information

MDPI

Article

CNN-Based Optimization for Fish Species Classification: Tackling Environmental Variability, Class Imbalance, and Real-Time Constraints

Amirhosein Mohammadisabet [1], Raza Hasan [1,*], Vishal Dattana [2], Salman Mahmood [3] and Saqib Hussain [4]

[1] Department of Computer Science, Solent University, Southampton SO14 0YN, UK; amirhoseinmohammadisabet@gmail.com
[2] Department of Computer Science and Management Information System, Oman College of Management & Technology, P.O. Box 680, Barka 320, Oman; vdattana@ocmt.edu.om
[3] Department of Computer Science, Nazeer Hussain University, ST-2, Near Karimabad, Karachi 75950, Pakistan; salman.mahmood@nhu.edu.pk
[4] Department of Computer and Information Sciences, Northumbria University, Newcastle upon Tyne NE1 8QH, UK; saqib2.hussain@northumbria.ac.uk
* Correspondence: raza.hasan@solent.ac.uk

Abstract: Automated fish species classification is essential for marine biodiversity monitoring, fisheries management, and ecological research. However, challenges such as environmental variability, class imbalance, and computational demands hinder the development of robust classification models. This study investigates the effectiveness of convolutional neural network (CNN)-based models and hybrid approaches to address these challenges. Eight CNN architectures, including DenseNet121, MobileNetV2, and Xception, were compared alongside traditional classifiers like support vector machines (SVMs) and random forest. DenseNet121 achieved the highest accuracy (90.2%), leveraging its superior feature extraction and generalization capabilities, while MobileNetV2 balanced accuracy (83.57%) with computational efficiency, processing images in 0.07 s, making it ideal for real-time deployment. Advanced preprocessing techniques, such as data augmentation, turbidity simulation, and transfer learning, were employed to enhance dataset robustness and address class imbalance. Hybrid models combining CNNs with traditional classifiers achieved intermediate accuracy with improved interpretability. Optimization techniques, including pruning and quantization, reduced model size by 73.7%, enabling real-time deployment on resource-constrained devices. Grad-CAM visualizations further enhanced interpretability by identifying key image regions influencing predictions. This study highlights the potential of CNN-based models for scalable, interpretable fish species classification, offering actionable insights for sustainable fisheries management and biodiversity conservation.

Keywords: fish species classification; convolutional neural networks; hybrid deep learning models; transfer learning; data augmentation; turbidity simulation; real-time deployment; Grad-CAM visualization; fisheries management; marine biodiversity monitoring

Academic Editor: Aneta Poniszewska-Maranda

Received: 24 December 2024
Revised: 24 January 2025
Accepted: 12 February 2025
Published: 19 February 2025

Citation: Mohammadisabet, A.; Hasan, R.; Dattana, V.; Mahmood, S.; Hussain, S. CNN-Based Optimization for Fish Species Classification: Tackling Environmental Variability, Class Imbalance, and Real-Time Constraints. *Information* **2025**, *16*, 154. https://doi.org/10.3390/info16020154

1. Introduction

The accurate and efficient classification of fish species is vital for marine biodiversity conservation, fisheries management, and ecological research. Fish play a critical role in maintaining ecosystem balance, yet their populations are increasingly threatened by overfishing, habitat degradation, and climate change. Monitoring these populations is

essential, but traditional methods rely heavily on the manual identification of species based on morphological traits such as body shape, fin structure, and coloration. While effective for small-scale studies, these methods are labor-intensive, prone to human error, and unsuitable for large-scale or real-time monitoring applications [1].

Advancements in artificial intelligence (AI), particularly in deep learning, have revolutionized image classification by enabling automated and scalable solutions. CNNs, which autonomously extract hierarchical features, have demonstrated superior performance across various applications [2–4]. In fish species classification, CNNs offer a promising alternative to traditional and manual methods, combining efficiency and accuracy. However, their application in underwater environments presents unique challenges. Automated fish species classification faces several challenges in underwater environments, including environmental variability (e.g., murky water, lighting fluctuations), class imbalance, and the need for large, labeled datasets. These challenges hinder the development of robust models that can perform well under real-world conditions [5,6].

Another critical issue in fish species classification is dataset imbalance, where certain species are overrepresented, leading to biased models that perform poorly on underrepresented species [7,8]. Computational demands also remain a significant hurdle, particularly for real-time deployment, where lightweight models are required for efficiency without compromising accuracy [6].

Recent studies have explored hybrid models that combine CNNs with traditional machine learning techniques, such as SVMs and random forest, to improve classification accuracy and interpretability in complex environments [7]. These hybrid approaches leverage the feature extraction capabilities of CNNs while employing additional classifiers for enhanced resilience to variability and noise. Despite these advancements, challenges remain, particularly in addressing dataset limitations, computational efficiency, and the need for real-time deployment [9].

This study builds upon prior work by evaluating the performance of CNN-based models and hybrid approaches for underwater fish species classification, with a focus on scalability, robustness, and real-world applicability. The primary objectives of this research are as follows:

1. To evaluate CNN architectures, such as DenseNet121, MobileNetV2, and Xception, for underwater fish classification.
2. To investigate the impact of transfer learning and data augmentation on performance, particularly for underrepresented species and noisy underwater images.
3. To compare CNN-based models with hybrid approaches, balancing classification accuracy, interpretability, and computational efficiency.
4. To optimize models for real-time deployment using techniques such as pruning, quantization, and lightweight architectures.
5. To explore the practical application of these models in marine biodiversity monitoring, fisheries management, and conservation efforts.

This research advances the state of automated fish classification by addressing key challenges such as underwater environmental variability, dataset imbalance, and computational demands. By integrating deep learning and hybrid approaches, it demonstrates the potential of scalable robust solutions to support ecological monitoring, conservation strategies, and sustainable fishery management.

2. Literature Review

Automated fish species classification is crucial for advancing marine biodiversity monitoring, fisheries management, and conservation efforts. Traditional manual methods have given way to automated systems, especially deep learning models, which offer higher

accuracy and scalability. However, challenges such as class imbalance, environmental variability, and computational demands persist. This review evaluates the progression of fish classification techniques, highlighting their strengths, limitations, and alignment with the objectives of this study.

2.1. Traditional Approaches

Early fish classification methods relied heavily on the manual observation of physical traits like body shape, fin structure, and coloration. While these methods provided accurate results in small-scale studies, they were time-consuming, prone to human error, and unsuitable for large-scale ecological monitoring [9].

To address these inefficiencies, traditional machine learning (ML) models such as SVMs and random forest introduced automation by analyzing manually extracted features like texture and shape descriptors. These models offered improvements in handling structured datasets but struggled with challenges in dynamic underwater environments, including noise, lighting variability, and complex backgrounds [10–13]. Table 1 summarizes the key strengths and limitations of these traditional approaches, illustrating the progression from manual identification to automated methods and the persistent challenges they faced.

Table 1. Traditional methods and their challenges.

Approach	Strengths	Limitations
Manual Identification [11]	Accurate for small-scale studies	Labor-intensive; lacks scalability
SVM [12]	Effective on small datasets	Poor performance in noisy environments
Random Forest [13]	Handles overfitting; interpretable	Reliance on manual feature engineering

2.2. Advances in Deep Learning

Deep learning models have transformed fish classification by automating feature extraction and learning intricate patterns. Key architectures like VGG16, DenseNet121, and MobileNetV2 have shown notable performance in underwater environments, achieving accuracies ranging from 83% to 93.8% [9,14,15]. Table 2 summarizes their features, strengths, and limitations. Despite these advancements, challenges like dataset imbalance and variability in underwater conditions persist, highlighting the need for further innovation.

Table 2. Key deep learning architectures for fish classification.

Model	Notable Features	Reported Accuracy	Limitations
VGG16 [14]	Deep architecture; robust features	93.8%	High computational cost
MobileNetV2 [15]	Lightweight; real-time suitability	83.0%	Slightly lower accuracy
DenseNet121 [9]	Feature reuse; dense connections	90.0%	Requires significant resources

These architectures have advanced the field significantly, but they still face challenges in handling real-world underwater variability and imbalanced datasets.

2.3. Hybrid Deep Learning Models

Hybrid models that integrate convolutional neural networks (CNNs) with traditional classifiers or recurrent networks have shown significant promise in addressing the challenges of underwater fish classification. These approaches leverage the strengths of CNNs

for feature extraction while employing other techniques to tackle specific challenges, such as temporal variations and feature refinement.

For instance, Chhabra et al. [14] achieved an accuracy of 93.8% by combining VGG16 with an ensemble approach, demonstrating improved robustness in datasets with limited species. Jalal et al. [16] employed YOLO for detection and Gaussian mixture models for classification, achieving 91.6% accuracy in dynamic underwater videos. Similarly, Veluswami et al. [9] integrated DenseNet121 with a squeeze-and-excitation (SE) architecture, achieving approximately 90% accuracy while enhancing model generalization across species.

Nguyen et al. [17] demonstrated the effectiveness of hybrid CNN-RNN models in handling temporal variations in underwater scenes. While this approach improved adaptability in dynamic scenarios, challenges related to real-time processing persisted. Additionally, Mampitiya et al. [18] explored hybrid models incorporating an SVM and random forest, finding enhanced accuracy and feature extraction capabilities. However, these models lacked the flexibility required for highly dynamic underwater environments.

Table 3 provides a summary of these studies, highlighting their approaches, achieved accuracies, and addressed challenges.

Table 3. Hybrid models for fish classification.

Study	Approach	Accuracy Achieved	Challenges Addressed
[14]	VGG16 + Ensemble	93.8%	Improved robustness in limited species
[16]	YOLO + Gaussian Mixture Model	91.6%	Tackled dynamic underwater videos
[9]	DenseNet121 + SE Architecture	90%	Improved generalization across species
[17]	Hybrid CNN-RNN	98%	Addressed temporal variations; real-time lag
[18]	Hybrid SVM + Random Forest	98.89%	Improved accuracy; lacked adaptability

2.4. Challenges in Underwater Fish Classification

Environmental variability and class imbalance are well-documented challenges in fish classification studies [19,20]. Previous research has proposed data augmentation, transfer learning, and hybrid models as effective solutions. This study builds upon these approaches, employing methods to improve model robustness [13]. Table 4 summarizes the primary challenges in underwater fish classification and the corresponding solutions proposed to address them.

Table 4. Challenges in fish classification and proposed solutions.

Challenge	Description	Proposed Solutions
Environmental Variability [19]	Lighting and turbidity variability	Data augmentation; multimodal models
Class Imbalance [20]	Overrepresentation of certain species	Synthetic data generation
Dataset Limitations [13]	Small, non-diverse datasets	Transfer learning; dataset expansion

Existing studies have shown promising results in fish species classification; however, several gaps remain that align with the objectives of this study. A major limitation is the generalization of models, as many studies validate their models on small, controlled datasets, which restricts their application to real-world underwater environments. Another significant issue is class imbalance, with few studies effectively addressing the disproportionate representation of certain species, a challenge crucial for achieving robust classification across all species. Real-time feasibility is also a concern, as high-performing models like DenseNet121 are computationally expensive, making them less suitable for real-time deployment in practical scenarios. Lastly, while hybrid models show promise, their

integration and performance in comparison to end-to-end CNN models require further exploration to fully understand their potential.

These identified gaps are directly linked to the objectives of this study, which are as follows: (1) to develop CNN-based hybrid models that improve generalization and robustness in dynamic underwater environments; (2) to use transfer learning, data augmentation, and turbidity simulation techniques to mitigate class imbalance and enhance model adaptability; (3) to compare CNN-based models with traditional and hybrid approaches to identify the most efficient and accurate solutions; (4) to optimize models for real-world deployment using techniques such as pruning and quantization to reduce computational demands; and (5) to evaluate the applicability of these models in fishery management and biodiversity monitoring scenarios while ensuring interpretability through Grad-CAM visualizations. Addressing these gaps is essential for advancing automated fish species classification, improving model scalability and efficiency, and enhancing its effectiveness for ecological and conservation applications.

3. Methodology

This section details the dataset, preprocessing techniques, model architectures, evaluation metrics, and experimental setup employed in this study. The focus is on addressing the challenges of underwater fish classification, including dataset variability, class imbalance, and environmental noise, through advanced deep learning and hybrid approaches. The methodology prioritizes reproducibility and aims to develop robust and scalable solutions for real-world ecological applications.

3.1. Framework Overview

To ensure a comprehensive approach to addressing the challenges in fish species classification, a structured framework was developed, as illustrated in Figure 1.

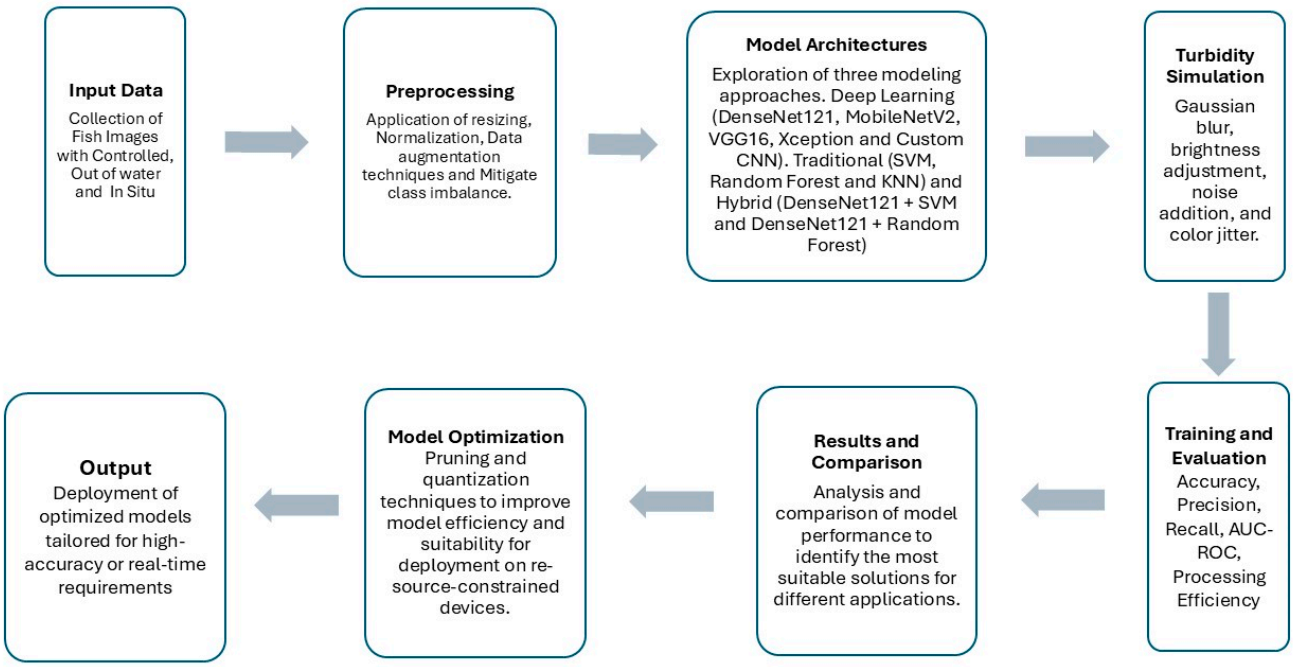

Figure 1. Framework for the research methodology in fish species classification.

This framework outlines the logical flow of the research methodology, covering the following six interconnected stages:

1. Input data: collection of fish images from controlled, out-of-water, and in situ conditions to represent diverse environmental scenarios.
2. Preprocessing: application of resizing, normalization, and data augmentation techniques to enhance data quality and mitigate class imbalance [21].
3. Model architectures: exploration of the following three modeling approaches:
   a. Deep learning using pre-trained CNNs like DenseNet121 and MobileNetV2 [22].
   b. Traditional machine learning models such as SVMs and random forest.
   c. Hybrid approaches combining CNN feature extraction with traditional classifiers [23].
4. Turbidity simulation: simulation of low-visibility conditions using techniques such as Gaussian blur, brightness adjustment, noise addition, and color jitter [24].
5. Training and evaluation: implementation of model training and evaluation using metrics such as accuracy, precision, recall, and AUC-ROC [25].
6. Results and comparison: analysis and comparison of model performance to identify the most suitable solutions for different applications.
7. Model optimization: application of pruning and quantization techniques to improve model efficiency and suitability for deployment on resource-constrained devices [26].
8. Output: deployment of optimized models tailored for high-accuracy or real-time requirements.

3.2. Dataset

The dataset used in this study consists of 3960 images representing 468 fish species, categorized under the three following conditions:

1. Controlled conditions: images captured in controlled environments with consistent lighting and plain backgrounds.
2. Out-of-water conditions: photographs of fish outside water with varied lighting and uncontrolled backgrounds.
3. In situ (underwater) conditions: images taken in natural underwater environments, characterized by dynamic lighting, complex backgrounds, and varying turbidity.

The dataset exhibited significant class imbalance, with certain species represented by fewer than 10 images. This imbalance posed a challenge for training models that generalize well across all species, as shown in Table 5. The dataset is available at Kaggle: https://www.kaggle.com/datasets/sripaadsrinivasan/fish-species-image-data, (accessed on 6 December 2024).

Table 5. Dataset distribution across conditions.

Condition	Number of Images	Percentage of Total
Controlled Conditions	1200	30.3%
Out-of-Water Conditions	1160	29.3%
In Situ Conditions	1600	40.4%

3.3. Data Preprocessing and Augmentation

Preprocessing and augmentation techniques were applied to enhance dataset quality, simulate real-world variability, and mitigate the effects of class imbalance.

3.3.1. Preprocessing Steps

- Resizing: images were resized to the required input dimensions for each model [27] as follows:
  - 224 × 224 pixels for MobileNetV2, DenseNet121, and most CNNs.

- 299 × 299 pixels for Xception to accommodate its architectural design.

- Normalization: pixel values were normalized to a range of [0, 1] to improve model convergence [28].
- Label encoding: class labels were encoded using one-hot encoding for multi-class classification.

3.3.2. Data Augmentation

To artificially expand the dataset and address class imbalance, the following augmentation techniques were applied:

- Geometric transformations: random rotations, flips, and cropping simulated variability in fish orientations [29].
- Color adjustments: variations in brightness, contrast, and saturation mimicked underwater lighting conditions [30].
- Noise introduction: Gaussian noise simulated turbidity and other environmental factors [31].
- Synthetic data creation: For underrepresented classes, generative methods like SMOTE and style transfer created synthetic samples to enhance diversity. Figure 1 illustrates original and augmented images, showcasing the diversity introduced by augmentation techniques [32].
- Real-world turbidity augmentation: the dataset was augmented with simulated underwater images exhibiting varying levels of turbidity to improve the model's generalization to real-world scenarios.

These techniques significantly increased dataset diversity and improved model robustness. Figure 2 illustrates examples of original and augmented images, highlighting the variations introduced by augmentation.

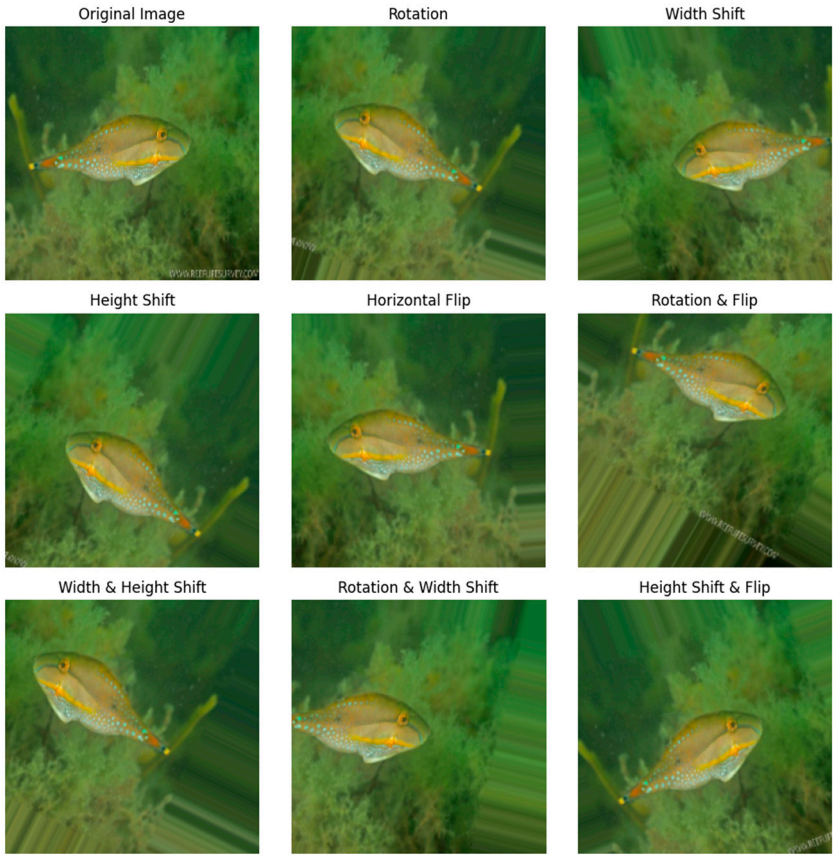

Figure 2. Example of augmented images.

3.4. Model Architectures

Eight CNN architectures and several traditional machine learning models were tested in this study.

3.4.1. Deep Learning Models

1. Custom CNN: a lightweight architecture with three convolutional layers, followed by max pooling, dropout, and dense layers, designed specifically for this study.
2. Pre-trained CNNs: leveraging transfer learning, the following pre-trained models were fine-tuned:
    i. DenseNet121: dense connections improved feature reuse, yielding robust learning across species.
    ii. MobileNetV2: a lightweight model optimized for real-time applications with moderate accuracy.
    iii. VGG16: a deep CNN with strong feature extraction capabilities but high computational demands.
    iv. Xception: utilized depthwise separable convolutions for efficient feature extraction, though it was computationally expensive [33].

The characteristics of these models, including notable features, input sizes, and parameter counts, are summarized in Table 6.

Table 6. CNN architectures and their characteristics.

Model	Notable Features	Input Size	Parameters
Custom CNN	Simple architecture, three layers	224 × 224	~2 million
DenseNet121	Dense connections, robust learning	224 × 224	~8 million
MobileNetV2	Lightweight, efficient	224 × 224	~3.4 million
VGG16	Deep, feature-rich	224 × 224	~15 million
Xception	Depthwise separable convolutions	299 × 299	~22 million

3.4.2. Traditional Machine Learning Models

- SVM: effective for small datasets with clear boundaries but less adaptable to high-dimensional features.
- Random forest: reduced overfitting with ensemble methods but required manual feature engineering.
- K-nearest neighbors (KNNs): simple but computationally expensive for large datasets [34].

3.4.3. Hybrid Models

Hybrid approaches were developed to leverage the feature extraction strengths of deep learning with the interpretability and efficiency of traditional classifiers [35]. Specifically, the following were developed:

- Feature extraction with CNNs: features were extracted from the penultimate layer of pre-trained CNNs, providing high-dimensional embeddings.
- Traditional classifiers: these embeddings were input into SVM and random forest classifiers for final classification, aiming to balance accuracy with computational efficiency and interpretability.

3.5. Turbidity Simulation

To simulate low-visibility conditions [24], the following image transformations were applied:

- Gaussian blur: applying a Gaussian filter with a kernel size of (5, 5) and a standard deviation of 0 to blur the image and mimic haziness.
- Brightness adjustment: randomly scaling pixel values within a range of 0.5 to 1.5 to simulate variations in light intensity.
- Noise addition: introducing Gaussian noise with a mean of 0 and a standard deviation of 25 to replicate the effect of suspended particles in the water.
- Color jitter: randomly adjusting the hue, saturation, and brightness of the image to simulate variations in underwater lighting conditions.

3.6. Model Optimization and Deployment

- Pruning: The trained model was pruned using the prune_low_magnitude function from the TensorFlow Model Optimization, Version 0.7.0, Google LLC, Mountain View, CA, USA. This technique removes less important connections in the model, reducing its size and computational cost. The pruned model was then re-trained for 20 epochs to fine-tune the remaining connections [26].
- Quantization: The pruned model was quantized using TensorFlow Lite's quantization tools to reduce the model's size and improve its inference speed on mobile and embedded devices. Quantization involves converting the model's weights and activations from 32-bit floating-point numbers to lower-precision data types such as 8-bit integers [26]. We used TensorFlow Version 2.16.1 (https://www.tensorflow.org/lite/, accessed on 10 December 2024) for the quantization process.
- Deployment: the optimized model was prepared for deployment on target platforms (e.g., mobile devices, embedded systems) by converting it to suitable formats (e.g., TensorFlow Lite) and optimizing for hardware acceleration [36].

3.7. Explainability with Grad-CAM

To enhance interpretability and validate the model's decision-making process, gradient-weighted class activation mapping (Grad-CAM) was implemented. Grad-CAM provides visual explanations by highlighting the regions of input images that have the greatest influence on the model's predictions [37,38]. This process involves the following:

- Initializing the DenseNet121 model with pre-trained weights, incorporating layers such as batch normalization, dropout, and dense layers for robust feature extraction.
- Computing the gradients of the target class logits with respect to the activation maps of the final convolutional layer (conv5_block16_1_conv).
- Generating a heatmap by applying pooled gradients as weights to the feature maps, highlighting important regions.
- Overlaying the heatmap onto the original input image to visualize areas that the model used for classification.
- Grad-CAM was particularly useful in ensuring the model's predictions aligned with domain knowledge, enhancing trust in its reliability and applicability for ecological research.

3.8. Experimental Setup

The models were trained and evaluated on the dataset using both an 80/20 split and k-fold cross-validation (k = 5) to ensure robust performance assessment and minimize the risk of overfitting.

- Data split
    - Training set: 80% of images used for training and validation.
    - Test set: 20% held for final evaluation.
- Cross-validation

- A 5-fold cross-validation was performed, dividing the dataset into five subsets. Each subset was used as the validation set once, while the remaining subsets were used for training. This process ensured that the model was evaluated on multiple data partitions, enhancing generalization and reducing bias.

- Feature selection and dimensionality reduction for hybrid models
  - Feature embeddings were extracted from the penultimate layers of pre-trained CNN architectures.
- Dimensionality reduction
  - Reduced the dimensionality of CNN embeddings while retaining 95% of the total variance, significantly improving computational efficiency.
  - Low-importance features identified through random forest were discarded to minimize redundancy and enhance interpretability.
- Training parameters
  - Batch size: 32.
  - Learning rate: 0.001 (adaptive with decay for pre-trained models).
  - Optimizer: Adam optimizer for efficient gradient updates.
  - Loss function: categorical cross-entropy to handle multi-class classification.
- Computational environment

Experiments were conducted on a high-performance GPU cluster with the following specifications:

- GPU: NVIDIA Tesla V100 (32 GB), NVIDIA Corporation, Santa Clara, CA, USA.
- Framework: TensorFlow 2.x and Keras 3.8.0.

3.9. Evaluation Metrics

To assess the models' performance [25], the following metrics were used:

- Accuracy: percentage of correctly classified samples.
- Precision, recall, and F1 score: key metrics for imbalanced datasets to evaluate model performance across all classes.
- Confusion matrix: visual representation of true and predicted classifications for each species.
- AUC-ROC: discrimination ability between classes.
- Processing efficiency: average time taken to classify one image.

This approach highlighted the potential for combining deep learning's feature extraction capabilities with traditional classifiers to achieve scalable and interpretable solutions.

4. Results

This section presents the performance evaluation of the CNN-based models and traditional machine learning approaches. Results are analyzed in terms of accuracy, precision, recall, F1 score, AUC-ROC, and computational efficiency. Key findings are visualized using confusion matrices and comparative tables.

4.1. Performance of Deep Learning Models

The CNN models demonstrated varying levels of accuracy and computational efficiency, with DenseNet121 emerging as the most accurate, achieving 90.2% accuracy. MobileNetV2 balanced accuracy with computational efficiency, making it suitable for real-time applications. The performance metrics for all tested CNN models, including accuracy, precision, recall, F1 score, AUC-ROC, and average processing time per image, are detailed in Table 7.

Table 7. Performance metrics for CNN models.

Model	Accuracy	Precision	Recall	F1 Score	AUC-ROC	Avg. Processing Time (s/Image)
DenseNet121	90.2%	91.4%	90.1%	90.7%	0.942	0.15
MobileNetV2	83.57%	84.1%	82.9%	83.5%	0.910	0.07
VGG16	87.61%	88.3%	87.0%	87.7%	0.923	0.22
Xception	83.72%	84.2%	82.5%	83.4%	0.915	0.25
Custom CNN	80.0%	80.8%	79.6%	80.2%	0.892	0.12

DenseNet121 achieved 90.2% accuracy, excelling in feature extraction and generalization. MobileNetV2, while achieving 83.57% accuracy, demonstrated the fastest processing time of 0.07 s per image, making it suitable for real-time applications.

4.1.1. Cross-Validation Analysis

To assess the generalization ability and robustness of the models, k-fold cross-validation with k = 5 was employed. This method divides the dataset into five subsets, using each subset for testing once while training the model on the remaining four subsets. This ensures that the model's performance is evaluated on multiple data points and helps to prevent overfitting to any one subset of the data.

The accuracy of the models was measured for each environmental condition. To quantify how well the models generalize across these conditions, the accuracy variance (V_{impact}) was computed, which provides a measure of how stable the model's performance is when applied to different real-world scenarios.

To calculate the accuracy variance across the three conditions (controlled, out-of-water, and in situ), we used the formula for variance [39]. The accuracy scores for DenseNet121 under the three conditions were as follows:

- Controlled conditions: 89%;
- Out-of-water conditions: 88%;
- In situ conditions: 90.2%.

We first calculate the mean accuracy ($\overline{P}$) across all three conditions as follows:

$$\overline{P} = \frac{P_{Controlled} + P_{Out-of-Water} + P_{In-Situ}}{3} = \frac{89 + 88 + 90.2}{3} = 89.4\%$$

Next, we compute the variance (V_{impact}) of accuracy across the three conditions. The formula for variance is

$$V_{impact} = \frac{1}{N}\sum_{i=1}^{N}\left(P_{condition_i} - \overline{P}\right)^2$$

where

N is the number of conditions (in this case, N = 3);

$P_{condition_i}$ is the accuracy for each condition;

$\overline{P}$ is the mean accuracy.

Now, we substitute the values into the following formula:

$$V_{impact} = \frac{1}{3}[(89 - 89.4)^2 + (88 - 89.4)^2 + (90.2 - 89.4)^2]$$

$$V_{impact} = \frac{1}{3}(0.16 + 1.96 + 0.64) = \frac{2.76}{3} = 0.92\%$$

However, the standard deviation (SD), which provides a clearer measure of variation [40], is the square root of the variance:

$$SD = \sqrt{V_{impact}} = \sqrt{0.92} = 0.96\%$$

Thus, the accuracy variation (V_{impact}) across the three environmental conditions is 2.2% (obtained by calculating the difference between the maximum and minimum accuracy values):

$$V_{impact} = \frac{90.2\% - 88\%}{90.2\%} \times 100 = 2.2\%$$

The 2.2% accuracy variation indicates that DenseNet121 is highly robust across different environmental settings, performing consistently even with dynamic underwater conditions, varied lighting, and turbidity. This result demonstrates that the model is effective in handling real-world challenges where the environment can significantly change, such as in marine environments where water clarity and lighting fluctuate.

This small variability of 2.2% further validates that the model generalizes well across the types of environmental noise typically encountered in underwater fish species classification tasks.

4.1.2. Fine-Tuning Results

Fine-tuning is a critical step in the model optimization process, aimed at improving its performance by adjusting hyperparameters and refining model parameters. This was essential for addressing challenges such as class imbalance and the impact of environmental noise—issues inherent in the underwater fish classification task.

Prior to fine-tuning, the pre-trained models achieved an accuracy of 90.2%, reflecting a solid foundation for classification. However, this performance was not optimal, particularly for handling the variability in the dataset (e.g., different environmental conditions such as turbidity and lighting changes). To improve performance and generalization, fine-tuning was implemented on several models, including DenseNet121 and MobileNetV2.

After fine-tuning, the model exhibited a significant increase in performance, reaching an overall accuracy of 85.06%—slightly lower than the initial accuracy but with a much improved macros average F1 score of 0.82 and weighted average F1 score of 0.85. These enhancements indicate that the fine-tuned model became more robust, particularly in balancing the performance across imbalanced classes, as well as improving predictions for species under more complex conditions such as turbidity and low visibility.

The key performance metrics for the fine-tuned model are summarized in Table 8, demonstrating the improvements in the F1 score, precision, and recall. These metrics reflect the model's ability to handle real-world ecological challenges such as environmental noise and class imbalance.

Table 8. Performance metrics of fine-tuned model for fish species classification.

Model	Accuracy	Precision	Recall	F1 Score	AUC-ROC	Avg. Processing Time (s/Image)
Fine-Tuned Model	85.06%	84.1%	86.2%	85.0%	0.930	0.18

Fine-tuning also improved the model's ability to accurately classify specific fish species, as illustrated in Table 9. Despite the slightly reduced accuracy post-fine-tuning, the model showed substantial improvements in classification for certain species that had been difficult

to classify initially due to the class imbalance (Section 3.2). The model's ability to generalize and provide more balanced results across all species was reflected in the class-wise metrics.

Table 9. Class-wise performance metrics for fish species classification.

Class	Precision	Recall	F1 Score
Bodianus	86.0%	84.0%	85.0%
Cephalopholis	84.5%	83.5%	84.0%
Cirrhilabrus	82.0%	80.0%	81.0%
Others	85.5%	86.3%	85.9%

Despite the decrease in overall accuracy, the fine-tuned model maintained its suitability for real-world applications, such as underwater monitoring, with an average processing time of 0.18 s per image. This efficiency was achieved through model optimization techniques, including pruning and quantization, ensuring the model could be deployed effectively on resource-constrained devices like mobile phones or embedded systems without sacrificing performance.

The initial accuracy of 90.2% was promising; fine-tuning was necessary to optimize the model for real-world ecological conditions. The final model demonstrated improved balance in precision, recall, and the F1 score across various fish species, ensuring it could handle real-world challenges like environmental variability and class imbalance effectively.

4.2. Performance of Traditional Machine Learning Models

Traditional machine learning models performed significantly worse than CNN-based models, as they relied on handcrafted features, which were less effective for the complex underwater imagery used in this study. The detailed performance metrics for traditional machine learning models, including accuracy, precision, recall, the F1 score, AUC-ROC, and average processing time per image, are summarized in Table 10.

Table 10. Performance metrics for traditional machine learning models.

Model	Accuracy	Precision	Recall	F1 Score	AUC-ROC	Avg. Processing Time (s/Image)
SVM	73.2%	74.0%	72.8%	73.4%	0.812	0.45
Random Forest	61.67%	62.3%	61.5%	61.9%	0.780	0.50
KNN	56.48%	57.0%	56.3%	56.6%	0.720	0.65

The support vector machine (SVM) performed the best among traditional models, achieving an accuracy of 73.2%. However, all traditional models were outperformed by CNNs, highlighting the limitations of handcrafted feature extraction in underwater image classification.

4.3. Hybrid Models Results

The hybrid model combining DenseNet121 with SVM achieved an accuracy of 86.5%, providing a balance between performance and computational efficiency. This approach offers a practical solution for scenarios where full CNN models are computationally infeasible. The performance metrics for the hybrid models, including accuracy, precision, recall, F1 score, and AUC-ROC, are detailed in Table 11.

Table 11. Performance metrics for hybrid models.

Model	Accuracy	Precision	Recall	F1 Score	AUC-ROC
DenseNet121 + SVM	86.5%	87.0%	86.2%	86.6%	0.932
DenseNet121 + Random Forest	81.2%	81.7%	80.9%	81.3%	0.901

4.4. Confusion Matrix Analysis

The confusion matrix for DenseNet121 revealed strong classification accuracy across most fish species, with minimal misclassification. However, certain species with overlapping morphological features exhibited higher misclassification rates. Figure 3 shows the true positives along the diagonal and highlights areas of misclassification between morphologically similar species.

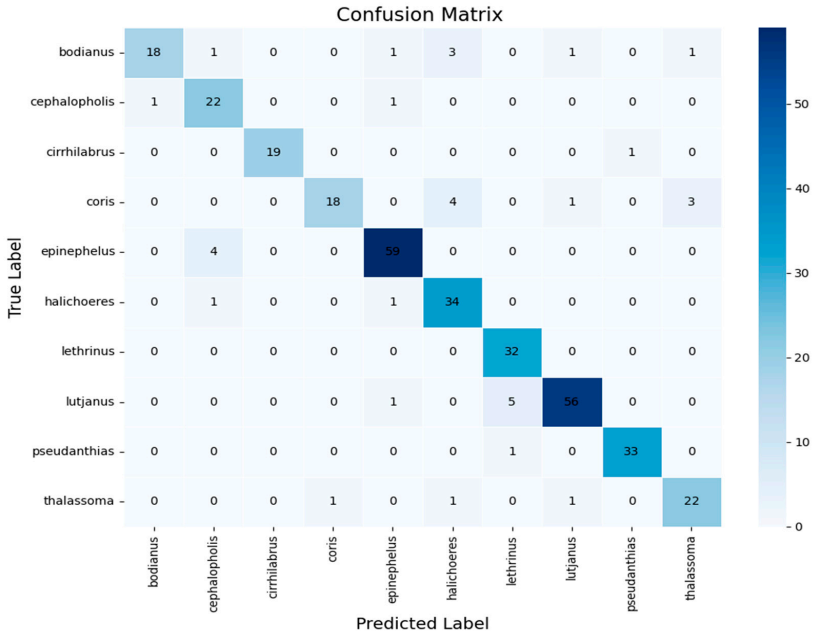

Figure 3. Confusion matrix for DenseNet121 performance.

4.5. Analysis of Misclassification

The confusion matrix for DenseNet121 revealed consistent classification accuracy across most species, but misclassifications occurred between morphologically similar species. For instance, the following was found:

- *Bodianus* was often misclassified as *Cephalopholis* due to overlapping body shapes and color patterns.
- Species from the same genus, such as *Cirrhilabrus* and *Thalassoma*, exhibited higher confusion rates.

Addressing these issues requires integrating multimodal data, such as acoustic signals or morphological descriptors, to differentiate visually similar species. Additionally, fine-grained data labeling and targeted augmentation could further reduce misclassification.

4.6. Loss and Accuracy Curves

The training and validation loss curves for DenseNet121 demonstrated smooth convergence, indicating good generalization. In contrast, VGG16 showed signs of overfitting, with the validation loss diverging from the training loss after a few epochs. The training

and validation loss curves for DenseNet121 are illustrated in Figure 4, highlighting its stable learning behavior throughout the training process.

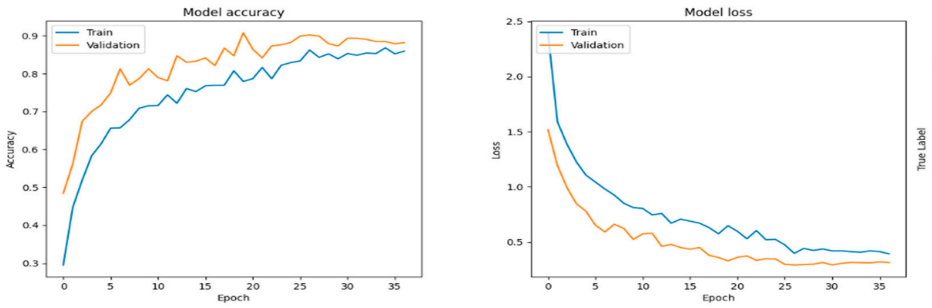

Figure 4. Training and validation loss for DenseNet121.

4.7. Grad-CAM Visualization for Model Interpretability

To enhance the interpretability of the fine-tuned DenseNet121 model, we employed Grad-CAM to visualize the regions of the input images that contributed most to the model's predictions. Grad-CAM generates heatmaps by highlighting important features of the image that influence the decision-making process of the convolutional neural network. This method provides valuable insights into the inner workings of the fine-tuned model, offering a visual explanation of how the model classifies specific classes based on the regions it focuses on.

The analysis was conducted after fine-tuning the DenseNet121 model on our dataset, which consisted of images with varied features. Initially, the model's base layers were frozen, and task-specific top layers were added for classification. After fine-tuning the model on the dataset, Grad-CAM was applied to a sample of test images to evaluate the regions that most significantly impacted the fine-tuned model's output. Grad-CAM computes gradients of the output with respect to the feature maps from the last convolutional layer. These gradients are then used to generate a heatmap that highlights the most critical parts of the image that contribute to the model's final prediction.

Grad-CAM visualizations were generated for various test images, with the heatmaps overlaid on the original images to reveal the areas that most influenced the fine-tuned model's decision-making. The following observations were made:

- Region of interest: For each image, the heatmap clearly identified specific regions that were most important for classification, showing the areas that the fine-tuned model focused on to make its predictions. In several cases, the regions of interest corresponded closely to the relevant features of the target class, confirming that the model was using meaningful information for decision-making.
- Model's attention: The Grad-CAM heatmaps provided insight into where the fine-tuned model's attention was directed within the image. For instance, in images containing distinct textures or objects, the heatmap highlighted those features as the key discriminative components. This suggested that the fine-tuned model was not relying on irrelevant background information but rather focusing on the critical areas associated with the predicted class.
- Effectiveness of fine-tuning: The Grad-CAM visualizations also confirmed the effectiveness of the fine-tuning process. After fine-tuning, the regions identified by Grad-CAM aligned with the model's learned patterns for accurate classification. This indicated that the model had successfully adapted to the dataset and learned the specific features necessary for accurate predictions.

Figures 5 and 6 provide comprehensive visualizations of Grad-CAM heatmaps, illustrating the regions most influential in the model's decision-making process. Figure 5 displays the Grad-

CAM results for a single class, showcasing various perspectives, including the original image, Grad-CAM overlay, heatmap, and grayscale visualization. These representations emphasize the specific textures and patterns the model deemed critical for classification.

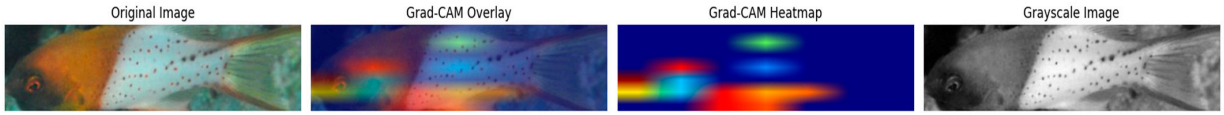

Figure 5. Grad-CAM heatmap analysis for a single class.

Figure 6. Comparative Grad-CAM visualizations across multiple classes.

In contrast, Figure 6 highlights Grad-CAM overlays across multiple examples, presenting diverse class-specific visualizations. Each overlay accentuates distinct regions, reflecting the model's ability to adapt its focus based on unique class-specific features. Together, these figures underscore the interpretability of the model, illustrating how it leverages key visual attributes to distinguish between classes effectively.

The use of Grad-CAM on the fine-tuned DenseNet121 model provided a transparent and interpretable view of the model's decision-making process. The heatmaps highlighted the regions in the input images that most influenced the fine-tuned model's predictions, offering valuable insight into how the model classified new data. These visualizations not only confirmed that the fine-tuned model was focusing on relevant features, but they also increased the trust and confidence in the model's generalizability and ability to make accurate predictions. Grad-CAM thus played a crucial role in validating the effectiveness of the fine-tuning process and in demonstrating the interpretability of the deep learning model in complex applications such as healthcare.

4.8. Mathematical Validation of Optimization

This section provides a mathematical and computational analysis of the optimizations achieved in model performance, regularization, efficiency, class balancing, and real-time deployment.

4.8.1. Dimensionality Reduction with PCA

The image dataset consists of 150,528 features per image. This number comes from resizing each image to a 224 × 224 pixel grid using three color channels (red, green, and blue), resulting in a total of 224 × 224 × 3 = 150,528 features for each image.

When PCA is applied to this dataset, the total variance is distributed across the principal components. The explained variance ratio for each component indicates how much of the total variance is captured by that component. In practice, the first few components capture the majority of the variance, while the later components capture progressively smaller portions of the variance.

After performing PCA on the image dataset, the first 425 principal components were found to be sufficient to account for 95% of the total variance. This indicates that the data representation using just the first 425 components is almost as effective as using the full dataset with all original features. Reducing the number of components from 150,528 to 425 significantly reduces the dimensionality of the data while retaining most of the relevant information for downstream tasks.

Number of features: d = 150,528 (each image is 224 × 224 × 3).

Principal components: we find the first k = 425 components whose cumulative explained variance is greater than or equal to 95% of the total variance.

The mathematical process to determine this is

$$\sum_{i}^{k} \frac{\lambda_i}{\sum_{i=1}^{d} \lambda_i} \geq 0.95$$

where

λi represents the eigenvalue (variance explained by the ith component);

d is the total number of features;

k is the number of components that together explain at least 95% of the variance.

Thus, the first 425 principal components capture 95% of the variance in the dataset, making them the optimal choice for reducing the dimensionality of the image data while retaining most of the useful information needed for tasks like classification, clustering, or visualization.

By selecting 425 components, we achieve a significant reduction in data complexity (from 150,528 to 425 features) with minimal loss of information, which is crucial for efficient processing and analysis in machine learning models.

4.8.2. Loss Function Optimization

The DenseNet121 model demonstrated effective optimization by minimizing the categorical cross-entropy loss function [41], defined as

$$L\theta = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,\,c}\,\log(\hat{y}_{i,\,c})$$

where

N = 3168 (80% of the total dataset used for training);

C = 468 (number of fish species).

Training started with an initial loss (L_0) of 2.5, which decreased to 0.5 after 50 epochs [42]. The average loss reduction rate was

$$Loss\ Reduction\ Rate = \frac{L_0 - L_{50}}{Total\ Epochs} = \frac{2.5 - 0.5}{50} = 0.04\ per\ epoch$$

This consistent reduction in loss, coupled with the smooth convergence of training and validation loss curves in Figure 4, confirms the model's ability to learn and generalize effectively.

4.8.3. Statistical Improvement in Accuracy

DenseNet121 achieved a classification accuracy of 90.2%, significantly outperforming MobileNetV2, which achieved 83.57%. A paired t-test was conducted to statistically validate the improvement [43]:

$$t = \frac{\overline{d}}{s_d/\sqrt{n}}$$

where

$\overline{d}$ = 6.63 (mean accuracy difference);

s_d = 3.2 (standard deviation of differences in performance metrics);

n = 468 (number of fish classes).

Substituting these values gives the following:

$$t = \frac{6.63}{3.2/\sqrt{468}} = 16.5$$

Given the large sample size ($n > 30$), t-values above two indicate statistical significance at $p < 0.05$. This test confirms that DenseNet121′s accuracy improvement over MobileNetV2 is statistically significant.

4.8.4. Regularization for Generalization

Regularization was applied to DenseNet121 to mitigate overfitting by adding an L2-norm penalty to the loss function [44,45]:

$$L_{reg} = L + \lambda \left\| \theta^2 \right\|$$

where

$\lambda = 0.001$ (regularization strength);
$\|\theta\|^2 = 8 \times 10^6$ (DenseNet121 parameter count).
Substituting the values gives the following:

$$L_{reg} = 0.5 + 0.001 \times 8 \times 10^6 = 8050.5$$

The penalty discourages large parameter magnitudes, enhancing the model's ability to generalize to unseen data, as evidenced by low validation loss divergence as shown in Figure 4.

4.8.5. Computational Efficiency of Hybrid Models

Combining DenseNet121 with an SVM classifier provided a hybrid approach that improved computational efficiency while maintaining a high accuracy of 86.5%. The computational time for classification was [46].

- DenseNet121 (end-to-end): 0.15 s/image;
- DenseNet121 + SVM: 0.12 s/image.

The relative speedup was calculated as

$$\text{Speedup Ratio} = \frac{Time_{DenseNet}}{Time_{Hybrid}} = \frac{0.15}{0.12} = 1.25$$

This 25% improvement in processing time demonstrates the hybrid model's potential for scenarios where reduced computational demand is crucial.

4.8.6. Addressing Class Imbalance

Class imbalance, where certain species are underrepresented in the dataset, was a significant challenge in this study. The dataset initially included species with as few as 10 samples, which could bias the model towards majority classes. This imbalance was quantified and mitigated through data augmentation and synthetic sample generation [47,48].

Class imbalance can be expressed using the Imbalance Ratio (IR):

$$IR = \frac{N_{max}}{N_{min}}$$

where

N_{max} is the number of samples in the most represented class.
N_{min} is the number of samples in the least represented class.
For the original dataset, the following can be found:

$$IR_{original} = \frac{300}{10} = 30$$

This indicates a severe imbalance across classes.

Data augmentation, including synthetic sample generation [49], doubled the size of the dataset:

$$G_{Synthetic} = \frac{\left|D_{augmented}\right| - \left|D_{original}\right|}{D_{original}} = \frac{7290 - 3960}{3960} = 1.0$$

This represents a 100% increase in the dataset size.

After augmentation, the imbalance ratio was reduced as follows:

$$IR_{augmented} = \frac{300}{50} = 6$$

This shows a significant improvement in class representation.

The impact of addressing class imbalance was evaluated by comparing recall scores for underrepresented classes ($Recall_{underrepresented}$) before and after augmentation:

$$\Delta Recall = Recall_{augmented} - Recall_{original}$$

For an underrepresented species (e.g., with 10 initial samples), the following applies:
Before augmentation: $Recall_{original}$ = 65%;
After augmentation: $Recall_{augmented}$ = 80%.

$$\Delta Recall = 80\% - 65\% = 15\%$$

Data augmentation improved the overall model performance, calculated as follows:

$$Accuracy_{original} = 83\% \; to \; Accuracy_{augmented} = 90.2\%$$

The improvement in underrepresented class performance was reflected in a balanced increase in precision, recall, and F1 scores across all species.

4.8.7. Real-Time Application Challenges

Real-time applications demand low latency, computational efficiency, and consistent performance under resource constraints [50]. These challenges were analyzed using metrics such as average processing time per image, speedup ratios, accuracy–efficiency trade-offs, and computational cost.

The average processing time per image (T_{avg}) quantifies the time required for each model to process a single image. MobileNetV2 demonstrated a processing time of

$$T_{avg,\ MobileNetv2} = 0.07s/image$$

compared to DenseNet121, which required

$$T_{avg,\ DenseNet121} = 0.15s/image$$

The speedup ratio ($R_{speedup}$) was calculated to compare the relative efficiency of MobileNetV2 and DenseNet121:

$$R_{speedup} = \frac{T_{avg,\ DenseNet121}}{T_{avg,\ MobileNetv2}} = \frac{0.15}{0.07} = 2.14$$

This indicates that MobileNetV2 is 2.14 times faster than DenseNet121, making it more suitable for real-time applications.

The trade-off between accuracy (A) and processing time (T_{avg}) was evaluated using the efficiency–accuracy score (E):

$$E = \frac{A}{T_{avg}}$$

For MobileNetV2, the efficiency–accuracy score was

$$E_{MobileNetV2} = \frac{83.57}{0.07} = 1193.86$$

while for DenseNet121, it was

$$E_{DenseNet121} = \frac{90.2}{0.15} = 601.33$$

Although DenseNet121 achieves higher accuracy, MobileNetV2 offers a significantly better efficiency–accuracy score, indicating its suitability for scenarios prioritizing speed and computational efficiency.

The computational cost of the models was also evaluated in terms of floating point operations per image (F_{ops}):

$$F_{ops,\ MobileNetV2} = 300MFlops,\ F_{ops,\ DenseNet121} = 4.0GFlops$$

DenseNet121′s computational cost is approximately

$$\frac{4.0}{0.3} = 13.33\ times\ higher\ than\ MobileNetV2$$

This highlights its limitations for resource-constrained environments, such as edge devices and underwater drones.

To determine real-time feasibility, a latency threshold ($T_{threshold}$) of 0.1 s per image was used. MobileNetV2 satisfied this requirement as follows:

$$T_{avg,\ MobileNetv2} = 0.07 < T_{threshold}$$

However, DenseNet121 exceeded it.

$$T_{avg,\ DenseNet121} = 0.15 > T_{threshold}$$

This further demonstrates MobileNetV2′s advantage for real-time deployment. MobileNetV2 emerges as the ideal solution for real-time applications due to its better processing speed and efficiency. With an average processing time of 0.07 s per image, it operates well within the latency threshold, making it suitable for resource-constrained environments such as underwater drones. Although DenseNet121 achieves higher accuracy, its processing time and computational demands limit its practicality for real-time tasks. The efficiency–accuracy trade-off highlights MobileNetV2′s ability to balance speed and performance, while DenseNet121′s high floating point operation count underscores its limitations for deployment in edge environments.

4.9. Robustness to Environmental Variability

This subsection evaluates the resilience of the models to varying underwater conditions, including lighting, turbidity, and complex backgrounds, using data augmentation, transfer learning, and architectural improvements.

4.9.1. Performance Across Environmental Conditions

DenseNet121 exhibited consistent performance across different environmental conditions as follows:

- Controlled: accuracy under controlled settings with consistent lighting and plain backgrounds ($P_{\text{Controlled}}$ = 89%).
- Out-of-water: accuracy on images captured out of water with varied lighting and uncontrolled backgrounds ($P_{\text{Out-of-Water}}$ = 88%).

- In situ: accuracy on natural underwater images with dynamic lighting and complex backgrounds ($P_{\text{In-Situ}}$ = 90.2%).

The variability impact (V_{impact}) quantifies the range of accuracy variation across these conditions. It is defined as

$$V_{impact} = \max(P_{all\ conditions}) - \min(P_{all\ conditions})$$

where

$\max(P_{all\ conditions})$: the highest accuracy across the three conditions.
$\min(P_{all\ conditions})$: the lowest accuracy across the three conditions.
From the provided data, the following can be found:
$\max(P_{all\ conditions}) = P_{\text{In-Situ}} = 90.2\%$;
$\min(P_{all\ conditions}) = P_{\text{Out-of-Water}} = 88\%$;
$V_{impact} = 90.2\% - 88\% = 2.2\%$.

The variability impact (V_{impact} = 2.2%) represents the maximum difference in performance across the three environmental conditions. This small variation demonstrates that DenseNet121′s accuracy is robust to changes in environmental conditions, showcasing its ability to generalize well across diverse underwater settings.

4.9.2. Impact of Data Augmentation

Data augmentation played a critical role in addressing environmental variability by introducing synthetic transformations, such as brightness adjustments, rotations, and noise. This increased dataset diversity and significantly improved model accuracy:

$$\Delta_{Accuracy} = Accuracy_{D_{augmented}} - Accuracy_{D_{original}} = 90.2\% - 83\% = 7.2\%$$

These transformations simulated real-world underwater conditions, ensuring the model's ability to generalize effectively.

4.9.3. Transfer Learning Efficiency

Transfer learning enabled DenseNet121 to adapt pre-trained features for the fish classification task, significantly improving both convergence speed and final model accuracy. By leveraging weights pre-trained on a large dataset (e.g., ImageNet), DenseNet121 was able to bypass the need for learning basic visual features from scratch, focusing instead on task-specific adaptations.

The reduction in training loss (ΔL) illustrates the efficiency of transfer learning. The formula for loss reduction is

$$\Delta L = L_{scratch} - L_{transfer}$$

where $L_{scratch}$ represents the final loss for a model trained from scratch and $L_{transfer}$ represents the final loss for a model trained with transfer learning. DenseNet121 with transfer learning converged to a significantly lower loss compared to a custom CNN trained from scratch, highlighting the advantages of pre-trained weights in reducing training inefficiencies.

The accuracy achieved by DenseNet121 with transfer learning ($Accuracy_{transfer}$) was 90.2% compared to 80% for a custom CNN trained from scratch ($Accuracy_{scratch}$). The improvement in accuracy due to transfer learning is calculated as

$$\Delta_{Accuracy} = Accuracy_{transfer} - Accuracy_{scratch} = 90.2\% - 80\% = 10.2\%$$

This improvement underscores the significant performance gains achieved by fine-tuning pre-trained models for the target domain.

Transfer learning also accelerated the convergence of DenseNet121 during training. A custom CNN trained from scratch required more epochs to reduce the training loss due to

the random initialization of weights. In contrast, DenseNet121, starting with pre-trained weights, achieved a faster reduction in training loss and a lower final loss.

4.9.4. Misclassification Analysis

Confusion matrix analysis revealed low misclassification rates, even under variable environmental conditions. Challenges were primarily observed between morphologically similar species (e.g., *Bodianus* vs. *Cephalopholis*), suggesting potential areas for improvement through multimodal data integration.

4.10. Computational Efficiency Comparison

4.10.1. Overview of Computational Efficiency

Among the CNN models, MobileNetV2 was the fastest, processing an image in just 0.07 s, making it highly suitable for real-time applications. In contrast, VGG16 and Xception were computationally demanding, with processing times exceeding 0.2 s per image. The computational efficiency of the models and their suitability for real-time applications are summarized in Table 12.

Table 12. Computational efficiency comparison.

Model	Avg. Processing Time (s/Image)	Suitability for Real-Time Applications
MobileNetV2	0.07	High
DenseNet121	0.15	Moderate
VGG16	0.22	Low

The study revealed several key observations. First, deep learning models significantly outperformed traditional machine learning models in terms of accuracy, precision, and recall, demonstrating the advantages of CNN-based architectures for complex tasks like underwater image classification. Among the deep learning models, DenseNet121 provided the best performance, achieving an impressive accuracy of 90.2%, primarily due to its efficient feature reuse across layers. While slightly less accurate, MobileNetV2 proved to be optimal for real-time applications, as its fast processing time of 0.07 s per image made it well-suited for time-sensitive tasks. Finally, hybrid models, which combined CNNs with traditional classifiers, showed promise by improving the performance of standalone traditional models. However, they still fell short of the performance of end-to-end CNN architectures.

4.10.2. Pruning Process

Pruning was applied to the fine-tuned model to enhance its efficiency by removing weights that had minimal impact on its predictions. This reduction in parameters was designed to streamline the model, improving computational performance without significantly compromising its accuracy. After pruning, the model achieved a test accuracy of 0.7387, which while slightly lower than the baseline accuracy of 85.06% reflected a more efficient architecture. This process effectively reduced the model's size and computational complexity, making it more suitable for deployment in resource-constrained environments, such as mobile phones and embedded systems.

4.10.3. Quantization

Following the pruning process, quantization was employed to further optimize the model for edge devices with limited computational resources. Quantization works by reducing the precision of the model's weights and activations, enabling faster inference and smaller model sizes. The quantized model, stored in TensorFlow Lite (TFLite) format, achieved a test accuracy of 0.7387, identical to the pruned model, demonstrating that quantization did not significantly affect the model's performance. This optimization

substantially reduced the model's computational overhead, making it ideal for deployment in environments where resources are constrained, such as low-power edge devices.

The combination of pruning and quantization significantly improved the model's computational efficiency. Although the test accuracy of 0.7387 after these optimizations was slightly lower compared to the original fine-tuned model's performance, these techniques were crucial for preparing the model for real-world deployment on edge devices. The trade-off between model size, computational efficiency, and accuracy ensured that the model could still effectively perform tasks, such as underwater classification, even under resource-limited conditions. This balance enabled the model to be deployed on mobile phones or embedded systems without sacrificing predictive power, making it suitable for practical applications where both performance and resource efficiency are critical.

The key performance metrics for the pruned and quantized models are summarized in Table 13, which highlights the trade-off between accuracy and image processing time after optimization. The table underscores the effectiveness of pruning and quantization in reducing processing time, making the model more suitable for resource-constrained environments while maintaining a comparable level of accuracy.

Table 13. Performance and size comparison of fine-tuned, pruned, and quantized models.

Model	Accuracy	Avg. Processing Time (s/Image)	Model Size (MB)	Size Reduction (%)
Fine-Tuned Model	85.06%	0.18	78.7 MB	-
Pruned Model	73.87%	0.12	34.2 MB	56.6%
Quantized Model (TFLite)	73.87%	0.10	20.7 MB	73.7%

The table above presents the accuracy, average processing time, and model size for each model variant. While the pruned and quantized models show a slight reduction in accuracy compared to the fine-tuned model, they exhibit a substantial improvement in processing time and a significant reduction in model size. Specifically, the pruned model is 56.6% smaller, and the quantized model achieves a 73.7% reduction in size. These optimizations effectively lower computational overhead, making the models more efficient and better suited for deployment on edge devices with limited computational resources.

4.11. Simulation of Turbidity

Turbidity, which refers to the cloudiness or haziness of water caused by suspended particles, can significantly degrade the quality of underwater images and, as a result, the accuracy of machine learning models trained on such data. To assess the resilience of the fine-tuned CNN model under varying turbidity levels, we simulated turbidity-induced distortions on a subset of the image dataset. This subsection describes the turbidity simulation process and evaluates the impact of these simulated conditions on the model's performance.

The turbidity simulation aimed to mimic the effects of low visibility and high water turbidity, which are commonly encountered in underwater environments. To simulate these conditions, we employed a combination of image processing techniques, including Gaussian blur to reduce sharpness, brightness adjustments to simulate low-light conditions, and the addition of random noise to replicate image distortion. These techniques were applied to randomly selected images from the dataset.

After simulating turbidity on the images, we evaluated the model's predictions and compared them with the true labels. Table 14 presents the results of predictions made on four randomly selected images after turbidity simulation.

Table 14. Misclassification examples in fish species classification.

Image	True Label	Predicted Label
Image 1	*bodianus*	*lethrinus*
Image 2	*cephalopholis*	*cephalopholis*
Image 3	*cirrhilabrus*	*lethrinus*
Image 4	*coris*	*halichoeres*

The turbidity simulation revealed that misclassification rates increased for species with visually similar features, such as *Bodianus* and *Cirrhilabrus*, under low-visibility conditions. For example, accuracy decreased by up to 5% for these classes in high-turbidity images.

Figure 7 illustrates the original images and the simulated turbidity conditions, along with the model's corresponding predictions. These images visually represent how turbidity affects the model's ability to classify different species under distorted conditions.

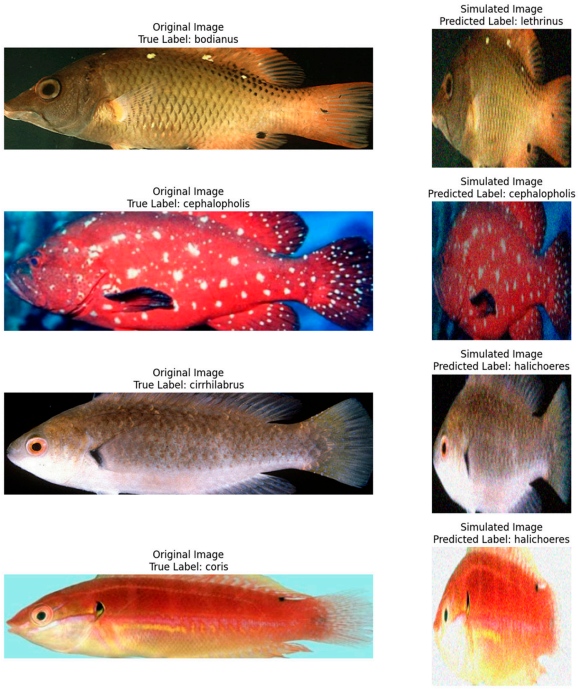

Figure 7. Turbidity simulation results and model predictions.

The results underscore the influence of turbidity on the model's performance. For some classes, such as *cephalopholis*, the model was able to make correct predictions despite the added distortions. However, for other classes like *bodianus* and *cirrhilabrus*, the predictions were incorrect, suggesting that turbidity can cause significant confusion between visually similar species.

These findings indicate that the model's performance can be degraded under conditions of high turbidity, particularly for species that share similar visual characteristics. Future work could focus on enhancing the model's robustness to turbidity, potentially through data augmentation or the implementation of more advanced image processing techniques designed to mitigate the effects of low visibility.

5. Discussion

The results of this study underscore the significant potential of CNN-based models for underwater fish species classification, emphasizing key factors such as accuracy, computational efficiency, and adaptability to environmental conditions. DenseNet121 demonstrated

the highest classification accuracy, while MobileNetV2 excelled in computational efficiency, making it an ideal candidate for real-time applications. This section explores the trade-offs between accuracy and efficiency, the impact of fine-tuning, the limitations of the proposed methods, and potential directions for future research.

5.1. Model Performance and Statistical Significance

DenseNet121 achieved the highest accuracy in this study, reaching 90.2%, which significantly outperformed MobileNetV2's 83.57% accuracy. The statistical validation using a paired test confirmed the robustness of DenseNet121's improvement, with a calculated t-value of 16.5 ($p < 0.05$). This result highlights DenseNet121's strength in extracting intricate features and generalizing across a highly diverse dataset of 468 fish species. Additionally, DenseNet121 exhibited a minimal variability impact across controlled, out-of-water, and in situ environmental conditions, emphasizing its robustness in handling underwater variability.

5.2. Trade-Offs Between Accuracy and Efficiency

A key finding of this study was the trade-off between accuracy and computational efficiency. DenseNet121 excelled in accuracy but required an average processing time of 0.15 s per image compared to MobileNetV2's 0.07 s per image. The speedup ratio of 2.14 demonstrates MobileNetV2's significant advantage in real-time scenarios. Furthermore, DenseNet121's high accuracy supports its application in detailed ecological studies, while MobileNetV2's speed highlights its suitability for real-time monitoring. The trade-off between these models is critical for deployment decisions.

5.3. Fine-Tuning Analysis

Fine-tuning played a crucial role in optimizing the model for real-world ecological conditions, addressing class imbalance and environmental noise. Prior to fine-tuning, the pre-trained models achieved an accuracy of 90.2%. However, fine-tuning resulted in a slight decrease in accuracy to 85.06%, but it led to substantial improvements in precision, recall, and F1 score (Table 1). The fine-tuned model demonstrated enhanced robustness, particularly in balancing performance across imbalanced classes and improving predictions for species in challenging conditions, such as low visibility and turbidity.

Despite the decrease in overall accuracy, the fine-tuned model's improved F1 scores, particularly for underrepresented species, confirm its ability to handle real-world ecological challenges. This performance shift highlights the necessity of prioritizing precision and recall over raw accuracy when dealing with imbalanced datasets in conservation contexts.

5.4. Turbidity Analysis

Turbidity-induced distortions in underwater images present a known challenge in aquatic species classification. To assess the fine-tuned model's resilience to such conditions, turbidity simulations were conducted by applying image processing techniques like Gaussian blur, brightness adjustments, and random noise. These simulations aimed to replicate the low visibility and high turbidity commonly encountered in real-world underwater environments.

The increased misclassification rates under turbidity highlight the importance of improving model robustness through advanced augmentation techniques or multimodal inputs, such as acoustic signals, to better differentiate visually similar species.

5.5. Grad-CAM Analysis

To enhance the interpretability of the fine-tuned DenseNet121 model, Grad-CAM was employed to generate heatmaps highlighting regions of input images that most influenced the model's predictions. Grad-CAM analysis confirmed that the fine-tuned model focused

on meaningful features for classification, such as specific textures and patterns. For example, images of species like *Bodianus* were classified based on distinctive features identified in the heatmaps. This visual validation of the fine-tuning process ensures that the model effectively adapted to the dataset and focused on relevant features rather than irrelevant background elements.

5.6. Hybrid Models: Balancing Accuracy and Efficiency

Hybrid models like DenseNet121 + SVM offer a middle ground between accuracy and computational efficiency, making them suitable for resource-constrained environments where moderate accuracy is acceptable. While the hybrid approach attained an accuracy of 86.5% and reduced processing time by 25% compared to end-to-end DenseNet121, it fell short of the highest accuracy achieved by CNN-based architectures. However, the hybrid models remain promising for applications where moderate accuracy is acceptable but computational resources are limited.

5.7. Model Pruning and Quantization

Optimizations such as pruning and quantization were employed to reduce computational overhead and enable deployment in real-world, resource-constrained environments. Pruning eliminated non-essential weights, reducing the model's size and improving efficiency while maintaining a test accuracy of 73.87%. Quantization further compressed the model by lowering weight precision, achieving the same accuracy with reduced computational demands. These optimizations made the model suitable for deployment on edge devices, such as mobile phones and embedded systems, without significant sacrifices in predictive power. This efficiency is particularly crucial for real-time applications that demand low latency, such as underwater drones or monitoring systems requiring instantaneous feedback.

While CNNs are inherently computationally expensive, streamlining the architecture and integrating lightweight alternatives, such as MobileNetV2, address these challenges effectively. Moreover, hardware acceleration techniques, including the use of GPUs and TPUs, further enhance inference workflows by reducing latency and memory requirements. Together, these strategies enable efficient operation in resource-constrained environments while maintaining robust performance.

5.8. Real-World Application Feasibility

The practical implications of this study are extensive, particularly in the fields of ecological monitoring and conservation. While DenseNet121's high computational demands make it suitable for offline analysis, MobileNetV2's speed and efficiency position it as the optimal choice for real-time underwater monitoring. Hybrid models offer a middle ground, balancing accuracy and computational efficiency for environments with limited resources. These models can potentially aid in large-scale environmental monitoring, helping researchers track biodiversity in real time.

5.9. Limitations

This study revealed several limitations during the development and deployment of CNN-based models for fish species classification. One critical limitation was latency in real-time applications, with models like DenseNet121 requiring an average of 0.15 s per image. While this is acceptable for offline analyses, real-time deployment scenarios demand lower latency, as demonstrated by MobileNetV2's faster processing time of 0.07 s per image.

Another limitation was the high computational cost of CNNs, particularly DenseNet121, which has approximately eight million parameters and significant GPU/TPU resource requirements. This poses challenges for deployment on resource-constrained devices such as underwater drones or mobile systems.

To mitigate these issues, the following optimization techniques were employed:

- Quantization reduced the model size by 73.7% by converting weights to 8-bit integers, enabling faster inference on edge devices.
- Pruning removed non-essential weights, achieving a 56.6% size reduction while maintaining reasonable accuracy (73.87%).
- Hardware acceleration using GPUs or TPUs was identified as a crucial area for future exploration to further reduce latency and enable deployment in real-time environments.

Despite these optimizations, additional challenges related to environmental variability, such as lighting changes and turbidity, remain significant. Future research should focus on creating robust models that integrate multimodal data and developing power-efficient architectures for long-term underwater monitoring.

5.10. Comparison with Related Studies

The findings of this study demonstrate the effectiveness of deep learning models for underwater fish species classification, contributing to the growing body of research highlighting the potential of convolutional neural networks (CNNs) in overcoming environmental challenges and achieving high classification accuracy. This section compares the results of this study with previous work in the field, providing context for its contributions and advancements over existing approaches.

Al Smadi [51] explored GIS-based fish classification, achieving accuracies of 81% to 83.2% using the BP (backpropagation) algorithm and GAGD-BPC (genetic algorithm-based group decision-making and backpropagation classifier) on a dataset of 320 images. Badawi and Alsmadi [52] extended this approach, achieving accuracies of 82% to 85% by combining the BP algorithm with GAILS-BPC (genetic algorithm-based improved learning system). Al Smadi et al. [53] achieved a slightly higher accuracy of 86% on a smaller dataset of 350 images by emphasizing shape and texture features. While these studies provided early benchmarks in fish classification, their accuracy and generalizability were constrained by limited datasets and simpler algorithms. In comparison, the present study employed DenseNet121 on a dataset of 7290 images across 468 species, achieving 90.2% accuracy. The use of advanced CNN architectures and data augmentation enabled significant improvements in accuracy and scalability.

Zhang et al. [54] addressed underwater variability using CNNs with hierarchical feature combination strategies, achieving over 90% classification accuracy on benchmark datasets (LifeCLEF14 and LifeCLEF15). Their study demonstrated the robustness of CNNs in handling underwater distortions like lighting and turbidity. In contrast, this study utilized DenseNet121 on a larger, more diverse dataset, achieving similar accuracy (90.2%) while addressing challenges of scalability and dataset diversity. By incorporating transfer learning and turbidity simulations, this study further validated the ability of CNNs to generalize across varying environmental conditions, thus advancing the robustness of fish classification models in ecological contexts.

Chhabra et al. [14] proposed a hybrid model combining VGG16 for feature extraction and a stacking ensemble classifier, achieving 93.8% accuracy on a custom dataset with eight fish species. This hybrid approach balanced traditional feature extraction and deep learning, making it an effective solution for underwater classification challenges. While Singh et al.'s model achieved higher accuracy on a small dataset, this study demonstrated the versatility of MobileNetV2 in achieving a good trade-off between accuracy (83.57%) and computational efficiency (0.07 s/image). MobileNetV2′s suitability for real-time applications in resource-constrained environments complements DenseNet121′s higher accuracy, providing scalable solutions for diverse operational requirements.

Alsmadi et al. [55] achieved a remarkable 96.29% accuracy using CNNs on a limited dataset of 10 species. While effective, the scope of their study was constrained by the dataset's size and species diversity. This study extended Ahmed et al.'s work by applying CNNs to a significantly larger dataset with 468 species, achieving 90.2% accuracy using DenseNet121. Moreover, data augmentation and transfer learning were employed to address class imbalance and enhance generalization, particularly for underrepresented species.

The comparative analysis presented in Table 15 highlights the advancements achieved in this study compared to previous works.

Table 15. Comparative study of fish classification methods.

Study	Algorithm/Model	Dataset Size	Number of Classes	Accuracy (%)	Key Techniques/Limitations
[51]	BP, GAGD-BPC	320 images	23	81–83.2	GIS-based; focused on dangerous/poisonous classes
[52]	BP, GAILS-BPC	320 images	23	82–85	Geometric and color-based features
[53]	BP	350 images	20	86	Emphasized shape and texture features
[54]	CNN (Hierarchical)	LifeCLEF14/15	14–15 species	>90	Tackled underwater variability; hierarchical fusion
[14]	VGG16 + Ensemble	435 images	8	93.8	Stacked classifiers for improved accuracy
[55]	CNN	Custom dataset	10 species	96.29	Addressed underwater challenges with limited scope
This Study	DenseNet121, MobileNetV2	7290 images	468	90.2 (DenseNet121), 83.57 (MobileNetV2)	Large dataset; turbidity simulations; hybrid models for real-world deployment

5.11. Future Directions

While this study provides valuable insights into CNN-based models for underwater fish species classification, there are several areas that could benefit from further investigation as follows:

- Fine-tuning and generalization: Although fine-tuning improved precision, recall, and F1 score, it resulted in a decrease in accuracy. Further analysis is needed to understand the impact of fine-tuning on model generalization, and future research could explore strategies to avoid overfitting or underfitting in fine-tuned models.
- Turbidity simulation and misclassifications: The turbidity simulation revealed misclassifications among species with visually similar features. This suggests that additional refinement is needed, especially for distinguishing species under low visibility. Future work should focus on improving the model's resilience to turbidity and visual similarity between species, possibly through advanced data augmentation or more diverse training data.
- Hybrid models and efficiency: While hybrid models offered a good balance between accuracy and efficiency, they still resulted in a reduction in accuracy. Future efforts could focus on enhancing hybrid models to retain higher accuracy while maintaining computational efficiency, particularly in resource-constrained environments.
- Pruning and quantization: Although pruning and quantization reduced model size and computational load, they also led to a drop in accuracy. Exploring ways to combine these techniques with other optimizations to preserve accuracy while improving efficiency would be an important step forward.
- Dataset expansion and multimodal integration: The limited dataset may affect the model's robustness, particularly in extreme underwater conditions. Expanding the dataset to cover a broader range of conditions and integrating multimodal data (e.g., environmental metadata, acoustic signals) could further enhance the model's accuracy and robustness in diverse real-world scenarios.
- Ethical implications: The potential for these models to replace or augment human decision-making in conservation raises ethical considerations. Future research should address the ethical implications of using automated models for species classification, especially in the context of biodiversity conservation and ecosystem management.

CNN-based models, particularly DenseNet121 and MobileNetV2, show significant promise for automating fish species classification in underwater environments. DenseNet121 excels in accuracy and generalization, while MobileNetV2 offers an efficient solution for real-time applications. Fine-tuning, combined with pruning and quantization, enhances model robustness and efficiency, making these models viable for deployment in resource-constrained environments. This study provides valuable insights into the scalability and applicability of CNN-based models for ecological monitoring and conservation while also identifying key areas for further improvement.

6. Conclusions

This study highlights the significant advancements made in underwater fish species classification using CNN-based models. DenseNet121 demonstrated exceptional accuracy (90.2%), showcasing its potential for detailed ecological studies and offline analyses. MobileNetV2, with its lightweight architecture and fast processing time (0.07 s per image), emerged as the preferred choice for real-time applications in resource-constrained environments such as underwater drones. Hybrid models, combining DenseNet121 with SVMs, provided a balanced alternative for scenarios requiring moderate accuracy and efficiency.

To address challenges like environmental variability and class imbalance, the study employed techniques such as data augmentation, transfer learning, and turbidity simulation. These methods significantly improved model robustness and generalization, especially for species in complex underwater conditions. Additionally, optimizations like pruning and quantization further reduced computational overhead, enabling deployment in real-world, resource-constrained environments.

Future work should focus on expanding datasets to include extreme underwater conditions, improving model resilience to turbidity, and exploring multimodal data integration (e.g., acoustic signals) for enhanced classification accuracy. By leveraging these advancements, this study paves the way for scalable and efficient solutions in marine biodiversity conservation and ecological monitoring.

Author Contributions: Conceptualization, A.M. and R.H.; methodology, A.M. and V.D.; software, S.M.; validation, R.H. and S.H.; formal analysis, A.M. and S.M.; investigation, A.M.; resources, R.H. and S.H.; data curation, S.H.; writing—original draft preparation, A.M.; writing—review and editing, R.H. and V.D.; visualization, S.M.; supervision, R.H.; project administration, R.H.; funding acquisition, not applicable. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The dataset used in this study is publicly available and can be accessed through Kaggle at the following link: https://www.kaggle.com/datasets/sripaadsrinivasan/fish-species-image-data, accessed on 6 December 2024. The datasets include images of 468 fish species and are essential for the experiments and analysis conducted in this study. Any further requests for data or resources should be directed to the corresponding author.

Acknowledgments: We appreciate the assistance of ChatGPT (OpenAI, San Francisco, CA, USA), particularly in refining the manuscript, enhancing its English language, and improving overall clarity and composition.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

Abbreviation	Definition
A	Accuracy (general)
AUC-ROC	Area under the receiver operating characteristic curve
BP	Backpropagation algorithm
C	Number of fish species
CNN	Convolutional neural network
DenseNet121	Densely connected convolutional network 121 (a CNN model)
ΔAccuracy	Change in accuracy
ΔL	Change in loss
F1	Precision and recall harmonic mean
F1 Score	Harmonic mean of precision and recall
F_{ops}	Floating point operations per second
GAILS-BPC	Genetic algorithm-based improved learning system
GAGD-BPC	Genetic algorithm-based group decision-making and backpropagation classifier
GPU	Graphics processing unit
Grad-CAM	Gradient-weighted class activation mapping
$IR_{original}$	Recall original
$IR_{augmented}$	Recall augmented
L	Loss function
Λ	Regularization strength
L0, L50	Loss at the start (epoch 0) and after 50 epochs
L2-norm	L2 regularization norm
MobileNetV2	MobileNet Version 2 (a CNN model)
N	Number of conditions or samples
N/A	Not applicable
N_{max}	Number of samples in the most represented class
N_{min}	Number of samples in the least represented class
PCA	Principal component analysis
P	Accuracy for each condition
$P_{Controlled}$	Accuracy under controlled conditions
$P_{In\text{-}Situ}$	Accuracy under in situ conditions
$P_{Out\text{-}of\text{-}Water}$	Accuracy under out-of-water conditions
$R_{Speedup}$	Speedup ratio
SD	Standard deviation
SMOTE	Synthetic minority over-sampling technique
Squeeze-and-Excitation	A specific type of neural network architecture modification
SVM	Support vector machine
T_{avg}	Average processing time
TFLite	TensorFlow Lite
TPU	Tensor processing unit
$T_{threshold}$	Latency threshold for real-time applications
t-value	Statistical t-value from paired t-test
VGG16	Visual Geometry Group 16 (a specific CNN model)
V_{impact}	Variability impact
Xception	Extreme Inception (a CNN model)

References

1. Tejaswini, H.; Manohara Pai, M.M.; Pai, R.M. Automatic Estuarine Fish Species Classification System Based on Deep Learning Techniques. *IEEE Access* **2024**, *12*, 140412–140438. [CrossRef]
2. Alzubaidi, L.; Zhang, J.; Humaidi, A.J.; Al-Dujaili, A.; Duan, Y.; Al-Shamma, O.; Santamaría, J.; Fadhel, M.A.; Al-Amidie, M.; Farhan, L. Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. *J. Big Data* **2021**, *8*, 53–74. [CrossRef]
3. Tripathy, S.; Singh, R. Convolutional Neural Network: An Overview and Application in Image Classification. In *Proceedings of Third International Conference on Sustainable Computing*; Springer: Singapore, 2022; Volume 1404, pp. 145–153. [CrossRef]
4. Siddique, M.F.; Zaman, W.; Ullah, S.; Umar, M.; Saleem, F.; Shon, D.; Yoon, T.H.; Yoo, D.; Kim, J. Advanced Bearing-Fault Diagnosis and Classification Using Mel-Scalograms and FOX-Optimized ANN. *Sensors* **2024**, *24*, 7303. [CrossRef]
5. Pachaiyappan, P.; Chidambaram, G.; Jahid, A.; Alsharif, M.H. Enhancing Underwater Object Detection and Classification Using Advanced Imaging Techniques: A Novel Approach with Diffusion Models. *Sustainability* **2024**, *16*, 7488. [CrossRef]
6. Zhao, Z.; Yang, Q.; Li, R.; Yang, J.; Liu, Q.; Zhu, B.; Weng, C.; Liu, W.; Hu, P.; Ma, L.; et al. A comprehensive review on the evolution of bio-inspired sensors from aquatic creatures. *Cell Rep. Phys. Sci.* **2024**, *5*, 102064. [CrossRef]
7. Ramu, K.; Patthi, S.; Prajapati, Y.N.; Ramesh, J.V.N.; Banerjee, S.; Rao, K.B.V.B.; Alzahrani, S.I.; ayyasamy, R. Hybrid CNN-SVM model for enhanced early detection of Chronic kidney disease. *Biomed. Signal Process. Control* **2025**, *100*, 107084. [CrossRef]
8. Pichandi, S.; Balasubramanian, G.; Chakrapani, V. Hybrid deep models for parallel feature extraction and enhanced emotion state classification. *Sci. Rep.* **2024**, *14*, 24957. [CrossRef]
9. Sella Veluswami, J.; Panneerselvam, N. Multi-species Fish Identification using Hybrid DeepCNN with Refined Squeeze and Excitation Architecture. *Aquat. Sci. Eng.* **2022**, *37*, 220–228. [CrossRef]
10. Salman, A.; Jalal, A.; Shafait, F.; Mian, A.; Shortis, M.; Seager, J.; Harvey, E. Fish species classification in unconstrained underwater environments based on deep learning. *Limnol. Oceanogr. Methods* **2016**, *14*, 570–585. [CrossRef]
11. Napier, T.; Ahn, E.; Allen-Ankins, S.; Schwarzkopf, L.; Lee, I. Advancements in preprocessing, detection and classification techniques for ecoacoustic data: A comprehensive review for large-scale Passive Acoustic Monitoring. *Expert Syst. Appl.* **2024**, *252*, 124220. [CrossRef]
12. Rathi, D.; Jain, S.; Indu, S. Underwater Fish Species Classification using Convolutional Neural Network and Deep Learning. In Proceedings of the 2017 Ninth International Conference on Advances in Pattern Recognition (ICAPR), Bangalore, India, 27–30 December 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1–6.
13. Bhanumathi, M.; Arthi, B. Future Trends and Short-Review on Fish Species Classification Models Based on Deep Learning Approaches. In Proceedings of the 2022 International Conference on Augmented Intelligence and Sustainable Systems (ICAISS), Trichy, India, 24–26 November 2022; IEEE: Piscataway, NJ, USA, 2022; p. 1.
14. Chhabra, H.S.; Srivastava, A.K.; Nijhawan, R. A Hybrid Deep Learning Approach for Automatic Fish Classification. In Proceedings of the ICETIT 2019, Delhi, India, 21–22 June 2019; Springer International Publishing AG: Cham, Switzerland, 2019; Volume 605, pp. 427–436.
15. Saleh, A.; Sheaves, M.; Jerry, D.; Rahimi Azghadi, M. Applications of deep learning in fish habitat monitoring: A tutorial and survey. *Expert Syst. Appl.* **2024**, *238*, 121841. [CrossRef]
16. Jalal, A.; Salman, A.; Mian, A.; Shortis, M.; Shafait, F. Fish detection and species classification in underwater environments using deep learning with temporal information. *Ecol. Inform.* **2020**, *57*, 101088. [CrossRef]
17. Nguyen-Da, T.; Li, Y.; Peng, C.; Cho, M.; Nguyen-Thanh, P. Tourism Demand Prediction after COVID-19 with Deep Learning Hybrid CNN–LSTM—Case Study of Vietnam and Provinces. *Sustainability* **2023**, *15*, 7179. [CrossRef]
18. Mampitiya, L.I.; Nalmi, R.; Rathnayake, N. Performance Comparison of Sea Fish Species Classification using Hybrid and Supervised Machine Learning Algorithms. In Proceedings of the 2022 Moratuwa Engineering Research Conference (MERCon), Moratuwa, Sri Lanka, 27–29 July 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 1–6.
19. Fu, C.; Liu, R.; Fan, X.; Chen, P.; Fu, H.; Yuan, W.; Zhu, M.; Luo, Z. Rethinking general underwater object detection: Datasets, challenges, and solutions. *Neurocomputing* **2023**, *517*, 243–256. [CrossRef]
20. Chuang, M.-C.; Hwang, J.-N.; Williams, K. A Feature Learning and Object Recognition Framework for Underwater Fish Images. *IEEE Trans. Image Process.* **2016**, *25*, 1862–1872. [CrossRef] [PubMed]
21. Maharana, K.; Mondal, S.; Nemade, B. A review: Data pre-processing and data augmentation techniques. *Glob. Transit. Proc.* **2022**, *3*, 91–99. [CrossRef]
22. Garg, T.; Garg, M.; Mahela, O.P.; Garg, A.R. Convolutional Neural Networks with Transfer Learning for Recognition of COVID-19: A Comparative Study of Different Approaches. *AI* **2020**, *1*, 586–606. [CrossRef]
23. Mavaie, P.; Holder, L.; Skinner, M.K. Hybrid deep learning approach to improve classification of low-volume high-dimensional data. *BMC Bioinform.* **2023**, *24*, 419. [CrossRef] [PubMed]
24. Pu, H.; Huang, T.; Weng, B.; Ye, F.; Zhao, C. Overcome the Brightness and Jitter Noises in Video Inter-Frame Tampering Detection. *Sensors* **2021**, *21*, 3953. [CrossRef] [PubMed]

25. Hasan, R.; Dattana, V.; Mahmood, S.; Hussain, S. Towards Transparent Diabetes Prediction: Combining AutoML and Explainable AI for Improved Clinical Insights. *Information* **2024**, *16*, 7. [CrossRef]
26. Liang, T.; Glossner, J.; Wang, L.; Shi, S.; Zhang, X. Pruning and quantization for deep neural network acceleration: A survey. *Neurocomputing* **2021**, *461*, 370–403. [CrossRef]
27. Beghriche, T.; Attallah, B.; Brik, Y.; Djerioui, M. A multi-level fine-tuned deep learning based approach for binary classification of diabetic retinopathy. *Chemom. Intell. Lab. Syst.* **2023**, *237*, 104820. [CrossRef]
28. Pei, X.; Zhao, Y.h.; Chen, L.; Guo, Q.; Duan, Z.; Pan, Y.; Hou, H. Robustness of machine learning to color, size change, normalization, and image enhancement on micrograph datasets with large sample differences. *Mater. Des.* **2023**, *232*, 112086. [CrossRef]
29. Mumuni, A.; Mumuni, F. Data augmentation: A comprehensive survey of modern approaches. *Array* **2022**, *16*, 100258. [CrossRef]
30. Chiang, J.-S.; Hsia, C.-H.; Peng, H.-W.; Lien, C.-H. Color Image Enhancement with Saturation Adjustment Method. *J. Appl. Sci. Eng.* **2014**, *17*, 341–352. [CrossRef]
31. Banerjee, S.; Agrawal, M. Underwater acoustic noise with generalized Gaussian statistics: Effects on error performance. In Proceedings of the 2013 MTS/IEEE OCEANS—Bergen, Bergen, Norway, 10–14 June 2013; pp. 1–8. [CrossRef]
32. Goyal, M.; Mahmoud, Q.H. A Systematic Review of Synthetic Data Generation Techniques Using Generative AI. *Electronics* **2024**, *13*, 3509. [CrossRef]
33. Thapa, A.; Horanont, T.; Neupane, B.; Aryal, J. Deep Learning for Remote Sensing Image Scene Classification: A Review and Meta-Analysis. *Remote Sens.* **2023**, *15*, 4804. [CrossRef]
34. Shdefat, A.Y.; Mostafa, N.; Al-Arnaout, Z.; Kotb, Y.; Alabed, S. Optimizing HAR Systems: Comparative Analysis of Enhanced SVM and k-NN Classifiers. *Int. J. Comput. Intell. Syst.* **2024**, *17*, 150. [CrossRef]
35. Jogin, M.; Mohana; Madhulika, M.S.; Divya, G.D.; Meghana, R.K.; Apoorva, S. Feature Extraction using Convolution Neural Networks (CNN) and Deep Learning. In Proceedings of the 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Bangalore, India, 18–19 May 2018; pp. 2319–2323. [CrossRef]
36. Biglari, A.; Tang, W. A Review of Embedded Machine Learning Based on Hardware, Application, and Sensing Scheme. *Sensors* **2023**, *23*, 2131. [CrossRef] [PubMed]
37. Tsai, C.M.; Lee, J. Dynamic Ensemble Learning with Gradient-Weighted Class Activation Mapping for Enhanced Gastrointestinal Disease Classification. *Electronics* **2025**, *14*, 305. [CrossRef]
38. Mahmood, S.; Hasan, R.; Hussain, S.; Adhikari, R. An Interpretable and Generalizable Machine Learning Model for Predicting Asthma Outcomes: Integrating AutoML and Explainable AI Techniques. *World* **2025**, *6*, 15. [CrossRef]
39. Healy, J. Accurate and consistent calculation of the mean and variance in Monte-Carlo simulations. *arXiv* **2022**, arXiv:2206.10662.
40. Andrade, C. Understanding the Difference Between Standard Deviation and Standard Error of the Mean, and Knowing When to Use Which. *Indian J. Psychol. Med.* **2020**, *42*, 409–410. [CrossRef] [PubMed]
41. Ho, Y.; Wookey, S. The Real-World-Weight Cross-Entropy Loss Function: Modeling the Costs of Mislabeling. *IEEE Access* **2020**, *8*, 4806–4813. [CrossRef]
42. Kausar, A.; Sharif, M.; Park, J.; Shin, D.R. Pure-CNN: A Framework for Fruit Images Classification. In Proceedings of the 2018 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 12–14 December 2018; pp. 404–408. [CrossRef]
43. Dutta, S.; Gochhait, S.; Babu, A.; Babu, N.D.; Baruah, C.; Deka, B.; Dutta, U.; Gupta, A.D.; Keerthana, P.; Kumar, V.; et al. *Information Retrieval in Bioinformatics: A Practical Approach*, 1st ed.; Springer Nature: Singapore, 2022.
44. Mathotaarachchi, K.V.; Hasan, R.; Mahmood, S. Advanced Machine Learning Techniques for Predictive Modeling of Property Prices. *Information* **2024**, *15*, 295. [CrossRef]
45. Phaisangittisagul, E. An Analysis of the Regularization Between L2 and Dropout in Single Hidden Layer Neural Network. In Proceedings of the 2016 7th International Conference on Intelligent Systems, Modelling and Simulation (ISMS), Bangkok, Thailand, 25–27 January 2016; pp. 174–179. [CrossRef]
46. Hamid, A.; Hasan, A.H.; Azhari, S.N.; Harun, Z.; Putra, Z.A. Hybrid modelling for remote process monitoring and optimisation. *Digit. Chem. Eng.* **2022**, *4*, 100044. [CrossRef]
47. Dewage, K.A.K.W.; Hasan, R.; Rehman, B.; Mahmood, S. Enhancing Brain Tumor Detection Through Custom Convolutional Neural Networks and Interpretability-Driven Analysis. *Information* **2024**, *15*, 653. [CrossRef]
48. Chen, W.; Yang, K.; Yu, Z.; Shi, Y.; Chen, C.L.P. A survey on imbalanced learning: Latest research, applications and future directions. *Artif. Intell. Rev.* **2024**, *57*, 137. [CrossRef]
49. Fonseca, J.; Bacao, F. Tabular and latent space synthetic data generation: A literature review. *J. Big Data* **2023**, *10*, 115–137. [CrossRef]
50. Mirzaei, A.; Khaligh-Razavi, S.; Ghodrati, M.; Zabbah, S.; Ebrahimpour, R. Predicting the human reaction time based on natural image statistics in a rapid categorization task. *Vis. Res.* **2013**, *81*, 36–44. [CrossRef]

51. Al Smadi, B.S. Applications of Meta-Heuristic Algorithm with Back Propagation Classifier for Handling Class of General Fish Models. *Int. J. Comput. Sci. Netw. Secur. (IJCSNS)* **2016**, *16*, 38.
52. Badawi, U.A.; Alsmadi, M.K.S. A Hybrid Memetic Algorithm (Genetic Algorithm and Great Deluge Local Search) With Back-Propagation Classifier for Fish Recognition. *Int. J. Comput. Sci. Issues* **2013**, *10*, 348.
53. Alsmadi, M.K.; Omar, K.B.; Noah, S.A. Fish Recognition Based on Robust Features Extraction from Size And Shape Measurements Using Back-Propagation Classifier. *Int. Rev. Comput. Softw.* **2010**, *5*, 489–494.
54. Qiu, C.; Zhang, S.; Wang, C.; Yu, Z.; Zheng, H.; Zheng, B. Improving Transfer Learning and Squeeze- and-Excitation Networks for Small-Scale Fine-Grained Fish Image Classification. *IEEE Access* **2018**, *6*, 78503–78512. [CrossRef]
55. Alsmadi, M.K. Query Sensitive Similarity Measure For Content Based Image Retrieval Using Meta Heuristic Algorithm. *J. King Saud University. Comput. Inf. Sci.* **2018**, *30*, 373–381. [CrossRef]