Science in Information Technology Letters
Vol. 5., No. 1, May 2024, pp. 16-25
ISSN 2722-4139
http://pubs2.ascee.org/index.php/sitech

Betta fish classification using transfer learning and fine-tuning of CNN models

Rihwan Munif [a,1], Adhi Prahara [a,2,*]

[a] Informatics Department, Universitas Ahmad Dahlan, Indonesia
[1] rihwan1900018415@webmail.uad.ac.id; [2]adhi.prahara@tif.uad.ac.id
* Corresponding Author

ARTICLE INFO

Article history
Received 1 April, 2024
Revised 8 April, 2024
Accepted 17 April, 2024

Keywords
Betta fish
Classification
Transfer learning
CNN
Computer vision

ABSTRACT

Betta fish, known as freshwater fighters, are in demand because of their beauty and characteristics. These betta fish such as Crowntail, Halfmoon, Doubletail, Spadetail, Plakat, Veiltail, Paradise, and Rosetail are hard to recognize without knowledge about them. Therefore, transfer learning of Convolutional Neural Network models was proposed to classify the betta fish from the image. The transfer learning process used a pre-trained model from ImageNet of VGG16, MobileNet, and InceptionV3 and fine-tuned the models on the betta fish dataset. The models were trained on 461 images, validated with 154 images, and tested on 156 images. The result shows that the InceptionV3 model excels with 0.94 accuracies compared to VGG16 and MobileNet which acquire 0.93 and 0.92 accuracy respectively. With good accuracy, the trained model can be used in betta fish recognition applications to help people easily identify betta fish from the image.

This is an open access article under the CC–BY-SA license.

1. Introduction

Betta fish is a freshwater fish that is known for its activeness and also aggressiveness because it tends to attack its fellow males. Betta fish have beautiful tail variations, and attractive body shapes with calm and graceful movements. Many people are interested in keeping betta fish as ornamental fish or as fighting fish. The original habitat of betta fish is spread across Southeast Asia, especially Indonesia. These fish are often found in swamps, lakes, and ponds. Most hobbyists recognize betta fish by various names, such as Crowntail, Doubletail, Paradise, Halfmoon, Plakat, Spadetail, Veiltail, and Rosetail. The name is given based on its shape. For example, the Halfmoon fish has fins that resemble a semicircle, while the Crowntail fish has jagged and pointed fins like a comb [1]. Betta fish also has alluring colors that come from the pigment cells (chromatophores) on its skin. The color of the betta fish's body, especially its pattern, can function as a type identifier as well as to protect itself from predators [1].

Betta fish attracts the attention of many ornamental fish fans because of the beautiful colors displayed by their bodies. This causes a high demand for betta fish in the market. Most people distinguish the types of betta fish based on their tails, color, and body shape. However, it is difficult for people without a decent knowledge of betta fish to classify them because of their variations [2]. One of the solutions is to use computer vision technology to make it easier to classify betta fish from images [1], [3].

Convolutional Neural Network (CNN) [4]–[8] is a type of artificial neural network that is specifically designed to process pixel data and visual images so that it is suitable for image classification. Various types of CNN models such as VGG16 [9]–[11], MobileNet [12]–[16], ResNet [17]–[19], Inception [20], and so on, already have pre-trained models available from the ImageNet [21] dataset. The pre-trained model is usually used for transfer learning in a smaller dataset. In this study, transfer learning and fine-tuning of CNN models were used for image classification of betta fish. The rest of the paper is explained as follows: Section 2 presents the proposed method, Section 3 presents the experiments, results, and discussion and the conclusion of this work is presented in Section 4.

2. The Proposed Method

In this research, a betta fish classification model was proposed. The general procedure of the proposed model is shown in Fig. 1. Based on Fig. 1, the model takes betta fish images as input and then performs pre-processing followed by augmentation. The training step involves transfer learning and fine-tuning CNN models to generate a trained model. In the test step, the model predicts the category of betta fish such as Crowntail, Halfmoon, Doubletail, Spadetail, Plakat, Veiltail, Paradise, and Rosetail using the trained model. The detail of each step is explained in the following sections.

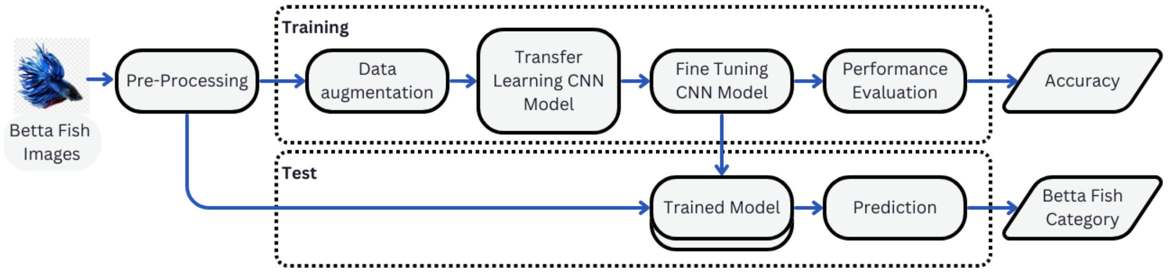


Fig. 1. The general procedure of the proposed betta fish classification method

2.1. Pre-Processing and Data Augmentation

Pre-processing is done by resizing the image to 224x224x3 with zero padding to maintain the ratio of the original image and rescaling the image pixel into the range of [0, 1] to fit the input of pre-trained CNN models. The dataset is then broken down into training data, validation data, and test data. Data augmentation uses various image transformation operations such as horizontal flip and random rotation to increase the quantity and variation in the dataset. Augmentation is only applied to training data and validation data. After augmentation, the dataset will be grouped into batches for training.

2.2. Transfer Learning and Fine-Tuning of CNN Models

Transfer learning is a machine learning approach where a pre-trained model can be used to solve other problems in similar or different domains. In transfer learning, a base model that already has general knowledge from large datasets such as ImageNet is used as a feature extractor. The classification head is added on top of it and trained on a smaller dataset to perform a specific task. Usually, all layers in the base model will be frozen, meaning that the layers will not be trained. This is done because these initial layers already have good general representations of common images. The training process only applied to the classification layers. With transfer learning, training time and resources can be reduced, as well as improving model performance on limited datasets [22]. In this research, transfer learning is performed using the pre-trained models from the ImageNet dataset such as VGG16, MobileNet, and InceptionV3 provided by the TensorFlow library.

VGG16 [10] model consists of a total of 19 layers, with 16 convolutional layers and 3 fully connected layers. Convolution is used on the input image to extract important features from the image. After convolution, max pooling is performed to reduce the spatial dimensions of the features produced by convolution. Then, fully connected layers act as the final classification layer to determine the class of the input. MobileNet [12] is a CNN architecture developed specifically for image recognition tasks on mobile devices with limited computational resources. The MobileNet architecture utilizes a technique called depthwise separable convolution to reduce the number of parameters and computational operations required, making it more computationally efficient. MobileNet has remarkable performance in achieving a balance between high accuracy, fast execution, and efficient resource usage. InceptionV3 [20] utilizes complex Inception modules, consisting of various convolution operations with different filter sizes. Its ability to extract features at various scales and levels of detail allows the network to accurately identify and represent different image features at various levels. InceptionV3 consists of 5 basic convolutional layers (stem) where each convolution operation is followed by ReLU and BatchNormalization. This is followed by 11 inception modules which are designed with factorized convolutions.

Fine-tuning involves adjusting some of the layers of a previously trained model. The layers in the base model will be unfrozen to be adjusted in the training process. Typically, this involves training steps with a smaller learning rate to prevent large changes to the initial layers. By fine-tuning, the model can update the weights to adjust the feature representations learned by the base model to the specific characteristics of the new task dataset. This process improves the model's performance and makes it more suitable for the specific task.

2.3. Evaluation Metrics

The results are evaluated using a multi-class confusion matrix [23]. The confusion matrix is an instrument used in evaluating the performance of a classification model to analyze the predicted results against the actual data. Based on the confusion matrix, we can compute accuracy, precision, recall, and F1-score to analyze the performance as shown in Equations (1), (2), (3), and (4) respectively where TP is True Positive, TN is True Negative, FN is False Negative, and FP is False Positive.

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \tag{1}$$

$$Precision = \frac{TP}{TP+FP} \tag{2}$$

$$Recall = \frac{TP}{TP+FN} \tag{3}$$

$$F1\ Score = 2\left(\frac{precision \cdot recall}{precision+recall}\right) \tag{4}$$

3. Results and Discussion

This research proposed a betta fish image classification method using transfer learning of CNN models. The proposed method was written in Python and trained on Google Colab using GPU. The pre-trained CNN models such as VGG16, MobileNet, and InceptionV3 were obtained from the TensorFlow library. The experiment and performance evaluation results are explained in the following sections.

3.1. Dataset

The dataset consists of 8 categories of betta fish e.g. Crowntail, Doubletail, Halfmoon, Paradise, Plakat, Rosetail, Spadetail, and Veiltail. The sample of betta fish images is shown in Fig. 2. The dataset was collected by searching betta fish images from the internet and having them verified by a betta fish expert. The total number of images in the dataset is 771. The data is divided into 461 training images, 154 validation images, and 156 test images.

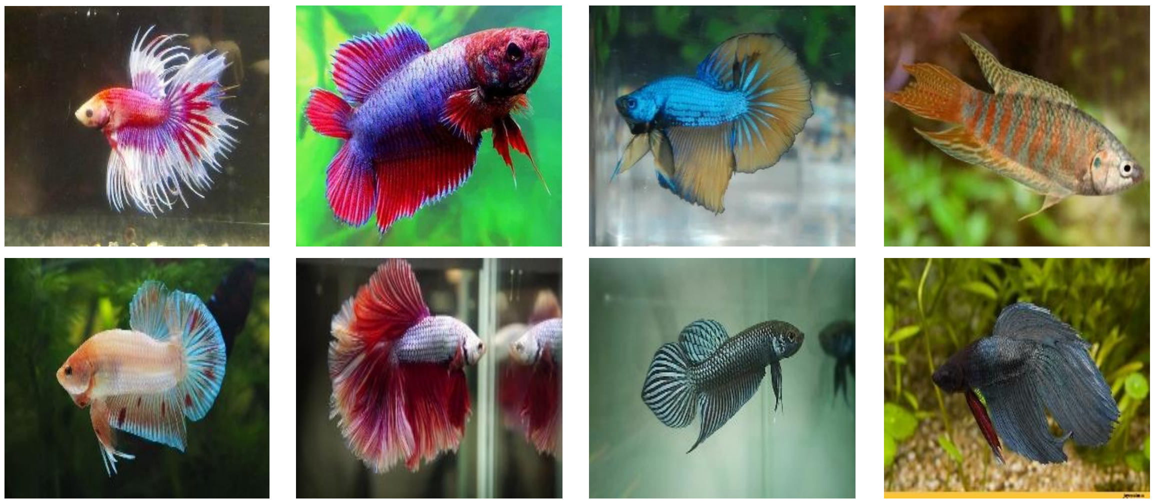


Fig. 2. Sample of betta fish images

[a] (source: the images taken from the internet may be subject to copyright)

3.2. Pre-Processing and Augmentation Result

Preprocessing is done to resize all images in the dataset to 224x224x3 with zero padding, apply image transformation such as horizontal flipping and random rotation by 20 degrees, and rescale the pixel into a range of [0, 1]. The dataset is then grouped into batches with a size of 32. The result of data augmentation is shown in Fig. 3.

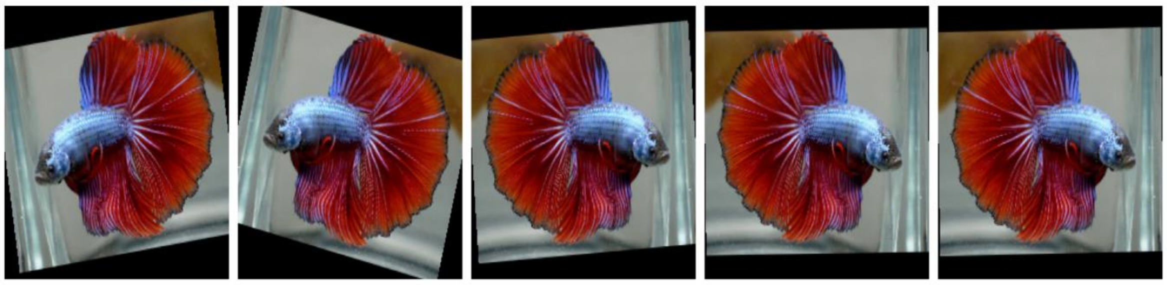


Fig. 3. The result of data augmentation

3.3. Transfer Learning and Fine-Tuning Result

The proposed method uses pre-trained VGG16, MobileNet, and InceptionV3 models from the ImageNet dataset as the base model. These base models are used as feature extractor layers. In the classification layers, we add dropout, one fully connected layer with ReLU activation function and L2 regularization followed by BatchNormalization and dropout. The last output layer is a fully connected layer with a softmax activation function. The architecture of the models is shown in Table 1 and the configuration of the hyperparameters that are applied to all models such as learning rate, optimizer, loss function, batch size, and epochs is shown in Table 2.

Table 1. The proposed model architectures

Layers	Configuration
Input	224, 224, 3
Base model	VGG16
freeze during transfer learning	MobileNet
unfreeze during fine-tuning	InceptionV3
Flatten/Global Average Pooling	Flatten for VGG16
	Global Avg Pooling for MobileNet and InceptionV3
Dropout	0.25
Fully connected	512, ReLU, L2 Regularization
BatchNormalization	
Dropout	0.5
Output	8, Softmax

Table 2. The configuration of hyperparameters

Hyperparameters	Value
Learning rate (transfer learning)	0.001
Learning rate (fine-tuning)	0.0001
Optimizer	Adam
Batch size	32
Loss function	Cross entropy
Evaluation metrics	Accuracy
Callback	Save the best model based on validation loss
Epoch	150 for both transfer learning and fine-tuning

The training process is divided into two phases namely transfer learning phase and fine-tuning phase. In the transfer learning phase, the model is trained for 150 epochs and monitored for the best performance on validation data according to the lowest validation loss. In this phase, only the classification layers are trained. The second training phase is fine-tuning the trained model for 150 epochs. In this phase, all layers are trained by unfreezing the base model to adjust the weights to the betta fish dataset with a lower learning rate.

The training graph of the VGG16 model is shown in Fig. 4 where (a) is the transfer learning phase and (b) is the fine-tuning phase. Transfer learning of the VGG16 model shows 0.9718 training accuracy, 0.5142 training loss, 0.9026 validation accuracy, and 0.8505 validation loss. The transfer learning phase has already shown a good result. Fine-tuning the VGG16 model shows 1.0000 training accuracy, 0.1602 training loss, 0.9026 validation accuracy, and 0.5747 validation loss. The higher the accuracy value and the lower the loss value are the desired result. Based on the result, it is clear that the validation loss is reduced during fine-tuning although the validation accuracy remains the same. The graph also seems unstable because of dropout and regularization applied to the model.

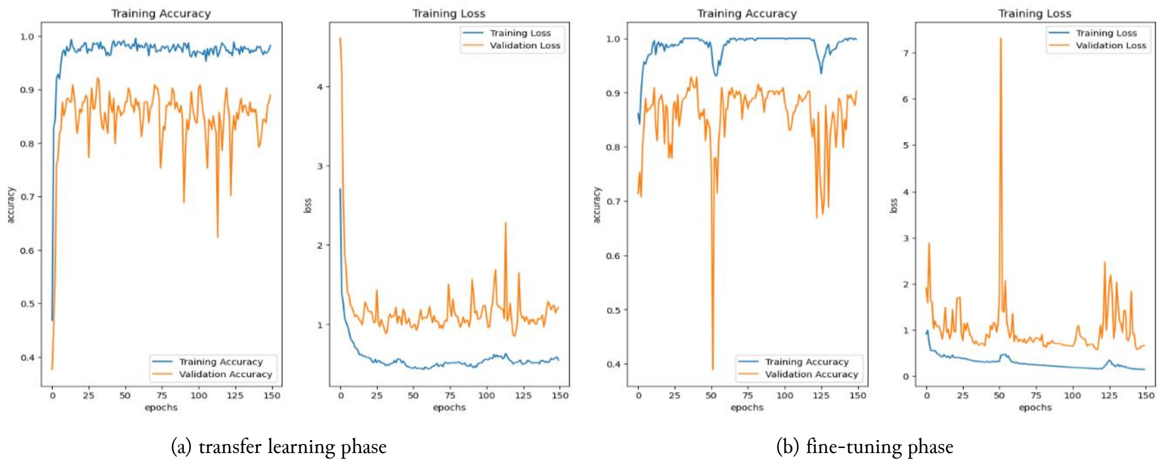


Fig. 4. Training graph of VGG16 model

The training graph of the MobileNet model is shown in Fig. 5 where (a) is the transfer learning phase and (b) is the fine-tuning phase. Transfer learning of the MobileNet model shows 0.9675 training accuracy, 0.3063 training loss, 0.8961 validation accuracy, and 0.6063 validation loss. This transfer learning result is similar to the VGG16 model. To further improve the performance, a fine-tuning phase is carried out with 150 epochs. After fine-tuning, the MobileNet model shows 0.9978 training accuracy, 0.1424 training loss, 0.9286 validation accuracy, and 0.5570 validation loss. The fine-tuning phase significantly improves the trained model and is better than the VGG16 model even with fewer model parameters.

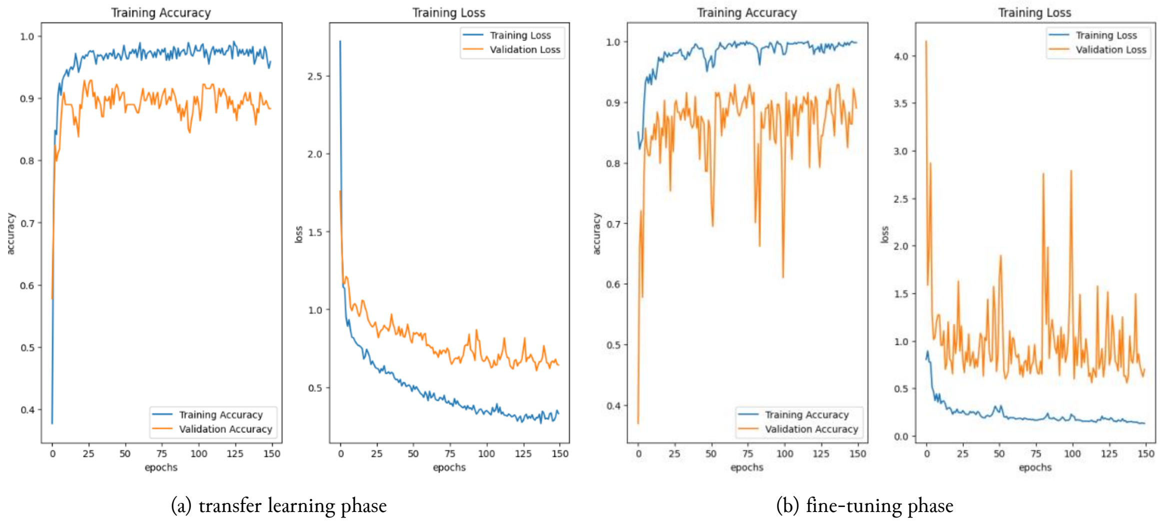
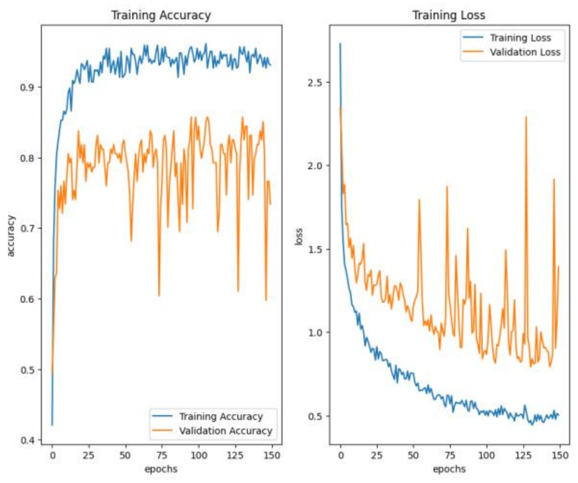


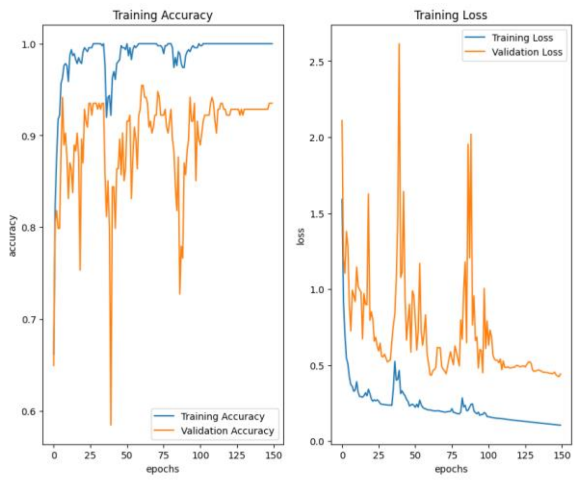
Fig. 5. Training graph of MobileNet model

The training graph of the InceptionV3 model is shown in Fig. 6 where (a) is the transfer learning phase and (b) is the fine-tuning phase. Transfer learning of the InceptionV3 model shows 0.9458 training accuracy, 0.4708 training loss, 0.8571 validation accuracy, and 0.7911 validation loss. The transfer learning result is the worst compared to VGG16 and MobileNet. However, after fine-tuning, the InceptionV3 model shows 1.0000 training accuracy, 0.1061 training loss, 0.9351 validation accuracy, and

0.4247 validation loss. Based on the result, the fine-tuning process on the InceptionV3 model adjusts the weights of each layer to the betta fish dataset better than the two previous models.



(a) transfer learning phase



(b) fine-tuning phase

Fig. 6. Training graph of InceptionV3 model

3.4. Performance Evaluation

Based on the trained models after transfer learning and fine-tuning phases, we test the performance on 156 test images. Table 3 shows the comparison of training results from VGG16, MobileNet, and InceptionV3 models. Based on Table 3, the InceptionV3 model performs better than the other models in the validation dataset during the fine-tuning phase with 0.9351 accuracy. During training for 150 epochs, the best model is saved according to the lowest validation loss. This scheme is used to prevent overfitting during training.

Table 3. Comparison of training performance of VGG16, MobileNet, and InceptionV3

Model	Transfer Learning Phase				Fine-Tuning Phase			
	Training accuracy	*Training loss*	*Validation accuracy*	*Validation loss*	*Training accuracy*	*Training loss*	*Validation accuracy*	*Validation loss*
VGG16	**0.9718**	0.5142	**0.9026**	0.8505	**1.0000**	0.1602	0.9026	0.5747
MobileNet	0.9675	**0.3063**	0.8961	**0.6063**	0.9978	0.1424	0.9286	0.5570
InceptionV3	0.9458	0.4708	0.8571	0.7911	**1.0000**	**0.1061**	**0.9351**	**0.4247**

Table 4 shows the result of inference on test images of each model. The result is evaluated using accuracy, precision, recall, and F1-Score. Based on Table 4, the result of the three models is similar but the InceptionV3 model achieves higher performance on all metrics with 0.94 test accuracy, 0.38 test loss, 0.95 precision, 0.94 recall, and 0.94 F1-score.

Table 4. Comparison performance of VGG16, MobileNet, and InceptionV3 on test data

Model	Accuracy	Loss	Precision	Recall	F1-Score
VGG16	0.93	0.54	0.93	0.93	0.93
MobileNet	0.92	0.57	0.93	0.92	0.92
InceptionV3	**0.94**	**0.38**	**0.95**	**0.94**	**0.94**

The confusion matrix of the InceptionV3 model on the test dataset is shown in Table 5. Based on Table 5, the InceptionV3 model performs better in the Crowntail, Paradise, and Plakat categories, moderately in Halfmoon, Spadetail, and Veiltail categories, and slightly worse in Doubletail and Rosetail categories. Some of the Rosetail images are recognized as Halfmoon and some of the Doubletail images are recognized as Plakat and Halfmoon. This failure may be caused by the position of the betta fish when captured making the shape of its tail difficult to be recognized.

Table 5. Confusion matrix of the InceptionV3 model

	Crowntail	Doubletail	Halfmoon	Paradise	Plakat	Rosetail	Spadetail	Veiltail
Crowntail	20	0	0	0	0	0	0	0
Doubletail	0	17	1	0	2	0	0	0
Halfmoon	0	0	19	0	0	1	0	0
Paradise	0	0	0	17	0	0	0	0
Plakat	0	0	0	0	20	0	0	0
Rosetail	0	0	3	0	0	16	0	0
Spadetail	0	0	0	0	0	0	19	1
Veiltail	1	0	0	0	0	0	0	19

4. Conclusion

A betta fish classification using transfer learning and fine-tuning of CNN models was proposed. The method performs pre-processing and data augmentation to the betta fish images. Pre-trained models such as VGG16, MobileNet, and InceptionV3 are used as the base model for feature extraction. The classification head consists of a fully connected layer. Transfer learning and fine-tuning of these models for 150 epochs produce trained models that are used to predict the test data. Based on the performance evaluation, the InceptionV3 model achieves 0.94 test accuracy, 0.38 test loss, 0.95 precision, 0.94 recall, and 0.94 F1-Score. These scores are better than the results of the VGG16 and MobileNet models. With this result, the proposed trained model can be used in the betta fish recognition applications to help people easily recognize the betta fish category from the images. For future works, the proposed transfer learning model can be improved to recognize more betta fish categories not only based on their shape but also their color, pattern, and purpose such as ornamental fish or fighting fish.

Declarations

Author contribution. All authors contributed equally to the main contributor to this paper. All authors read and approved the final paper.
Funding statement. None of the authors have received any funding or grants from any institution or funding body for the research.
Conflict of interest. The authors declare no conflict of interest.
Additional information. No additional information is available for this paper

References

[1] F. Shidiq, E. W. Hidayat, and N. I. Kurniati, "Application of K-nearest Neighbor (Knn) Method to Determine Cupang Fish Using Canny Edge Detection and Invariant Moment," *J. Tek. Inform.*, vol. 3, no. 1, pp. 11–20, 2022. [Online]. Available at: 10.37058/innovatics.v3i2.3093.

[2] C. C. F. Pleeging and C. P. H. Moons, "Potential welfare issues of the Siamese fighting fish (Betta splendens) at the retailer and in the hobbyist aquarium," *Vlaams Diergeneeskd. Tijdschr.*, vol. 86, no. 4, p. 213, Aug. 2017, doi: 10.21825/vdt.v86i4.16182.

[3] N. Rafi, Z. Zainuddin, and I. Nurtanio, “Betta Fish Classification Using Faster R-CNN Approach with Multi-Augmentation,” in *2024 International Seminar on Intelligent Technology and Its Applications (ISITIA)*, Jul. 2024, pp. 500–505, doi: 10.1109/ISITIA63062.2024.10668167.

[4] I. J. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available at: http://alvarestech.com/temp/deep/Deep%20Learning%20by%.

[5] D. M. Hibban and W. F. Al Maki, “Classification of Ornamental Betta Fish Using Convolutional Neural Network Method and Grabcut Segmentation,” in *2021 International Conference on Data Science and Its Applications (ICoDSA)*, Oct. 2021, pp. 102–109, doi: 10.1109/ICoDSA53588.2021.9617213.

[6] X. Lan, J. Bai, M. Li, and J. Li, “Fish Image Classification Using Deep Convolutional Neural Network,” in *Proceedings of the 2020 International Conference on Computers, Information Processing and Advanced Education*, Oct. 2020, pp. 18–22, doi: 10.1145/3419635.3419643.

[7] D. V, J. R, D. M, N. S, H. M, and S. Johnson, “Fish Classification Using Convolutional Neural Network,” in *2023 International Conference on Research Methodologies in Knowledge Management, Artificial Intelligence and Telecommunication Engineering (RMKMATE)*, Nov. 2023, pp. 1–4, doi: 10.1109/RMKMATE59243.2023.10369927.

[8] N. Hasan, S. Ibrahim, and A. Aqilah Azlan, “Fish diseases detection using convolutional neural network (CNN),” *Int. J. Nonlinear Anal. Appl.*, vol. 13, no. 1, pp. 1977–1984, Mar. 2022. [Online]. Available at: https://ijnaa.semnan.ac.ir/article_5839.html.

[9] K. Auliasari, M. Wasef, and M. Kertaningtyas, “Leveraging VGG16 for Fish Classification in a Large-Scale Dataset,” *Brill. Res. Artif. Intell.*, vol. 3, no. 2, pp. 316–328, Dec. 2023, doi: 10.47709/brilliance.v3i2.3270.

[10] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *3rd Int. Conf. Learn. Represent. (ICLR 2015)*, Apr. 2015. [Online]. Available at: http://www.robots.ox.ac.uk/.

[11] M.-H. Chen, T.-H. Lai, Y.-C. Chen, and T.-Y. Chou, “A Robust Fish Species Classification Framework: FRCNN-VGG16-SPPNet,” Apr. 2023, doi: 10.21203/RS.3.RS-2825927/V1.

[12] A. G. Howard *et al.*, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” no. April 2017, 2017, [Online]. Available at: http://arxiv.org/abs/1704.04861.

[13] P. D. Hung and N. N. Kien, “SSD-Mobilenet Implementation for Classifying Fish Species,” in *Advances in Intelligent Systems and Computing*, vol. 1072, Springer, Cham, 2020, pp. 399–408, doi: 10.1007/978-3-030-33585-4_40.

[14] E. Suharto, Suhartono, A. P. Widodo, and E. A. Sarwoko, “The use of mobilenet v1 for identifying various types of freshwater fish,” *J. Phys. Conf. Ser.*, vol. 1524, no. 1, p. 012105, Apr. 2020, doi: 10.1088/1742-6596/1524/1/012105.

[15] X. Liu, Z. Jia, X. Hou, M. Fu, L. Ma, and Q. Sun, “Real-time Marine Animal Images Classification by Embedded System Based on Mobilenet and Transfer Learning,” in *OCEANS 2019 - Marseille*, Jun. 2019, vol. 2019-June, pp. 1–5, doi: 10.1109/OCEANSE.2019.8867190.

[16] G. Yu, L. Wang, M. Hou, Y. Liang, and T. He, “An adaptive dead fish detection approach using SSD-MobileNet,” in *2020 Chinese Automation Congress (CAC)*, Nov. 2020, pp. 1973–1979, doi: 10.1109/CAC51589.2020.9326648.

[17] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-December, pp. 770–778, Dec. 2016, doi: 10.1109/CVPR.2016.90.

[18] M. Mathur and N. Goel, “FishResNet: Automatic Fish Classification Approach in Underwater Scenario,” *SN Comput. Sci.*, vol. 2, no. 4, p. 273, Jul. 2021, doi: 10.1007/s42979-021-00614-8.

[19] X. Xu, W. Li, and Q. Duan, “Transfer learning and SE-ResNet152 networks-based for small-scale unbalanced fish species identification,” *Comput. Electron. Agric.*, vol. 180, p. 105878, Jan. 2021, doi: 10.1016/j.compag.2020.105878.

[20] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, vol. 2016-Decem, pp. 2818–2826, doi: 10.1109/CVPR.2016.308.

[21] O. Russakovsky *et al.*, "ImageNet Large Scale Visual Recognition Challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015, doi: 10.1007/s11263-015-0816-y.

[22] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," in *Artificial Neural Networks and Machine Learning--ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III 27*, 2018, pp. 270–279, doi: 10.1007/978-3-030-01424-7_27.

[23] M. Grandini, E. Bagli, and G. Visani, "Metrics for Multi-Class Classification: an Overview," *arXiv*, pp. 1–17, Aug. 2020. [Online]. Available at: https://arxiv.org/abs/2008.05756v1.