IJCCS (Indonesian Journal of Computing and Cybernetics Systems)
Vol.20, No.1, January 2026, pp. 13~24
ISSN (print): 1978-1520, ISSN (online): 2460-7258
DOI: https://doi.org/10.22146/ijccs.108820

Classification of Infected Salmon Using CNN Deep Features and Optuna-Optimized SVM

Agus Hendra Pradita*[1], Putu Desiana Wulaning Ayu[2], Dandy Pramana Hostiadi[3]
[1]Magister Program, Department of Magister Information Systems, Institut Teknologi dan Bisnis STIKOM Bali, Indonesia,
[2, 3]Department of Magister Information Systems, Institut Teknologi dan Bisnis STIKOM Bali, Indonesia
e-mail: ***[1]232011012@stikom-bali.ac.id**, [2]wulaning_ayu@stikom-bali.ac.id ,
[3]dandy@stikom-bali.ac.id

Abstrak

Penyakit ikan merupakan tantangan besar dalam industri akuakultur yang berdampak pada produktivitas dan ekonomi terutama dalam budidaya salmon. Meskipun banyak metode klasifikasi telah dikembangkan, masih terdapat celah dalam mengembangkan sistem yang tidak hanya akurat tetapi juga efisien secara komputasi. Penelitian ini bertujuan untuk mengembangkan kerangka kerja klasifikasi gambar yang efisien untuk salmon terinfeksi, menggunakan pendekatan hibrid yang menggabungkan fitur dalam jaringan saraf konvolusional (CNN) dan klasifikasi mesin vektor pendukung (SVM) yang dioptimalkan secara otomatis dengan Optuna. Dataset terdiri dari 1.208 gambar, yang diseimbangkan melalui augmentasi sebelum dibagi menjadi 70% data pelatihan dan 30% data uji. Fitur diekstraksi pada lapisan tengah dari tiga arsitektur CNN yang telah dilatih sebelumnya: EfficientNetB1, ResNet50, dan VGG16, lalu fitur diseleksi menggunakan metode Least Absolute Shrinkage and Selection Operator (LASSO) untuk mengurangi redundansi dan kompleksitas. Model klasifikasi SVM dilatih menggunakan 5-fold stratified cross validation dan dioptimalkan menggunakan Optuna. Hasil menunjukkan bahwa model dengan fitur dari EfficientNetB1 yang dioptimalkan oleh Optuna mencapai akurasi tertinggi sebesar 99,34%, melampaui studi sebelumnya. Analisis beban komputasi menunjukkan bahwa pendekatan ini menghasilkan model ringan dengan waktu inferensi cepat, menjadikannya praktis untuk implementasi di dunia nyata.

Kata kunci*—CNN, kerangka kerja optuna, klasifikasi salmon, SVM*

Abstract

Fish disease presents a significant challenge in the aquaculture industry, adversely affecting productivity and economic outcomes, particularly in salmon farming. Although many classification methods have been developed, a gap remains in creating an accurate and computationally efficient system. This research aims to develop an efficient image classification framework for infected salmon, utilizing a hybrid approach that combines features from a convolutional neural network (CNN) and a support vector machine (SVM) classifier, which is automatically optimized with Optuna. The dataset consists of 1,208 images, balanced through augmentation before being split into 70% training and 30% test data. Features were extracted from the middle layers of three pre-trained CNN architectures: EfficientNetB1, ResNet50, and VGG16. Subsequently, these features were selected using the Least Absolute Shrinkage and Selection Operator (LASSO) method to reduce redundancy and complexity. The SVM classification model was trained using 5-fold stratified cross-validation and optimized with Optuna. The results show that the model with features from EfficientNetB1, optimized by Optuna, achieved the highest accuracy of 99.34%, outperforming previous studies. A computational load

Received July 5th,2025; Revised August 12th, 2025; Accepted January 25th, 2026

analysis indicates that this approach yields a lightweight model with fast inference times, making it practical for real-world implementation.

Keywords*— CNN, optuna framework, salmon classification, SVM.*

1. INTRODUCTION

Salmon aquaculture is a global food sector with high economic value. [1], [2]. However, this industry faces a serious challenge from the spread of disease in intensive farming systems, where high population density leads to financial losses and decreased productivity. [3]. Conventional fish disease detection methods, such as visual inspection, are inefficient, thus requiring a faster and more accurate approach. [4].

Several previous studies have tried to address this challenge. Ahmed et al. used a conventional method of statistical and texture feature extraction, the Gray-Level Co-occurrence Matrix (GLCM), combined with SVM, achieving an accuracy of 94.12% [5]. Further development by Nguyen et al., implementing transfer learning with frozen backbone MobileNet and manual hyperparameter tuning, achieved an accuracy of 96.30% [6]. In another study, Afridiansyah et al. utilized the EfficientNetB1 architecture to detect salmon disease, resulting in an accuracy of 99.18% [7], while Prasanna Kumar et al. developed a Pseudo Hamiltonian Neural Network (PHNN) optimized with the Philippine Eagle Optimization Algorithm (PEOA), achieving an accuracy of 99.24% [8]. Although these results are impressive, a research gap remains for a highly accurate, lightweight, and computationally efficient framework.

In recent years, advanced architectures like the Vision Transformer (ViT) have become the new standard. A Vision Transformer (ViT) 's performance heavily depends on the size of the training data; it outperforms Convolutional Neural Networks (CNNs) on large datasets but performs worse on smaller datasets.[9]. Still, their high computational resource demands make them less feasible for certain applications such as aquaculture. Therefore, this research suggests a more efficient hybrid approach. This method extracts features using pre-trained CNNs (EfficientNetB1, ResNet50, VGG16), selects these features with LASSO to reduce redundancy, and classifies them using an SVM whose configuration is automatically optimized by Optuna. The goal is to develop a classification system for infected salmon that is not only highly accurate but also practical and efficient.

2. METHODS

2.1 Research Stages

This research was carried out through several systematic stages visualized by the flowchart in Figure 1.

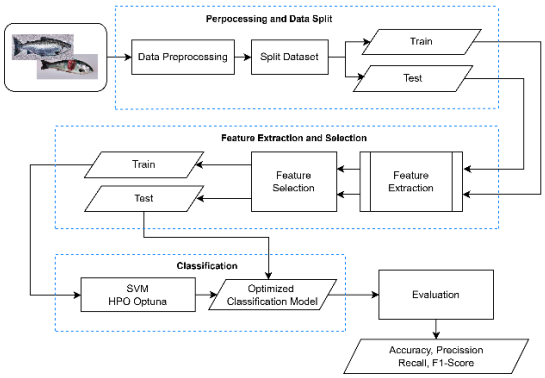

Figure 1 Flowchart of Research

Figure 1 illustrates the outline of the research flow which includes: dataset collection, data preprocessing, dividing the dataset into training datasets and test datasets, feature extraction test scenarios using three *pretrained* CNN architectures (EfficientNetB1, ResNet50, and VGG16), feature selection using LASSO, classification model training using SVM with two scenarios namely default and tuning using Optuna, followed by model performance evaluation using evaluation metrics.

2.2 DataSet

The dataset used in this study is *SalmonScan,* sourced from Mendeley Data. It is a salmon image dataset developed to support the development of disease detection systems in the artificial intelligence-based aquaculture sector. [10].

Table 1 Number of Salmon Image Datasets

Class	Total Data
Fresh	456
Infected	752
Total Data	1,208

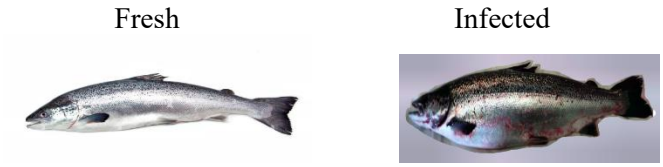

Figure 2 Sample Image of Salmon Fish

The dataset used in this study consists of 1,208 color images of salmon fish with a resolution of 250 × 600 pixels in PNG format. These images were classified into 752 infected fish images and 456 fresh fish images, as shown in Table 1. Data were collected from various online sources and in collaboration with salmon farming companies and annotated based on fish health conditions. Sample images of salmon fish are shown in Figure 2.

2.3 Preprocessing

The preprocessing stage begins by converting images to the LAB color space and applying the CLAHE (Contrast Limited Adaptive Histogram Equalization) technique to enhance contrast.[11]. To address the data imbalance, the minority class (fresh fish) is augmented using horizontal flipping and ±15-degree rotations. Augmentation is performed before the data is split, ensuring balanced class proportions are achieved from the outset. Subsequently, the balanced dataset is divided into two subsets using a stratified split to maintain class proportions. [12]. The split is performed with a 70:30 ratio and a fixed seed (42) to ensure consistent replication of the experiments.

2.4 Feature Extraction

Features were extracted using three pre-trained CNN architectures. The intermediate layer of each model was chosen to balance the trade-off between the simple features of shallow layers and the highly abstract features of deep layers. This layer is selected because it represents complex patterns while retaining relevant visual information. [13].

2 .4.1 EfficientNetB1

EfficientNetB1 is used for feature extraction because its compound scaling method effectively balances accuracy with computational efficiency. [14]For this purpose, the model is loaded with its layers frozen and the top fully connected layer removed (include_top=False),

accepting an input image size of 240x240 pixels. The (8x8x192) feature tensor is extracted from the block6e_add layer and then reduced to a 192-dimensional vector using Global Average Pooling (GAP).

2.4.2 ResNet50

ResNet50 was chosen for feature extraction due to its stability in deep networks, a benefit of its residual learning approach. [15]. For this task, the model was configured with its layers frozen, the top fully-connected layer removed (include_top=False), and an input image size of 224x224 pixels. A (14x14x1024) feature tensor was extracted from the conv4_block6_out layer and then reduced to a 1,024-dimensional vector using GAP.

2.4.3 VGG16

The VGG16 model was chosen for its simple, sequential architecture, simplifying the interpretation of its feature representations. [16]The model was configured with its pre-trained layers frozen, the top fully connected layer removed (include_top=False), and an input image size of 224x224 pixels. Feature extraction was focused on the block4_pool layer, selected for its balance of semantic depth and spatial information. The resulting (14x14x512) tensor was then reduced to a 512-dimensional vector using GAP.

After feature extraction, standardization is performed using StandardScaler with z-score transformation to standardize the scale of features with different value ranges. Standardization is important because the SVM algorithm is sensitive to scale differences. [17]. The process is performed on the training data, and the same parameters are applied to the test data to avoid data leakage.

2.5 Feature Selection

The LASSO (Least Absolute Shrinkage and Selection Operator) method was implemented using Scikit-learn's LassoCV for feature selection. LASSO is effective because its L1 penalty performs both regularization and feature selection simultaneously by shrinking the coefficients of less informative features to zero [18]. The optimal regularization parameter (alpha) was automatically identified through a 5-fold cross-validation process applied exclusively to the training data to determine the most relevant feature set.

2.6 Classification

A Support Vector Machine (SVM) was used for classification, chosen for its ability to effectively handle high-dimensional data by maximizing the margin between classes. [19]. Using a Radial Basis Function (RBF) kernel, the model was trained on the LASSO-selected features. The SVM was trained using a stratified 5-fold cross-validation scheme applied only to the training data to ensure a robust evaluation with a balanced class distribution in each fold. This process tests the separability of the CNN-generated features using the SVM's non-linear decision boundary.

2.7 Hyperparameter Optimization

Hyperparameter optimization in SVM models is performed using Optuna, a *Python* library for *Bayesian-based* optimization and TPE (*Tree-structured Parzen Estimator*) [20]. Optuna efficiently searches for optimal hyperparameters by running parallel trials and implementing pruning algorithms. The search space included parameters C, gamma, and kernel type, with each trial evaluated using stratified 5-fold cross-validation on the training data.

The process ran for 100 trials, with the ASHA (Asynchronous Successive Halving Algorithm) pruning technique applied to terminate unpromising trials early, thereby accelerating the search. [21]. The optimal hyperparameters discovered were then used to retrain the final SVM model on the entire training dataset before its evaluation on the test data.

2.8 Evaluation

The model's performance was evaluated using standard classification metrics to provide a comprehensive overview of its predictive ability. The primary metrics—including Accuracy, Precision, Recall, and the F1 Score—are each calculated from the values in the Confusion Matrix. [22]. The calculation formula for each metric is as follows:

$$Accuracy = \frac{T_N+T_P}{T_N+T_P+F_P+F_N} \quad (1)$$

$$Precision = \frac{T_P}{T_P+F_P} \quad (2)$$

$$Recall = \frac{T_P}{T_P+F_N} \quad (3)$$

$$F1\ Score = 2 * \frac{Precision*Recall}{Precision+Recall} \quad (4)$$

Terms used:

a. True positive (TP): An image of an infected salmon is correctly predicted.
b. True negative (TN): An image of a fresh salmon that is correctly predicted.
c. False positive (FP): An image of a fresh salmon incorrectly predicted as infected.
d. False negative (FN): An image of an infected salmon is incorrectly predicted as fresh.

In addition, to evaluate the model's ability to distinguish between classes, a Confusion Matrix visualization facilitates the interpretation of classification results.

3. RESULTS AND DISCUSSION

3.1 Preprocessing and Augmentation

CLAHE was selectively applied to channel L (luminance) with clipLimit = 1.5 and tileGridSize = 4.4 to enhance adaptive contrast and emphasize the lesion texture. Channel A (red-green) using CLAHE with clipLimit = 1.0 and tileGridSize = (8,8) to highlight the reddish color elements associated with infection while maintaining the color stability of the image. The preprocessing results are shown in Figure 3.

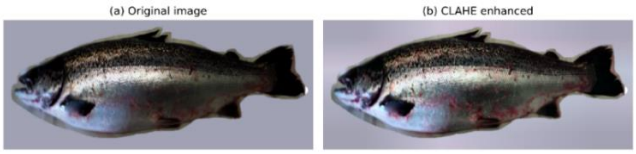

Figure 3 CLAHE Process Result

Table 2 shows the implementation of augmentation and split at a ratio of 70:30, which resulted in a consistent proportional distribution in the training and testing sets.

Table 2 Dataset after Augmentation and Split

Class	Training Data	Test Data	Total Data
Fresh	526	226	752
Infected	526	226	752
Total Data	1052	452	1504

3.2 Feature Extraction and Feature Selection

Feature extraction is performed using three scenarios and three pre-trained CNN architectures. It involves taking the output of the middle layer, reducing it using Global Average

Pooling (GAP), and selecting it using LASSO to minimize redundancy and identify the most relevant features.

Table 3 Summary of CNN Feature Extraction and Selection Results

Feature	Extraction Layer	Total Features	Selected Features	Percentage Selected	Alpha
EfficientNetB1	block6e_add	192	142	74,00%	0.001874
ResNet50	conv4_block6_out	1024	471	46,00%	0.000811
VGG16	block4_pool	512	227	44,33%	0.002477

Based on Table 3, which summarizes the results of CNN feature extraction and selection, it can be observed that the three CNN models used show different characteristics in the feature extraction process. The EfficientNetB1 model with block6e_add extraction layer successfully extracted 192 features, and after going through the selection process, 142 features were selected, or about 74.00% of the total extracted features, with an alpha value of 0.001874. ResNet50 using layer conv4_block6_out produced the highest number of features at 1024 features, but after selection, only 471 features were selected with a selection percentage of 46.00% and an alpha value of 0.000811. Meanwhile, VGG16 with layer block4_pool extracted 512 features and selected 227 features with the lowest selection percentage of 44.33% and the highest alpha value of 0.002477. These results show that EfficientNetB1 has the highest feature selection efficiency, where most of the extracted features are considered relevant and informative for use in the next stage.

3.3 Classification Performance Evaluation

Model training and evaluation were conducted in two scenarios: a baseline implementation with default parameters and hyperparameter optimization using the Optuna framework. The research aims to identify potential performance improvements through hyperparameter tuning. The optimization process uses a search space with C parameters in the range of 1e-9 to 1e8, gamma in the range of 1e-9 to 1e2, three types of kernels (RBF, Polynomial, Sigmoid), and polynomial degree 1-10, which are explored through 100 optimization trials for each model.

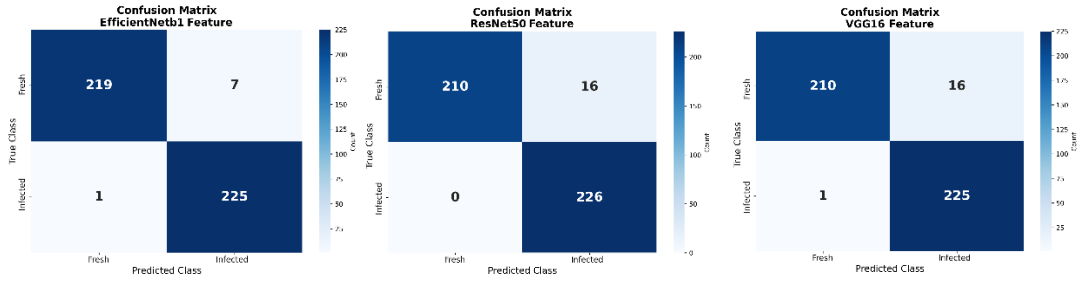

Figure 4 Confusion Matrix of SVM Classification based on EfficientNetB1, ResNet50 and VGG16 Features without hyperparameter tuning.

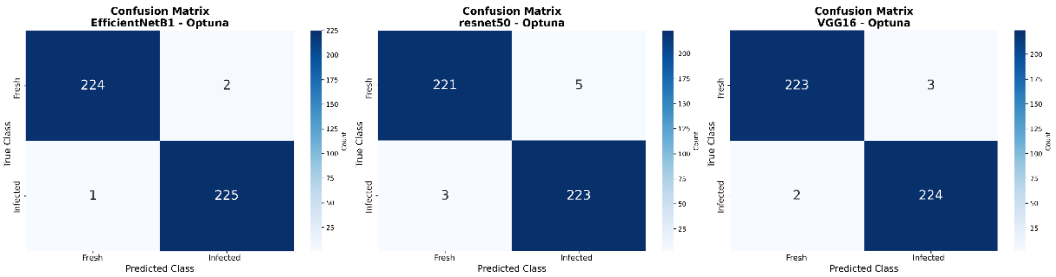

Figure 5 Confusion Matrix of SVM Classification based on EfficientNetB1, ResNet50 and VGG16 Features with hyperparameter tuning.

The confusion matrix in Figure 4 shows each model's different classification characteristics. The EfficientNetB1 baseline has a good error distribution (7 FP, 1 FN). ResNet50 achieves perfect sensitivity with zero false negatives but has 16 high false positives, while VGG16 shows a similar pattern with 16 false positives and one false negative.

In Figure 5, after hyperparameter optimization using Optuna, all models have significantly improved results. EfficientNetB1 achieved the best performance with only two false positives and maintained one false negative. ResNet50 experienced strategic balancing where false negatives increased from 0 to 3, but false positives dropped dramatically from 16 to 5 cases. VGG16 also saw significant improvement with false positives dropping from 16 to 3, although false negatives rose slightly from 1 to 2.

In the application to detect infected fish, all models achieved a detection failure rate of less than 2% after optimization, which means that almost no sick fish were missed, which is very important to prevent the spread of disease. On the other hand, the models also drastically reduced the error in classifying fresh fish as infected.

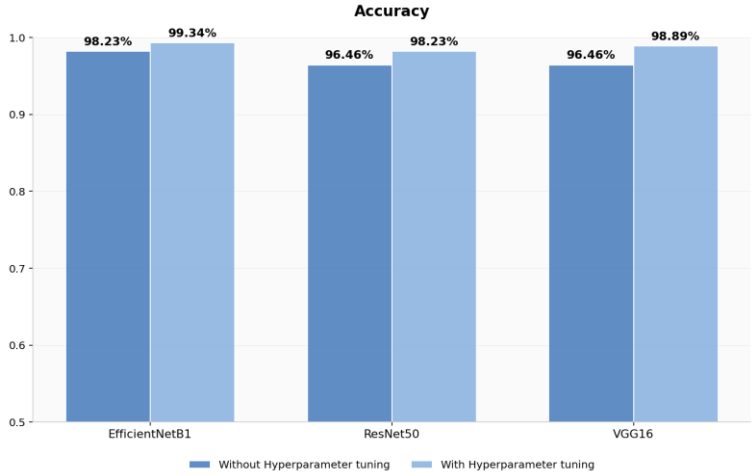

Figure 6 Comparison of Model Accuracy Without and with Hyperparameter Tuning

Table 4 Comparison of Model Evaluation Metrics without and with Hyperparameter Tuning

Model	Scenario	Accuracy	Precision	Recall	F1-Score	FPR	FNR
EfficientNetB1	Baseline	98.23%	96.98%	99.56%	98.25%	3.10%	0.44%
EfficientNetB1	Optuna	**99.34%**	99.56%	99.12%	99.34%	0.88%	0.44%
ResNet50	Baseline	96.46%	93.39%	100.00%	96.58%	7.08%	0.00%
ResNet50	Optuna	98.23%	97.81%	98.67%	98.24%	2.21%	1.33%
VGG16	Baseline	96.46%	93.39%	100.00%	96.58%	7.08%	0.44%
VGG16	Optuna	98.89%	98.68%	99.12%	98.90%	1.33%	0.88%

The visualization results in Figure 6 show the comparison and varying patterns of performance improvement when comparing the no-optimization condition with the hyperparameter optimization results. VGG16 displays the most striking gap between the two conditions, while EfficientNetB1 shows the highest achievement after optimization.

In Table 4, ResNet50 experienced a favorable trade-off, with recall dropping from 100% to 98.67%, but precision increased significantly from 93.39% to 97.81%, resulting in a more balanced and applicable F1 Score. VGG16 showed the highest responsiveness to tuning with 2.43% improvement, reaching 98.89% from a baseline of 96.46%. EfficientNetB1 achieved the most optimal accuracy of **99.34%** with the highest precision of 99.56% and the lowest false positive rate of 0.88%, demonstrating excellent discrimination ability.

Table 5 Optimal Hyperparameter Configuration (100 trials)

Feature	C Value	Gamma Value	Kernel	Trials Completed	Trials pruned	Best Trial
EfficientNetB1	9400127.674	0.003697	RBF	80	20	49
ResNet50	1926.417	0.000143	RBF	76	24	90
VGG16	1339440.003	0.000502	RBF	80	20	85

Based on Table 5, the three Deep Features CNNs show different optimal parameter configurations after optimization. All models use the RBF (Radial Basis Function) kernel. EfficientNetB1 achieved optimal performance on the 49th trial with a C value of 9400127.674 and a gamma value of 0.003697. ResNet50 obtained the best results in the 90th trial with the lowest C value of 1926.417 and the lowest gamma value of 0.000143. At the same time, VGG16 achieved optimal performance in the 85th trial with a C value of 1339440.003 and a gamma value of 0.000502. The difference in hyperparameter values shows that each model has different characteristics and sensitivity to parameter settings, where ResNet50 tends to require stronger regularization with smaller C and gamma to achieve optimal performance.

Table 6 Summary of 5-Fold Cross-Validation Average Evaluation Metrics

Feature	Scenario	CV Accuracy ± Std	CV Precision± Std	CV Recall	CV F1-Score
EfficientNetB1	Baseline	96.86% ± 0.71%	96.92% ± 0.69%	96.87% ± 0.71%	96.86% ± 0.71%
EfficientNetB1	Optuna	98.00% ± 0.35%	98.03% ± 0.35%	98.01% ± 0.35%	98.00% ± 0.35%
ResNet50	Baseline	94.96% ± 0.88%	95.22% ± 0.80%	94.6% ± 0.88%	94.96% ± 088%
ResNet50	Optuna	97.91% ± 0.65%	97.95% ± 0.61%	97.91% ± 0.65%	97.91% ± 0.65%
VGG16	Baseline	95.91% ± 0.82%	96.07% ± 0.83%	95.92% ± 0.82%	95.91% ± 0.82%
VGG16	Optuna	98.10% ± 0.52%	98.10% ± 0.52%	98.10% ± 0.52%	98.10% ± 0.52%

Cross-validation confirmed the improvement's stability with consistent standard deviation reduction. The EfficientNetB1 feature shows an improvement in CV accuracy from 96.86% ± 0.71% to 98.00% ± 0.35%, ResNet50 from 94.96% ± 0.88% to 97.91% ± 0.65%, and VGG16 from 95.91% ± 0.82% to 98.10% ± 0.52%. These results reflect an improvement in average performance and excellent robustness.

Table 7 Comparison with previous research

Researcher	Method/Architecture	Optimization Technique	Accuracy
Ahmed et al. (2022)	Manual Feature Extraction (Statistical + GLCM) + SVM Classification	Default hyperparameters	94.12%
Nguyen et al. (2024)	MobileNetV1 Fine-Tuned	Manual hyperparameter tuning	96.30%
Afridiansyah & Setiadi (2024)	EfficientNetB1	Manual hyperparameter tuning	99.18%
Prasanna Kumar et al. (2024)	SKCET Feature Extraction + Pseudo Hamiltonian Neural Network (PHNN) + PEOA	Philippine Eagle Optimization Algorithm	99.24%
This research (2025)	EfficientNetB1 Deep Features + LASSO + SVM Classification	Optuna	99.34%

The SalmonScan dataset has been used in several studies to evaluate salmon disease detection systems, with method developments showing consistent performance improvements. Table 7 shows the progression of research specific to the SalmonScan dataset in the domain of classification/detection of infected salmon. Starting from the traditional approach of Ahmed et al. (2022) with manual feature extraction and default hyperparameter on SVM, which achieved 94.12% accuracy, then progressed to a deep learning approach with Nguyen et al. (2024) using MobileNetV1 and manual hyperparameter tuning reached 96.30%, followed by Afridiansyah & Setiadi (2024) with EfficientNetB1 and manual tuning reached 99.18%. Prasanna Kumar et al. (2024), who used SKCET feature extraction with PHNN and PEOA optimization, achieved 99.24%. This research proposes an innovative hybrid approach that combines EfficientNetB1 deep features, LASSO feature selection, and SVM classification optimized with Optuna, achieving the highest accuracy of 99.34%.

3.4 Qualitative Analysis

A qualitative analysis was performed on prediction samples from the test set using the best-performing model (EfficientNetB1—Optuna) to better understand the model's performance and limitations. Figure 7 displays examples of both correct and incorrect classifications.

(a) (b) (c) (d)

Figure 7 Qualitative Examples of Classification Results

a) True Positive (Correct): The model accurately identified an infected salmon showing visible reddish lesions along its body. This highlights the model's ability to recognize common visual symptoms of the disease.
b) True Negative (Correct): A fresh salmon with uniform color and intact scales was correctly classified as "Fresh". This confirms the model's capacity to distinguish fresh fish from infected ones.
c) False Positive (Type I Error): In this case, a fresh salmon was misclassified as "Infected". Visual inspection reveals a strong light reflection on the fish's scales. The CLAHE pre-processing step, intended to enhance contrast, likely amplified this light artifact, causing the model to misinterpret it as a lesion. This type of error highlights the model's sensitivity to unusual lighting conditions.

d) False Negative (Type II Error): This is the most significant error, where an infected salmon was incorrectly classified as "Fresh." The signs of infection on this sample might have been very early or too subtle, making them hard to detect from the extracted features. This failure indicates that the model may still struggle with ambiguous cases and highlights the need for training data covering all disease stages.

This qualitative analysis provides valuable insight that, despite the high quantitative accuracy, the model has limitations that can guide future improvements.

3.5 Computational Load Analysis

Beyond accuracy, practical aspects such as model speed and efficiency are crucial for real-world deployment. Therefore, the proposed approach's computational load was analyzed by measuring the inference time and comparing the feature dimensions before and after LASSO selection. The results of this analysis are presented in Table 8.

Table 8 Computational Load and Efficiency Analysis

Feature Model	Initial Dimensions	Dimensions after LASSO	Total Time	Average Time per Image	memory consumption
EfficientNetB1	192	142	31.68 s	69.75 ms	725.31 MB
ResNet50	1024	471	43.83 s	96.56 ms	873.55 MB
VGG16	512	227	72.37 s	159.39 ms	675.09 MB

The analysis in Table 8 shows the effectiveness of the proposed framework. The LASSO method greatly reduced feature dimensions across all models, especially for VGG16 (55.7%) and ResNet50 (54%). This efficiency gain highlights a clear trade-off between processing speed and memory use. While the EfficientNetB1 model provided the fastest inference time at 69.75 ms per image, VGG16 was the most memory-efficient, using only 675.09 MB. ResNet50 offered a balanced performance between these two factors.

These results have practical implications for real-world deployment. Combining a reduced feature set with an SVM classifier creates a highly efficient system. EfficientNetB1 is the best choice for applications where speed is critical, such as real-time monitoring. Conversely, VGG16 is recommended for deployment on hardware with significant memory constraints.

4. CONCLUSIONS

Based on the research results, combining deep learning feature extraction, LASSO feature selection, and SVM classification proved highly effective for salmon disease detection. Hyperparameter optimization using the Optuna framework significantly improved all models, with the EfficientNetB1 feature model reaching the highest accuracy of 99.34%. This result represents an improvement from its 98.23% baseline and exceeds the accuracy reported in previous studies, which ranged from 94.12% to 99.24%.

This research's primary contribution is achieving high accuracy and demonstrating a systematic, efficient, and reproducible framework. The 5-fold cross-validation results showed excellent stability and robustness with a low standard deviation. The computational load analysis also confirmed that the proposed method produces a lightweight and fast model, making it a practical solution for real-world applications. For future work, it would be interesting to compare this hybrid approach with modern, end-to-end architectures like the Vision Transformer (ViT) to analyze further the trade-offs between accuracy, model complexity, and data requirements in fish disease detection.

REFERENCES

[1] R. L. Naylor *et al.*, "A 20-year retrospective review of global aquaculture," *Nature*, vol. 591, no. 7851, pp. 551–563, 2021, doi: 10.1038/s41586-021-03308-6.

[2] I. Y. Gudbrandsdottir, N. M. Saviolidis, G. Olafsdottir, G. V Oddsson, H. Stefansson, and S. G. Bogason, "Transition Pathways for the Farmed Salmon Value Chain: Industry Perspectives and Sustainability Implications," *Sustainability*, vol. 13, no. 21, 2021, doi: 10.3390/su132112106.

[3] A. G. Murray, M. Moriarty, B. Rabe, and D. Tulett, "Bio-physical models for the management of micropathogens in Scottish aquaculture: a preliminary view to farming further offshore," *Mar Ecol Prog Ser*, vol. 679, pp. 133–147, 2021, [Online]. Available: https://www.int-res.com/abstracts/meps/v679/meps13875

[4] V. A, P. S. Rathore, S. B. E, V. N, and H. S, "Fish Disease Detection Using Machine Learning," in *2024 International Conference on Science Technology Engineering and Management (ICSTEM)*, 2024, pp. 1–4. doi: 10.1109/ICSTEM61137.2024.10560522.

[5] M. S. Ahmed, T. T. Aurpa, and M. A. K. Azad, "Fish Disease Detection Using Image Based Machine Learning Technique in Aquaculture," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 8, pp. 5170–5182, Sep. 2022, doi: 10.1016/j.jksuci.2021.05.003.

[6] H. Van Nguyen, T. Q. Huynh, N. M. Nguyen, A. K. Su, and H. T. Nguyen, "Comparative Analysis of Fine-Tuned MobileNet Versions on Fish Disease Detection," in *Lecture Notes on Data Engineering and Communications Technologies*, vol. 214, Springer Science and Business Media Deutschland GmbH, 2024, pp. 201–212. doi: 10.1007/978-3-031-64766-6_20.

[7] R. Afridiansyah and D. R. I. M. Setiadi, "Comparison of EfficientNetB1 Model Effectiveness in Identifying Fish Diseases in South Asian Fish Diseases and Salmon Fish Diseases," *JAIC*, vol. 8, no. 2, pp. 453–462, 2024.

[8] P. kumar M, S. K. K, K. P, and S. C, "Fish disease detection in aquaculture using pseudo hamiltonian neural network optimized with Philippine eagle optimization algorithm," *Knowl Based Syst*, vol. 317, May 2025, doi: 10.1016/j.knosys.2025.113374.

[9] I. B. Akkaya, S. S. Kathiresan, E. Arani, and B. Zonooz, "Enhancing performance of vision transformers on small datasets through local inductive bias incorporation," *Pattern Recognit*, vol. 153, p. 110510, 2024, doi: https://doi.org/10.1016/j.patcog.2024.110510.

[10] M. S. Ahmed and S. M. Jeba, "SalmonScan: A novel image dataset for machine learning and deep learning analysis in fish disease detection in aquaculture," *Data Brief*, vol. 54, p. 110388, Jun. 2024, doi: 10.1016/j.dib.2024.110388.

[11] S. R. Borra, N. P. Tejaswini, V. Malathy, B. Magesh Kumar, and M. I. Habelalmateen, "Contrast Limited Adaptive Histogram Equalization based Multi-Objective Improved Cat Swarm Optimization for Image Contrast Enhancement," in *2023 International Conference on Integrated Intelligence and Communication Systems (ICIICS)*, 2023, pp. 1–5. doi: 10.1109/ICIICS59993.2023.10420959.

[12] F. C. Farias, T. B. Ludermir, and C. J. A. B. Filho, "Similarity Based Stratified Splitting: an approach to train better classifiers," *ArXiv*, vol. abs/2010.06099, 2020, [Online]. Available: https://api.semanticscholar.org/CorpusID:222310611

[13] A. Holliday and G. Dudek, "Pre-trained CNNs as Visual Feature Extractors: A Broad Evaluation," in *2020 17th Conference on Computer and Robot Vision (CRV)*, 2020, pp. 78–84. doi: 10.1109/CRV50864.2020.00019.

[14] P. Utami, R. Hartanto, and I. Soesanti, "The EfficientNet Performance for Facial Expressions Recognition," in *2022 5th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, 2022, pp. 756–762. doi: 10.1109/ISRITI56927.2022.10053007.

[15] Y. Jing and L. Zhang, "An Improved Dual-Path Attention Mechanism With Enhanced Discriminative Power for Asset Comparison Based on ResNet50," in *2025 5th International Conference on Advances in Electrical, Electronics and Computing Technology (EECT)*, 2025, pp. 1–10. doi: 10.1109/EECT64505.2025.10966991.

[16] N. Zakaria and Y. M. M. Hassim, "A Review Study oftheVisual Geometry Group Approachesfor Image Classification," *Journal of Applied Science, Technology and Computing*, 2024, [Online]. Available: https://api.semanticscholar.org/CorpusID:273421268

[17] A. Ambarwari, Q. J. Adrian, and Y. Herdiyeni, "Analysis of the Effect of Data Scaling on the Performance of the Machine Learning Algorithm for Plant Identification," *Jurnal Resti Vo.4 No 1*, vol. 1, pp. 117–122, 2020.

[18] S. Suprapto and Y. L. Nikmah, "Ridge and Lasso Regression for Feature Selection of Overlapping Ibuprofen and Paracetamol UV Spectra," *Morrocan Journal of Chemistry Vol. 11 No. 1 (2023)*, vol. 11, 2023.

[19] K. L. Du, B. Jiang, J. Lu, J. Hua, and M. N. S. Swamy, "Exploring Kernel Machines and Support Vector Machines: Principles, Techniques, and Future Directions," Dec. 01, 2024, *Multidisciplinary Digital Publishing Institute (MDPI)*. doi: 10.3390/math12243935.

[20] H. Almarzooq and U. bin Waheed, "Automating hyperparameter optimization in geophysics with Optuna: A comparative study," *Geophys Prospect*, vol. 72, no. 5, pp. 1778–1788, 2024, doi: https://doi.org/10.1111/1365-2478.13484.

[21] C. Deng, A. Kim, and W. Lu, "A generic battery-cycling optimization framework with learned sampling and early stopping strategies," *Patterns*, vol. 3, no. 7, Jul. 2022, doi: 10.1016/j.patter.2022.100531.

[22] R. Yacouby and D. Axman, "Probabilistic Extension of Precision, Recall, and F1 Score for More Thorough Evaluation of Classification Models," in *Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems*, S. Eger, Y. Gao, M. Peyrard, W. Zhao, and E. Hovy, Eds., Online: Association for Computational Linguistics, Nov. 2020, pp. 79–91. doi: 10.18653/v1/2020.eval4nlp-1.9.