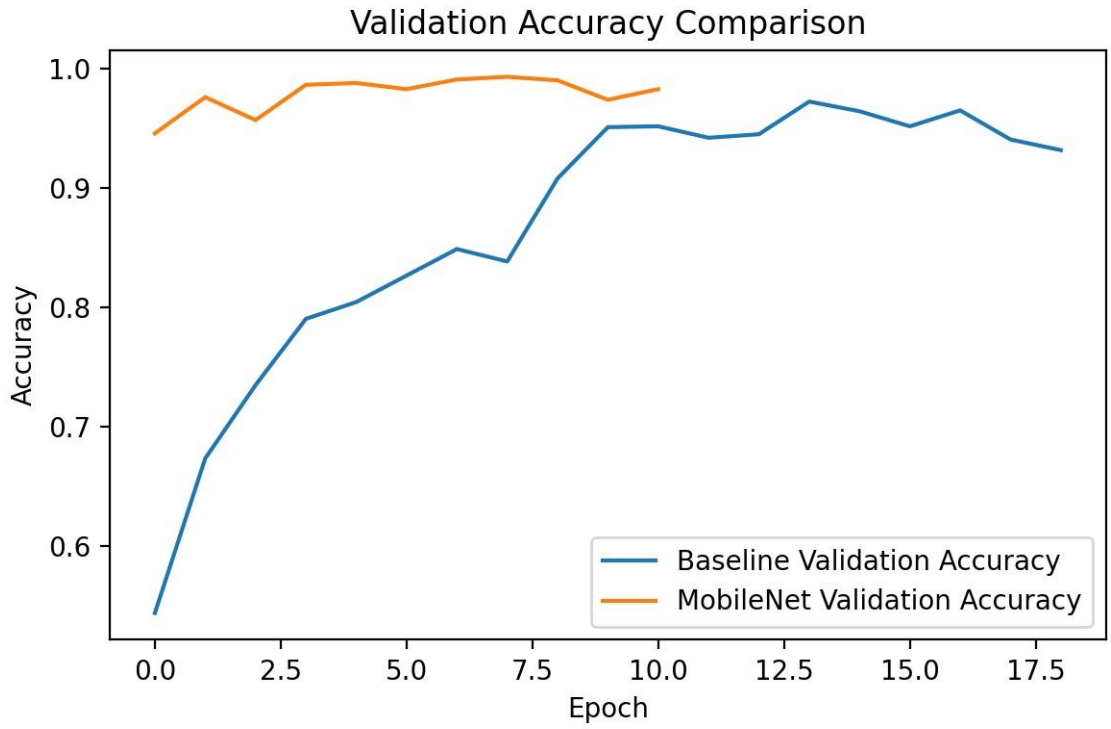
- Pretrained backbone captures general visual features (edges, textures, shapes).
- Less data required to learn strong representations.
- Better generalization under real-world variation.

If MobileNet performance is similar to baseline, that can still be explained as:

- Dataset is clean and consistent; baseline is already strong.

- Classes are visually separable; transfer learning provides marginal gains.





---

10. Error Analysis and Discussion

Even with strong performance, misclassifications can occur due to:

- **Visual similarity** between certain species.
- **Pose variation** (fish orientation, partial visibility).
- **Lighting/background differences**.
- **Texture and color overlap** across species.

11. Conclusion

This project successfully demonstrates a full deep learning pipeline for fish species recognition, including:

- EDA and data cleaning
- Preprocessing and augmentation
- Two modeling approaches (baseline CNN and MobileNet transfer learning)
- Evaluation using accuracy, classification report metrics, and confusion matrices

The final results show strong performance on the test set, and confusion matrices confirm that most predictions are correct with limited confusion mainly among visually similar species.

---

12. Future Improvements

Possible extensions to improve robustness and real-world performance:

1. **Fine-tuning MobileNet** (unfreeze top layers with low learning rate).
2. Try **EfficientNet** or **ResNet** for stronger feature extraction.
3. Add **background segmentation** to reduce background bias.
4. Increase dataset diversity (different lighting, environments, camera angles).
5. Add systematic **hard-example mining** to focus training on difficult cases.

---

13. References

TensorFlow Developers. TensorFlow Documentation.
https://www.tensorflow.org/ Keras Team.
Keras Documentation.
https://keras.io/

Kaggle Dataset. A Large-Scale Fish Dataset.
https://www.kaggle.com/datasets/crowww/a-large-scale-fish-dataset

Howard, A. G., Zhu, M., Chen, B., et al. (2017).
MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.
https://arxiv.org/abs/1704.04861

Pedregosa, F., Varoquaux, G., Gramfort, A., et al. (2011).
Scikit-learn: Machine Learning in Python.
Journal of Machine Learning Research, 12, 2825–2830.