Journal of Science and Technology
ISSN: 2456-5660 Volume 5, Issue 5, Sep-October 2020

www.jst.org.in *DOI: https://doi.org/10.46243/jst.2020.v5.i5.pp227-236*

Performance Comparison of Convolutional Neural Network-based model using Gradient Descent Optimization algorithms for the Classification of Low Quality Underwater Images

Shamsudheen Marakkar[1], Mithun Haridas T.P.[2], Supriya M.H.[3]
[1, 2, 3]*(Department of Electronics, Cochin University of Science And Technology, Cochin, Kerala, India)*
[2]*Corresponding Author: mithuntp@cusat.ac.in*

To Cite this Article
Shamsudheen Marakkar, Mithun Haridas T.P. and Supriya M.H., "Performance Comparison of Convolutional Neural Network-based model using Gradient Descent Optimization algorithms for the Classification of Low Quality Underwater Images", Journal of Science and Technology, Vol. 05, Issue 05, Sep-October 2020, pp227-236

Article Info
Received: 15-06-2020 Revised: 22-08-2020 Accepted: 05-09-2020 Published: 18-09-2020

Abstract: *Underwater imagery and analysis plays a major role in fisheries management and fisheries science helping developing efficient and automated tools for cumbersome tasks such as fish species identification, stock assessment and abundance estimation. Majority of the existing tools for analysis still leverage conventional statistical algorithms and handcrafted image processing techniques which demand human interventions and are inefficient and prone to human errors. Computer vision based automated algorithms need a better generalisation capability and should be made efficient to address the ambiguities present in the underwater scenarios, and can be achieved through learning based algorithms based on artificial neural networks. This paper research about utilising the Convolutional Neural Network (CNN) based models for under water image classification for fish species identification. This paper also analyses and evaluates the performance of the proposed CNN models with different optimizers such as the Stochastic Gradient Descent (SGD),Adagrad, RMSprop, Adadelta, Adam and Nadam on classifying ten classes of images from the Fish4Knowledge(F4K) database.*

Keywords*: Underwater Image, Deep Learning, Image Classification, Convolutional Neural Network (CNN), Fish Species Classification, SGD,Adagrad, RMSprop, Adadelta, Adam, Nadam*

I. Introduction

Fisheries management and fisheries science nowadays rely mostly on computer vision based tools to assist in many of the problems in the field. Manual analysis of underwater data such as images and videos is costly, time consuming, labour intensive, prone to fatigue errors and requires experts' assistance. Hence there exists a high need for the development of highly reliable automation tools for detection, classification and tracking of fish species and various marine organisms from underwater video footage with minimum human intervention. Such challenges in underwater imagery and analysis are tackled by the computer vision community in collaboration with marine engineers, scientists, and biologists. Earlier, the computer vision tools and techniques were leveraging conventional statistical algorithms and handcrafted image processing algorithms like Support Vector Machines(SVM) and its variants, Linear models such as regression models, etc. These tools are mainly challenged by the variability in unconstrained underwater scenarios which are highly affected by variations in lighting conditions, turbulence, turbidity and occlusion. Research was also conducted on a few techniques that use a combination of statistical and learning-based algorithms.

After the unprecedented growth and wide acceptance of artificial intelligence algorithms, mainly deep neural net-based computer-vision techniquesutilising CNN and FCN (Fully Convolutional Network) have been applied in underwater imagery and analysis as well. These models outperform Conventional statistical algorithms and handcrafted image processing algorithms because of their ability to learn hidden patterns from the data and to do efficient generalization. These learning models address and solve a huge number of problems in underwater imagery ranging from abundance estimation and stock assessment to environmental surveillance and overfishing, under fishing monitoring in both artificial and natural habitats. Fish stock assessment helps both research and business at the same time, using methods such as monitoring and capturing the temporal dynamics of fish abundance through evolutionary algorithms for content analysis in underwater images.

Moreover, environmental monitoring and surveillance are important as the underwater species are getting endangered as their habitats are getting hostile due to pollution, climate change and unrestricted commercial fishing. The process involves abundance estimation, health monitoring of individual species or the whole ecosystem and estimation of number and length of species present. This paper research on modeling convolution neural networks based deep learning models for fish species classification from low quality underwater images from the Fish 4 Knowledge (F4K) database [1],[2],[3]. The paper also presents the performance comparison of Convolutional Neural Network-based underwater image classification models trained on ten classes of fish species images from the dataset and performance evaluation and analysis is carried out by experimenting the models with a handful of different optimizers such as the Stochastic Gradient Descent (SGD), Adagrad, RMSprop, Adadelta, Adam and Nadam.

II. Related Works

Image Classification has been a highly explored area, but research in underwater image analytics was not common probably due to the highly dynamic scenarios like lighting conditions, turbulence, turbidity, occlusion, etc. More over the applications of deep neural network based models for solving the problem in the underwater scenarios are also less. However, some notable research in the underwater image analytics with the special focus in the fish and other marine organisms are explored here such as detection and classification of fish species, fisheries stock estimation, fish biomass monitoring, detection and classification of other underwater objects such as starfishes.

Salman et al. [4] compares conventional algorithms like Support Vector Machines, K-nearest Neighbors algorithm and Sparse Representation Classifier with the deep neural net-based Convolutional Neural Network (CNN). The research applied underwater fish datasets like LifeCLEF14 as well as LifeCLEF15 on these different models for performance comparison. Villon et al. in [5], uses Histogram Oriented Gradients (HOG) based feature descriptors for the training of SVM model for classifying coral reef fish. This combined model is compared with a fine-tuned CNN model which removes this feature descriptor part. Both of these researches show that the CNN based model outperforms the statistical and image processing based models by a huge margin. Siddiqui *et al*. [6] proposed a cross-layer pooling based CNN for enhanced discriminative ability in order to handle the problem of limited labeled training data. Salman et.al recognise, quantifies and measures the abundance by measuring the cover as well as size of underwater flora and fauna [7].

Khalid et al. uses a simplified four layer CNN architecture for aquarium family fish species classification [8]. The first two layers are convolutional layers and the next two are dense layers. The paper compares the proposed architecture with CNN architectures likeAlexNet and VGGNet with untrained benchmark dataset. Rathi et al. developed a method to classify the fish species from the F4K database[2],[3], using CNN and digital image processing techniques [9]. Before training the model, the dataset is passed through a series of digital image processing-based preprocessing techniques like thresholding, noise removal, histogram equalization, and some morphological operations such as erosion and dilation to finally come up with images having perfect fish-background separation, and thus CNN model is trained with this preprocessed data, to achieve an accuracy of 96.29%.

Ahsan Jalal and Ahmed Salman in [10] addressed the problem of variability of unconstrained underwater scenarios and demonstrates that CNN architecture utilising hierarchical features learns unique visual features of fish species and proved to be efficient, with an accuracy of 90% using LifeCLEF14 and LifeCLEF15 dataset. Frederik and Helmut et.al [11] describe how CNN makes the modelling free of hand-engineered image features. The paper also investigates, if the classification accuracy can be increased by adding additional meta-information to the network and achieves a test accuracy of 93% on the CNN model.

Suxia Cui and Yu Zhou, in [12] explains the CNN model proposed for fish detection with three optimization approaches. Data augmentation to increase training samples; network simplification by accommodating deep neural networks; and measures to speed up the training are incorporated to make the model more efficient.

Dmitry A. Konovalov et al. [13] proposes alablelling-efficient method utilisingXCeption architecture based model with less dataset of fish and non fish classes taken from 20 different habitats and achieved 0.17% and 0.61% for false-positives and false-negatives respectively with AUC of 99.94%.

III. Proposed Model

Deep Learning based image Classification requires feature extraction and decision making networks through learning. The capabilities such as translation and spacial invariance of convolutional neural networks is utilised for feature extraction and classification power of fully connected neural networks is used for prediction of different classes of fish species.

A. Convolutional Neural Networks

CNN nowadays has become very useful and efficient for image-based tasks such as detection, classification and segmentation. CNN was developed through the inspiration from Fukusima's Neo-cognitron [14] and one of the efficient implementations was proposed by Yann Lecun [15] in 1998. TheLeNet-5 model [15] for classifying the MNIST hand-written character recognition was proved successful, and ever since there has been a rapid advancement in the development of efficient CNN architectures for various applications.

The convolutional neural network(CNN) architecture consists of three types of layers namely: Convolution layers, Activation layer and Pooling layer. Series of such cascaded convolutional blocks extracts the features from the input and produces abstract features at higher layers, extracting the most relevant information from the images. Thus extracted abstract features are analysed in all possible combinations and appropriate prediction is made using fully connected layers of neurons for classifying images [15]. The CNN architecture learns the kernel weights of convolutional layers along with the weights of the fully connected layers through the training using the labelled dataset. The hyper-parameters of the different layers such as the kernel size, number of kernels, stride and padding, number of units in each layer, activation functions can be pre-defined and the model weights are learned using gradient descent based optimizers.

B. Proposed Architecture

The proposed model utilises cascaded three convolutional blocks followed by three fully connected layers with ten neurons in the output layer. The architecture of the proposed model is as given in figure 1. Convolution layers are designed with half padding and unity stride so that feature maps of the same input dimension will be obtained and the Rectified Linear Unit (ReLU) activation functions are used for better learning.MaxPooling layer with pool-size 2x2 and stride of pool size 2 is added for subsampling the derived feature maps. The output layers are provided with the Softmax activation function to get the probability values of the class prediction.

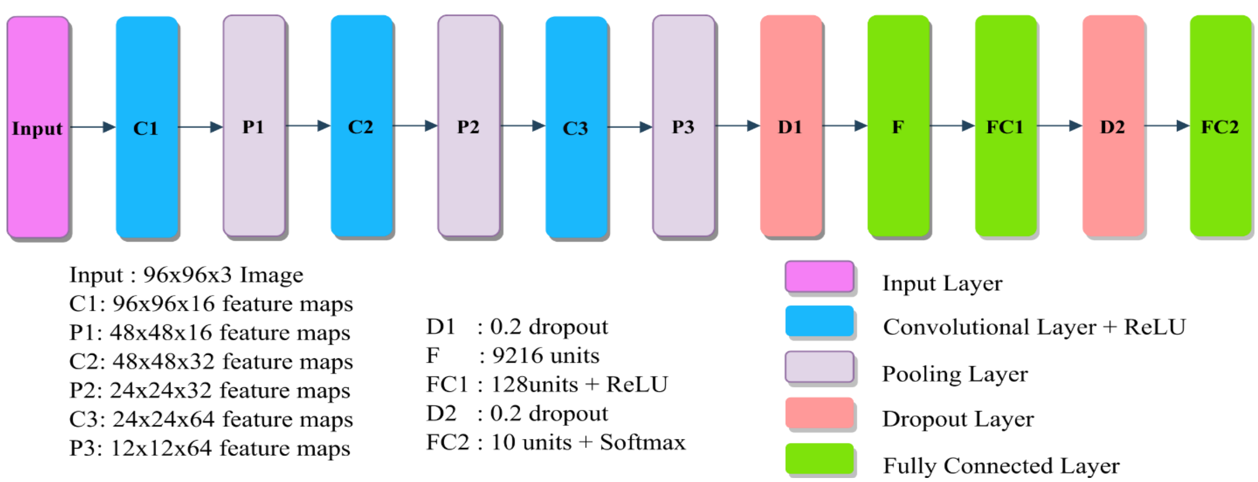

Fig. 1.: Architecture of the proposed fish species underwater image classification model.

The architecture accepts the input images as 96x96x3 RGB image which is passed through the Convolutional layer (C1)having 16 filter kernels of size 3x3 to generate feature maps of size 96x96 which is then activated using ReLU function. The Max pooling layer(P1) subsamples the feature maps and reduce to the size 48x48. Thus generated set of 48x48 feature maps are then subjected to second set of Convolutional layer(C2) having

32 filter kernels of size 3x3 to generate feature maps of 48x48 which is ReLU activated. The Max pooling layer (P2) subsamples the feature map to size 24x24. These feature maps are then passed to the third convolutional layer(C3) having 64 kernels of size 3x3 to generate feature maps of size 24x24 which is ReLU activated and subsampled using Max pooling layer (P3) to128 units generate 64 feature maps of size 12x12. The feature maps are subjected to a dropout of 20% to reduce the possibility of overfitting by a Dropout layer (D1).

These obtained feature maps are then flattened to a 1D vector using a 9216 sized Flatten layer (F) which are then connected with a fully connected layers with 128 units (FC1) with ReLU activation followed by a dropout of 20%using the second Dropout layer (D2). The output classification result is provided with fully connected layer of 10 units (FC2) with Softmax activation function which produces the probability of the fish species being predicted as a particular class out of 10 classes in the dataset. The configuration details and the number of trainable parameters in each layer of the proposed architecture is given in the Table 1 with the total trainable parameters of 12,04,650.

Table 1: Model Configuration of the proposed architecture

Layers	Configuration	Number of Trainable Parameters
Convolution 2D (C1)	16 filters, 3x3 kernel, half padding, stride 1, ReLU	448
Max Pooling 2D (P1)	2x2 kernel, no padding, stride 2	0
Convolution 2D (C2)	32 filters, 3x3 kernel, half padding, stride 1, ReLU	4640
Max Pooling 2D (P2)	2x2 kernel, no padding, stride 2	0
Convolution 2D (C3)	64 filters, 3x3 kernel, half padding, stride 1, ReLU	18496
Max Pooling 2D (P3)	2x2 kernel, no padding, stride 2	0
Dropout (D1)	Dropout Rate = 0.2 (20%)	0
Flatten (F)	9216 units	0
Dense (FC1)	128 units, ReLU	1179776
Dropout (D2)	Dropout Rate = 0.2 (20%)	0
Dense (FC2)	10 units, Softmax	1290
	Total Trainable Parameters	1204650

IV. Methodology

Artificial Neural Networks based learning models provide learning parameters and hyper parameters in their layered structure. Parameters are randomly initiated and loss functions are minimized using gradient descent based optimizations. Back Propagation algorithm help in effectively training the parameters in the deep layered structure of convolutional neural network. The underwater images of ten classes of fish species from Fish4Knowledge [2][3]database is used as the dataset to train the CNN model.

***Data Preprocessing*:** The images having varied size ranges around 115 to 87 pixels in the Fish4Knowledge dataset, are resized into size 96x96 with 3 channels. The underwater images from selected ten classes of fish species are used for the work. Dataset details as given in the Table 2. A set of 20 images is reserved exclusively for testing and the remaining images are split to training and validation of model, with validation split ratio of 20%. The train, valid and test split of each class is also mentioned in the Table 2.Sample image data of 10 classes of fish species is shown in the Figure 2.

Table 2: Ten selected classes of fish species image dataset from Fish for Knowledge [2],[3]

Class. No.	Fish Species	Total number of Images	Number of Images for Training	Number of Images for Vallidation	Number of Images for Testing
0	Amphiprionclarkii	4049	3223	806	20
1	Chromischrysura	3593	2859	714	20
2	Plectroglyphidodondickii	2683	2131	532	20
3	Chaetodon lunulatus	2539	2016	503	20
4	Myripristiskuntee	450	344	86	20
5	Neoniphonsammara	299	223	56	20
6	Hemigymnusfasciatus	241	177	44	20
7	Acanthurusnigrofuscus	218	158	40	20
8	Lutjanusfulvus	206	148	38	20
9	Chaetodon trifascialis	190	136	34	20
	Total	14468	11415	2853	200

Fig. 2. Sample image of 10 classes of fish species selected from Fish for Knowledge database.[2],[3]

Loss Function:Sparse Categorical Cross Entropy Loss function is used to train the CNN model. Categorical cross entropy is the cross entropy loss function with the softmax activation as the prediction as shown in the equation (1), where J is the loss function with respect to the parameter θ, t_i is the ground truth and x is the input data, i and $j$are the training labels over C classes. Sparse Categorical Cross entropy represents the labels of images as the arguments of the one-hot embeddings.

$$J(\theta) = -\sum_{i}^{C} t_i . log\left(\frac{e^{(\theta^T x)_i}}{\sum_j^C e^{(\theta^T x)_j}}\right) \quad (1)$$

Optimisers:The CNN model is optimized with different gradient descent based functions such as SGD, Adagrad, RMSprop, Adadelta, Adam and Nadam which are described below [16]:

Stochastic Gradient Descent (SGD): SGD update the parameters for every training example $(x^{(i)},y^{(i)})$ and thus by performing faster updation leading to online learning, but leads to high fluctuation in the value of the function. The SGD parameter updation function is as given in equation (2), where η is the learning rate.

$$\theta = \theta - \eta . \nabla_\theta J\left(\theta; x^{(i)}; y^{(i)}\right) \quad (2)$$

Adagrad:Adagrad adapts the learning rate throughout the weight updation process, by largeupdates for infrequent and small updates for frequent parameters, making it suitable for sparse data, by tuning the learning rate.Adagrad parameter updation function is given by the equation (3), where t is the time step, G_t is the diagonal matrix containing sum od squares of the past gradients with respect to all the parameters θ, ϵ *is the smoothing term* and g_t is the gradient of the objective function with respect to the parameter θ at time step t. $\odot$ denotes the element wise matrix vector multiplication[17].

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{(G_t + \epsilon)}} \odot g_t \quad (3)$$

Adadelta: Adadelta is the modification to Adagrad by minimizing the aggressiveness of learning rate change by restricting accumulation of past gradients by fixing the window size w, by calculating the decaying average of all past squared gradients $E[g^2]_t$ at every time step t, as shown in the equation (4) and (5) with γ as the momentum decay term. [18].

$$E[g^2]_t = \gamma E[g^2]_{t-1} + (1-\gamma)g_t^2 \tag{4}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{(E[g^2]_t + \epsilon)}} g_t \tag{5}$$

RMSprop: RMSprop is also an adaptive learning method proposed by Hinton et.al. [19] similar to Adadelta with $\gamma = 0.9$ as shown in the equation (6).

$$E[g^2]_t = 0.9E[g^2]_{t-1} + 0.1\, g_t^2 \tag{6}$$

Adam: Adaptive Momentum Estimation (Adam) is also an adaptive learning rate based optimization which utilizes the exponentially decaying average of past gradients (m_t) along with the past squared gradients (v_t). m_t is the first moment denoting the mean and v_t is the second moment representing the variance, β_1 *and* β_2 are the decaying rates with values close to 1, as given in equations (7) and (8). Bias corrected moments are given by equations (9) and (10). The Adam parameter update rule is given by equation (11) [20]

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)g_t \tag{7}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2)g_t^2 \tag{8}$$

$$\widehat{m}_t = \frac{m_t}{(1-\beta_1^t)} \tag{9}$$

$$\hat{v}_t = \frac{v_t}{(1-\beta_2^t)} \tag{10}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\left(\sqrt{\hat{v}_t} + \epsilon\right)} \widehat{m}_t \tag{11}$$

Nadam: Nesterov-acclerated Adaptive Moment Estimation (Nadam) [21] incorporates Nesterov Accelerated Gradient (NAG)[22] with Adam. NAG provides more accurate updation in the gradient direction by providing momentum step before computing the gradient given by equation (12), (13) and (14) and the Nadam parameter update rule by equation (15).

$$v_t = \gamma v_{t-1} + \eta \nabla_{\theta_t} J(\theta_t - \gamma v_{t-1}) \tag{12}$$

$$m_t = \gamma m_{t-1} + \eta \nabla_{\theta_t} J(\theta_t - \gamma m_{t-1}) \tag{13}$$

$$g_t = \nabla_{\theta_t} J(\theta_t - \gamma m_{t-1}) \tag{14}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\left(\sqrt{\hat{v}_t} + \epsilon\right)} \left(\beta_1 \widehat{m}_t + \frac{(1-\beta_1)g_t}{(1-\beta_1^t)}\right) \tag{15}$$

Performance Evaluation: Evaluation of Performance in Testing phase is done using Confusion Matrix and the classification metrics such as Precision, Recall and F1 score, calculated from the True Positives (TP), False Positives (FP), True Negatives (TN) and False Negatives (FN) from the confusion matrix.

Precision provides the correctness of the positive predictions and is calculated as the ratio of True Positives to all positive predictions as in equation (16). The value range is [0,1]

$$Precision = \frac{TP}{TP + FP} \tag{16}$$

*Recall*provides the score of predicting the actual positives in the dataset as positives, and is calculated as the ratio of True Positives to Actual positives (combining True Positives and False Negatives as in equation (17). The value range is [0,1]

$$Recall = \frac{TP}{TP + FN} \tag{17}$$

F1 score provides the score of trade-off between Precision and Recall, and is calculated as the harmonic mean of Precision and Recall, as in equation (18). The value range is [0,1]

$$F1\ score = \frac{2Precision\ x\ Recall}{Precision + Recall} = \frac{2TP}{2TP + FP + FN} \tag{18}$$

Python is used for developing the CNN model by utilizing the Jupyter Notebook environment. Deep Learning libraries used includes TensorFlow, Keras and Scikit-Learn along with libraries such as numpy, matplotlib, pandas and seaborn. Tensor board is utlised for analyzing the training performance. Callback functions are called for monitoring loss function and early stopping criteria is set for the minimum delta of 0.01 at a patience of 10 epochs. The best training parameter values/weights are saved every epoch monitoring the improvements in the loss function value.

V. Results and Discussions

The CNN based model is trained with loss function as Sparse Categorical Cross-entropy and optimized based on different functions gradient descent based functions such as Stochastic Gradient Descent (Model M1), Adagrad (Model M2), RMSprop (Model M3), Adadelta (Model M4), Adam (Model M5) and Nadam (Model M6) for 50 epochs with early stopping criteria. Each model is analysed with their training, validation and testing performance.

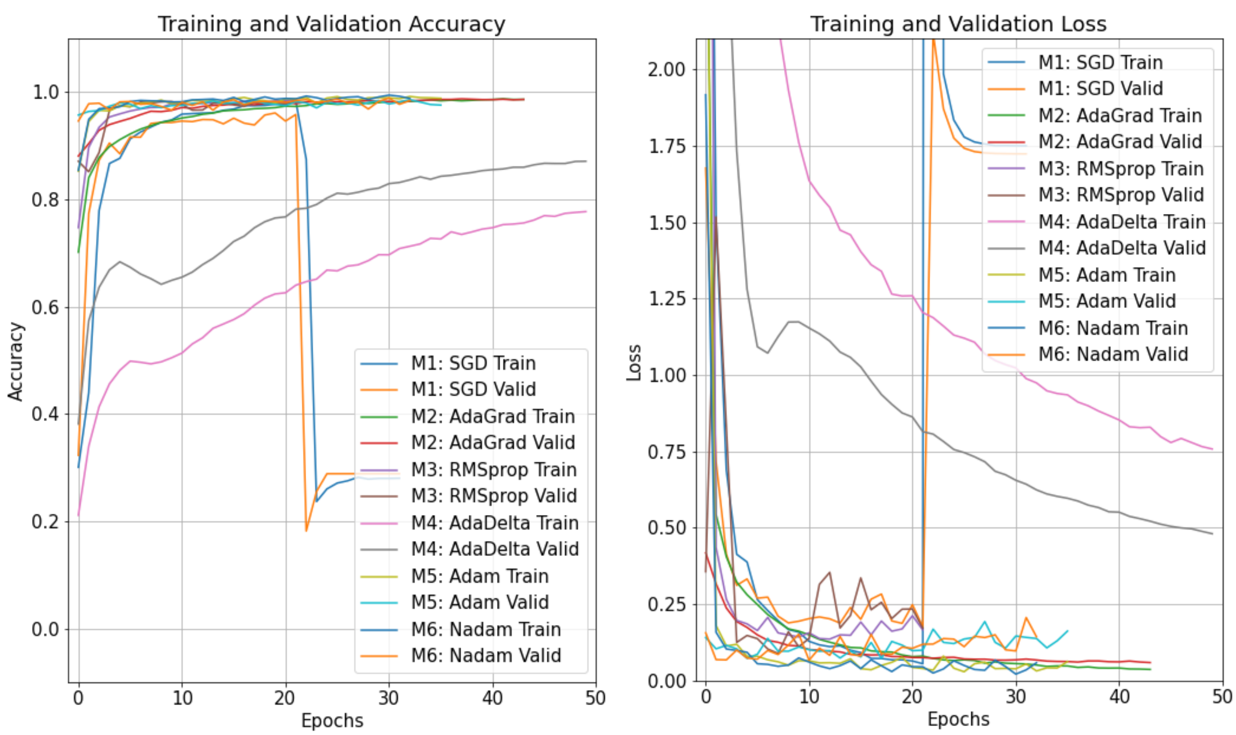

Fig. 3. Accuracy and Loss plot with respect to epochs for both Training and Validation of six models

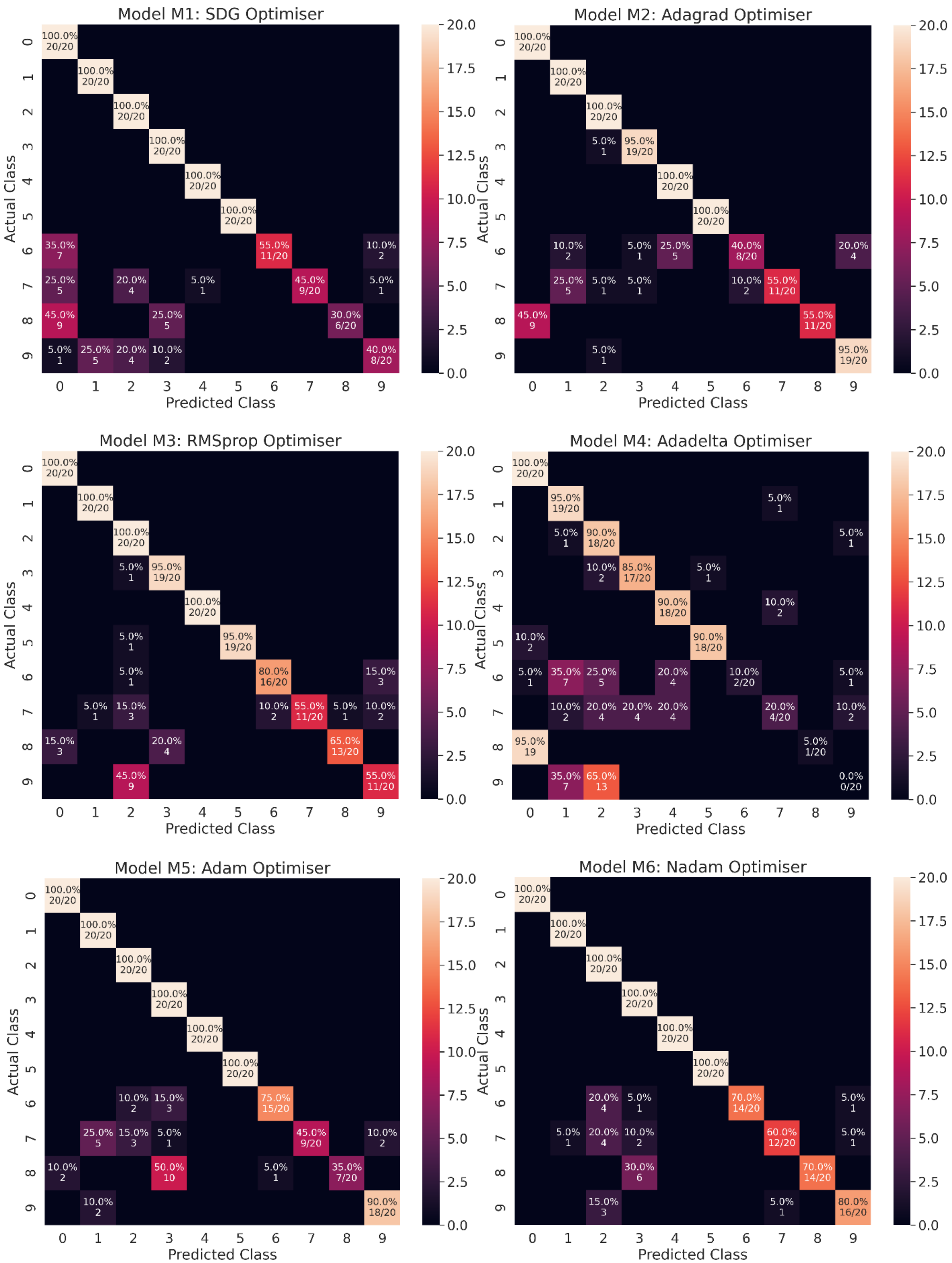

Figure 4. Confusion Matrix showing the testing results of all models

Accuracy and Loss function values obtained while Training and Validation is analysed with respect to the number of epochs as shown in the figure 3. It is observed that the Model M6 with Nadam optimizer performs better in training phase and the model M2 with Adagrad optimizer perform better in validation phase, considering the training accuracy alone. It was also observed that Model M2 with Adagrad optimize the model better with minimum loss both in the training and validation phase, but takes 11 more epochs to converge than Model M6 with Nadam optimizer.

Nadam optimize the model to reach the maximum accuracy 0.9907 and 0.9824 for training and validation respectively in 32 epochs and gets saturated. It is also observed that Nadam also performs well in the loss plot with 0.04529 and 0.1477 for training and validation respectively with 32 epochs. Adagrad (M2) optimizes the model with minimum loss of 0.03815 and 0.06074 for training and validation, but training accuracy less than that of Nadam with 0.9866, however the validation accuracy is more with value 0.9865 by taking 43 epochs. RMSprop (M3) takes the least number of epoch to get saturated with the accuracy of 0.9843 and 0.9815 and loss of 0.178 and 0.2088 for training and validation respectively in 21 epochs. Adam (M5) also performs well, with accuracy of 0.9895 and 0.9787 along with loss of 0.04948 and 0.1412 for training and validation respectively in 35 epochs. Adadelta (M4) take all 50 epochs to reach the accuracy of 0.7738 and 0.8693, and loss of 0.7716 and 0.491 for training and validation, to observe as the least performing model, but continue to improve over the epochs in both accuracy and loss. SGD optimizer makes the model (M1) over-fitted after 21 epochs, with highest accuracy of 0.9794 and 0.954 along with lowest loss of 0.0640 and 0.2065, making the model accuracy decrease to 0.2857 and 0.2918 along with loss increased to 1.892 and 1.718 for training and validation respectively by 31st epoch. The training performance is tabulated in the Table 3.

Table 3: Training and Validation Performance with respect to accuracy and loss of all models

Models	Model M1	Model M2	Model M3	Model M4	Model M5	Model M6
Optimiser	SGD	Adagrad	RMSprop	Adadelta	Adam	Nadam
Training Accuracy	0.97940	0.98660	0.98430	0.77380	0.98950	**0.99070**
Validation Accuracy	0.95400	**0.98650**	0.98150	0.86930	0.97870	0.98240
Training Loss	0.06400	**0.03815**	0.17800	0.77160	0.04948	0.04529
Validation Loss	0.20650	**0.06074**	0.20880	0.49100	0.1412	0.14770
Epochs taken to converge	21	43	**21**	50	35	32

All the models are also analysed with their testing performance by plotting the confusion matrix for all 10 classes by taking 20 images from each class of fish species. The confusion metrics of all models are plotted in the figure 4. Actual fish species and the predicted species names are plotted in confusion matrix and cells denotes the number of images predicted along with the percentage. The model performance is also analysed with the different classification performance metrics such as Precision, Loss and F1 score and is tabulated in the Table 4. Since the dataset is highly unbalanced Precision, Recall and F1 score is evaluated for each class and average of overall class is also calculated. It is observed that the Model M6 with Nadam performs better compared to other models in terms of Precision, Recall and F1 score with values 0.91, 0.88 and 0.88 respectively.

Table4: Classification Evaluation Metrics for Testing Performance of all models

Class No.	Fish Species (10 classes)	Model M1 (SGD)			Model M2 (Adagrad)			Model M3 (RMSprop)			Model M4 (Adadelta)			Model M5 (Adam)			Model M6 (Nadam)		
		Precision	Recall	F1 score	Precision	Recall	F1 score	Precision	Recall	F1 score	Precision	Recall	F1 score	Precision	Recall	F1 score	Precision	Recall	F1 score
0	Amphiprionclarkii	0.48	1.00	0.65	0.69	1.00	0.82	0.87	1.00	0.93	0.48	1.00	0.65	0.91	1.00	0.95	1.00	1.00	1.00
1	Chromischrysura	0.80	1.00	0.89	0.74	1.00	0.85	0.95	1.00	0.98	0.53	0.95	0.68	0.74	1.00	0.85	0.95	1.00	0.98
2	Plectroglyphidodondickii	0.71	1.00	0.83	0.87	1.00	0.93	0.57	1.00	0.73	0.43	0.90	0.58	0.80	1.00	0.89	0.65	1.00	0.78
3	Chaetodon lunulatus	0.74	1.00	0.85	0.90	0.95	0.93	0.83	0.95	0.88	0.81	0.85	0.83	0.59	1.00	0.74	0.69	1.00	0.82
4	Myripristiskuntee	0.95	1.00	0.98	0.80	1.00	0.89	1.00	1.00	1.00	0.69	0.90	0.78	1.00	1.00	1.00	1.00	1.00	1.00
5	Neoniphonsammara	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.95	0.97	0.95	0.90	0.92	1.00	1.00	1.00	1.00	1.00	1.00
6	Hemigymnusfasciatus	1.00	0.55	0.71	0.80	0.40	0.53	0.89	0.80	0.84	1.00	0.10	0.18	0.94	0.75	0.83	1.00	0.70	0.82
7	Acanthurusnigrofuscus	1.00	0.45	0.62	1.00	0.55	0.71	1.00	0.55	0.71	0.57	0.20	0.30	1.00	0.45	0.62	0.92	0.60	0.73
8	Lutjanusfulvus	1.00	0.30	0.46	1.00	0.55	0.71	0.93	0.65	0.76	1.00	0.05	0.10	1.00	0.35	0.52	1.00	0.70	0.82
9	Chaetodon trifascialis	0.73	0.40	0.52	0.83	0.95	0.88	0.69	0.55	0.61	0.00	0.00	0.00	0.90	0.90	0.90	0.89	0.80	0.84
	Average	0.84	0.77	0.75	0.86	0.84	0.82	0.87	0.84	0.84	0.65	0.58	0.50	0.89	0.84	0.83	**0.91**	**0.88**	**0.88**

VI. Conclusion

This paper applies a three layered convolutional neural network for classifying fish species from low quality underwater image data of selected ten classes of species from Fish4Knowledge Database. The classifier model is provided with sparse categorical cross-entropy loss function and optimized with different gradient descent based function such as Stochastic Gradient Descent (SGD), Adagrad, RMSprop, Adadelta, Adam and Nadam as different models. The performance comparison for each model for training and validation is done and it was observed that even though the model M6 with Nadam optimizer converges to the maximum training accuracy of 0.9907, the model M2 with Adagrad optimizes the model to converge at maximum validation accuracy of 0.98650 and minimum loss of 0.03815 and 0.06074 for training and validation respectively.Testing evaluation of all models with Confusion matrix and the classification metrics such as precision, recall and F1 score shows that the model M6 with Nadam optimizer performs well with values 0.91, 0.88 and 0.88 respectively. Class imbalance in the dataset makes the precision and recall fluctuated for different classes with respect to different optimisers. Data augmentation methods can be incorporated as future work for better learning. The work can also be extended to video dataset for real-time fish detection and classification to help the underwater resource monitoring and exploration more effectively.

References

1. A P. X. Huang, B. J. Boom, R. B. Fisher, "Hierarchical Classification for Live Fish Recognition", BMVC student workshop paper, September 2012.
2. B. J. Boom, P. X. Huang, C. Spampinato, S. Palazzo, J. He, C. Beyan, E. Beauxis-Aussalet, J. van Ossenbruggen, G. Nadarajan, J. Y. Chen-Burger, D. Giordano, L. Hardman, F.-P. Lin, R. B. Fisher, "Long-term underwater camera surveillance for monitoring and analysis of fish populations", Proc. Int. Workshop on Visual observation and Analysis of Animal and Insect Behavior (VAIB), in conjunction with ICPR 2012, Tsukuba, Japan, 2012.
3. B. J. Boom, P. X. Huang, J. He, R. B. Fisher, "Supporting Ground-Truth annotation of image datasets using clustering", 21st Int. Conf. on Pattern Recognition (ICPR), 2012.
4. Villon S., Chaumont M., Subsol G., Villéger S., Claverie T., Mouillot D. “Coral Reef Fish Detection and Recognition in Underwater Videos by Supervised Machine Learning: Comparison Between Deep Learning and HOG+SVM Methods” International Conference on Advanced Concepts for Intelligent Vision Systems, pp.160--171, 2016
5. A. Salman, A. Jalal, F. Shafait, A. Mian, M. Shortis, J. Seager and E. Harvey, “Fish species classification in unconstrained underwater environments based on deep learning,” Limnology and Oceanography: Methods , vol. 14, p. 570–585, 2016.
6. S.A. Siddiqui, A. Salman, M.I. Malik, F. Shafait, A. Mian, M.R. Shortis, E.S. Harvey. “Automatic fish species classification in underwater videos: exploiting pre-trained deep neural network models to compensate for limited labelled data”.ICES Journal of Marine Science,Vol.75,No.1, pp.374-389.2018
7. Salman A, Siddiqui SA, Shafait F, Mian A, Shortis MR, Khurshid K, Ulges A, Schwanecke U, “Automatic fish detection in underwater videos by a deep neural network-based hybrid motion learning system”. ICES Journal of Marine Science,vol.77, No.4,pp.1295-1307,2020.
8. Khalifa, N.E.M.; Taha, M.H.N.; Hassanien, A.E. “Aquarium Family Fish Species Identification System Using Deep Neural Networks”. In International Conference on Advanced Intelligent Systems and Informatics; Springer: Cham, Switzerland, pp. 347–356,2018
9. D. Rathi, S. Jain and S. Indu, "Underwater Fish Species Classification using Convolutional Neural Network and Deep Learning," 2017 Ninth International Conference on Advances in Pattern Recognition (ICAPR), Bangalore, 2017, pp. 1-6,2017
10. Jalal A, Salman A, Mian A, Shortis M, Shafait F (2020), “Fish detection and species classification in underwater environments using deep learning with temporal information”,. Ecological Informatics,Vol.57, pp.101088,2020
11. Kratzert, F., Mader, H. “Fish species classification in underwater video monitoring using Convolutional Neural Networks”, OpenKratzert, Frederik, and Helmut Mader. Fish Species Classification in Underwater Video Monitoring Using Convolutional Neural Networks. EarthArXiv, vol. 15, 2018.
12. Suxia Cui and Yu Zhou. “Fish Detection In Deep Learning”. Hindawi Applied Computational Intelligence and Soft Computing, Vol.2020,
13. Dmitry A. Konovalov et al. “Underwater Fish Detection with Weak Multi-Domain Supervision” International Joint Conference on Neural Networks (IJCNN),pp.1-8, 2019.
14. Fukushima, Kunihiko and Miyake, Sei, “Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition”, Competition and cooperation in neural nets, Springer, pp.267-285,1982
15. Yann LeCun, Léon Bottou, YoshuaBengio, and Patrick Haffner. “Gradient-based learning applied to document recognition”. Proceedings of the IEEE, 86(11):2278–2324, 1998.
16. Ruder, Sebastian, “An overview of gradient descent optimization algorithms”, arXiv preprint arXiv:1609.04747, 2016
17. John Duchi, EladHazan, and Yoram Singer., “Adaptive Subgradient Methods for Online Learning and Stochastic Optimization”,. Journal of Machine Learning Research, Vol.12,No.7,pp.2121–2159, 2011.
18. Matthew D. Zeiler. ADADELTA: An Adaptive Learning Rate Method. arXiv preprint arXiv:1212.5701, 2012.
19. Hinton, Geoffrey and Srivastava, Nitish and Swersky, Kevin, “Neural networks for machine learning lecture 6a overview of mini-batch gradient descent”, cs.toronto.edu, 2012
20. Diederik P. Kingma and Jimmy Lei Ba. “Adam: a method for Stochastic Optimization”, International Conference on Learning Representations, pages 1–13, 2015.
21. Timothy Dozat. “Incorporating Nesterov Momentum into Adam”. ICLR Workshop, pp.1-4, 2016
22. YuriiNesterov. “A method for unconstrained convex minimization problem with the rate of convergence o(1/k2)”. Doklady ANSSSR, vol 269, pp.543–547, 1983.