A Novel Feature Extraction Model to Enhance Underwater Image Classification

Muhammad Irfan[1(✉)], Jiangbin Zheng[1], Muhammad Iqbal[2], and Muhammad Hassan Arif[3]

[1] School of Software, Northwestern Polytechnical University, Xian, China
mirfan@mail.nwpu.edu.cn, zhengjb@nwpu.edu.cn
[2] Faculty of Computer and Information Science, Higher Colleges of Technology, Fujairah, UAE
miqbal1@hct.ac.ae
[3] CESAT, Islamabad, Pakistan
mhassanarif@gmail.com

Abstract. Underwater images often suffer from scattering and color distortion because of underwater light transportation characteristics and water impurities. Presence of such factors make underwater image classification task very challenging. We propose a novel classification convolution autoencoder (CCAE), which can classify large size underwater images with promising accuracy. CCAE is designed as a hybrid network, which combines benefits of unsupervised convolution autoencoder to extract non-trivial features and a classifier, for better classification accuracy. In order to evaluate classification accuracy of proposed network, experiments are conducted on Fish4Knowledge dataset and underwater synsets of benchmark ImageNet dataset. Classification accuracy, precision, recall and f1-score results are compared with state-of-the-art deep convolutional neural network (CNN) methods. Results show that proposed system can accurately classify large-size underwater images with promising accuracy and outperforms state-of-the-art deep CNN methods. With the proposed network, we expect to advance underwater image classification research and its applications in many areas like ocean biology, sea exploration and aquatic robotics.

Keywords: Convolutional autoencoder · Deep learning · Convolutional neural network · Underwater images

1 Introduction

Underwater image classification has recently attracted many computer vision researchers because of its applications in marine sciences and autonomous underwater vehicles (AUV). Underwater image classification is a challenging task because of complex underwater environment and poor lighting conditions. Factors such as wavelength dependent absorption and scattering of light degrade the visibility of underwater images and make them poorly contrasted and blur [1].

C. Brito-Loeza et al. (Eds.): ISICS 2020, CCIS 1187, pp. 78–91, 2020.
https://doi.org/10.1007/978-3-030-43364-2_8

These factors hinder the performance of underwater image classification applications used in sea exploration, aquatic robotics and sea environmental surveillance [2].

In recent years, most of the studies used for underwater image classification are deep learning based. Hongwei et al. [3] used combination of CNN, support vector machine (SVM), principal component analysis (PCA) and spatial pyramid pooling (SPP) for classification of fish species from Fish4Knowledge dataset. Xin et al. [4] used same Fish4Knowledge dataset for fish classification and tracking from videos. Xu et al. [5] used pre-trained GoogleNet along with augmentation techniques for underwater image classification. Limited labelled data is used by Siddiqui et al. [6] for automatic classification of fish through pre-trained neural networks. Jin et al. [7] used pre-trained AlexNet, trained on ImageNet dataset, for underwater image recognition in small sample size situations.

Convolutional Autoencoder (CAE) is a type of unsupervised learning [8]. CAE extends the basic structure of the autoencoder by using convolution layers instead of the fully connected layers to preserve the spatial locality of input data [9]. It consists of two parts, encoder and decoder [10]. Encoder part consists of pooling layers and convolutional layers. Decoder part consists of up-sampling layers and deconvolution layers. CAE tries to regenerate input at the output by using learned convolution filters. Learned convolutional filters extract non trivial features from the input. Extracted features can be used for classification.

In this study we propose a novel deep classification convolutional autoencoder, named CCAE. CCAE is designed as a hybrid classification convolutional autoencoder architecture. The main objective behind architecture of CCAE is the better feature extraction mechanism by combining classification layer with deep encoder-decoder network during the training as well as the testing phase. It extracts spatially localized features with hint of class. It combines capability of CNN to extract features from images [11] and capability of autoencoder (AE) to compress high dimensional data to low dimension [12] and a classifier to extract features with hint of class. Experiments are conducted, on underwater synsets of benchmark ImageNet dataset [13] and Fish4Knowledge dataset [14] (fish recognition ground truth dataset made by the Fish4-Knowledge project), to verify the architecture of CCAE. Results show that CCAE can correctly classify underwater images with promising accuracy. Classification accuracy, precision, recall and f1-score results of the proposed method are compared with state-of-the-art deep CNN methods such as ResNet [15], DenseNet [16], Inception [17] and Xception [18].

The rest of this paper is organized as follows. In Sect. 2, we review CAE. In Sect. 3, we present details of the proposed deep network. Experimental design is discussed in Sect. 4. Experiment results are presented and discussed in Sect. 5. Conclusion is drawn in Sect. 6.

2 Background

2.1 Convolutional Autoencoder

AE is a special type of neural network mainly used for data dimension reduction [19]. The AE consists of an encoder and a decoder. The encoder part encodes the input data to a hidden representation by using a deterministic nonlinear function, and then the decoder part reconstructs the input back to the original input space [20]. The connections between layers are fully connected.

CAE is categorized as unsupervised learning algorithm [8]. It combines benefits of CNN with unsupervised pre-training of AE [21,22]. It also consists of two parts, encoder and decoder [10]. Encoder part mainly consists of convolutional layers and pooling layers. The encoder convolutionally transforms the input into a latent space. Contrary to traditional AE, CAE preserves the spatial relationship of the input through a bank of 2-D convolutional filters [23]. CAE architecture is mainly used for semantic segmentation [24], image denoising/dehazing [25], deep clustering [26] and for feature extraction [23].

Encoder part is used to extract non-trivial (compressed) features [27]. Decoder part consists of deconvolution layers and up-sampling layers. The decoder tries to regenerate the input by mapping the latent layer data back to the original input space by minimizing the reconstruction error. Extracted compressed features can also be used for classification [23].

For a given input image X the encoder function is defined as

$$Y_i = encoder(X) = \sigma(X * W^i + b) \tag{1}$$

Where b is encoder bias, Y_i is encoding of the input X in a low dimensional space, W^i is 2-D convolutional filter, * is 2-D convolution and σ denotes activation function such as ReLU [28].

In decoding phase, Y_i is the input of the decoding function, which can be defined as

$$G = decoder(Y_i) = \sigma(Y_i * \tilde{W}^i + \tilde{b}) \tag{2}$$

Where $\tilde{W}^i$ is 2-D convolutional filter in decoder and $\tilde{b}$ is bias of decoder,

Mean square error (MSE) function E, as described below, is used as cost function to make output G as close as possible to input X.

$$E(\theta) = 1/n \sum_{i=1}^{m} (G_i - X_i)^2 \tag{3}$$

Back propagation algorithm is used to calculate the gradient of the error function with respect to the parameters as

$$\frac{\delta E(\theta)}{\delta W^i} = X * \partial Y_i + \tilde{Y}_i * \partial G \tag{4}$$

∂G and ∂Y are deltas of the reconstruction and latent space.

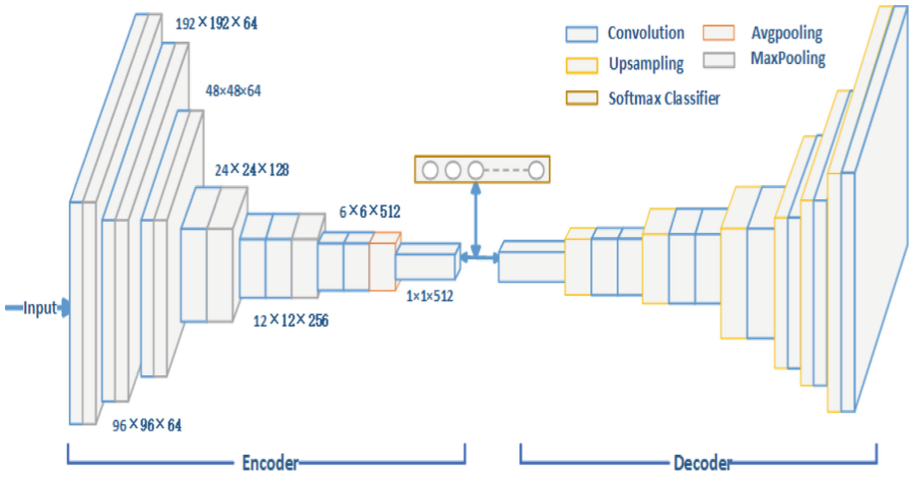

Fig. 1. CCAE architecture diagram

3 Proposed Method

Proposed deep neural network CCAE combines benefits of unsupervised learning and supervised learning. It consists of deep encoder network and corresponding decoder network, combined with a classification layer. CCAE layer wise architecture is elaborated in Fig. 1. Each convolution layer consists of different number of channels. CCAE encoder part takes an image as an input, convolves the input with convolution filters trained during the training phase and extracts non-trivial features of the input image. After the last layer (average pooling) of encoder, a classification layer is attached. The classification layer i.e. softmax, is an integral part of encoder-decoder network during the training as well as during the testing phase. This architecture is used to leverage better features with hint of class. Output of the average pooling layer of encoder is taken as an input to the decoder module. Proposed layers configuration of the encoder module and number of channels in each layer, offer improved feature extraction capability for better classification results. Table 1 summarizes configuration of CCAE layers in terms of number of channels, filter sizes, layer type, output size and input to each layer. Layers of encoder and decoder are separately arranged in Table 1. Keeping in view limitation of space, batch normalization layer used after every convolution layer is not mentioned in Table 1.

As convolution autoencoder is a type of unsupervised learning, so it is trained in an unsupervised way to extract features. In proposed network encoder-decoder network is trained through ensemble of a classification layer with the encoder during the training phase. Moreover, the decoder part is an integral part of the network during the training and testing phases. This training mechanism improves feature extraction by extracting non-trivial features with hint of class, which results in improvement in classification accuracy.

The following subsections describe the proposed module in detail.

3.1 Encoder

The encoder part of CCAE takes an image as an input and extracts the non-trivial features of the input image as described in Eq. (1). The encoder module consists of 8 convolution layers as shown in Fig. 1. Size of feature map output of each layer decreases from the first conv1 (192 × 192) to the last convolution layer con8 (6 × 6). Number of channels of convolution layers of encoder module increases from the first convolution layer conv1 (64 channels) to the last convolution layer conv8 (512 channels), as shown in Table 1. Each convolution layer performs the convolution operation to generate a set of feature maps. For element-wise non-linearity ReLU $max(0, x)$ [28] is used with each convolution layer. Regularization of network is done by decreasing the inner co-variate shift of data by using batch normalization layer after every convolution layer [29].

Pooling is used in the encoder network to filter noisy activations in lower layers and to retain robust activations in upper layers [30]. It also reduces the number of connections between convolutional layers which results in computational efficiency. Max-pooling layer with stride 2 (non-overlapping) and window size 2 × 2 is used after each convolution block to sub-sample output by factor of 2. For robust feature extraction, translation in-variance is achieved by using several max-pooling layers. After the last convolution layer (conv8) of the encoder module, an average pooling layer is used instead of a max pooling layer. Softmax classifier is used as activation non-linearity at the end of the encoder module. Output of the average pooling layer is first flattened (reshaped) and then it is given as an input to softmax classifier.

3.2 Decoder

The decoder module tries to reconstruct the same input at output layer through the combination of up-sampling and de-convolution as described in Eq. (2). It up-samples the encoded feature maps, by using a trainable filter bank. In the decoder, there are de-convolutional and up-sampling layers corresponding to each convolution and pooling layer in encoder. Hence, decoder part also consists of 8 de-convolution layers.

In the decoder, de-convolution operation is achieved using convolution operation along with up-sampling [31]. Contrary to convolution and pooling operations, the output of the de-convolutional operation is an enlarged and dense activation map. The output of an unpooling layer is an enlarged, yet sparse activation map. The convolution layer is used to densify the sparse activations obtained as output of unpooling. Each convolution layer performs convolution operation to up-sampled input by a factor of stride value with padding through trainable decoder filter banks. A batch normalization layer is employed after each convolution layer. Inspired by DeepLab [32], bilinear interpolation algorithm is employed in up-sampling layer.

Table 1. CCAE layers configuration

Encoder			
Layer Type	Filter	Output Size	Connected to
Input	–	192 × 192 × 3	–
conv1	3 × 3	192 × 192 × 64	Input
maxpooling1	2 × 2	96 × 96 × 64	conv1
conv2	3 × 3	96 × 96 × 64	maxpooling1
maxpooling2	2 × 2	48 × 48 × 64	conv2
conv3	3 × 3	48 × 48 × 64	maxpooling2
maxpooling3	2 × 2	24 × 24 × 64	conv3
conv4	3 × 3	24 × 24 × 128	maxpooling3
maxpooling4	2 × 2	12 × 12 × 128	conv4
conv5	3 × 3	12 × 12 × 256	maxpooling4
conv6	3 × 3	12 × 12 × 256	conv5
maxpooling5	2 × 2	6 × 6 × 256	conv6
conv7	3 × 3	6 × 6 × 512	maxpooling5
conv8	3 × 3	6 × 6 × 512	conv7
avgpooling1	6 × 6	1 × 1 × 512	conv8
softmax	.		avgpooling1
Decoder			
conv9	1 × 1	1 × 1 × 512	avgpooling1
upsampling1	6 × 6	6 × 6 × 512	conv9
conv10	3 × 3	6 × 6 × 512	upsampling1
conv11	3 × 3	6 × 6 × 512	conv10
upsampling2	2 × 2	12 × 12 × 512	conv11
conv12	3 × 3	12 × 12 × 256	upsampling2
conv13	3 × 3	12 × 12 × 256	conv12
upsampling3	2 × 2	24 × 24 × 256	conv13
conv14	3 × 3	24 × 24 × 128	upsampling3
upsampling4	2 × 2	48 × 48 × 128	conv14
conv15	3 × 3	48 × 48 × 64	upsampling4
upsampling5	2 × 2	96 × 96 × 64	conv15
conv16	3 × 3	96 × 96 × 64	upsampling5
upsampling6	2 × 2	192 × 192 × 64	conv16
conv17	3 × 3	192 × 192 × 3	upsampling6

In decoder module, the trained filters in convolutional layers correspond to bases to reconstruct shape of an input image. Therefore, similar to the encoder module, a hierarchical structure of convolutional layers are used to capture different level of input image details.

3.3 Loss Function

CCAE has two output layers, so it is a challenging task to train this deep network. Let L represents the total loss function of CCAE, which is can be calculated as follow:

$$L = L_r + L_c \tag{5}$$

Where, L_r represents mean square error (MSE) and used to make output of convolutional autoencoder as close as possible to input, as discussed in Eq. (3). L_c represents categorical cross entropy loss used for classification.

$$L_c = -\sum_{i=1}^{N} \sum_{k=1}^{K} t_{ik} log(y_{ik}) \tag{6}$$

Where, t_{ik} is the target, y_{ik} is the calculated output probability. During the training process, in order to minimize loss L, CCAE tries to minimize L_r and L_c simultaneously.

4 Experimental Design

We conducted experiments on underwater synsets of benchmark ImageNet dataset [13] and Fish4Knowledge dataset to demonstrate the effectiveness of proposed method for better classification. We perform comparisons with state-of-the-art deep CNN methods such as ResNet [15], DenseNet [16], Inception [17] and Xception [18].

4.1 Datasets

Experiments are conducted on Fish4Knowledge dataset [14] and underwater synsets of benchmark ImageNet dataset. Scattering and absorption of underwater light causes color distortion and visibility degradation to underwater images. Water impurities and high water density augment the complexity of underwater image classification task. In both datasets images vary significantly in size, object orientation, scale and underwater opacity level [33].

Fish4Knowledge. Fish4Knowledge dataset consists of underwater images of different fish species. The detail of classes included in this dataset is shown in Table 2. This underwater live fish dataset is prepared from live videos recorded from the under sea. There are total 23 species and total 27,370 fish images. Images in dataset are manually labeled by expert marine biologists. Images vary in size ranging from about 20 × 20 to about 200 × 200 pixels. The number of fish images in different classes are quite imbalanced. The number of images in big class is almost 1000 times of the least one. So it is quite difficult to achieve a high accuracy over the whole dataset.

Table 2. Fish4Knowledge dataset

ID	Class	Images	ID	Class	Images
1	Dascyllus reticulatus	12112	13	Plectroglyphidodon dickii	3683
2	Chromis chrysura	3593	14	Amphiprion clarkii	4049
3	Chaetodon lunulatus	253	15	Chaetodon trifascialis	190
4	Myripristis kuntee	450	16	Acanthurus nigrofuscus	218
5	Neoniphon sammara	299	17	Abudefduf vaigiensis	98
6	Hemigymnus fasciatus	241	18	Pomacentrus moluccensis	181
7	Canthigaster valentini	147	19	Hemigymnus melapterus	42
8	Lutjanus fulvus	206	20	Scolopsis bilineata	49
9	Scaridae	56	21	Pempheris vanicolensis	29
10	Zanclus cornutus	21	22	Neoglyphidodon nigroris	16
11	Zebrasoma scopas	90	23	Balistapus undulatus	41
12	Siganus fuscescens	25	...	...	

ImageNet Underwater Synsets. There are forty five (45) synsets related to underwater environment in ImageNet dataset [5]. These synsets consists of images both from underwater and on water surface. The detail of these underwater synsets is shown in Table 3. We take all forty five underwater synsets for experiments, and divided them in three (03) groups i.e. G I, G II and G III. So, each group consists of fifteen (15) synsets. One group is taken at a time for a single experiment.

Table 3. ImageNet underwater synsets

Group I synsets	Images	Group II synsets	Images	Group III synsets	Images
Great White Shark	1242	Lobster	1206	Steller Seal	1342
Scubadiving	1507	Hammer head shark	1094	Liner	1385
Australian Seal	1200	Snorkel Diving	1257	Nuclear submarine	1161
Star Fish	1396	California seal	1245	Whale Shark	1185
Sea Fan	1270	Sea slug(Hermissenda)	2211	Dugong	1018
Attack Submarine	1014	Prawn	1156	Skin diving	1211
Sea snake	1108	Corel Reef	1706	Lion fish	1513
king crab	1019	Wreck	1240	American lobster	1123
Shrimp	1236	Sea pen	1084	Rock crab	1215
Sea turtle	1485	Sea cucumber	1167	Sea urchin	1186
Sea bed	1071	Sea horse	1272	Walrus	1101
Aircraft career	1321	Battle ship	1185	Sea otter	1385
Sea hare	1138	Manatee	1360	Sea slug	711
Chiton	971	Sea cow	424	Bivalve	865
Sea lampery	670	Oster	882	Rockfish	1415

4.2 Parameters Settings

To leverage computational efficiency all images of ImageNet underwater synsets are resized to 192 × 192 pixels. All images of Fish4Knowledge dataset are resized to 96 × 96 pixels.

For fair comparison all methods are trained only on synsets used in experiments without any data augmentation technique. CCAE is trained using ADAM [34] with learning rate=0.001 and batch size 32. For fair comparison, all methods used for comparison are trained with same parameters. All methods run for 120 epochs for ImageNet dataset and for 50 epochs for Fish4Knowledge dataset. In all experiments 70% of the available data is randomly allocated for training and remaining 30% for testing. We implemented CCAE using TensorFlow with Keras as deep learning framework on Google Colab having Tesla K80 GPU.

5 Results and Discussion

5.1 Results

To make comparison, we implemented ResNet [15], DenseNet [16], Inception [17] and Xception [18] as per settings recommended by respective authors.

Table 4 shows the results of classification accuracy, precision, recall and f1-score of all methods for ImageNet underwater synsets of Group I. Results show that proposed approach achieved accuracy of 73.75% and outperformed all other methods. DenseNet performed better than other CNN methods and achieved accuracy of 70.54%. Accuracy of Inception remained low among all methods with score of 58.95%. CCAE achieved f1-score of .72, whereas both DenseNet and Xception net achieved f1-score of .71.

Table 4. Classification accuracy, precision, recall and f1-score comparison for ImageNet synsets group I

Method	Accuracy	Precision	Recall	F1-Score
CCAE	0.7375	0.73	0.72	0.72
DenseNet	0.7054	0.72	0.71	0.71
ResNet	0.6129	0.67	0.63	0.65
Inception	0.5895	0.61	0.60	0.60
Xception	0.6988	0.73	0.70	0.71

Classification accuracy, precision, recall and f1-score of ImageNet synsets of Group II are elaborated in Table 5. Proposed method achieved 70.98% accuracy, best among all methods. Among other methods DenseNet achieved better accuracy of 68.38%. Inception remained again at bottom level with 57.14% accuracy.

Results for classification accuracy, precision, recall and f1-score Group III synsets are depicted in Table 6. Proposed method again achieved best accuracy

Table 5. Classification accuracy, precision, recall and f1-score comparison for ImageNet synsets group II

Method	Accuracy	Precision	Recall	F1-Score
CCAE	0.7098	0.72	0.71	0.71
DenseNet	0.6838	0.70	0.69	0.69
ResNet	0.5966	0.64	0.62	0.63
Inception	0.5714	0.60	0.59	0.59
Xception	0.6718	0.69	0.68	0.68

Table 6. Classification accuracy, precision, recall and f1-score comparison for ImageNet synsets group III

Method	Accuracy	Precision	Recall	F1-Score
CCAE	0.7137	0.71	0.70	0.70
DenseNet	0.6735	0.69	0.69	0.69
ResNet	0.6149	0.65	0.63	0.64
Inception	0.5826	0.58	0.58	0.58
Xception	0.6538	0.66	0.65	0.65

with score 71.37%. DenseNet achieved accuracy of 67.35%, which is better than remaining methods.

Table 7 shows the classification accuracy, precision, recall and f1-score results of Fish4Knowledge dataset. All twenty three (23) classes are taken simultaneously for classification. It is evident that proposed method CCAE performed better than other methods and achieved accuracy of 99.28%. Among other methods DeepFish net with SVM classifier using data augmentation and scaling proposed by Hongwei achieved better accuracy with score 98.64%. Whereas, classification accuracy of 98.59% was achieved by DeepFish net with data augmentation. It is not worthy that our method achieved best accuracy among all methods without using any augmentation technique. Among other deep CNN methods Xception achieved better accuracy of 98.34%. Classification accuracy of Inception remained low among all used methods with scorer of 90.50%. CCAE achieved 0.99 precision score, the highest among all methods. Recall score of three methods such as CCAE, DenseNet and Xception remained same i.e. 0.98. Similarly CCAE, DenseNet and Xception achieved the same f1-score i.e. 0.98.

As observed, proposed method achieves best classification accuracy in all experiments for both datasets, which also highlights the robustness of our method. Whereas, it can also be observed that generally DenseNet performed better classification than ResNet, Inception and Xception in ImageNet underwater synsets classification. Whereas, DeepFish achieved better classification accuracy results as compared to state-of-the-art deep CNN methods. It is noteworthy that all five methods achieved best accuracy results for classification of

Table 7. Classification accuracy, precision, recall and f1-score comparison for Fish4Knowledge dataset

Method	Accuracy.	Precision	Recall	F1-Score
CCAE	0.9928	0.99	0.98	0.98
DenseNet	0.9758	0.98	0.98	0.98
ResNet	0.9480	0.96	0.95	0.95
Inception	0.9050	0.92	0.91	0.91
Xception	0.9834	0.98	0.98	0.98
DeepFish-SVM-aug-scale [3]	0.9864	...	...	...
DeepFish-SVM-aug [3]	0.9859	...	...	...
Deep-CNN [3]	0.9857	...	...	...
DeepFish-Softmax-aug-scale [3]	0.9849	...	...	...

fish4knowledge dataset as compared to ImageNet underwater synsets classification. Whereas, generally all five methods achieved worst accuracy results for classification of ImageNet underwater synsets group II.

5.2 Discussion

CCAE architecture takes advantage of both unsupervised learning as well as the localized spatial features enabled by convolutional filtering. It outperformed other state-of-the-art deep CNN architectures such as Deepfish, Inception, ResNet, DenseNet and Xception. The training of CCAE is carried out by ensemble of a softmax classifier during training time with the convolutional autoencoder architecture. This training strategy enabled the network to extract better feature extraction, which resulted in improved classification accuracy results.

Experiments are also conducted to assess the impact of number of layers and number of channels in each layer, on classification accuracy. Various experiments are conducted to determine the most appropriate number of layers and filters for best classification results. ImageNet underwater synsets of group I are taken for experiments. Table 8 shows the result of comparison of classification accuracy for different number of layers and filters.

Table 8. Impact of no. of layers and no. of filter on accuracy

Sr. no.	Layers	Filters in each layer	Accuracy
1	08	64-64-64-128-256-256-512-512	0.7375
2	08	64-128-256-256-512-512-512-512	0.7216
3	09	64-64-128-256-256-512-512-512-512	0.6269
4	10	x64-64-128-128-256-256-512-512-512-512	0.6114

Results as shown in Table 8 suggest that best classification accuracy with score 73.75% is achieved with CCAE encoder architecture which consists of 08 convolution layers (serial no 1 in Table 8). When number of channels in few layers changed by keeping the number of layers same, as shown at serial no 2 of Table 8, the classification accuracy of 72.16% achieved. This configuration of layers and channels in the encoder module is similar to VGG11 architecture. Classification accuracy keeps decreasing when one more convolution layer having 64 channels added to the serial 2 configuration. New configuration is shown at serial number 3 in Table 8. Configuration of layers and channels of the encoder at serial no 4 in Table 8 is similar to the VGG13 architecture. It consists of 10 convolution layers. Classification accuracy of 61.14% is achieved by using this configuration. From this experiment it can be inferred that increasing number of channels and number of layers in design of the encoder of CCAE does not result in increase in classification accuracy. And CCAE encoder architecture is most optimized to achieve best classification accuracy results.

Max pooling layer is used after every convolution layer of the encoder module. Avg pooling layer is used after last convolution layer of the encoder module. Two commonly used pooling layers in deep CNN are avg pooling layer and max pooling layer. We conducted the experiments to find better pooling layer to be used after last convolution layer of encoder. It was found during the experiments that classification accuracy of 73.75% was achieved by using Avg pooling layer. Whereas, classification accuracy dropped to 57.38% when max pooling layer is used after last convolution layer of the encoder module. It can be inferred that important information of features of the input image is retained by using avg pooling layer at the end of last convolution layer of the encoder module, which in result improves the classification accuracy.

6 Conclusion

In this paper, a novel feature extraction technique is proposed to improve the accuracy of image classifiers. Proposed model uses the strength of unsupervised deep convolutional autoencoder to learn useful features from large training data set to train a supervised softmax classifier. Experiments on underwater image data sets demonstrate that proposed model has remarkably improved classification accuracy. Classification accuracy, precision, recall and f1-score results showed that proposed model has out performed state-of-the-art deep CNN methods. The proposed model can easily be used to other image classification and object recognition tasks.

References

1. Schettini, R., Corchs, S.: Underwater image processing: state of the art of restoration and image enhancement methods. EURASIP J. Adv. Signal Process. **2010**(1), 1–14 (2010). 746052

2. Anwar, S., Li, C., Porikli, F.: Deep Underwater Image Enhancement. CoRR abs/1807.03528 (2018). arXiv:1807.03528
3. Qin, H., et al.: DeepFish: accurate underwater live fish recognition with a deep architecture. Neurocomputing **187**(C), 49–58 (2016)
4. Sun, X., et al.: Transferring deep knowledge for object recognition in low-quality underwater videos. Neurocomputing **275**(C), 897–908 (2018)
5. Xu, Y., et al.: Underwater image classification using deep convolutional neural networks and data augmentation. In: 2017 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC), pp. 1–5 (2017)
6. Siddiqui, S.A., et al.: Automatic fish species classification in underwater videos: exploiting pre-trained deep neural network models to compensate for limited labelled data. ICES J. Mar. Sci. **75**(1), 374–389 (2017)
7. Jin, L., Liang, H.: Deep learning for underwater image recognition in small sample size situations. In: OCEANS 2017, Aberdeen, pp. 1–4 (2017)
8. Noh, H., Hong, S., Han, B.: Learning deconvolution network for semantic segmentation. In: Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), ICCV 2015, pp. 1520–1528. IEEE Computer Society, Washington (2015)
9. Luo, W., et al.: Convolutional sparse autoencoders for image classification. IEEE Trans. Neural Netw. Learn. Syst. **29**(7), 3289–3294 (2018)
10. Guo, Y., et al.: Deep learning for visual understanding. Neurocomputing **187**(C), 27–48 (2016)
11. Masci, J., Meier, U., Cireşan, D., Schmidhuber, J.: Stacked convolutional auto-encoders for hierarchical feature extraction. In: Honkela, T., Duch, W., Girolami, M., Kaski, S. (eds.) ICANN 2011, part I. LNCS, vol. 6791, pp. 52–59. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21735-7_7
12. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science **313**(5786), 504–507 (2006)
13. Deng, J., et al.: ImageNet: a large-scale hierarchical image database. In: CVPR 2009 (2009)
14. Boom, B.J., et al.: Supporting ground-truth annotation of image datasets using clustering. In: 2012 21st International Conference on Pattern Recognition (ICPR 2012), November 2012, pp. 1542–1545. IEEE Computer Society, Los Alamitos (2012)
15. He, K., et al.: Deep residual learning for image recognition. CoRR abs/1512.03385 (2015). arXiv: 1512.03385
16. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. CoRR abs/1608.06993 (2016). arXiv:1608. 06993
17. Szegedy, C., et al.: Going deeper with convolutions. CoRR abs/1409.4842 (2014). arXiv: 1409.4842
18. Chollet, F.: Xception: deep learning with depthwise separable convolutions. CoRR abs/1610.02357 (2016). arXiv: 1610.02357
19. Zhang, C., et al.: Deep sparse autoencoder for feature extraction and diagnosis of locomotive adhesion status. J. Control. Sci. Eng. **2018**, 8676387:1–8676387:9 (2018)
20. Yanming, G., et al.: A review of semantic segmentation using deep neural networks. Int. J. Multimedia Inf. Retrieval **7**(2), 87–93 (2018)
21. Baldi, P.: Autoencoders, unsupervised learning and deep architectures. In: Proceedings of the 2011 International Conference on Unsupervised and Transfer Learning Workshop - Volume 27, UTLW 2011, Washington, USA, pp. 37–50. JMLR.org (2011)

22. Szegedy, C., et al.: Going deeper with convolutions. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1–9 (2015)
23. Luo, W., et al.: Convolutional sparse autoencoders for image classification. IEEE Trans. Neural Netw. Learn. Syst. **29**, 3289–3294 (2018)
24. Badrinarayanan, V., Kendall, A., Cipolla, R.: SegNet: a deep convolutional encoder-decoder architecture for image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **39**(12), 2481–2495 (2017)
25. Zhang, H., Patel, V.M.: Densely connected pyramid dehazing network. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018
26. Guo, X., et al.: Deep clustering with convolutional autoencoders. In: ICONIP (2017)
27. Vincent, P., et al.: Extracting and composing robust features with denoising autoencoders. In: Proceedings of the 25th International Conference on Machine Learning, Helsinki, Finland, ICML 2008, pp. 1096–1103. ACM (2008)
28. Maas, A.L., Hannun, A.Y., Ng, A.Y.: Rectifier nonlinearities improve neural network acoustic models. In: ICML Workshop on Deep Learning for Audio, Speech and Language Processing (2013)
29. Santurkar, S., et al.: How does batch normalization help optimization? In: Bengio, S., et al. (eds.) Advances in Neural Information Processing Systems 31, pp. 2483–2493. Curran Associates Inc., (2018)
30. Jiuxiang, G., et al.: Recent advances in convolutional neural networks. Pattern Recogn. **77**(C), 354–377 (2018)
31. Dumoulin, V., Visin, F.: A guide to convolution arithmetic for deep learning. CoRR abs/1603.07285 (2016)
32. Chen, L.-C., et al.: DeepLab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. IEEE Trans. Pattern Anal. Mach. Intell. **40**(4), 834–848 (2018)
33. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Pereira, F., et al. (eds.) Advances in Neural Information Processing Systems 25, pp. 1097–1105. Curran Associates Inc., (2012)
34. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. CoRR abs/1412.6980 (2015)