frontiers | Frontiers in Computer Science

TYPE Original Research
PUBLISHED 29 February 2024
DOI 10.3389/fcomp.2024.1326452

Automatic labeling of fish species using deep learning across different classification strategies

Javier Jareño[1], Guillermo Bárcena-González[1]*, Jairo Castro-Gutiérrez[2], Remedios Cabrera-Castro[2] and Pedro L. Galindo[1]

[1]Computer Science Department, University of Cádiz, Cádiz, Spain, [2]Biology Department, University of Cádiz, Cádiz, Spain

Check for updates

OPEN ACCESS

EDITED BY
Nicola Strisciuglio,
University of Twente, Netherlands

REVIEWED BY
Bojan Žlahtič,
University of Maribor, Slovenia
Carlos Vilas,
Spanish National Research Council (CSIC),
Spain

*CORRESPONDENCE
Guillermo Bárcena-González
✉ guillermo.barcena@uca.es

RECEIVED 23 October 2023
ACCEPTED 05 February 2024
PUBLISHED 29 February 2024

CITATION
Jareño J, Bárcena-González G, Castro-Gutiérrez J, Cabrera-Castro R and Galindo PL (2024) Automatic labeling of fish species using deep learning across different classification strategies.
Front. Comput. Sci. 6:1326452.
doi: 10.3389/fcomp.2024.1326452

Convolutional neural networks (CNNs) have revolutionized image recognition. Their ability to identify complex patterns, combined with learning transfer techniques, has proven effective in multiple fields, such as image classification. In this article we propose to apply a two-step methodology for image classification tasks. First, apply transfer learning with the desired dataset, and subsequently, in a second stage, replace the classification layers by other alternative classification models. The whole methodology has been tested on a dataset collected at Conil de la Frontera fish market, in Southwest Spain, including 19 different fish species to be classified for fish auction market. The study was conducted in five steps: (i) collecting and preprocessing images included in the dataset, (ii) using transfer learning from 4 well-known CNNs (ResNet152V2, VGG16, EfficientNetV2L and Xception) for image classification to get initial models, (iii) apply fine-tuning to obtain final CNN models, (iv) substitute classification layer with 21 different classifiers obtaining multiple F1-scores for different training-test splits of the dataset for each model, and (v) apply *post-hoc* statistical analysis to compare their performances in terms of accuracy. Results indicate that combining the feature extraction capabilities of CNNs with other supervised classification algorithms, such as Support Vector Machines or Linear Discriminant Analysis is a simple and effective way to increase model performance.

KEYWORDS

supervised learning, classification, fish species, SVM, LDA, deep learning, multiple comparison analysis

1 Introduction

Convolutional neural networks (CNNs) have radically transformed the field of image processing and computer vision over the past decade. Leveraging their unique architecture, which mimics human visual processing, CNNs have consistently demonstrated their ability to extract intricate patterns and features from images, often outperforming human performance on specific tasks. This, together with data augmentation techniques—through rotation, zooming, panning, and other techniques—that allow the size and diversity of the training dataset to be expanded, is a well-established approach for similar tasks, widely recognized in the scientific community.

Several studies, such as those by Norouzzadeh et al. (2018), Allken et al. (2019), Barbedo (2019), Kaya et al. (2019), Montalbo and Hernandez (2019), and Palmer et al. (2022), highlight the effectiveness of CNNs in animal species classification. The adoption of pre-trained models such as ResNet-50, VGG16, and Xception underlines a fundamental trend in the deep learning community: transfer learning. Instead of training a model from scratch, researchers leverage pre-trained models on large datasets such as ImageNet to benefit from their already learned features. These models are then refined on specific datasets, often yielding superior results in a fraction of the original training time. We could highlight the following studies in this field ResNet-50 (He et al., 2016a), ResNet152V2 (He et al., 2016b), VGG16 (Simonyan and Zisserman, 2014), EfficientNetV2L (Tan and Le, 2021), Xception (Chollet, 2017), AlexNet (Krizhevsky et al., 2012), and GoogleNet (Szegedy et al., 2014).

The works cited employ a range of pre-trained models, each with its architecture and strengths. For instance, ResNet models, with their residual connections, are known to alleviate the vanishing gradient problem in deep networks (He et al., 2016a; Huang et al., 2017). On the other hand, EfficientNet architectures scale all dimensions of the model (width, depth, and resolution) based on a set compound coefficient, making them extremely efficient. These models, when paired with strategic data augmentation, can discern subtle differences among animal species, even those imperceptible to the human eye (Ibraheam et al., 2021).

Despite the prowess of CNNs, innovation in the field of machine learning continues unabated. The classification model presented in this paper represents this evolutionary process. Rather than relying solely on end-to-end training of CNNs, the model presented here employs a two-step process. First, feature extraction is performed using a CNN. These extracted features, rich in information content, are fed into a supervised learning model, which can be either a traditional machine learning algorithm or another neural network variant. This hybrid approach aims to harness the strength of CNNs in feature extraction and combine it with the robustness of other classification algorithms, potentially providing more accurate and interpretable results. Crucially, the selection of the definitive model will be based on a rigorous hypothesis testing process, ensuring that the chosen approach not only performs well in theory but also stands up to empirical scrutiny.

The supervised learning models used in this paper range from dimensionality reduction algorithms such as linear discriminant analysis (LDA) and quadratic discriminant analysis (QDA), through the generation of single and multiple decision trees, the latter known as Random Forest (RF), the use of Kernel methods such as support vector machines (SVM), the use of non-parametric classification methods such as the k-nearest neighbor (K-NN) algorithm and the use of probabilistic classification methods such as the Gaussian Naive Bayes (GNB) algorithm. All of them have been used independently for the classification of different biogeochemical species (Franco et al., 1990; Cutler et al., 2007; Munoz et al., 2013; Pundlik, 2016; Saberioon et al., 2018; Shang and Li, 2018; Deep and Dash, 2019; Knauer et al., 2019; Luan et al., 2020; Nuraini, 2022).

In essence, while CNNs have paved the way for unparalleled advancements in image classification, the ever-evolving landscape of machine learning ensures that newer, potentially more efficient methods will be needed.

Transfer learning is a common and effective approach in this field and has many advantages, such as a great reduction in training times and a better performance, specially for small datasets, where training from scratch may result in a severe overfitting. The usual approach consists on removing the final layers responsible for classification and replace them with new layers (usually fully-connected). The pre-trained model's weights are frozen, and only the weights of the new layers are trained on the specific dataset.

Previous models, which utilized the entire neural network, usually have a higher computational cost and possibly, a higher risk of overfitting when compared to some classic classification algorithms. The proposed technique allows for the exploration of different approaches and the identification of the classifier that best suits the characteristics of our data.

This work proposes a methodology for the design of a classification model using transfer learning and replacing the final layers by other alternative classifiers instead of using a set of fully-connected layers. In order to statistically assess the performance, a multiple comparison analysis has been applied to check whether there are differences in performance between the different classification models. This methodology has allowed to improve the generalization ability of the final model in a real-world scenario (fish market image recognition), getting a significant increase in performance.

2 Materials and methods

2.1 Dataset

The southern Spanish fish market has a rich history and a wide variety/range of seafood products. Most auction facilities, taking a leap toward modernity, have adopted digital platforms for their operations. These platforms primarily use photography to showcase their products to prospective buyers. However, by analysing these images, researchers can determine the size, species and weight, ensuring that fishing practices conform to sustainable standards. Many studies such as Dobeson (2016) and Jarek and Mazurek (2019) show that the integration of technological advances can significantly improve the traceability of sales and auction processes.

The dataset used for this study was gathered from sales conducted at the Conil de la Frontera (36°17'44.1"N 6°08'16.9"W) fish market, where each sold box is associated with an image. The images are captured within the sales box where only specimens of the same species of fish appear. Once the fishing vessel has unloaded its cargo and placed the catches in their sales boxes, they are transferred to a conveyor belt where they are weighed, and an image is captured for publication on the auction portal of the fish market. The photos have dimensions of 800 × 480 pixels with a resolution of 96 pixels per inch (ppi) at a fixed height of approximately 1 meter at the point where the box is weighed. Additionally, no additional lighting or flash is used beyond what is present in the room where these measurements are conducted.

These images are stored alongside auction sales data, including size, weight, and the Food and Agriculture Organization of the United Nations Code (FAO) which refers to the abbreviated nomenclature of the species, among other information, though private data of both buyers and sellers has been anonymized. The original raw dataset comprises 12,525 images representing 80 different species across 38 distinct

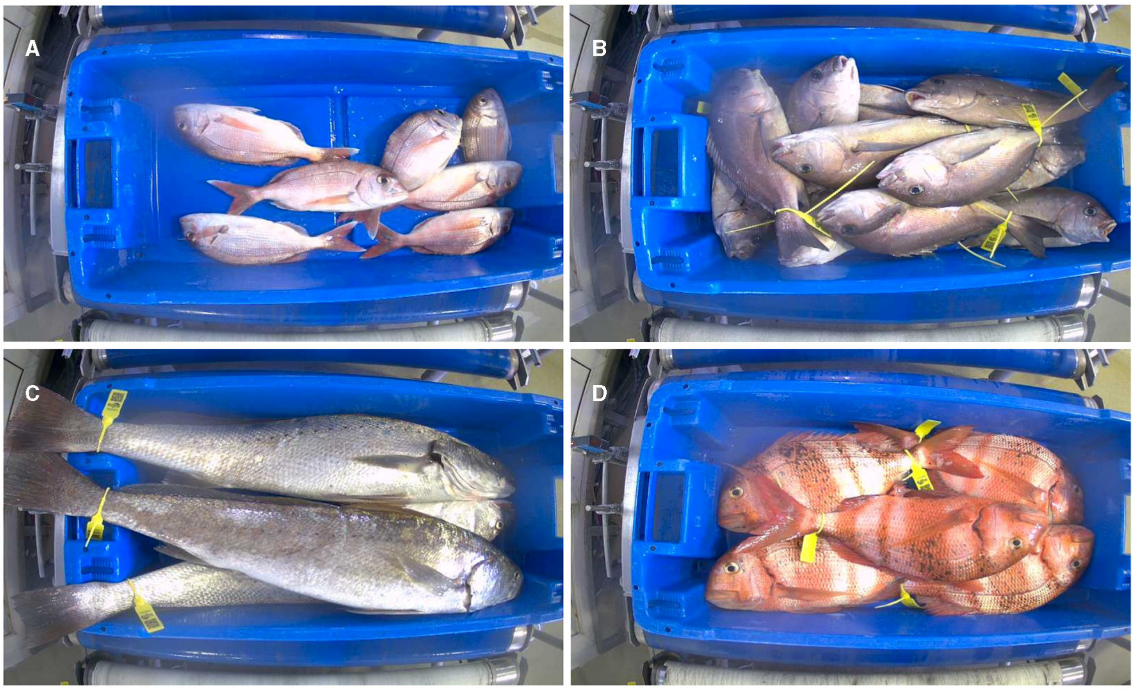

FIGURE 1
Main species of the fishing port of Conil de la Frontera. **(A)** *Pagrus pagrus*. **(B)** *Plectorhinchus mediterraneus*. **(C)** *Argyrosomus regius*. **(D)** *Pagrus auriga*.

days. However, there's a noticeable class imbalance issue, with some species having fewer than 30 sample images, while more common species such as those depicted in Figures 1A–D have 1.217, 2.167, 836 and 1.120 instances, respectively.

To address class imbalance, the study focuses on the 19 species that have over 200 sales instances. The dataset is divided in a manner such that every species has 80% of their instances in training, 10% for validation and 10% for testing, henceforth keeping the balance in all the stages. Data augmentation techniques (Shorten and Khoshgoftaar, 2019) are applied to the training data, with the aim to increase instances for each species to a minimum of 500, greatly improving system performance. This approach enables the network to learn from image variations, including fish distribution, caliber differences, blood stains, camera water droplets, and snow in boxes, thus facilitating knowledge extrapolation. The original image set is retained, and new images are generated using transformations such as mirroring, rotation, blur, optical distortion, and hue-saturation adjustments. These are performed using the `Albumentations` framework (Buslaev et al., 2020), a Python library for fast and flexible image augmentations.

Therefore, the resulting dataset consists of 10,632 instances, representing 19 target species, with an average of 640 images per class. Furthermore, 20% of the dataset is reserved for test and validation purposes. Therefore, it is divided into training (80%–8,505 original instances, 12,095 with data augmentation), validation (10%–1,064 instances), and testing (10%-1,063 instances).

As established in the introduction, this study aims to conduct a comparative analysis among various classification models based on convolutional neural networks (CNNs) and distinct supervised learning models that do not utilize neural networks for classification.

2.2 CNN models and supervised learning

A CNN is a deep learning model designed for the processing of grid-structured data, such as images or matrix data. In contrast to conventional neural networks, CNNs employ a specialized architecture that leverages the spatial correlations in data by applying convolutional filters across successive layers (Goodfellow et al., 2016). These filters autonomously acquire local features, such as edges, textures, and shapes, which accumulate to construct increasingly abstract and meaningful representations as one delves deeper into the network.

One of the main advantages of CNNs is their inherent capacity to autonomously extract features from input data without necessitating prior preprocessing. Instead of requiring the manual design and selection of pertinent features for a specific task, CNNs dynamically and hierarchically acquire the most discriminative features during the training process. This attribute renders them exceptionally potent for various computer vision tasks, including but not limited to image classification, object detection, segmentation, and face recognition.

Pre-trained networks refer to CNN models that have undergone training on extensive datasets, such as ImageNet, and have garnered widespread popularity within the deep learning community (Hussain et al., 2019). The current study will employ these pre-trained models, including ResNet152V2 (He et al., 2016b), VGG16 (Simonyan and Zisserman, 2014), EfficientNetV2L (Tan and Le, 2021), and Xception (Chollet, 2017), as a foundation. These models have acquired the ability to discern a diverse array of visual features, rendering them a robust basis for various computer vision tasks. By harnessing the wealth of knowledge embedded in these models, substantial time and resources can be conserved, as there is no need to initiate training from scratch, leading to expedited access to high-quality results.

Let the VGG16 CNN architecture shown in Figure 2 work as an example. It can't be used without applying transfer learning (Goodfellow et al., 2016), a technique used to extrapolate pre-trained CNN models to new classification problems. CNNs can be divided in two main parts: the convolution layers where the feature extraction is performed, and the fully connected layers (Multi Layer Perceptron MLP) where the classification is performed based on the features extracted. Since the original number of classes is different of ours, the last part of the CNN architecture is replaced with a new Fully Connected Neural Network (FCNN) which matches our number of fish species, 19.

The initial feature extraction layers remain unchanged, as they retain the knowledge acquired from the ImageNet dataset. However, the classification layers are substituted with a *GlobalAveragePooling2D* layer followed by a *Dense-Softmax* output layer which will serve as the classification layer.

To perform the comparison, the same number of features has been employed for all algorithms. These features are extracted by the CNN models in their convolutional stage. Given this, VGG16 results in 14 × 14 × 512 = 100,352 (as seen in Figure 2) neurons after the final pooling and flattening of neurons across all convolutional layers. Therefore, it will be the number of features extracted by VGG16 for the comparison among all supervised learning models. Ultimately, the goal is to utilize the feature extractor of CNN models in their convolutional stage, which was employed in the fully-connected layers of the original model for the classification, as a feature extractor that will feed the features to the supervised learning models. Thus, within the same CNN model, comparisons are made with the same number of extracted features, and among different CNN models, the number of these features will vary.

The workflow for training and validating the models is outlined in Algorithm 1. The system begins by initially training and evaluating the proposed CNN-based model. Subsequently, the final classification layers are removed, and the model's output is set to the features extracted from the convolutional layers. This modified model is then used as a feature extractor for the proposed supervised learning classifier models.

During the supervised learning process, all images are preprocessed using the feature extractor obtained from the original CNN, which will serve as input for the proposed models. The results of each combination of the CNN model and the supervised learning model are evaluated through the mean of five executions of 10-fold Cross-Validation. The algorithms employed are those mentioned in the Introduction, namely: Linear Support Vector Machine (SVM), Radial kernel SVM, Polynomial SVM, Random Forest, Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis, Gaussian Bayes, Decision Tree, and K-NN with K values ranging from 1 to 29 in steps of 2.

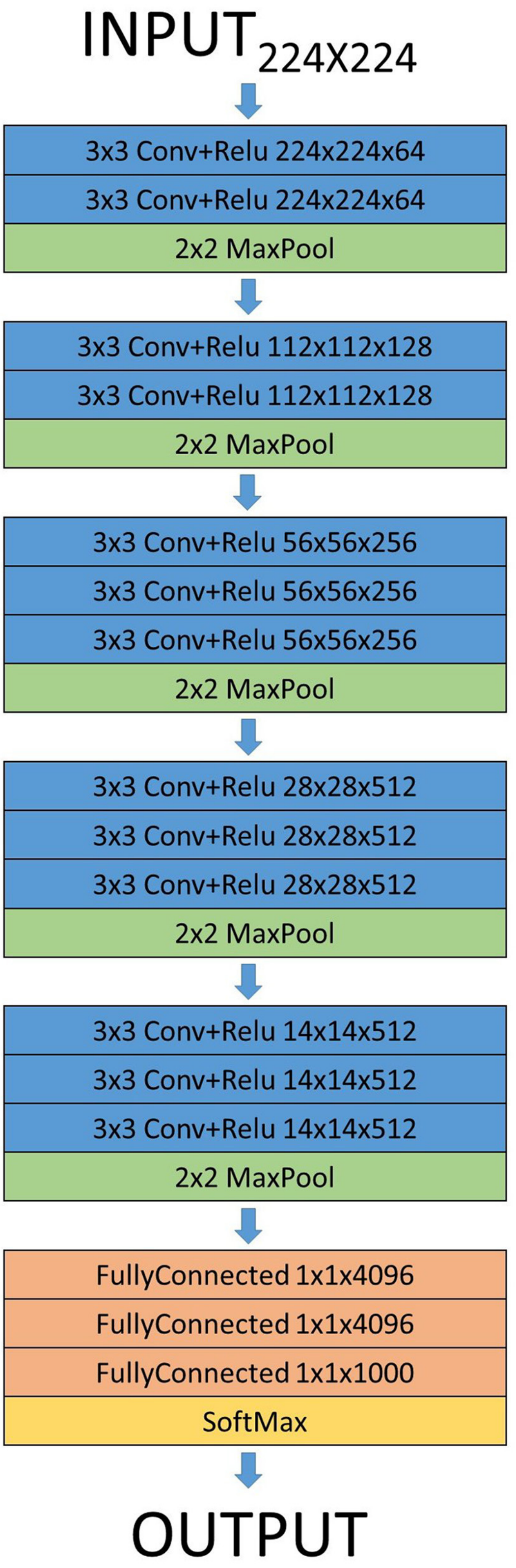

FIGURE 2
VGG16 CNN architecture. An input image of size 224 × 224 is supplied, followed by a sequence of convolutional and pooling layers, culminating in a single hidden layer comprising 4,096 neurons that feed into the output classification layer for the desired number of classes. When applying transfer learning, only the input layer and the final fully-connected layer are modified.

```
Input: models_list = ["ResNet152V2", "VGG16", "EfficientNetV2L", "Xception"]
classifiers_list = ["SVMlin",“SVMrbf", “RandomForest", "LDA", "QDA", “1-NN", "2-NN"...]

for i in [1,30] do
   trn_data, val_data, tst_data = create_dataset()
   for each CNN_MODEL in models_list do
      train(CNN_MODEL, trn_data, val_data)
      evaluate(CNN_MODEL, tst_data)
      FEAT_CNN_MODEL = remove_classification_FCN_layers(CNN_MODEL)
      trn_feat, val_feat, tst_feat = extract_features(FEAT_CNN_MODEL, trn_data, val_data, tst_data)
      for each CLASSIFICATION_MODEL in classifiers_list do
         CV_scores(CLASSIFICATION_MODEL, trn_feat, val_feat, tst_feat)
      end for
   end for
end for
```

Algorithm 1. Workflow for training and validation.

TABLE 1 Training hyperparameters summary: the batch size was chosen according to the memory constraints of the GPU.

	Optimizer	Loss function	Learning rate	Epochs	Batch size
Frozen layers	Adam	Categorical crossentropy	1e-3	35	64
Fine-tuning			1e-5	5	16

The selection of epoch values for both training phases was made following an analysis of the learning curves, where clear signs of overfitting were observed at those specific points. Notably, in the fine-tuning phase, an increase in the number of epochs had a substantial impact on the validation error.

Therefore, we have a total of 23 algorithms and 4 pre-trained CNN models, resulting in $23 \cdot 4 + 4 = 96$ distinct models. The remaining 4 models correspond to the original CNN+MLP (CNN with a classification layer based on a MLP) models that were trained separately.

2.3 Model training and evaluation

In this work, the training and evaluation of all models was developed in Python 3.10 using the scikit-learn package v1.2.2. From this package, four modules were used for modeling (tree, neural_networks, mixture, and ensemble), while the metrics module was used for performance calculations. Since the Transfer Learning technique has been used, the training stage is performed in two phases in order to improve the performance of the results. First, the convolutional layers of the model are frozen so that the training will affect just the Fully Connected (FC) layers. We use "categorical crossentropy" as the loss function and the Adam optimizer with its default hyperparameters of learning rate and decay (Goodfellow et al., 2016). The model is trained in 35 epochs and with a batch size of 64. The batch size has been selected to ensure optimal placement of the model within the GPU's shared memory. Using a larger batch size would risk memory overflow, while a smaller batch size tends to result in overfitting. The GPU is an NVIDIA GeForce RTX 3090Ti with 24GiB of memory, of which 22.4GiB is used for model storage.

Once the model has been fully trained with these parameters, a fine-tuning training phase is performed in order to adapt the whole network to this specific problem and increase its performance. In this stage, all the layers are unfrozen, so that the training changes all the weights of the model. However, a low learning rate, specifically 1e-5, is set to ensure only minor adjustments are made to the model's weights, avoiding drastic changes. Furthermore, as we are updating all layers, the size of the model in the GPU increases, necessitating a reduction in the batch size to 16. All these parameters are shown at Table 1. The selection of values is motivated by hyperparameter tuning through grid search, aiming to enhance the f-score and reduce the training time for all proposed CNN models.

To assess the performance of our model, we will focus on the concept of model evaluation, with a particular emphasis on its relevance to CNN models. We will delve into the common metric used to gauge the effectiveness of CNNs in various computer vision tasks, providing crucial insights into their object classification capabilities. By comprehending and interpreting these evaluation metric, we can make informed decisions concerning model selection, optimization, and deployment. The performance of our models will be quantified using the mean F_1-Score of all the species, which is defined by Equations 1–3:

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$F_1 Score = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{3}$$

TABLE 2 Multiple models are subjected to pairwise comparisons using the Games-Howell test.

Model$_1$	Model$_2$	Mean difference	Std. error	*t*-value	*p*-value	Upper limit	Lower limit
EfficientNetV2L	ResNet152V2	−0.0642	0.0017	26.5849	0.001	8.7991	−8.9275
EfficientNetV2L	VGG16	−0.0099	0.0022	3.1598	0.9	9.0081	−9.0278
EfficientNetV2L	Xception	−0.0634	0.0018	24.896	0.001	8.8531	−8.9798
ResNet152V2	VGG16	0.0543	0.0024	15.8547	0.001	8.922	−8.8135
ResNet152V2	Xception	0.0008	0.002	0.2722	0.9	8.8166	−8.815
VGG16	Xception	−0.0535	0.0025	15.2051	0.001	8.7947	−8.9017

The classification models based on CNN were compared using the four proposed pretrained networks. Models with a p-value of 0.9 are indistinguishable from each other, while those with a *p*-value of 0.001 are considered different. It is observed that the models grouped into two pairs, and these pairs are distinguishable from each other, as confirmed by the differences in their means values. The resulting pairs are EfficientNetV2L-VGG16 and ResNet152V2-Xception. Mean difference is computes as $\mu = \mu_2 - \mu_1$.

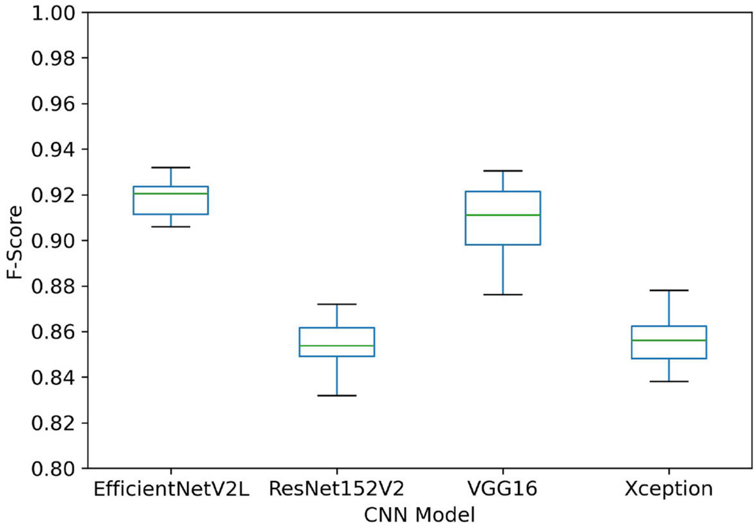

FIGURE 3
Sample distribution for F-Score results using CNN-MLP models. It is observed that EfficientNetV2L and VGG16 get better results, while ResNet152V2 and Xception results are not so competitive, in good agreement with the statistical analysis results. It is also evident that the variances of the distributions of F-score for each model are not equal (i.e., VGG16 clearly shows a more dispersed distribution than EfficientNetV2L).

Where *TP* stands for True Positives, *FP* for False Positives and *FN* as False Negatives.

For a correct evaluation of the proposed models, we carefully assembled a total of 30 distinct datasets, dividing each one into training, validation, and test sets. First, each model was trained using the training and validation sets and the F1 metric was then calculated using test data for each dataset. Finally, the average F1-score was obtained. This method enables us to cover a diverse array of scenarios, representing various sets of instances that can be fed into the network, and statistically demonstrate the performance of the models. Henceforth, the metrics presented for each experiment are the outcomes of 30 iterations of each proposed model.

3 Results and discussion

This section presents the results from 30 executions of the 96 proposed models. Among these, four are associated with the original CNN+MLP pretrained CNN models, while the remaining 92 pertain to the CNN+supervised learning proposed algorithms. The final objective of this work is to perform a well-founded comparison between the proposed models. To achieve this, we employ the comparative model analysis discussed in Pizarro et al. (2002). This paper introduces a novel approach to model selection that is based on hypothesis testing.

When comparing the models, the first thing that comes to mind is to determine whether there are statistically significant differences between the means. In this case, the usual approach is apply Analysis of Variance (ANOVA). However, ANOVA has a number of assumptions that should hold for valid inferences: normality, homoscedasticity(uniform variance across all groups) and independence of cases among others.

The Shapiro-Wilk test is a well-known test to evaluate whether the dataset is normally distributed within each model. If the p-value obtained exceeds the significance level, the null hypothesis (the population is normally distributed) cannot be rejected. This test is applied to each group, so the test generates a *p*-value for each of the models, allowing us to determine which models individually adhere to a normal distribution. With a significance level of 0.01, it was obtained that all p-values were greater than 0.01, and the null

TABLE 3 Games-Howell model comparison for the top CNN+MLP models, in comparison with their sub-models utilizing supervised learning algorithms for classification results.

Model$_1$	Model$_2$	Mean difference	Std. error	*t*-value	*p*-value	Upper limit	Lower limit
EfficientNetV2L	EfficientNetV2L-Decision tree	−0.1692	0.0014	86.5853	0.001	9.4111	−9.7494
EfficientNetV2L	EfficientNetV2L-GaussBayes	−0.0032	0.0012	1.882	0.9	9.7162	−9.7225
EfficientNetV2L	EfficientNetV2L-K-NN k = 1	−0.0312	0.0012	19.1629	0.001	9.7694	−9.8318
EfficientNetV2L	EfficientNetV2L-K-NN k = 3	−0.0251	0.0012	15.4471	0.001	9.7753	−9.8255
EfficientNetV2L	EfficientNetV2L-K-NN k = 5	−0.0193	0.0012	11.4817	0.001	9.6988	−9.7375
EfficientNetV2L	EfficientNetV2L-K-NN k = 7	−0.017	0.0012	9.8946	0.001	9.666	−9.7
EfficientNetV2L	EfficientNetV2L-K-NN k = 9	−0.0156	0.0012	9.0705	0.001	9.6659	−9.6971
EfficientNetV2L	EfficientNetV2L-K-NN k = 11	−0.0146	0.0012	8.5709	0.001	9.6807	−9.71
EfficientNetV2L	EfficientNetV2L-K-NN k = 13	−0.0139	0.0012	8.2055	0.001	9.6884	−9.7163
EfficientNetV2L	EfficientNetV2L-K-NN k = 15	−0.0134	0.0012	7.7069	0.9	9.6478	−9.6746
EfficientNetV2L	EfficientNetV2L-K-NN k = 17	−0.0129	0.0012	7.3586	0.9	9.6378	−9.6636
EfficientNetV2L	EfficientNetV2L-K-NN k = 19	−0.0121	0.0012	6.888	0.9	9.6425	−9.6666
EfficientNetV2L	EfficientNetV2L-K-NN k = 21	−0.0119	0.0012	6.7677	0.9	9.6379	−9.6617
EfficientNetV2L	EfficientNetV2L-K-NN k = 23	−0.0116	0.0012	6.5889	0.9	9.6351	−9.6583
EfficientNetV2L	EfficientNetV2L-K-NN k = 25	−0.0116	0.0012	6.5662	0.9	9.632	−9.6551
EfficientNetV2L	EfficientNetV2L-K-NN k = 27	−0.0111	0.0012	6.3741	0.9	9.6474	−9.6696
EfficientNetV2L	EfficientNetV2L-K-NN k = 29	−0.011	0.0012	6.3265	0.9	9.6473	−9.6693
EfficientNetV2L	EfficientNetV2L-LDA	**0.0314**	0.0012	18.8338	**0.001**	9.7713	−9.7085
EfficientNetV2L	EfficientNetV2L-Linear SVM	**0.0331**	0.0011	20.3917	**0.001**	9.8355	−9.7692
EfficientNetV2L	EfficientNetV2L-Poly SVM	**0.0306**	0.0011	19.1619	**0.001**	9.8828	−9.8215
EfficientNetV2L	EfficientNetV2L-QDA	−0.8589	0.001	612.2118	0.001	9.6248	−11.3426
EfficientNetV2L	EfficientNetV2L-RBF SVM	**0.0386**	0.0011	24.0823	**0.001**	9.8821	−9.805
EfficientNetV2L	EfficientNetV2L-RandForest	−0.35	0.0014	173.8726	0.001	9.2309	−9.9308
VGG16	VGG16-Decision Tree	−0.2188	0.0022	71.2309	0.001	9.7734	−10.2111
VGG16	VGG16-GaussBayes	−0.0691	0.0021	23.7781	0.001	10.1981	−10.3362
VGG16	VGG16-K-NN k = 1	−0.0786	0.002	27.2227	0.001	10.2245	−10.3817
VGG16	VGG16-K-NN k = 3	−0.0689	0.002	23.8557	0.001	10.2295	−10.3674
VGG16	VGG16-K-NN k = 5	−0.0636	0.002	22.1344	0.001	10.2718	−10.399
VGG16	VGG16-K-NN k = 7	−0.062	0.002	21.6174	0.001	10.2775	−10.4016
VGG16	VGG16-K-NN k = 9	−0.0615	0.002	21.4153	0.001	10.276	−10.399
VGG16	VGG16-K-NN k = 11	−0.0618	0.002	21.4104	0.001	10.2395	−10.3631
VGG16	VGG16-K-NN k = 13	−0.0619	0.002	21.4012	0.001	10.2281	−10.3519
VGG16	VGG16-K-NN k = 15	−0.0624	0.002	21.5769	0.001	10.2257	−10.3506
VGG16	VGG16-K-NN k = 17	−0.0629	0.0021	21.5605	0.001	10.1793	−10.3051
VGG16	VGG16-K-NN k = 19	−0.0634	0.0021	21.747	0.001	10.1819	−10.3087
VGG16	VGG16-K-NN k = 21	−0.0641	0.0021	21.9499	0.001	10.174	−10.3021
VGG16	VGG16-K-NN k = 23	−0.0647	0.0021	22.2589	0.001	10.1936	−10.323
VGG16	VGG16-K-NN k = 25	−0.0655	0.0021	22.4896	0.001	10.1827	−10.3137
VGG16	VGG16-K-NN k = 27	−0.066	0.0021	22.7513	0.001	10.209	−10.341

(Continued)

TABLE 3 (Continued)

Model_1	Model_2	Mean difference	Std. error	*t*-value	*p*-value	Upper limit	Lower limit
VGG16	VGG16-K-NN k = 29	−0.0669	0.0021	23.0788	0.001	10.2066	−10.3405
VGG16	VGG16-LDA	−0.0118	0.002	4.1593	0.9	10.4159	−10.4394
VGG16	VGG16-Linear SVM	−0.0169	0.0021	5.7614	0.9	10.2432	−10.2095
VGG16	VGG16-Poly SVM	−0.0065	0.0021	2.2003	0.9	10.1645	−10.1775
VGG16	VGG16-QDA	−0.574	0.002	198.0999	0.001	9.7063	−10.8544
VGG16	VGG16-RBF SVM	−0.0094	0.0021	3.2381	0.9	10.2645	−10.2833
VGG16	VGG16-RandForest	−0.4587	0.0024	134.1629	0.001	9.2351	−10.1526

Mean difference is computed as $\mu = \mu_2 - \mu_1$. Four sub-models demonstrate significantly superior performance when compared to their MLP counterparts (p-values and mean differences highlighted in bold). Models that utilize support vector machines or linear discriminant analysis algorithms have shown a significant performance improvement in comparison to their MLP versions.

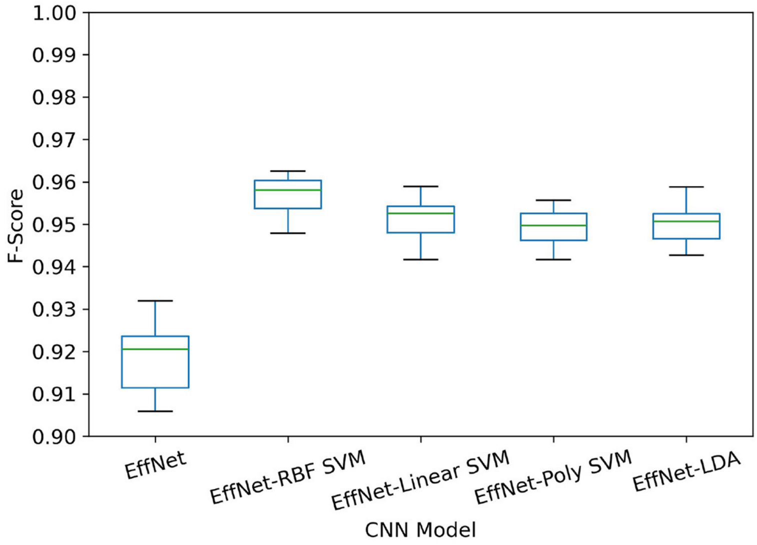

FIGURE 4
Sample distribution for F-score results using EfficientNetV2L model (shown as "EffNet"), CNN+MLP and four sub-models that exhibit significantly better performance than the MLP version (see Table 3) models are shown. Models employing SVM or LDA classification algorithms have shown improved performance compared to their MLP versions. This statistically proven improvement is clearly evident in the figure, where models utilizing the proposed supervised learning algorithms exhibit a higher mean and a more concentrated distribution of F-Score.

hypothesis for each group that the data are normally distributed could not be rejected.

The second tested assumption was the homogeneity of variances. In this case, the Levene test was conducted with the null hypothesis that the variances of all groups are equal. The Levene test applied to all groups gave a *p*-value of 3.4154e-70, leading to the rejection of the null hypothesis, indicating the presence of different variances among the groups.

Post hoc tests, such as Bonferroni (1936), Tukey (1949), Duncan (1955), Dunnett (1955), etc. Galindo et al. (2000) allow testing for differences between multiple group means while also controlling for the family-wise error rate, that is, the probability of at least one false conclusion in a series of hypothesis tests. However, most of these tests rely on the equal variance assumption. When the homoscedasticity assumption is not met, which was our case, alternative tests might be used. In such situation, the Games-Howell, Tamhane's T2, Dunnett's T3, and Dunnett's C tests can be applied (Shingala and Rajyaguru, 2015). The Games-Howell test is similar to Tukey's test, but it does not assume equal variances and sample sizes (provided that there are more than five samples in each group) (Games et al., 1979). In our case, this last prerequisite is also easily met, as there are 30 runs for each model. The Games-Howell used routine performs a pairwise comparison among groups which returns a p-value bounded between 0.001 and 0.9. This is due to the fact that the calculation of the *p*-value uses scalar minimization and results are bound to be between 0.001 and 0.9, as described in the documentation of `statsmodel` package (Seabold and Perktold, 2010). Therefore, a *p*-value = 0.001 in the results should be interpreted as *p*-value $\leq$ 0.001, and a *p*-value = 0.9 should be interpreted as *p*-value $\geq$ 0.9.

Consequently, a pairwise model comparison using Games-Howell routine was conducted among the CNN-MLP models (see Table 2) using four different pre-trained models (EfficientNetV2L, ResNet152V2, VGG16, and Xception). From the results, there is

TABLE 4 F-Score performance obtained by different models in 30 independent executions.

Model	μ	σ	max	min
EfficientNetV2L	0.91862	0.00727	0.93193	0.90588
EfficientNetV2L-Decision Tree	0.74943	0.00709	0.75856	0.73174
EfficientNetV2L-GaussBayes	0.91545	0.00488	0.92988	0.90443
EfficientNetV2L-K-NN k = 1	0.88745	0.00432	0.89655	0.8794
EfficientNetV2L-K-NN k = 11	0.904	0.00507	0.91381	0.89284
EfficientNetV2L-K-NN k = 13	0.90468	0.00501	0.91334	0.89342
EfficientNetV2L-K-NN k = 15	0.9052	0.0054	0.91557	0.89334
EfficientNetV2L-K-NN k = 17	0.9057	0.00551	0.91634	0.89361
EfficientNetV2L-K-NN k = 19	0.90657	0.00547	0.91738	0.89367
EfficientNetV2L-K-NN k = 21	0.90673	0.00552	0.91702	0.89479
EfficientNetV2L-K-NN k = 23	0.90702	0.00556	0.91802	0.8958
EfficientNetV2L-K-NN k = 25	0.90703	0.0056	0.91785	0.89571
EfficientNetV2L-K-NN k = 27	0.9075	0.00543	0.91868	0.89561
EfficientNetV2L-K-NN k = 29	0.90758	0.00543	0.91849	0.89621
EfficientNetV2L-K-NN k = 3	0.89349	0.00432	0.90196	0.88486
EfficientNetV2L-K-NN k = 5	0.89927	0.00488	0.90743	0.88971
EfficientNetV2L-K-NN k = 7	0.90162	0.00518	0.91354	0.8921
EfficientNetV2L-K-NN k = 9	0.90303	0.0052	0.91294	0.89165
EfficientNetV2L-LDA	**0.95003**	**0.00472**	**0.95882**	**0.94268**
EfficientNetV2L-Linear SVM	**0.95177**	**0.00431**	**0.95889**	**0.94166**
EfficientNetV2L-Poly SVM	**0.94923**	**0.00401**	**0.95564**	**0.94167**
EfficientNetV2L-QDA	0.0597	0.00059	0.06095	0.05848
EfficientNetV2L-RBF SVM	**0.9572**	**0.00406**	**0.96251**	**0.94792**
EfficientNetV2L-RandForest	0.56865	0.00752	0.5802	0.54518
VGG16	0.90873	0.01456	0.93036	0.87622
VGG16-Decision Tree	0.68988	0.00656	0.70418	0.67646
VGG16-GaussBayes	0.83967	0.00398	0.84483	0.83097
VGG16-K-NN k = 1	0.83014	0.00362	0.83743	0.82391
VGG16-K-NN k = 11	0.8469	0.00364	0.85531	0.84085
VGG16-K-NN k = 13	0.84681	0.00376	0.85553	0.83988
VGG16-K-NN k = 15	0.84629	0.00377	0.85517	0.83853
VGG16-K-NN k = 17	0.84585	0.00422	0.85406	0.8371
VGG16-K-NN k = 19	0.84534	0.00419	0.85413	0.83727
VGG16-K-NN k = 21	0.84466	0.00426	0.85388	0.83628
VGG16-K-NN k = 23	0.84399	0.00407	0.85157	0.8352
VGG16-K-NN k = 25	0.8432	0.00416	0.85097	0.8349
VGG16-K-NN k = 27	0.84274	0.0039	0.8509	0.83496
VGG16-K-NN k = 29	0.84178	0.00392	0.85016	0.83328
VGG16-K-NN k = 3	0.83981	0.00367	0.84739	0.83248
VGG16-K-NN k = 5	0.84516	0.00329	0.85145	0.83701
VGG16-K-NN k = 7	0.84669	0.00325	0.85294	0.8405
VGG16-K-NN k = 9	0.84725	0.00327	0.85503	0.84089
VGG16-LDA	0.89695	0.00218	0.90229	0.8932
VGG16-Linear SVM	0.92558	0.00437	0.93284	0.91601
VGG16-Poly SVM	0.90223	0.00488	0.9125	0.89286
VGG16-QDA	0.33471	0.00385	0.34262	0.3287
VGG16-RBF SVM	0.89933	0.00392	0.90939	0.8915
VGG16-RandForest	0.45	0.01019	0.47693	0.43398

Columns correspond to mean (μ), standard deviation (σ), maximum (max), and minimum (min) values. The EfficientNetV2L and VGG16 models are displayed, as they are considered statistically superior in F-Score performance compared to the other proposed models (seen at Table 2). Models whose performance is statistically indistinguishable from each other are highlighted. It can be seen that the highlighted models are distinctly set apart from the others in terms of performance; the minimum value achieved by these models surpasses the maximum of the remaining models by 0.01, as evidenced by the maximum achieved by EfficientNetV2L. Therefore, the results obtained from the Games-Howell (shown at Table 3) comparison are deemed sufficiently representative.

no evidence that EfficientNetV2L and VGG16 are significantly distinguishable from each other (p-value $\geq$ 0.9), as it happens with ResNet152V2 and Xception (p-value $\geq$ 0.9). However, these pairs are significantly distinguishable from each other(p-value $\leq$ 0.01), resulting in two distinct pairs of models, with one pair apparently outperforming the other (see Figure 3). The boxplot displays the sample distribution for each model. It is also notable that the models' variances are not equal; EfficientNetV2L and VGG16 exhibit notably different standard deviations, being VGG16 results more dispersed than those obtained by EfficientNetV2L.

Finally, the CNN-MLP models were compared with their sub-models using supervised learning algorithms for classification to determine which ones serve as the best classifiers. We validated models that, while demonstrating a better average than their MLP classifier counterparts, were significantly distinct from them. Therefore, it was observed that four sub-models had significantly better performance than the MLP version (see Table 3 and Figure 4). Models employing SVM or LDA algorithms have shown improved performance compared to their MLP versions, with an average F-Score enhancement of 0.03. Furthermore, these four models are deemed significantly distinguishable from the MLP model while also being statistically indistinguishable from each other.

The F-Score performance of various models was assessed through 30 independent executions (Tables 4, 5). Notably, the EfficientNetV2L and VGG16 models are presented as they are deemed statistically superior in F-Score performance compared to the other proposed models (refer to Table 2). Models with statistically indistinguishable performance are emphasized. It can be seen that these highlighted models distinctly outperform the others; the minimum value achieved by these models exceeds the maximum of the remaining models by 0.01, as demonstrated by the maximum achieved by EfficientNetV2L. Consequently, the results obtained from the Games-Howell comparison (refer to Table 3) are considered representative proposed models' performance.

TABLE 5 F-Score performance obtained by different models in 30 independent executions.

Model	μ	σ	*max*	*min*
ResNet152V2	0.85446	0.01022	0.87183	0.83194
ResNet152V2-Decision Tree	0.60688	0.00578	0.61867	0.59539
ResNet152V2-GaussBayes	0.84648	0.00414	0.85391	0.83523
ResNet152V2-K-NN k = 1	0.79454	0.00471	0.80279	0.78353
ResNet152V2-K-NN k = 11	0.80992	0.00515	0.81805	0.79556
ResNet152V2-K-NN k = 13	0.80894	0.0055	0.81714	0.79605
ResNet152V2-K-NN k = 15	0.80781	0.00553	0.81715	0.79499
ResNet152V2-K-NN k = 17	0.80645	0.00572	0.81576	0.79188
ResNet152V2-K-NN k = 19	0.80472	0.00583	0.81485	0.78915
ResNet152V2-K-NN k = 21	0.80306	0.00552	0.8128	0.78939
ResNet152V2-K-NN k = 23	0.80169	0.006	0.81181	0.78645
ResNet152V2-K-NN k = 25	0.79998	0.00599	0.80972	0.7848
ResNet152V2-K-NN k = 27	0.79859	0.00613	0.80955	0.78396
ResNet152V2-K-NN k = 29	0.79705	0.00603	0.80752	0.78369
ResNet152V2-K-NN k = 3	0.8001	0.00485	0.80922	0.78911
ResNet152V2-K-NN k = 5	0.80874	0.00414	0.81663	0.79916
ResNet152V2-K-NN k = 7	0.81048	0.00507	0.81862	0.79877
ResNet152V2-K-NN k = 9	0.81039	0.00518	0.81812	0.79682
ResNet152V2-LDA	**0.9353**	**0.0021**	**0.93925**	**0.93018**
ResNet152V2-Linear SVM	**0.93648**	**0.00224**	**0.94016**	**0.9323**
ResNet152V2-Poly SVM	0.929	0.00271	0.93479	0.92453
ResNet152V2-QDA	0.0711	0.00196	0.07545	0.0685
ResNet152V2-RBF SVM	**0.93147**	**0.00233**	**0.93585**	**0.9266**
ResNet152V2-RandForest	0.37039	0.00554	0.38028	0.36217
Xception	0.85525	0.01085	0.87798	0.83806
Xception-Decision Tree	0.53283	0.00458	0.5424	0.52474
Xception-GaussBayes	0.81987	0.00302	0.82475	0.81359
Xception-K-NN k = 1	0.76913	0.00326	0.77555	0.76229
Xception-K-NN k = 11	0.79191	0.00379	0.80112	0.78491
Xception-K-NN k = 13	0.79135	0.00383	0.80062	0.78518
Xception-K-NN k = 15	0.79049	0.00403	0.80017	0.78353
Xception-K-NN k = 17	0.78897	0.00419	0.798	0.78109
Xception-K-NN k = 19	0.78763	0.00426	0.79704	0.78025
Xception-K-NN k = 21	0.78612	0.0041	0.79439	0.77685
Xception-K-NN k = 23	0.78461	0.00455	0.7949	0.77488
Xception-K-NN k = 25	0.783	0.00425	0.79157	0.77484
Xception-K-NN k = 27	0.78136	0.00452	0.79181	0.77156
Xception-K-NN k = 29	0.77999	0.00454	0.79012	0.7696
Xception-K-NN k = 3	0.77678	0.00378	0.78602	0.7699
Xception-K-NN k = 5	0.78817	0.00358	0.79674	0.78135
Xception-K-NN k = 7	0.79136	0.00408	0.80258	0.78244
Xception-K-NN k = 9	0.79222	0.00366	0.80125	0.78505
Xception-LDA	0.91502	0.00241	0.91826	0.91084
Xception-Linear SVM	0.92613	0.00269	0.93137	0.92122
Xception-Poly SVM	0.92063	0.00201	0.92478	0.91646
Xception-QDA	0.0743	0.00195	0.07785	0.07126
Xception-RBF SVM	0.91806	0.00204	0.92256	0.91471
Xception-RandForest	0.33876	0.00349	0.34749	0.32937

Columns correspond to mean (μ), standard deviation (σ), maximum (*max*), and minimum (*min*) values. The ResNet152V2 and Xception models are displayed, they are considered statistically inferior in F-Score performance compared to the other proposed models (seen at Table 2). Models whose performance is statistically indistinguishable from each other are highlighted. Although it is seen that they are statistically superior, there are other models that, while achieving decent results, are indistinguishable from the highlighted models and other models that are statistically inferior to the top-performing models. Consequently, they have not been highlighted as they are surpassed by the highlighted models.

4 Conclusions

In this research paper, we have tackled the challenging task of automating the labeling of fish species through the application of deep learning techniques. Specifically, we examined the efficacy of four different pre-trained neural networks, including ResNet, VGG16, EfficientNetV2L, and Xception, using transfer learning. After transfer learning, we harnessed the knowledge these networks had gained from large-scale datasets and fine-tune them to our specific fish image dataset.

The initial phase of our investigation involved transfer learning, where all the pre-trained layers were kept frozen, allowing only the last layer to be customized to our dataset. Subsequently, we employed fine-tuning, which permitted us to update the pre-trained layers. This two-step approach aimed to leverage the powerful representations learned by these networks while adapting them to the nuances of our fish image dataset.

To further enhance the performance of our system, we decided to replace the final layers of the pre-trained networks with 23 distinct classification models, including Support Vector Machines (SVM), Linear Discriminant Analysis (LDA), Random Forests, and k-Nearest Neighbors (K-NN), among others. This diverse set of classifiers allowed us to explore which model might be most suitable for the classification task using the features extracted from the pre-trained CNN.

To determine the best-performing model among the extensive pool of candidates, we conducted a rigorous comparative analysis. It's worth noting that we encountered a deviation from the homoscedasticity assumption in our data. To address this issue, we applied the Games-Howell method, which is an improved and robust alternative to the Tukey-Kramer method. The Games-Howell method was specifically selected because of its suitability for scenarios where the homoscedasticity assumption is violated.

Our comparative analysis, rooted in the Games-Howell test, has demonstrated that not all pre-trained networks are created equal when it comes to fish species recognition. Specifically, EfficientNet and VGG16 emerged as the top performers, significantly outshining ResNet and Xception in our dataset. This highlights the importance of carefully selecting a pre-trained network that aligns with the specific nuances of the task at hand, and it underscores that not all deep learning architectures are universally applicable.

Moreover, our exploration of the final stage of the CNN architecture revealed a promising strategy for performance enhancement. The integration of classifiers like SVM and LDA as the concluding layer of the CNN framework led to a substantial improvement in the F-score, namely, from 0.92 to 0.95. This result suggests that the incorporation of sophisticated classifiers can serve as a powerful tool to boost the accuracy and reliability of CNNs in image classification tasks. This strategy should certainly be considered and explored in similar cases, opening up opportunities for enhancing the capabilities of deep learning models in a variety of applications.

Lastly, the comparison between the CNN-MLP models and their sub-models employing supervised learning algorithms unveiled a set of four models that exhibited significantly superior performance to their MLP counterparts. The integration of support vector machines and linear discriminant analysis algorithms led to an average F-score enhancement of 0.03, showcasing the potential of these classifiers in image classification tasks.

We may conclude that our study provides valuable insights into the intricate world of deep learning and image classification, emphasizing the importance of model selection, the strategic integration of classifiers, and the careful consideration of statistical testing techniques. These findings not only contribute to the field of automatic fish species labeling but also offer a roadmap for researchers in diverse domains seeking to leverage deep learning and statistical analysis to enhance their own classification tasks. This work paves the way for more robust and efficient approaches to image classification and serves as a foundation for further exploration in the realm of computer vision and machine learning.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Ethics statement

Ethical approval was not required for the study involving animals in accordance with the local legislation and institutional requirements because the research was made on photographs obtained at fish market.

Author contributions

JJ: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Project administration, Resources, Software, Validation, Visualization, Writing—original draft, Writing—review & editing. GB-G: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Resources, Software, Validation, Visualization, Writing—original draft, Writing—review & editing. JC-G: Investigation, Validation, Writing—review & editing. RC-C: Funding acquisition, Investigation, Resources, Validation, Writing—review & editing. PG: Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing—original draft, Writing—review & editing.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. The work was supported by "Ministerio de Agricultura, Pesca y Alimentación–Fondos NextGenerationEU" (DIGIPESCA Project) and "Junta de Andalucía–Grupos PAI" (TIC-145 and RNM-243 Research Groups).

Acknowledgments

The authors would like to thank the Lonja de Conil (fish market) for providing the data used in this research.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

Allken, V., Handegard, N. O., Rosen, S., Schreyeck, T., Mahiout, T., and Malde, K. (2019). Fish species identification using a convolutional neural network trained on synthetic data. *ICES J. Mar. Sci.* 76, 342–349. doi: 10.1093/icesjms/fsy147

Barbedo, J. G. A. (2019). Plant disease identification from individual lesions and spots using deep learning. *Biosyst. Eng.* 180, 96–107. doi: 10.1016/j.biosystemseng.2019.02.002

Bonferroni, C. E. (1936). Teoria statistica delle classi e calcolo delle probabilita. *Pubbl. R. Ist. Super. di Sci. Econom. Commer. Firenze.* 8, 3–62.

Buslaev, A., Iglovikov, V. I., Khvedchenya, E., Parinov, A., Druzhinin, M., and Kalinin, A. A. (2020). Albumentations: fast and flexible image augmentations. *Information* 11:125. doi: 10.3390/info11020125

Chollet, F. (2017). "Xception: deep learning with depthwise separable convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 1251–1258. doi: 10.1109/CVPR.2017.195

Cutler, D. R., Edwards Jr, T. C., Beard, K. H., Cutler, A., Hess, K. T., Gibson, J., et al. (2007). Random forests for classification in ecology. *Ecology* 88, 2783–2792. doi: 10.1890/07-0539.1

Deep, B. V., and Dash, R. (2019). "Underwater fish species recognition using deep learning techniques," in *2019 6th International Conference on Signal Processing and Integrated Networks (SPIN)* (IEEE), 665–669. doi: 10.1109/SPIN.2019.8711657

Dobeson, A. (2016). Scopic valuations: how digital tracking technologies shape economic value. *Econ. Soc.* 45, 454–478. doi: 10.1080/03085147.2016.1224143

Duncan, D. B. (1955). Multiple range and multiple f tests. *Biometrics* 11, 1–42. doi: 10.2307/3001478

Dunnett, C. W. (1955). A multiple comparison procedure for comparing several treatments with a control. *J. Am. Stat. Assoc.* 50, 1096–1121. doi: 10.1080/01621459.1955.10501294

Franco, M., Seeber, R., Sferlazzo, G., and Leardi, R. (1990). Classification and prediction ability of pattern recognition methods applied to sea-water fish. *Analyt. Chim. Acta* 233, 143–147. doi: 10.1016/S0003-2670(00)83471-6

Galindo, P. L., Pizarro-Junquera, J., and Guerrero, E. (2000). "Multiple comparison procedures for determining the optimal complexity of a model," in *Advances in Pattern Recognition: Joint IAPR International Workshops SSPR 2000 and SPR 2000 Alicante, Spain, August 30-September 1, 2000 Proceedings* (Springer), 796–805. doi: 10.1007/3-540-44522-6_82

Games, P. A., Keselman, H. J., and Clinch, J. J. (1979). Tests for homogeneity of variance in factorial designs. *Psychol. Bull.* 86:978. doi: 10.1037//0033-2909.86.5.978

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. London: MIT press.

He, K., Zhang, X., Ren, S., and Sun, J. (2016a). "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 770–778. doi: 10.1109/CVPR.2016.90

He, K., Zhang, X., Ren, S., and Sun, J. (2016b). "Identity mappings in deep residual networks," in *Computer Vision-ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV* (Springer), 630–645. doi: 10.1007/978-3-319-46493-0_38

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. (2017). "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 4700–4708. doi: 10.1109/CVPR.2017.243

Hussain, M., Bird, J. J., and Faria, D. R. (2019). "A study on cnn transfer learning for image classification," in *Advances in Computational Intelligence Systems: Contributions Presented at the 18th UK Workshop on Computational Intelligence, September 5–7, 2018, Nottingham, UK* (Springer), 191–202. doi: 10.1007/978-3-319-97982-3_16

Ibraheam, M., Li, K. F., Gebali, F., and Sielecki, L. E. (2021). A performance comparison and enhancement of animal species detection in images with various R-CNN models. *AI* 2, 552–577. doi: 10.3390/ai2040034

Jarek, K., and Mazurek, G. (2019). Marketing and artificial intelligence. *Central Eur. Bus. Rev.* 8, 46–55. doi: 10.18267/j.cebr.213

Kaya, A., Keceli, A. S., Catal, C., Yalic, H. Y., Temucin, H., and Tekinerdogan, B. (2019). Analysis of transfer learning for deep neural network based plant classification models. *Comput. Electr. Agric.* 158, 20–29. doi: 10.1016/j.compag.2019.01.041

Knauer, U., von Rekowski, C. S., Stecklina, M., Krokotsch, T., Pham Minh, T., Hauffe, V., et al. (2019). Tree species classification based on hybrid ensembles of a convolutional neural network (CNN) and random forest classifiers. *Rem. Sens.* 11, 2788. doi: 10.3390/rs11232788

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems* 25.

Luan, J., Zhang, C., Xu, B., Xue, Y., and Ren, Y. (2020). The predictive performances of random forest models with limited sample size and different species traits. *Fisher. Res.* 227:105534. doi: 10.1016/j.fishres.2020.105534

Montalbo, F. J., and Hernandez, A. (2019). "Classification of fish species with augmented data using deep convolutional neural network," in *2019 IEEE 9th International Conference on System Engineering and Technology (ICSET)* (IEEE), 396–401. doi: 10.1109/ICSEngT.2019.8906433

Munoz, F., Pennino, M. G., Conesa, D., Lopez-Quilez, A., and Bellido, J. M. (2013). Estimation and prediction of the spatial occurrence of fish species using bayesian latent gaussian models. *Stochastic Environ. Res. Risk Assess.* 27, 1171–1180. doi: 10.1007/s00477-012-0652-3

Norouzzadeh, M. S., Nguyen, A., Kosmala, M., Swanson, A., Palmer, M. S., Packer, C., et al. (2018). Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning. *Proc. Natl. Acad. Sci.* 115, E5716–E5725. doi: 10.1073/pnas.1719367115

Nuraini, R. (2022). Identification of freshwater fish types using linear discriminant analysis (lda) algorithm. *IJICS* 6, 147–154. doi: 10.30865/ijics.v6i3.5565

Palmer, M., Álvarez Ellacuría, A., Moltó, V., and Catalán, I. A. (2022). Automatic, operational, high-resolution monitoring of fish length and catch numbers from landings using deep learning. *Fisher. Res.* 246:106166. doi: 10.1016/j.fishres.2021.106166

Pizarro, J., Guerrero, E., and Galindo, P. L. (2002). Multiple comparison procedures applied to model selection. *Neurocomputing* 48, 155–173. doi: 10.1016/S0925-2312(01)00653-1

Pundlik, R. (2016). "Comparison of sensitivity for consumer loan data using gaussian naïve bayes (gnb) and logistic regression (lr)," in *2016 7th International Conference on Intelligent Systems, Modelling and Simulation (ISMS)* (IEEE), 120–124. doi: 10.1109/ISMS.2016.57

Saberioon, M., Císař, P., Labbé, L., Souček, P., Pelissier, P., and Kerneis, T. (2018). Comparative performance analysis of support vector machine, random forest, logistic regression and k-nearest neighbours in rainbow trout (oncorhynchus mykiss) classification using image-based features. *Sensors* 18:1027. doi: 10.3390/s18041027

Seabold, S., and Perktold, J. (2010). "Statsmodels: econometric and statistical modeling with python," in *Proceedings of the 9th Python in Science Conference* (Austin, TX), 10–25080. doi: 10.25080/Majora-92bf1922-011

Shang, Y., and Li, J. (2018). "Study on echo features and classification methods of fish species," in *2018 10th International Conference on Wireless Communications and Signal Processing (WCSP)* (IEEE), 1–6. doi: 10.1109/WCSP.2018.8555591

Shingala, M. C., and Rajyaguru, A. (2015). Comparison of post hoc tests for unequal variance. *Int. J. New Technol. Sci. Eng.* 2, 22–33. Available online at: https://www.ijntse.com/upload/1447070311130.pdf

Shorten, C., and Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *J. Big Data* 6, 1–48. doi: 10.1186/s40537-019-0197-0

Simonyan, K., and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., et al. (2014). "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 1–9. doi: 10.1109/CVPR.2015.7298594

Tan, M., and Le, Q. (2021). "Efficientnetv2: smaller models and faster training," in *International Conference on Machine Learning* (PMLR), 10096–10106.

Tukey, J. W. (1949). Comparing individual means in the analysis of variance. *Biometrics* 5, 99–114. doi: 10.2307/3001913