A Survey of Deep Learning Techniques for Underwater Image Classification

Preprint · February 2021

DOI: 10.13140/RG.2.2.25098.59846

3 authors, including:

Sparsh Mittal
Indian Institute of Technology Roorkee
222 PUBLICATIONS **5,904** CITATIONS

SEE PROFILE

Srishti Srivastava
PolyMage Labs
5 PUBLICATIONS **174** CITATIONS

SEE PROFILE

A Survey of Deep Learning Techniques for Underwater Image Classification

Sparsh Mittal[1], *IEEE Senior Member*, Srishti Srivastava[2], J Phani Jayanth[3]

***Abstract*—In recent years, there has been an enormous interest in using deep learning to classify underwater images to identify various objects like fishes, plankton, coral reefs, seagrass, submarines, and gestures of sea-divers. This classification is essential for measuring the water bodies' health and quality and protecting the endangered species. Further, it has applications in oceanography, marine economy and defense, environment protection, and underwater exploration and human-robot collaborative tasks. This paper presents a survey of deep learning techniques for performing the underwater image classification. We underscore the similarities and differences of several methods. We believe that underwater image classification is one of the killer application that would test the ultimate success of deep learning techniques. Towards realizing that goal, this survey seeks to inform researchers about state-of-the-art on deep learning on underwater images and also motivate them to push its frontiers forward.**

***Index Terms*—Deep neural networks, artificial intelligence, autonomous underwater vehicle, transfer learning.**

1. Introduction

Recent years have witnessed a massive growth in the interest in processing underwater images. Studying the behavior and number of various species of aquatic plants and animals is helpful in marine biology, economy and biodiversity management. It can help in analyzing the differences in species and protecting endangered species. For example, plankton have high sensitivity to the changes in surroundings and the environment. Hence, the study of their well-being provides an early indication of climatic events, e.g., pollution and global warming. They are a crucial link in the ocean food chain and connect the ocean to the atmosphere. Plankton produces more than 80% of the world's oxygen, and hence, a low level of plankton is harmful. At the same time, an excessive amount of plankton leads to toxins. Therefore, the level of plankton needs to be carefully controlled.

Similarly, Posidonia Oceanic live only in clean water and contribute to biodiversity, reduce erosion of beaches, and enhance water quality. Studying the well-being of underwater organisms can help analyze the impact of global warming and excessive human activity on the water bodies and marine life, thus guiding preservation campaigns. Image processing can complement other techniques such as physio-chemical analysis of water and sonar-based detection.

Challenges in the automated classification of underwater images: Although essential, the automated classification of underwater images is fraught with many challenges. Energy loss during the propagation of light diminishes its intensity. This decrease leads to low and variable illumination and visibility, especially in deeper waters. Additionally, compared to still waters such as ponds or swimming pools, oceans have ocean currents, which change the luminosity. These changes and the impurities and suspended solids lead to complex noise in underwater images, especially in ocean images. These images also have low contrast and deteriorated edges and details. Further, the non-uniform spectral propagation distorts color depending on the distance. To mitigate some of these limitations, sophisticated yet costly cameras are required.

Underwater images are unconstrained and have complex backgrounds. For example, sea cucumbers' color is inherently similar to its surroundings, such as sediments and seaweed. This feature complicates their separation from their background. Fishes move freely and quickly in 3D space. They tend to hide behind other fishes, corals or sand. Hence, determining their pose/orientation/size and performing semantic segmentation is challenging. The significant variations in the mix of animal/plant life lead to large variations in the backgrounds. Further, even foreground appearance can vary due to variations in transparency, color and dissolved objects. Visual classification of plankton is challenging due to the presence of several phylogenetic species in plankton images and the tiny size of organisms that form plankton. In underwater exploration missions, sea divers' gestures need to be recognized in a wide range of environments/situations and from multiple distances and viewpoints. Also, differentiating the gestures from regular driving motions is difficult.

Furthermore, large-scale public datasets are not available for underwater images. The existing datasets are highly imbalanced and need data augmentation. For example, in F4K dataset, the number of images of different classes may differ by $1000\times$ [45]. Scarce training data makes it infeasible to train a deep CNN [17, 42]. Similarly, in coral reef image datasets, non-coral elements appear abundantly while coral elements are limited. Further, the distinction between different coral reef classes is not strict, but significant differences exist within a single class. This motivates the need for fine-grained classification, which has received less attention than coarse-grained classification. Underwater imaging systems have lower resolution than those used for ground imaging. Hence, they have limited distinguishing features, which can confuse conventional techniques or even humans.

Limitations of conventional computer-vision techniques: Conventional techniques use hand-crafted features, which capture visual characteristics such as blobs, texture, curves, edges

[1]IIT Roorkee, [2]IIT Dharwad, [3]IIT Madras, India. Dr Sparsh is the corresponding author (sparshfec@iitr.ac.in). Sparsh and Srishti are co-first authors. "This work was supported in part by Semiconductor Research Corporation (SRC), grant number 2020-IR-2972".

or corners. Examples of such features are SIFT[1], HOG and and "local binary patterns". These features do not generalize to different classes, scenarios and datasets. Their accuracy saturates with the increasing size of the training data. Also, extracting these features requires domain expertise and time. Hence, most of the previous work on fish recognition has been done on dead fishes, fishes taken outside water or in unnatural conditions, e.g., swimming pools or tanks with sufficient lighting. Overall, conventional techniques provide low accuracy and hence, are ineffective.

Contributions: The success of deep learning models has motivated researchers to apply them for underwater image processing. In fact, CNNs have already shown better predictive performance than conventional image-processing or machine learning techniques [7, 10, 30, 33, 45–48] and even humans [31]. In this paper, we present a survey of deep learning techniques for underwater image classification. Figure 1 gives an overview of the rest of the paper. Section 2 presents a classification of research works. Section 3 reviews the techniques that focus on image pre-processing, training and the datasets used for training. Section 4 discusses the techniques for designing and optimizing CNNs. Finally, Section 5 concludes this paper with a discussion of future works.

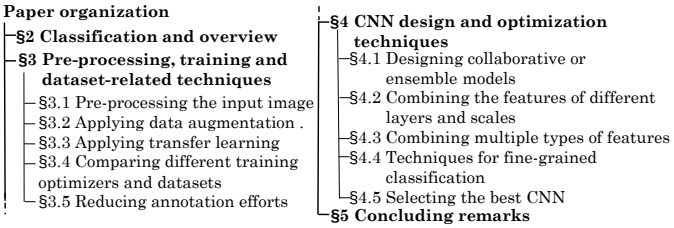

Fig. 1: Organization of the paper

2. Classification and overview

Table I classifies the works based on the key challenge addressed by them and the importance of that challenge or design-issue.

Table II classifies the works based on the characteristics of the test dataset and the evaluation metrics used by them. It also shows the FPS values achieved and the deep learning framework used by them. Table III summarizes the properties of different underwater datasets. Table IV shows various hyperparameters and their values.

[1]The following acronyms are used frequently: "adaptive moment estimation" (Adam), "area under the ROC curve" (AUC), "autonomous underwater vehicles" (AUVs), "average precision" (AP), "average recall" (AR), "bag of visual words" (BOVW), batch normalization (BN), "cognitive autonomous diving buddy underwater gestures" (CADDY-UG), "convolution" (CONV), "artificial/convolutional neural network" (ANN/CNN), "correct classification rate" (CCR), "deconvolutional" (DECONV), "fast direct super-resolution" (FDSR), "Fish4Knowledge" (F4K), "frames per second" (FPS), "fully connected" (FC), generative adversarial network (GAN), "histogram of oriented gradients" (HOG), "k-nearest neighbour" (k-NN), "LifeCLEF 2014/2015 Fish" (LifeCLEF14/15), "matrix power normalized co-variance" (MPN-COV), "mean/peak signal to noise ratio" (MSNR/PSNR), "Moorea labeled coral" (MLC), "multi-layer perceptron" (MLP), "network in network" (NiN), "principal component analysis" (PCA), "root mean square propagation" (RMSProp), "scale-invariant feature transform" (SIFT), "sparse representation classifier" (SRC), "squeeze-and-excitation" (SE), "stochastic gradient descent" (SGD), and "support vector machine" (SVM).

TABLE I: Important aspects considered by the authors.

Important Aspect	Why it is important
Pre-processing input images [2, 5, 9, 13, 26, 34, 36, 39, 45, 46]	Complex nature/ low-resolution of underwater images make further classification hard
Transfer learning [2, 3, 5, 8–11, 18, 20–22, 29, 30, 30, 31, 34, 37, 43, 46–48]	Reducing training overheads and utilizing pretrained CNNs
Data augmentation [2–4, 7–10, 14, 15, 20, 21, 24, 26, 27, 30, 31, 37, 39, 42, 44, 45, 47]	Underwater datasets are unbalanced and not large enough for training deep networks
Efficiently extracting input image features [5, 6, 11, 13, 15, 20, 24, 25, 47]	Effective feature extraction leads to accurate classification
Design-space exploration to find the optimal configuration/network [2–5, 8, 11, 17, 21–23, 26, 27, 29, 33, 39, 42–47]	Optimization objectives of different scenarios are different (e.g., accuracy or speed)

TABLE II: Test dataset characteristics and evaluation metrics.

Category	References
Input image	Grayscale [6, 9, 13, 15, 18, 23–26, 26, 27, 39, 42, 47], color [2–5, 8, 10, 11, 14, 17, 20–22, 27, 29–31, 33, 34, 36, 37, 43–46, 48]
Input image (in pixels)	256x256 [7, 39, 42, 46], 227x227 [30], 224x224 [4, 6, 43], 222x222 [23], 200x200 [20], 128x128 [3, 7, 44], 100x100 [24, 36], 64x101 [8], 64x64 [44], 48x48 [14], 47x47 [45], 32x32 [33, 47, 48]
Dataset used for testing	F4K [4, 11, 14, 20, 21, 30, 34, 36, 45], LifeCLEF15 [30, 47, 48], SIPPER [23], CADDY-UG [10, 22], MLC [43], WHOI-Plankton [18, 25, 40], ZooScan [18, 39], LifeCLEF14 [47], underwater images from ILSVRC [37], temperate fish species [20], ILES [24], CZECH [24], ZOOVIS [13], Fish-gres [4], ANT-XXVIII/2 and PS103: [9], ISIIS [26], PlanktonSet 1.0 [6, 44], Zooglider [15], ZooLake [3], FlowCam imaging system [8]
Dataset collected from	sea [2, 4, 6, 8, 9, 13–15, 25, 26, 30, 31, 33, 34, 39, 44], lake [3, 17, 24], test tube filled with alcohol [46], internet [27, 29]
Number of classes	1611 [5], 121 [6, 42, 44], 108 [26], 104 [8], 103 [25], 77 [23], 52 [23], 51 [9], 45 [37], 35 [3], 29 [46], 27 [15], 23 [4, 11, 14, 21, 30, 45], 21 [31, 36], 17 [10], 16 [22], 15 [30, 47, 48], 13 [39], 10 [34, 47], 9 [43], 8 [4], 7 [13, 23], 4 [20, 24, 27], 3 [2, 29], 2 [17, 33]
Objects classified	fish species [2, 4, 11, 14, 20, 21, 27, 30, 31, 34, 36, 45, 47, 48], phyto/zoo-plankton species [3, 5–8, 13, 15, 18, 23–26, 39, 42, 44], Diatoms [9], Posidonia Oceanica meadows [33], coral reefs [43], macroinvertebrate benthos [46], sea cucumbers [17], underwater objects [29, 37], diver gestures [10, 22],
Metrics used (Accuracy = Classification accuracy)	
Accuracy	[2–8, 10–12, 14, 15, 17, 18, 20, 21, 23–25, 29, 30, 34, 36, 39, 40, 42, 43, 45, 46]
Precision/recall/F1	precision/recall [2, 3, 5, 5, 11, 13, 14, 17, 18, 26, 30, 47, 48], "average class precision" (ACP) [26, 43], F1-score [2, 3, 5, 8, 9, 11, 14, 17, 18, 24, 26]
Others	CCR [22, 31], PSNR [34], detection hit ratio (TP/P) [33], recognition rate [27], AUC [18], average count [47], predicted class score [37], kappa [8, 11], inference time [4, 13], softmax loss [44]
Training time	[22, 27, 33, 39], training time per epoch [20]
FPS	[10, 22, 27, 31, 36, 42, 47]
FPS achieved	≤ 10 [42], > 10 and ≤ 100 [10, 22, 27, 31], > 100 [36, 47]
Framework/library	Caffe [21, 31, 34, 37, 44, 46, 48], MATLAB [6, 13, 18, 22, 29, 30, 45, 47, 47], LibSVM [46–48], LibLinear [45], Tensorflow [4, 6], Keras [4, 11, 14], Theano [15]

TABLE III: Commonly used underwater datasets

Name	Object	#Class	#Image	Where collected
Sipper [57]	Zooplankton	81	>750K	Gulf of Mexico
WHOI [53]	Plankton	70	>3.4M	Atlantic ocean
ZooScan [51]	Zooplankton	20	3,771	Bay of Villefranche
ZOOVIS [32]	Zooplankton	6	>685K	Bering Sea
ISIIS [58]	Plankton	108	>42K	Gulf of Mexico
PlanktonSet [50]	Plankton	121	>60K	Straits of Florida
ZooLake [3]	Plankton	35	17,943	Greifensee lake
F4K [49]	Tropical Fish	23	27,370	Taiwan sea
LifeCLEF14	Tropical Fish	10	19,868	Subset of F4K
LifeCLEF15	Tropical Fish	15	>20K	Subset of F4K
Temperate fish [20]	Temperate Fish	4	619	Southern Norway
Fish-gres [1]	Fish	8	3248	Gresik
MLC [56]	Coral Reef	9	2055	Moorea
CADDY [16]	Diver gestures	16	10,322	Croatia sea, pool

TABLE IV: Hyperparameters set by the authors.

Category	References
Training epochs	10 [22, 24], 13 [8], 25 [30], 30 [3, 18], 40 [15], 50 [9, 20], 60 [4], 70 [7], 136 [23], 148 [23], 150 [23, 26], 200 [17], 300 [11]
Base learning rate	1e-5 [3, 4], 0.0001 [2, 12, 15, 24, 29], 0.001 [2, 8, 11, 18, 20, 22, 29, 34, 42], 0.01 [2, 7, 45], 0.05 [17], 0.1 [17]
Momentum	0.1 [29], 0.2 [29], 0.9 [7, 34, 42, 45]
Mini-batch size	8 [2, 20], 10 [29], 16-128 [18], 20 [4, 29], 32 [9, 11, 24], 40 [34], 64 [2, 45], 100 [17], 256 [7], 500 [8, 22]
Others	Weight decay: $4e-5$ [24], 0.0005 [42, 45], 0.005 [34], dropout ratio: 80% [36], 50% [7, 15, 24], 30% [3], 20% [23]

3. Pre-processing, training and dataset-related techniques

Given the poor quality of underwater images (Section 1), pre-processing them is absolutely essential. The scarcity of underwater datasets and the high class imbalance necessitate the use of data augmentation and transfer learning. Transfer learning also reduces the computational requirements during training. Similarly, the microscopic size of objects/organisms in underwater images, along with the scarcity of dataset, necessitates reducing the annotation effforts. We now discuss the techniques that apply pre-processing to the images (Section 3.1), use data augmentation (Section 3.2) and employ transfer learning (Section 3.3). Then, we review the works that compare different training optimizers and datasets (Section 3.4). Finally, we review techniques for reducing annotation efforts (Section 3.5). Table V highlights the approaches used for image pre/post-processing and data augmentation.

A. Pre-processing the input image

Jin et al. [34] propose a model for live-fish species classification. They first de-noise the image while preserving its essential details. The standard median filtering technique computes the median of the value of the pixels in a sliding window and replaces the original value of the pixel in the center of the window with this median value. It helps in suppressing the noise. Unlike standard median filtering, which processes all the pixels of an image, they first detect the polluted pixels and then process only the pixels polluted by impulse noise.

To detect polluted pixels, they first classify all the pixels with values not equal to 0 or 255 as unpolluted. This classification is done because impulse noise usually changes a pixel's value to either 0 or 255. Then, the image is processed using a 5x5 standard median filter to obtain an initial processed image. The pixel values of the initial processed image are subtracted from the corresponding pixel values of the original image. The average pixel value obtained after this subtraction is used as a threshold. In the original image, all the pixels that have not already been classified as unpolluted are examined. Among these, all pixels with values greater than the threshold are classified as polluted. Then, the polluted pixels are processed using a standard median filter of size 5x5 pixels to obtain the final de-noised image.

TABLE V: Image processing and augmentation techniques.

Pre-processing techniques (1 to 4)	
1. Denoising	improved median filtering method [34], morphological opening, and pyramid mean shifting [36], connected component analysis and cropping [46], Otsu's global thresholding [13, 24, 36, 46], bilateral filter [40], removing non-uniformities using flat-fielding [26], Gaussian filtering [25], thresholding [5, 34]
2. Image-scaling	bicubic interpolation [7], bilinear transformation [47], SqR and Pad [18]
3. Image sharpening	modified Laplacian kernel [14], contrast-enhancement [2, 40], ROI enhancement [13], logarithmic image enhancement [25], Scharr operator for highlighting shape [40]
4. Foreground extraction	sparse and low-rank matrix decompostion [45], Suzuki and Abe's algorithm [5], colony merging algorithm [5] Segment high and low contrast images using MSER and Sauvola methods respectively [13], Extracting texture using Canny edge detector [40]
Post-processing	changing output class based on confidence score [31], not making any prediction for low-confidence images [26], using confidence based criteria in active learning [24]
Data augmentation techniques	Rotation [2–4, 7–10, 14, 20, 24, 26, 27, 39, 42, 44, 45], flipping [2–4, 7–9, 15, 20, 21, 24, 30, 31, 39], translation [2, 4, 8, 10, 20, 39, 42, 44], rescaling [20, 26, 30, 39, 42], cropping [7, 21, 31], shearing [2–4, 8, 9, 20, 39], zooming [2, 3, 8, 9], width and height shifting [9], stretching [42], blurring using Gaussian filter [27, 47], downsampling and affine transformation [21], image distortion, noise addition, changing of light intensities, sharpening [47], scale, aspect ratio and color augmentations, photometric distortions, artificial image generation using a GAN [37]

They use an AlexNet-like CNN with ten output classes. To avoid overfitting on a small-sized dataset, this CNN is first pretrained on the large-sized ILSVRC dataset. This pretraining makes the CNN sensitive to edges, textures, and colors in natural images. Then, the CNN is finetuned on the smaller-sized F4K dataset. The proposed two-step denoising strategy achieves a PSNR of 24.36 dB, which is much better than the values 22.40 dB and 21.25 dB achieved by the standard 3x3 and 5x5 median filtering strategies, respectively. Further, the CNN achieves a classification accuracy of 85.08%.

Rathi et al. [36] propose a CNN for live-fish classification. They first pre-process the images to de-noise them by removing obstacles, such as non-fish objects and dirt. The pre-processing step consists of Otsu's thresholding, morphological erosion followed by morphological dilation, and pyramid mean shifting, in that order. Morphological erosion decreases the thickness of objects, thereby eliminating small objects. However, it also breaks a few objects. Morphological dilation helps join the broken parts of the object, most notably of the live fish. The input to their CNN is an image of size $100 \times 100 \times 4$ pixels, which is the $100 \times 100 \times 3$ sized, original color-image stacked with the $100 \times 100 \times 1$ sized output of the

pre-processing step. The CNN has three CONV blocks, which have CONV layers with 5x5 filters, ReLU and max-pooling layers. Then, there are two FC layers. Their CNN achieves an accuracy of 96.3% and an FPS of 546 on the F4K dataset.

Deep et al. [14] propose a CNN model and two CNN+shallow models for live-fish species classification. In their dataset, most images are noise-free but blurry. Hence, they first pre-process images using an image sharpening method to enhance the edges in the images. They employ a slightly modified Laplacian kernel, in which the sum of all the kernel elements is one, instead of zero. This kernel produces a color image while the original Laplacian kernel produces a binary image. After this pre-processing, they obtain a sharpened image of size 48×48 pixels, which is fed to the CNN for feature extraction.

The CNN has three CONV layers, followed by max-pooling since they ensure better detection of edges than average-pooling. Average-pooling averages the pixel values and hence fails to provide accurate detection of edges. Then, there are two FC layers, each preceded by a dropout layer to avoid overfitting. Finally, there is a softmax activation function. In addition, they propose two models that use CNN for feature extraction and an SVM or a k-NN algorithm for classification. These models are called CNN-SVM and CNN-kNN, respectively. SVM uses a linear kernel with the "one vs. rest strategy". On the F4K dataset, their CNN, CNN-kNN and CNN-SVM models achieve accuracies of 98.6%, 98.8% and 98.3%, respectively. By contrast, random forest, kNN and SVM models achieve accuracies of 81.4%, 79.0% and 78.8%, respectively.

Sun et al. [48] propose a model to classify low-resolution underwater images. They use FDSR method to convert "low-resolution" images to "high-resolution images". This method is trained using a self-training strategy. To extract distinctive features from these high-resolution images, they use PCANet and NiN. PCANet and NiN are simpler than CNNs and hence, require less training time. They employ pretrained NiN and finetune it on the LifeCLEF15 dataset. The extracted features from each deep network are fed to a linear SVM, which performs the final classification. The proposed models are FDSR+PCANet+SVM and FDSR+NiN+SVM, and are shown in Figure 2.

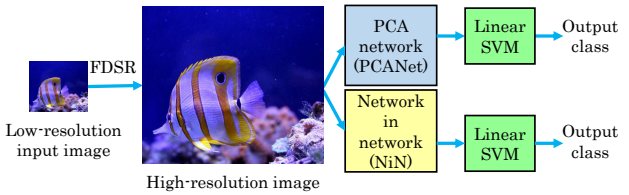

Fig. 2: Two models proposed by Sun et al. [48].

They compare the features extracted by PCANet and NiN with two other features: "gabor" and "dense SIFT". Using these, following models are formed: FDSR+Gabor+SVM, FDSR+DenseSIFT+SVM, PCANet+SVM, NiN+SVM, Gabor+SVM, and DenseSIFT+SVM. The results are summarized in Table VI. The precision values achieved by FDSR+PCANet+SVM are higher compared to those achieved by PCANet+SVM. Also, the best precision values are obtained using a color-image of size 32×32 as input. Further, the precision values achieved by FDSR+NiN+SVM, FDSR+Gabor+SVM, FDSR+DenseSIFT+SVM, NiN+SVM, Gabor+SVM, and DenseSIFT+SVM are 69.8%, 38.3%, 28.6%, 68.3%, 23.6%, and 21.4%, respectively.

TABLE VI: Precision results of Sun et al. [48].

Image	FDSR+PCANet+SVM	PCANet+SVM
28×28, grayscale	67.82%	67.77%
32×32, color	77.27%	75.63%
40×40, color	76.57%	74.44%

Cheng et al. [13] employ techniques for enhancing the ROIs from plankton images by alleviating background noise and intensifying target features. Otsu's global threshold method to extract ROIs misses several target objects. Further, "maximally stable extremal regions" (MSER) and "Sauvola's local binarization" methods can result in more than desired ROIs due to non-uniform illumination and variable "contrast ratio" within the same image. To avoid this, they segment images with "high contrast" (MSNR>0.1) using the MSER technique and those with low contrast (MSNR<=0.1) using Sauvola's method.

Then, they employ a denoising algorithm for noise suppression. A small rectangular window is slid across the image to calculate a suitable threshold value and identify biological boundary feature points. Based on the fact that a true feature point usually has more feature points around it, the algorithm identifies valid/true feature points and discards the background noise. After segmentation, they nullify the background pixels. Then, they apply a grayscale transformation, where pixels lower than a suitable threshold value are attenuated, and those higher than the threshold are amplified. The enhanced ROIs are fed into a CNN for "feature extraction", and the output from FC layers is passed to a multi-class SVM model to obtain the prediction. Their architecture is shown in Figure 3.

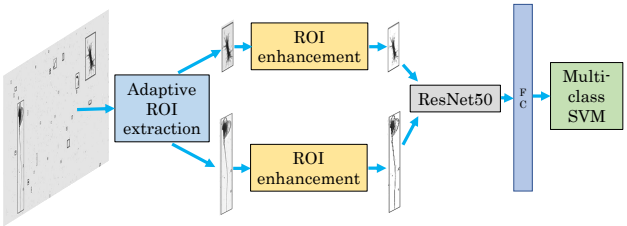

Fig. 3: Architecture of Cheng et al. [13]

The dataset has 685,520 images across seven classes. As for CNN, they evaluate AlexNet, VGG, GoogLeNet, and ResNet. The CNN + multi-class SVM model achieves a recall rate of 94.13% compared to 92.76% by the model without ROI enhancement. The best model, ResNet50 + SVM with ROI enhancement, achieves precision and recall rates of 94.52% and 94.13%, respectively.

Luo et al. [26] evaluate "preprocessing", "segmentation", and postprocessing for accurate classification across 108 plankton classes. The authors use grayscale images, which have noise and non-uniformities across gray-level and contrast. The authors employ radiometric calibration called *flat-fielding* to overcome the non-uniformities. Flat-fielding subtracts a

calculated calibration frame from the raw image. They use "histogram normalization" to normalize the contrast in each frame, which allows better segmentation of ROIs. They discard extremely noisy images, i.e., those with a signal-to-noise (SNR) ratio below 25. Then, they utilize K-harmonic means clustering for detecting and segmenting the ROIs.

In plankton images, background pixels are more numerous than plankton pixels. Hence, they use a sparseConvNet which ignores background pixels to improve efficiency. Their sparseConvNet has 13 CONV layers separated by 12 fractional max-pooling layers with a scaling factor of $1/\sqrt{2}$. The model is then followed by "multinomial logistic regression" for output class prediction. They employ probability filtering, which categorizes "low-confidence" images into an 'unknown' category. This filtering removes 30% images. This allows for an overall increase in F1-Score and precision, though it reduces the recall rate by 23 percentage points. The model achieves precision and recall rates of 84% and 40%, respectively.

Kloster et al. [9] propose a CNN-based method for taxonomic classification of microalgal groups of diatoms. The authors employ several techniques to obtain high-quality image data from microscopic slides of diatoms. They apply a focus stacking technique which acquires focus-enhanced image regions from each diatom slide. These highly enhanced image regions are combined into gigapixel-sized 'virtual slides' by slide stitching.

Then, they annotate objects of interest (OOIs) in each virtual slide. They employ automatic segmentation software to refine the OOIs to the exact object shapes. The authors extract 10K OOIs across 51 classes for training the classifier. They perform data augmentation by rotation, shift, shear, zoom, and flipping on the training data. Using a VGG with one FC layer, their background masking technique increases the F1-score from 97% to 99%.

Dai et al. [39] deal with the challenge of a small dataset with class imbalance. Their dataset contains zooplankton images of sizes ranging from 40x40 pixels to 400x400 pixels. They first rescale the images to a size of 256x256 pixels and then normalize them by subtracting the mean value over the training set from each pixel. They also use data augmentation. Their CNN has 8 CONV and 3 FC layers. Parametric ReLU (PReLU) and local response normalization are used to improve performance. Their CNN achieves an accuracy of 93.7%, whereas GoogleNet and VGG achieve 91.9% and 92.2%, respectively.

B. Applying data augmentation

Villon et al. [31] use GoogLeNet to classify images of live-fish species. They do not apply background subtraction as a pre-processing step because background knowledge can help correctly classify fishes associated with their environments. Also, automated background subtraction can be inaccurate for generic purposes and requires human intervention. To avoid overfitting, they use dropout while training. To design robust CNN, they collect a variety of underwater images covering 20 fish species. These images differ in lighting conditions, visibility, depth, and the amount of soft and hard corals in the background. They call this collected dataset T0. From T0, they build four different datasets for realizing various benefits, as explained in Table VII. The "part of fish" and "part of species" classes are described for two sample images of different species in Figure 4.

TABLE VII: The datasets designed by Villon et al. [31] and their benefits. Ti inherits the benefits of T$(i-1)$.

Dataset	How the dataset is designed	Benefits of using the dataset
T1	By taking each image of the T0 dataset along with its horizontally-flipped version.	Helps CNN to correctly classify a fish irrespective of its swimming direction (left or right) with respect to the camera.
T2	By taking each image of the T1 dataset along with a separate image for each of its four parts, namely, "upper", "lower", "left", and "right". Each of these four parts is kept into a single class called "part of fish".	Teaches the CNN that it should not wrongly classify a fish based on partial information. It will learn to classify each image with partial information as "part of fish", rather than classifying it into a wrong category.
T3	By taking each image of the T2 dataset along with the images from the "environment" class (that do not contain any fish).	Helps CNN to correctly classify a fish irrespective of its surroundings.
T4	By removing "part of fish" class from T3 and replacing it with 20 "part of species" classes, one for each species.	Helps CNN to correctly recognize a fish species even if it is partially occluded by say, a coral.

The authors train their network separately on each of the four datasets. These networks are referred to as N1 to N4. T2, T3, and T4 datasets have many images corresponding to the "environment" class. Hence, N2, N3 and N4 tend to classify a fish image as "environment" with a high confidence score (up to 99%). To address this issue, they propose two post-processing rules, which are summarized in Table VIII.

TABLE VIII: Post-processing rules of Villon et al. [31].

Rule	Applies to	Description
R1	T2, T3, and T4	When the "environment" class's confidence score is the highest but is less than 99%, the final output is given as the class with the second-highest score
R2	T4	If the "part of species X" class has the highest score, then the final output is given as the "species X" class

Before applying the post-processing rules, the CCR values obtained by N1, N2, N3, and N4 are 87.6%, 87.9%, 87.7%, and 86.9%, respectively. N4 achieves the CCR value of 90.2% on applying the R1 rule and 94.1% on applying both R1 and R2 rules. On another test set, this network with R1 and R2 rules achieves 95.7% CCR, whereas humans (biology students) achieve an average CCR value of 89.3%. Their network performs better than humans when the images are tiny or blurry.

Xu et al. [37] use GoogLeNet to classify underwater images taken from the ILSVRC dataset. From these images, they create a dataset comprising 45 classes of underwater objects. To prevent overfitting, the authors compare various methods: L1 and L2 regularization, max-norm regularization and dropout. Dropout provides the best performance, and hence, they use it in GoogLeNet.

As for data augmentation, their first technique doubles the training dataset size, using the scale, aspect ratio and color augmentations, and photometric distortions [55]. The second

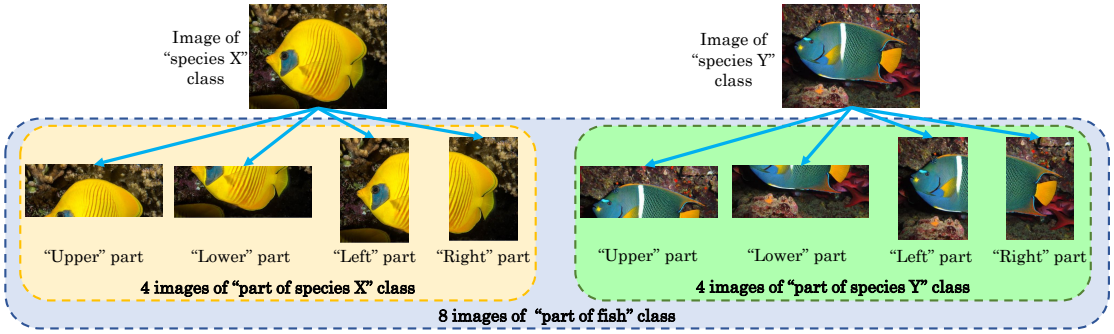

Fig. 4: The "part of fish" and "part of species" classes of Villon et al. [31].

technique triples the training dataset size by using a GAN to generate artificial images. While the second data augmentation technique performs better than the first one, the best class scores are achieved when both techniques are used. The class scores on using both the techniques are 0.620, 0.892, 0.885, and 0.027, on the "submarine", "wreck", "coral reef", "scuba diver" images, respectively.

Martija et al. [10] use a ResNet50-like CNN for classifying diver gestures on the CADDY dataset. The CNN has 17 outputs, including one non-gesture and 16 gestures. To mitigate class imbalance, they augment the number of images of minority classes. They compare against two traditional techniques for feature extraction: (1) HOG and (2) SIFT+BOVW. Both these techniques use an SVM for classification. Table IX shows their results. ResNet50 outperforms traditional techniques and is especially effective in distinguishing between similar gestures. However, it sometimes incorrectly classifies positive gestures as none or vice-versa. HOG has a lower FPS value because it uses sliding windows.

TABLE IX: Results obtained by Martija et al. [10]

	Accuracy	FPS (system)
HOG	84.50%	0.03 (on i7-7500 CPU)
SIFT+BOVW	64.0%	10 (on i7-7500 CPU)
Deep learning (ResNet50)	97.1%	84 (on RTX 2070 GPU)

C. Applying transfer learning

Lumini et al. [18] propose a model for classifying underwater grayscale images into several species of plankton. They fuse multiple deep learning networks by the sum rule. They compare nine different pretrained networks, namely, AlexNet, GoogLeNet, InceptionV3, VGG16, VGG19, ResNet50, ResNet101, DenseNet, and SqueezeNet, and their fusions. They also employ transfer learning, which helps avoid overfitting in case of scarce training data.

For all these networks, they compare three different finetuning strategies: "one round tuning" (1R), "two rounds tuning" (2R), and "pre-processing tuning" (PR). The PR strategy preprocesses the images before feeding them into the network. They evaluate various pre-processing methods, namely, "gradient", "orientation", "local binary patterns", "local ternary patterns", "local phase quantization" and "wavelet". Preprocessing helps highlight those features of images that would make the classification task easier.

In transfer learning, from each pretrained and finetuned network, a set of those layers are selected that would output the "best" feature vectors for classification. For selecting the layers, they employ the "sequential floating forward selection" (SFFS) method [59], which uses an SVM. In plankton classification, the distributions of different classes are generally different in the training and the test sets. They emulate such "dataset drift" in their data by keeping the class distributions different in the training and test sets. The authors also compare the "SqR" and "Pad" strategies for resizing the input image before it is fed into the network. These strategies are summarized in Table X.

TABLE X: Image resizing strategies of Lumini et al. [18].

Strategy	Action
SqR	Pad the input image to a "square size" and then resize it to the required size.
Pad	If input image's size is less than the required size, "pad" it to the required size. Otherwise, apply SqR.

The authors compare the various fusion models used by them with FUS_Hand [19], Baseline [38], Gaussian SVM [38], and "multiple kernel learning" (MKL) (with three kernels) [38]. Among their fusion models, the FUS_2R+FUS_1R model performs the best, and its architecture is explained in Table XI. Further, Table XII shows the results. Clearly, their model outperforms conventional models.

TABLE XI: Some models evaluated by Lumini et al. [18]. (They evaluate both x=1 and $x = 2$ models)

Model	Fusion of
FUS_RS_SqR (xR)	All the nine networks except SqueezeNet, using the xR strategy and a resizing strategy of "SqR".
FUS_RS_Pad (xR)	All the nine networks except SqueezeNet, using the xR strategy and a resizing strategy of "Pad".
FUS_xR	FUS_RS_SqR (xR) and FUS_RS_Pad (xR) models.
FUS_2R+FUS_1R	FUS_2R and FUS_1R models.

TABLE XII: Results of Lumini et al. [18]

Dataset	FUS_2R+FUS_1R (proposed model)			Highest value among FUS_Hand, Baseline, Gaussian SVM and MKL
	AUC	Accuracy	F1-score	F1-score
WHOI	99.9%	95.3%	95.3%	90.3%
ZooScan	99.5%	88.3%	89.7%	89.4%
Kaggle	99.8%	94.1%	92.6%	84.9%

Tamou et al. [30] note that compared to GoogleNet, VGG or ResNet, AlexNet has a simple architecture, which allows

easy finetuning, especially with limited training data. Further, AlexNet, with an input size of 227×227 pixels, is faster and requires less computing power. Hence, they adapt AlexNet to propose multiple models for live-fish species classification. (1) *Alex-FE:* Last few FC layers of a pretrained AlexNet are removed, and an SVM is used as the classifier. (2) *Alex-FT:* They finetune AlexNet by retraining the last FC layer from scratch on their dataset. (3) *Alex-FT-FE:* They first finetune the last two FC layers of AlexNet by retraining them from scratch on their dataset. Then, an SVM is used as the classifier. Of all the models, Alex-FT-FE achieves the highest accuracy and AP, which is shown in Table XIII. The accuracy is lower on the LifeCLEF15 dataset because it is a more challenging dataset with blurry and noisy images and low lighting conditions.

TABLE XIII: Results obtained by Alex-FT-FE model [30].

Dataset	Accuracy	AP
LifeCLEF15 (with data augmentation)	76.4%	58.8%
F4K	99.3%	96.0%

Olsvik et al. [20] propose a CNN to classify underwater images into four temperate fish species. Their network, named CNN-SENet, uses the SE block. Such a CNN is noise-tolerant, and thus, to avoid losing any important information of the input images, the authors do not pre-process them. The input image is resized to 200×200 pixels and then fed into the CNN-SENet, which is shown in Figure 5. The authors model various channels ' interdependencies using the SE block to increase the network's sensitivity towards more informative features and reduce it towards the less informative ones. This block recalibrates the filter responses using the "squeeze" and "excitation" operations. Using global average pooling, the "squeeze" operation obtains contextual information for each channel and "squeezes" it into its descriptors. The "excitation" operation uses this "squeezed" information to capture the channel interdependencies.

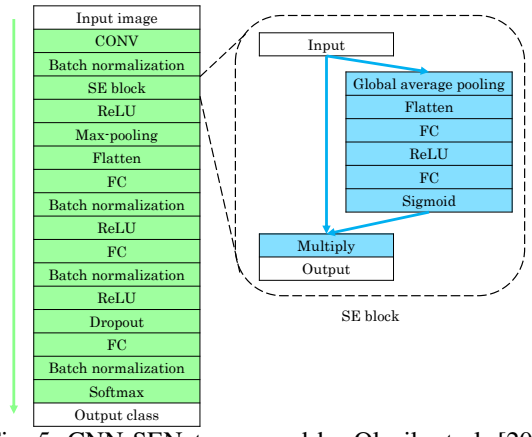

Fig. 5: CNN-SENet proposed by Olsvik et al. [20].

They employ transfer learning, wherein they first pretrain the CNN on the F4K dataset having tropical fish species and then finetune it on the temperate fish species dataset (refer Table III). They compare CNN-SENet with the InceptionV3, ResNet50, Inception-ResNetV2, and "CNN-SENet without SE block" networks. All the networks achieve very close classification accuracy values after pretraining (and before finetuning), with CNN-SENet achieving the highest value of 99.3%. After finetuning, the accuracy of CNN-SENet is 83.7%. This value is the second-highest (after finetuning), with InceptionV3 achieving the highest value of 85.4%. Finetuning reduces accuracy because before finetuning, the accuracy is measured on the test set of the F4K dataset, while after finetuning, it is measured on the test set of the temperate fish species dataset. The temperate fish species dataset is more challenging than the F4K dataset because it closely represents unrestricted underwater environments, with more occlusions and high variations in depth, visibility, background, and distance and angle of the fish from the camera. Also, its training dataset is much smaller.

They further perform data augmentation. After fine-tuning on the augmented temperate fish species dataset, the accuracy of InceptionV3, ResNet50, Inception-ResNetV2, CNN-SENet and "CNN-SENet without SE block" networks increase to 88.5%, 90.2%, 82.4%, 87.7%, and 83.6%, respectively. The last two networks have a much simpler architecture, and hence, they take the lowest time per epoch for both pretraining and finetuning.

D. Comparing different training optimizers and datasets

Al et al. [23] present a CNN for classifying underwater grayscale images of various plankton species. They first resize each image to 256x256 pixels and then crop them to a size of 222x222 pixels. These images are fed to a CNN that comprises five CONV layers followed by two FC layers and the final output layer. They use the Sipper dataset, which has 81 classes of plankton comprising low-resolution images with high noise, low inter-class and high intra-class variations, deformations and occlusions. From this, they design three datasets, which are described in Table XIV.

TABLE XIV: Datasets and results of Al et al. [23].

Dataset	Dataset creation	Accuracy
Sipper-7	7 classes, viz., "acantharia", "calanoid", "chaetognath", "doliolid", "larvacean", "radiolaria", and "trichodesmium"	98.2%
Sipper-52	Classes with $>$ 1000 and $<$ 2000 images per class	81.8%
Sipper-77	Classes with $>$ 10 and $<$ 2000 images per class	80.5%

In Sipper-52 and Sipper-77 datasets, they restrict the maximum number of images per class to 2000 to reduce the overfitting of the network in the classes with more images. This restriction slightly reduces the class imbalance, although this imbalance still exists in all three experiments. The Sipper-7 dataset has the least intra-class and the highest inter-class variations, which explains the superior classification accuracy. Compared to Sipper-52, the accuracy is lower on Sipper-77 because of the higher class imbalance. However, the insignificant difference in the two accuracies hints at the model's effectiveness while dealing with class imbalance.

Gonzalez et al. [33] compare shallow learning and deep learning methods to classify images of "Posidonia Oceanica meadows". They use two shallow learning models, namely, an SVM and an "artificial neural network" (ANN), along with multiple variations of each. In the deep learning method, the original image is first divided into 32×32 pixel patches. Then,

each of these patches is fed into a CNN. The CNN has 3 CONV layers that use 5×5 filters and 2 FC layers.

They use dataset1 with 69 images and dataset2, which is designed by adding 180 images to dataset1. However, these 180 images have many similarities, and hence, dataset2 is less diverse. Among the shallow learning models, the highest recall and the least training time is achieved by the "SVM using co-occurrence matrix" (SVMC) model. Table XV shows the results. The CNN has a higher recall on dataset1 than on dataset2. Also, it has higher recall but higher training time than SVMC.

TABLE XV: The results obtained by Gonzalez et al. [33].

	CNN (deep learning)		SVMC (shallow learning)	
	Recall	training time (s)	recall	training time (s)
Dataset1	96.4%	733	94.8%	165
Dataset2	94.0%	5043	91.8%	1326

Given the scarcity of training data, Guo et al. [17] use a 13-layer ResNet for classifying sea cucumbers in underwater images. They train this architecture from scratch rather than employing transfer learning. They compare ReLU and leaky-ReLU activation functions and the SGD and Adam optimizers to select the best option for this classification task. The highest classification accuracy, precision and f1-score values they achieve are 89.5%, 90.1% and 89.9% with SGD optimizer and ReLU. However, the highest recall they obtain is 92.9% with SGD and leaky-ReLU. This value is very close to the recall value of 92.4% obtained using SGD and ReLU. Thus, the SGD optimizer shows a more remarkable generalization ability as compared to Adam. Further, "ReLU activation function" outperforms leaky-ReLU.

Szymak et al. [29] use the pretrained AlexNet to classify underwater images into "diver", "fishes" and AUV categories. They download 50 images for each category from the internet. These 150 images are highly diverse with respect to the water, photographer and scale, thus providing a rich database for training. They evaluate three different training optimizers, namely, "SGD with momentum", RMSProp and Adam, and observe that they give comparable results.

E. Reducing annotation efforts

Manual annotation of plankton video and image datasets poses a challenge. Bochinski et al. [24] propose a "cost-effective active learning" (CEAL) for training CNN-based zooplankton classification systems. They assess the effectiveness of CEAL in making the classifier robust against background noise and adaptable to newer image data.

Through active learning, the classifier trains iteratively using the most informative data samples, thereby minimizing the required labeled training samples. Samples with low confidence are considered most informative and proceed to further training steps. This iterative process repeats until the training loss is converged. The authors employ the least confidence, margin sampling, and entropy as the confidence criteria. In CEAL, "high-confidence samples" act as "pseudo-labels" and contribute to the training process. They utilize Otsu thresholding to obtain the approximate shape and perform rotation and cropping of the image for background removal. They use AlexNet-based CNN with dropout and data augmentation.

The authors train the classifier on the ILES dataset consisting of 840K images across four classes. On the entire dataset, the CNN classifier (without CEAL) obtains an accuracy of 83.84%. However, the application of CEAL reliably achieves the same accuracy with just 48K training images. This reduces the manual annotation effort. The model achieves satisfactory performance on the CZECH dataset consisting of images captured in a different lake environment and camera setup. With only one additional CEAL iteration, the model achieves similar accuracy on the CZECH dataset as the ILES dataset.

4. CNN DESIGN AND OPTIMIZATION TECHNIQUES

Given the challenges of underwater image classificaion, using a single CNN or a single type, level or scale of features is not sufficient. Hence, researchers have proposed intelligently combining multiple networks and different types, scales and levels of features. Further, the different species of underwater organisms are distinguished by minute variations in their appearance, size and shape and hence, ascertaining the species is a challenging task. Since the deeper layers of a CNN lose the low-level information present in images, the fine-grained classification task is challenging for a CNN and requires special optimizations. Section 4.1 discusses techniques for designing collaborative or ensemble networks. Section 4.2 discusses techniques for combining the features of different layers and scales and Section 4.3 discusses techniques for using multiple types of features. Section 4.4 focuses on approaches used for fine-grained classificaion. Section 4.5 reviews the approaches that choose the best CNN from a design-space exploration approach. Table XVI classifies CNN architectures, design features, optimizations and training-related settings.

A. Designing collaborative or ensemble models

In ensemble learning, the output class "predictions" of multiple CNNs in the ensemble are taken and processed in a suitable way to obtain the final output class prediction. In collaborative learning, multiple CNNs operate on an image. Then, the feature maps of their final CONV layer are concatenated. These are fed to a small neural network having an FC layer to obtain the final predictions.

Kerr et al. [8] use multiple collaborative models for improved classification performance on datasets with class imbalance. This system combines pre-trained CNNs followed by an additional learning phase. To mitigate class imbalance, they employ the strategies of data standardization, data augmentation, and usage of "class weights". Additionally, the authors integrate training using geometric (dimensions, area, etc.) and environmental data (temperature, salinity, season, time, etc.) into the classification system by concatenating with the extracted feature maps from CONV layers.

The authors investigate VGG, InceptionNet, ResNet, and DenseNet architectures (termed as *learners*) for constructing the collaborative model. The learners are trained in isolation and are loaded with frozen weights before forming the collaboration. For this, each learner's last layers (without softmax) are concatenated and followed by an FC layer and a softmax layer. The FC layer works as a novel function for the model to learn how efficiently every learner contributes.

TABLE XVI: Network architectures and optimizations

Network architecture and characteristics	
CNNs used	AlexNet [6, 7, 7, 13, 18, 21, 22, 25, 27, 29, 30, 34, 39, 40], ResNet13 [17], ResNet18 [22], ResNet32 [42], ResNet50 [3, 5, 6, 8, 10, 11, 13, 18], ResNet101 [6, 8, 18], GoogLeNet [8, 13, 18, 22, 27, 31, 37, 40, 44], VGG [2, 4, 4, 6, 8, 9, 13, 15, 15, 18, 18, 22, 39, 43, 44], LeNet [27], PCANet [48], NiN [44, 48], InceptionV3 [3, 6, 12, 18], DenseNet [3, 8, 18], SparseConvNet [26], EfficientNet [3], Xception [2], MobileNet [2, 3]
Features used in CNNs	color [3, 5, 43], texture [5, 25, 33, 40, 43], shape [25, 40], geometric [3, 8, 15], environmental [8], hydrographic [15], geotemporal [15]
ML models used/compared	SVM [5, 13–15, 18, 21, 30, 33, 43, 45–48], k-NN [14, 45, 47], MLP or ANN [3, 8, 15, 33, 43], PCA [45, 47], k-means [5], SRC [47], random forest [5, 7, 14, 15], multinomial logistic regression [26], XRT [15], GBC [15]
Features compared	DenseSIFT [45, 48], ImageJ [46], HOG [10, 46], Gabor [5, 33, 48], SIFT [10, 46], BOVW [5, 10]
Kernel size except 5x5, 3x3, 2x2 and 1x1	13x13 [18, 21, 39, 45], 11x11 [21, 22, 27, 29, 30, 34], 7x7 [10, 11, 18, 22, 27, 31, 37, 39], 4x4 [47]
Skip connections	[4, 10, 11, 17, 18, 22, 42]
Normalization schemes	BN [2, 4, 6, 22], local response normalization [39], histogram normalization [26]
Pooling (except max/average)	spatial pyramid pooling [43, 45], global average pooling [5, 12, 20], fractional max-pooling [26], cross convolutional layer pooling [11], feature pooling [45], cyclic pooling [15], MPN-COV pooling [6]
In deep+shallow hybrid networks, using shallow networks for	classification [5, 13, 14, 21, 26, 30, 45–47], pre-processing and classification [48], feature extraction [3, 8], feature extraction and classification [3, 43]
Optimizer	SGD [17, 23, 34, 37, 42, 45], "SGD with momentum" [29], Adam [8, 10, 15, 17, 20, 29, 31, 36], AdaMax [11], Adagrad [2], Adadelta [2], RMSProp [12, 29]
Loss function	Cross-entropy [2, 12, 17, 20, 36], hinge [45]
Avoiding overfitting	limiting max number of images per class [23], dropout [2, 8, 12, 14, 15, 31, 37]
Handling class-imbalance	class reweighting [3, 8], data standardization [8], data augmentation (Table V)
Other design features and optimizations	leaky-ReLU [17], parametric ReLU [39], no data augmentation [17, 18, 23, 25], fusing low-level and high-level features [4, 47], depthwise separable CONV [4], asymmetric CONV [4], multi-scale images [44], SE block [20], SFFS [18], DECONV layers [21], SPP layer [45],

Training on geometric data is done separately using an MLP configuration before feeding to the collaborative network. The MLP configuration is tuned using feature engineering of input data and adding Dropout layers. They discard DenseNet due to its negative impact on the F1 accuracy. Their architecture is shown in Figure 6. The top collaborative model with three CNNs (VGG, Inception and ResNet) and MLP achieves a 3.2 percentage-point boost in overall accuracy and a 6 percentage-point gain in F1 score against the best single learner.

Kyathanahally et al. [3] use transfer learning, along with collaborative and ensemble learning for CNN-based classification of zooplankton. They experiment with several combinations of 12 models - EfficientNets B0 through B7, InceptionV3, Dense121, MobileNet, and ResNet50.

The authors also extract and integrate 110 morphological and color features such as aspect ratio, eccentricity, orientation, solidity, area, volume, and intensity to improve model performance. They utilize an MLP to train these features. They employ pre-trained CNN models with frozen layers.

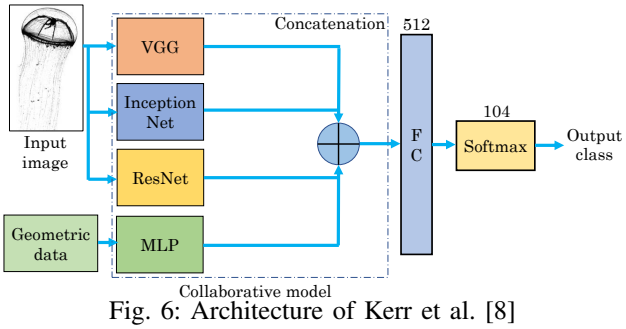

Fig. 6: Architecture of Kerr et al. [8]

They replace the last layers of pretrained DNNs with a dense layer, which is preceded and followed by dropout. They also train a 4-layer CNN, which combines features extracted from MLP and CNN to form a 'mixed' model for output class prediction. The 'mixed' models perform only slightly better than individual CNN models, with a 0.3% increase in classification accuracy. They also assess various ensemble combinations using the averaging and stacking ensembling methods. To overcome the issue of class imbalance, they adopt class reweighting. It gives more weight to minority classes by highly penalizing minority class misclassification.

They choose 6 'best' models based on validation performance and generalization ability on the ZooLake dataset - DenseNet121, MobileNet, EfficientNets B2, B5, B6, and B7. EfficientNet-B7 attains the best F1-score of 90.0%. The lightest model, MobileNet achieves an F1-score of 89.1%. The authors find that stacking ensembling of the 6 'best' models achieves the highest accuracy (97.9%) and F1-score (92.7%). Furthermore, their ensemble model outperforms the previously best models on ZooScan, Kaggle, and WHOI datasets by 1.3%, 1.0%, and 0.3%, respectively.

Jose et al. [2] propose a region-based CNN ensemble architecture for classification of tuna fish species. They separate the fish images into three regions corresponding to the fish's head, abdomen, and tail. They employ "contrast stretching" to enhance these three images. The contrast stretching process operates on every pixel in the sub-region by nullifying pixels with an intensity below a threshold. This threshold is chosen as the mean intensity value of the sub-region. They utilize the "mean fusion" method to merge the enhanced and original images. They perform data augmentation using rotation, translation, zooming, flipping, and shearing.

They construct an ensemble with four CNN models (VGG16, VGG19, Xception, and MobileNet) pre-trained on ImageNet. They use BN and dropout in these CNNs. The three enhanced image sub-regions are fed into three individual ensembles, as shown in Figure 7. Individual scores from each model are summed up class-wise, and the maximum score is considered the final prediction. Then, the "majority voting" method is applied to the predictions of each ensemble for obtaining the final prediction. Their technique achieves an accuracy of 95.65% and an F1-score of 96%. By contrast, the best individual model (VGG19) achieves an accuracy of 88.33%.

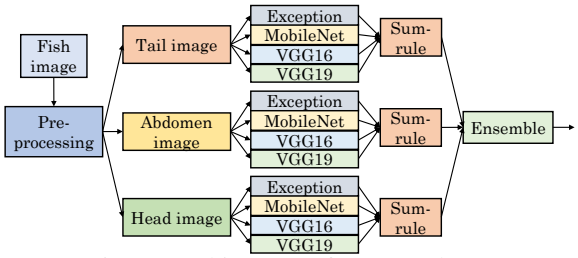

Fig. 7: Architecture of Jose et al. [2]

B. Combining the features of different layers and scales

Mathur et al. [11] propose a "cross convolutional layer pooling" [52] based deep network. They perform transfer learning on a pretrained CNN. For the pretrained CNN, they choose ResNet50 over VGG16 since VGG16 performs poorly due to its smaller number of layers and lack of skip connections. They apply cross convolutional layer pooling, shown in Figure 8, on ResNet50 for performing better feature extraction.

This technique extracts "local features" from a lower CONV layer. Then, it pools them by taking activations from the higher CONV layer as guidance. This creates one pooling channel for every detected area. The activations of layers being pooled are multiplied point-wise, and then they are added to create a feature vector for every feature map. Feature vectors of various pooling channels undergo concatenation to create the final features. These features are fed into an FC layer.

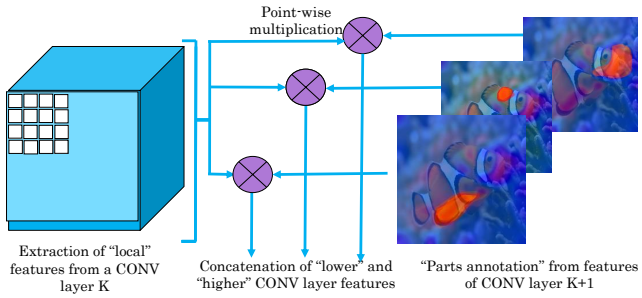

Fig. 8: Cross convolutional layer pooling algorithm [11].

A design-space exploration shows that the best accuracy is achieved using the tanh activation function, the AdaMax optimizer during training and by cross-pooling layers 154 and 157. On the F4K dataset, their model achieves 98.0% accuracy, 92.0% precision, 85.0% F1-score, and 82.0% kappa score.

Salman et al. [47] propose a model for classifying images of multiple species of live fishes. The authors employ data augmentation to increase variations in the images. The input image is resized to 32x32 pixels using bilinear transformation, converted to grayscale, and then fed into the CNN. The three CONV layers have eight 5x5 filters, 24 3x3 filters and 80 4x4 filters, respectively. All CONV layers use max-pooling and the sigmoid activation function. Finally, there is an FC layer.

Features are extracted from all the three CONV layers rather than just from the output layer to preserve the low-level information. This information is necessary for classification when the inter-species variations are small. Each of the first and second CONV layers' outputs goes through PCA. The dimensions of the principal components corresponding to each layer are kept equal to the dimensions of the feature vector that is output by the third CONV layer. The principal components obtained from the first and second CONV layers are concatenated hierarchically with the third CONV layer's feature vector. This concatenated output is the final feature vector fed into the classifier (SVM or k-NN). The model is named CNN-SVM or CNN-KNN, depending on the classifier used.

Using LifeCLEF14 and LifeCLEF15 datasets, they perform four experiments, as described in Table XVII. In experiments (3) and (4), the models' performance is measured on those species of live fishes common to the two datasets. The authors compare their proposed models (CNN-SVM and CNN-KNN) with SVM, k-NN, SRC, PCA-SVM, and PCA-KNN. Of all these models, CNN-SVM or CNN-KNN models achieve the highest average count, AP and AR values in all four experiments. The average count is defined as the ratio of the sum of true positives of every class and (sum of true positives of every class + sum of false positives of every class).

TABLE XVII: Comparison of CNN-SVM and CNN-KNN [47] with five models (SVM, k-NN, SRC, PCA-SVM, and PCA-KNN) (L14 = LifeCLEF 2014, L15 = LifeCLEF 2015).

Experiment	CNN-SVM		CNN-KNN		Highest among other five models	
	AP (%)	AR (%)	AP (%)	AR (%)	AP (%)	AR (%)
(1) Train & test on L14	94.5	95.7	93.4	95.0	86.9	85.6
(2) Train and test on L15	91.6	91.0	92.0	91.3	82.7	83.5
(3) Train on L14, test on L15	65.4	74.5	63.9	75.7	44.6	61.4
(4) Train on L15, test on L14	97.2	98.4	96.9	98.0	88.1	94.3

Table XVII shows the results. LifeCLEF15 dataset is more challenging than LifeCLEF14 because of blurriness, light-intensity attenuation and confusing background. Roughly, the authors observe that from best to worst, the models' performance order is experiment 4, 1, 2, and 3. Expectedly, a model trained on a less challenging dataset performs very poorly when tested on a more challenging dataset (experiment 3). Further, a model trained on a more challenging dataset performs very well when tested on a less challenging dataset (experiment 4). When the train and test sets are the same (experiment (1) and (2)), a model performs better when trained and tested on a less challenging dataset since the classification task is relatively easy. Both CNN-SVM and CNN-KNN models achieve an FPS of around 1000. They further observe that even when Gaussian blur or Gaussian white noise is added to LifeCLEF14 images, CNN-SVM and CNN-KNN achieve higher accuracy than conventional machine learning models.

Prasetyo et al. [4] propose a multi-level residual (MLR) network that fuses low-level and high-level features across the depth of CNN by employing "depthwise separable convolution" (DSC). Incorporating the low-level traits (lines, points, and textures), representing spines, gills, fins, and skin textures of fish, improves the CNN performance. To minimize the parameters, they replace the fifth CONV stack of VGG by asymmetric convolutional (AC) layers. These AC layers use two pairs of asymmetric filters of sizes 1x3 and 3x1. They

add BN with AC layers to prevent "internal covariate shifts" and fasten the training process.

The authors add skip connections from every intermediate feature map to the final 'concatenation' layer after the last CONV layer. However, the feature map sizes of each CONV layer are different. Therefore, they employ DSC to adjust the feature map size before concatenation. Three AC + BN layers follow the concatenated feature maps from all four CONV layers. The authors also add residual connections across the three AC + BN layers. This allows retaining low-level, middle-level, and high-level features across the depth of the model. Figure 9 shows their architecture. They use pre-trained VGG with frozen weights in the four CONV blocks to reduce the computational cost and model size. Additional AC + BN layers and residuals are completely trained. On the Fish-gres dataset, their proposed MLR-VGG16 model achieves an accuracy of 98.46%, whereas ResNet50 achieves an accuracy of 97.84%.

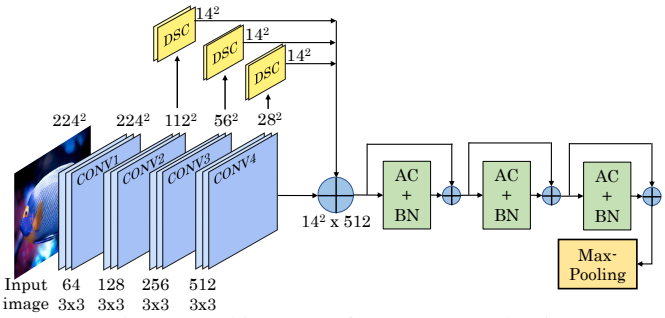

Fig. 9: Architecture of Prasetyo et al. [5]

Py et al. [44] propose a CNN which takes multi-size imagery input for plankton image classification. They define a c-value metric, which quantitatively measures the learning capacity of complex features by CONV layers. C-value is the ratio of filter and receptive field sizes. As the receptive field exceeds the size of filter, the capacity of convolutional layer in learning more complex features declines. They apply two constraints to network design. Firstly, the c-value of every CONV layer must not be smaller than the threshold (1/6). Secondly, the receptive field of the first CONV layer should not be greater than the input image. This allows the network the ability to learn high-level features.

The authors design an 'inception' module that allows the model to process images of various sizes to minimize distortion effects and information loss caused by resizing of images. The inception module extracts multi-scale features from the input image across the sizes 32x32, 48x48, 64x64, 96x96 and 128x128, by CONV with kernels of suitable sizes. These extracted features are then concatenated before feeding into the CNN. Figure 10 shows their approach. They use GoogLeNet, with the last two FC layers replaced by CONV layers with smaller kernels. They employ data augmentation with translational and rotational transforms. Their model achieves lower softmax loss than VGG, NiN and GoogLeNet. Also, the model that takes multi-scale (128x128 and 64x64-sized) input images performs better than the models that take single-sized (64x64 or 128x128) input images.

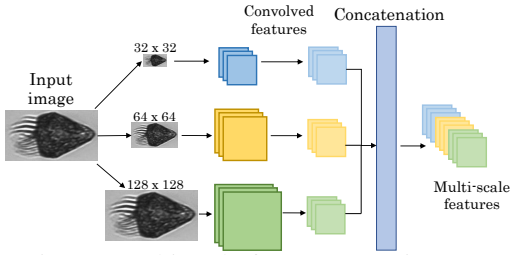

Fig. 10: Multi-scale feature extraction [44]

Du et al. [6] propose a CNN architecture with second-order pooling layers to classify plankton images. Plankton images have high inter-class similarity and intra-class variance. Therefore, they use covariance pooling instead of first-order pooling for fine-grained image classification and achieve better accuracy than baselines. The authors integrate "matrix power normalized co-variance" (MPN-COV) Pooling [35] into ResNet-50 and ResNet-101. In the original ResNet, downsampling is performed after the conv5_1 layer. The authors connect this last layer to a 1x1x256 CONV layer for dimensionality reduction, followed by MPN-COV Pooling Layer.

MPN-COV takes the feature map from the last CONV layer and sequentially performs global covariance pooling followed by Eigen decomposition to compute matrix power, which is input to the FC layers. Let d be the feature dimension of the last CONV layer. By replacing first order pooling such as max/average pooling with MPN-COV, the model captures d(d+1)/2 dimensional image representation, i.e., it learns second-order features for improved generalization and better classification. On PlanktonSet 1.0 dataset, their technique of inserting MPN-COV Pooling increases the accuracy of ResNet-50 by 2.07% and that of ResNet-101 by 2.24%.

C. Combining multiple types of features

Mahmood et al. [43] note that the coral classification datasets are not sufficiently large for training a CNN from scratch. Also, pretrained CNNs are trained on a dataset that is very different from the coral classification datasets, both in texture and shape. Hence, it is not sufficient to use just a pretrained CNN. Further, the RGB channels are not enough for storing the color information of underwater images [56], and CNNs rely entirely on the RGB channels of an image for feature extraction. Hence, they combine hand-crafted features with CNN-extracted features to classify underwater coral reef images into five coral genera and four non-coral classes. Figure 11(a) shows their overall flow.

They employ VGG16 as the CNN and finetune its weights using the MLC dataset. Since the MLC dataset contains point annotations rather than bounding box annotations, the authors apply a novel feature extraction scheme before feeding the features as an input to VGG16. This scheme, referred to as local-SPP, extracts four patches of sizes 28×28, 56×56, 84×84, and 112×112 pixels around each labeled point. Each of these four patches is resized to 224×224 pixels and fed to the VGG16 network, as shown in Figure 11(b).

For each labeled point, VGG16 outputs a 4096-sized feature vector. These vectors are max-pooled together to obtain the final 4096-sized CNN-extracted feature vector for that point. The conventional SPP technique divides the whole image into

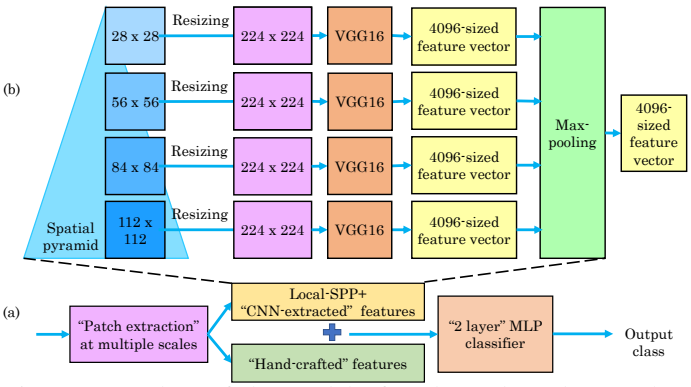

Fig. 11: (a) Flow of the model of Mahmood et al. [43] (b) Local-SPP followed by CNN-based feature extraction [43].

multiple non-overlapping tiles and pools together the features in each tile to produce a single, "global" feature depiction of the entire image. By contrast, their local-SPP approach draws patches at various scales around the same point. They are max-pooled to ascertain the most scale-independent response. Thus, the result vector captures the most eminent features in the local-neighborhood of the pixel. It also makes the infrequently present coral classes more prominent in the final CNN-extracted feature vector, thus solving the class imbalance problem.

As for hand-crafted features, they use the color and texton based features of Beijbom et al. [56]. The normalized 540-sized hand-crafted "feature vector" is concatenated with the normalized 4096-sized CNN-extracted "feature vector" to produce the final 4636-sized feature vector. This vector is fed into an MLP with two FC layers followed by the output layer (with softmax activation). The MLC dataset consists of images collected from 2008 to 2010. The authors conduct three experiments described in Table XVIII. "Average class precision" (ACP) is the average of the diagonal of a confusion matrix, and a higher value is better. Their proposed model outperforms the model of Beijbom et al. [56] in all three experiments.

TABLE XVIII: Comparision between the results of Mahmood et al. [43] and Beijbom et al. [56]. (Accu. = accuracy)

#	Training dataset	Test dataset	Mahmood		Beijbom	
			Accu.	ACP	Accu.	ACP
1	$2/3^{rd}$ of 2008 images	$1/3^{rd}$ of 2008 images	77.9%	69%	74.3%	60%
2	2008 images	2009 images	70.1%	63%	67.3%	56%
3	2008 and 2009 images	2010 images	84.5%	68%	83.1%	60%

Ellen et al. [15] devise a method of incorporating metadata of several types into CNN classifiers. Additionally, they assess five classifiers: random forest, "extremely randomized trees", "gradient boosted classifier", MLP, and SVM. The authors study the impact of integrating hydrographic, geo-temporal, and geometric metadata into the model. Hydrographic features include salinity, density, chlorophyll a fluorescence, temperature, and hydrostatic pressure. Geotemporal features are location, season, time of day, sample depth, and distance from shore. Geometric features are mean intensity, skeletal area, perimeter, height, width, circularity, and symmetry.

These metadata measurements are normalized and concatenated into the FC layers. The authors experiment with multiple architectures to incorporate the metadata, as shown in Figure 12. The approach shown in Figure 12(b) achieves higher integration of metadata than simple concatenation, shown in Figure 12(a). The CNN model comprising five CONV layers is based on the VGG-16. Dropout is employed on image data only since employing it on metadata harms performance. Hyperparameter tuning is employed to obtain optimal learning rate and regularization strength across all feature-based classifiers and the CNN model.

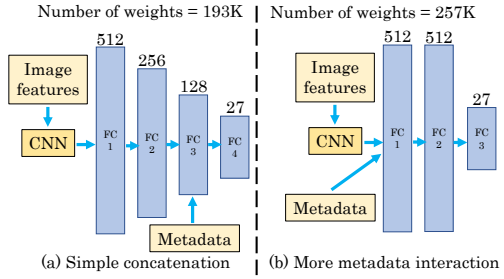

Fig. 12: Architecture of Ellen et al. [13]

They evaluate 12 different architectures with various filter sizes, number of layers, changes in dropout, and metadata configurations. Using all three types of metadata provides better accuracy for all feature-based classifiers. Further, doubling the number of layers and utilizing "cyclic pooling and rolling" [41] boosts the accuracy. Their best CNN model achieves an accuracy of 92.28%. Metadata inclusion also improves CNN execution time and reduces time to convergence.

Most plankton classes have different shapes, and those with similar shapes show variation in texture. Hence, using multiple features is important for achieving high accuracy. Cui et al. [25] integrate texture and shape features of plankton to improve CNN performance. To obtain and utilize these features, they devise a "feature extraction" scheme based on Gaussian filtering. They employ Gaussian low-pass filtering, which allows the passage of shape features without sharp details. They use "Gaussian high-pass filtering" to obtain the texture features from the plankton images. However, high-pass filtering degrades the intensity of images, making them much darker. To overcome this issue, the authors adopt the logarithmic image enhancement method. It maps pixel values from source to target through a logarithmic function. This method enhances the contrast of darker areas in the image.

They evaluate an AlexNet-based CNN trained using the WHOI-Plankton dataset. The original plankton image and the images with enhanced texture and shape features are concatenated and fed into the model, as shown in Figure 13(a). The model achieves a classification accuracy of 94.32%, whereas AlexNet trained on original images obtains an accuracy of 93.58%. The model also offers slightly better performance on minority classes. However, low-pass filtering leads to blurred borders, which reduces the efficacy of shape features. Similarly, their image enhancement technique creates salt noise.

Dai et al. [40] propose a feature-fusion approach, where they use both global features (shape and setae) and local

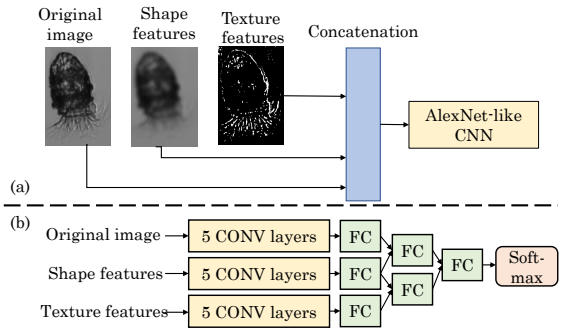

Fig. 13: Architecture of (a) Cui et al. [25] (b) Dai et al. [40]

features (texture, including cytoplasm and nucleus), along with the original image. To extract global features, they first use a bilateral filter for removing noise, then a Scharr operator for marking plankton appearance, and finally, contrast-enhancement for highlighting shape information. To extract local features, they use a canny edge detector. The original image, local features, and global features are fed in a network, shown in Figure 13(b). On a subset of the WHOI-Plankton dataset, a single AlexNet trained on original images achieves 94.8% accuracy, and their network shown in Figure 13(b) achieves 95.8% accuracy.

Rivas-Villar et al. [5] propose a method for accurate detection and segmentation of phytoplankton specimens in freshwater samples containing similar objects like zooplankton, minerals, bubbles, or dirt. To separate the background, they utilize adaptive gaussian thresholding. It thresholds each of the RGB channels of input image separately and combines them through an OR operator to extract a high amount of information. The authors implement Suzuki and Abe's algorithm for object segmentation in the foreground image. They employ a colony merging algorithm, which measures color similarities for the suitable detection of sparse specimens.

The authors also integrate color and texture features into the system using a BoVW model. They extract base texture features as complex responses to a Gabor filter bank before feeding them to the BoVW model. However, they feed the RGB information directly to the BoVW model. The authors use deep features from the global average pooling layer of a pre-trained ResNet50. They combine these color, texture, and deep features from ResNet50 and feed them to a classifier. Their architecture is shown in Figure 14. They evaluate a random forest classifier and an SVM classifier on a dataset with 293 microscopic images with 1611 phytoplankton specimens. Their best SVM model with color, texture and deep features achieves 84.07% precision at 90% recall. SVM model with only color or texture features achieves a precision of only 62.38% and 69.89%, respectively. The model has an overall accuracy of 87.50%.

D. Techniques for fine-grained classification

Wang et al. [21] propose an encoding-decoding architecture for classifying noisy and less illuminated underwater images. Their network architecture is shown in Figure 15. The CONV layers extract deep "discriminative features" from the noisy images, and the DeCONV layers extract fine-grain features. CONV layer parameters are set from AlexNet. DeCONV2

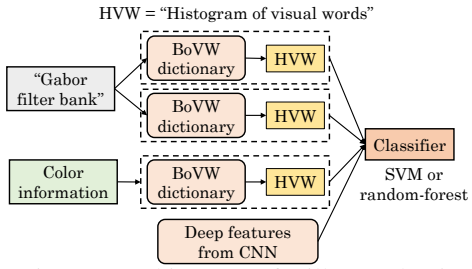

Fig. 14: Architecture of Villar et al. [5]

parameters are the same as that of CONV2, and DeCONV1 parameters are set based on feature-map dimensions to keep the dimensions manageable. Their network can use either an FC layer with a softmax activation function or an SVM for the final classification. With the FC layer and SVM, the network is called ED-Net-Soft and ED-Net-SVM, respectively.

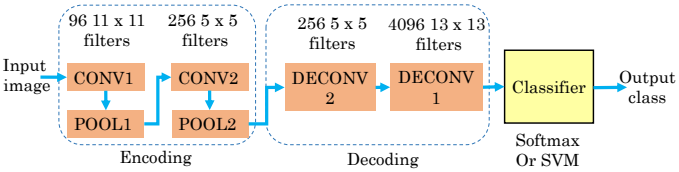

Fig. 15: Architecture of Wang et al. [21].

They finetune the pretrained AlexNet on the F4K dataset. They compare their models with DeepFish [45], and UW-CNN [28] models. ED-Net-SVM achieves the best classification accuracy of 99.4%. For ED-Net-Soft, DeepFish and UW-CNN, these values are 95.8%, 90.1% and 97.1%, respectively.

Riabchenko et al. [46] propose a deep-learning method for the classification of images into 29 categories of Finnish river macroinvertebrate benthos. Here, the variations in size, shape and appearance of different classes' objects are very subtle. The authors first pre-process the images to remove any undesired objects. For this, they apply Otsu's thresholding on the images to create binary masks. On these binary masks, they perform connected components analysis. Then, they crop a square region around each component from the original images. Each square region has a 20-pixels margin around the component and is resized to 256x256 pixels. These images are input into a CNN to extract deep features from each image. These extracted features are input into an SVM for the final classification.

The use of CNN-extracted features achieves higher accuracy than using hand-crafted features like SIFT, HOG and ImageJ. Further, finetuning a pretrained CNN increases the classification accuracy by 8-9% compared to training the CNN from scratch, even though the pretraining data differs significantly from the macroinvertebrates data.

Qin et al. [45] propose models for live-fish classification. Figure 16 shows their workflow. They first extract a foreground fish mask from each image to eliminate the complex backgrounds of such unconstrained images [54]. Then, the extracted foreground is resized to 47×47 pixels and fed into the DeepFish-SVM model, shown in Figure 16. Here, the CONV layers use PCA filters. "Low-level" and "high-level" features are extracted by color and grayscale filters, respectively. The feature pooling layer produces block-wise

histograms, with a block-size of 8×8 pixels and an overlapping ratio of 0.6. This layer offers translation-invariance, just like a "max-pooling" layer. The SPP layer helps extract information that is invariant to large poses. This extraction is beneficial since the unconstrained images consist of complex poses.

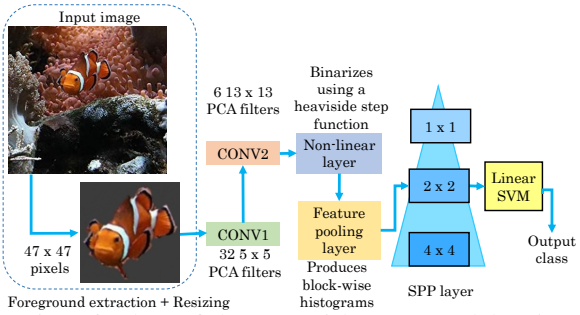

Fig. 16: Flow of the DeepFish-SVM model [45].

Compared to a softmax layer's objective function, the objective function of an SVM (hinge loss) has a higher "positive correlation" with "test error". Further, the hinge loss facilitates faster training. Hence, linear SVM is found to be superior to a softmax layer for final classification. They augment the number of training images of those fish species that have less than 300 images in the F4K dataset. They also scale the feature vector size to 100 times before feeding it into the SVM or softmax layer. By combining these options, they create different networks, named DeepFish-x-y-z, as shown in Table XIX. For example, a network using augmentation, scaling and SVM classifier is called DeepFish-SVM-aug-scale. The authors also design a CNN, called Deep-CNN, which is shown in Figure 17. As shown in Table XIX, all the models have very close classification accuracy values, with the DeepFish-SVM-aug-scale model having the highest.

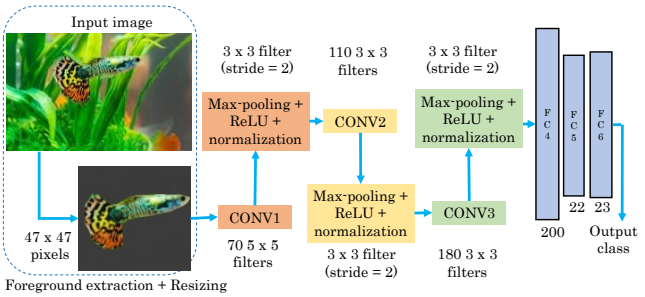

Fig. 17: Flow of the Deep-CNN model [45].

TABLE XIX: Results obtained by Qin et al. [45].

Model	Classifier	Aug. used?	Scaling used?	Accuracy
VLFeat Dense-SIFT (Best conventional model)	SVM	No	No	93.58%
DeepFish-SVM	Linear SVM	No	No	98.23%
DeepFish-SVM-aug	Linear SVM	Yes	No	98.59%
DeepFish-SVM-aug-scale	Linear SVM	Yes	Yes	98.64%
DeepFish-Softmax-aug	Softmax	Yes	No	92.55%
DeepFish-Softmax-aug-scale	Softmax	Yes	Yes	98.49%
Deep-CNN	FC layer	No	No	98.57%

E. Selecting the best CNN

Yang et al. [22] compare multiple CNNs for classifying underwater images into 15 diver gesture categories and one "no-gesture" category. Since the CADDY-UG dataset is not too large, the authors employ transfer learning to circumvent the overfitting problem. Table XX shows the results of using pretrained AlexNet, VGG16, ResNet18 and GoogLeNet for the required task. Compared to AlexNet (with eight layers), VGG16 has double the number of layers (16) and smaller filter sizes. Hence, VGG16 preserves more information of the input images but takes a longer time and a higher memory to train. ResNet18 enjoys the benefits of both the shallow and deep networks due to the presence of skip connections in it. GoogLeNet (with 22 layers) has the least number of parameters despite having the highest number of layers since it extensively uses pooling layers and BN.

TABLE XX: The results obtained by Yang et al. [22].

	Training time on the CADDY-UG dataset	CCR	FPS
AlexNet	11.7 hours	82.9%	27.8
VGG16	15.1 hours	95.0%	14.3
ResNet18	7.0 hours	88.6%	21.3
GoogLeNet	12.5 hours	90.1%	16.4

Li et al. [42] compare ResNet19, ResNet32, and ResNet50, and VGG19 for classifying underwater grayscale-images into various plankton species. ResNet19 has skip connections and fewer FC layers than VGG19. Hence, ResNet19 has fewer parameters and converges faster while training. ResNet32, with an input size of 256×256 pixels and an output layer of 121 neurons, achieves the highest classification accuracy. ResNet50 achieves a lower accuracy than ResNet32 because the authors' dataset is too small to effectively train the large ResNet50 network. Hence, they choose ResNet32, which displays a top-5 accuracy of 95.8%. Note that the winning team of the National Data Science Bowl on the Kaggle platform has achieved an accuracy of over 98%. Further, ResNet32 achieves an FPS of 9.1, which is much higher than that of the winning team (1.4 FPS).

Meng et al. [27] evaluate various CNNs for live-fish classification in natural lakes. They collect the images from the internet. Table XXI shows their results. LeNet has the least average recognition rate because, as opposed to the other networks, it takes grayscale images as input, which carry less information than color images. Moreover, its input size is much smaller. Overall, AlexNet provides a good balance between the average recognition rate and FPS, which makes it suitable for real-time applications.

TABLE XXI: The results obtained by Meng et al. [27].

	Average recognition rate	Training time	FPS
LeNet	67%	2 minutes	38.3
GoogLeNet	85%	120 minutes	23
AlexNet	87%	32 minutes	19.2

Eerola et al. [7] present a technique for phytoplankton classification. They reduce image size using "bicubic interpolation", which gives better accuracy than "nearest-neighbor interpolation". Then, padding is done with the most frequent pixel value (i.e., mode), which provides better accuracy than

padding with black or white color. They use an AlexNet-like network, which has only 3M parameters compared to 61M parameters of AlexNet trained on the ImageNet dataset. This is because of factors such as smaller image size (128x128) and lower number of output classes (34, compared to 1000 in the ILSVRC dataset). Their CNN achieves nearly 89% accuracy. Not using dropout or augmentation brings a large loss in accuracy. Increasing the node count in FC layers increases accuracy at the cost of increased parameter count and training time.

5. Conclusion and future outlook

In this paper, we presented a survey of deep learning techniques to classify underwater images. We compared them on critical parameters and highlighted their similarities and differences. We reviewed the works related to datasets and training and those related to the design and optimization of CNNs. We close this paper with a brief mention of future research challenges.

Deep learning models require a large amount of data for achieving high accuracy. While data augmentation overcomes the scarcity of training data, it also reduces the robustness of the network. Some of the existing datasets are small and distinguish between only a few species. There is a need for large-scale, realistic and standardized datasets, including other essential sea animals such as dolphins, whales and sharks. Research is also required for fast and cost-effective approaches for dataset acquisition.

Similarly, there is a need for competitions similar to ILSVRC to stimulate further interest in this area. Further, instead of retrofitting the existing CNNs, there is a need to design novel CNNs for underwater image classification. For example, CNNs have a limited receptive field, and this limitation can be overcome by using an attention mechanism, which focuses on certain factors while processing the data. Also, there is a need to design and optimize CNNs for 3D underwater images to provide novel insights. Recently, there has been an increased emphasis on explainable artificial intelligence. This research can help immensely in adapting the CNNs for underwater images and also design novel CNNs. Finally, to use these techniques in mission-critical applications such as defense, ensuring deep networks' reliability and security is required. Overall, we believe that the field of deep learning on underwater images is still developing, and concerted efforts are needed from both industry and academia to establish this as a fully mature field.

References

[1] Fish-gres dataset. https://data.mendeley.com/datasets/76cr3wfhff/1, 2021.

[2] Jisha Anu Jose, C Sathish Kumar, and S Sureshkumar. An ensemble of region-based CNN models combined by sum rule for tuna classification. In *International Conference on Communication, Control and Information Sciences (ICCISc)*, volume 1, pages 1–6, 2021.

[3] Sreenath Pruthviraj Kyathanahally, Tommy Hardeman, Ewa Merz, Thea Kozakiewicz, M Reyes, P Isles, F Pomati, and M Baity-Jesi. Deep learning classification of lake zooplankton. *arXiv:2108.05258*, 2021.

[4] Eko Prasetyo, Nanik Suciati, and Chastine Fatichah. Multi-level residual network vggnet for fish species classification. *Journal of King Saud University - Computer and Information Sciences*, 2021.

[5] David Rivas-Villar, José Rouco, Rafael Carballeira, Manuel Gonzalez, and Jorge Novo. Fully automatic detection and classification of phytoplankton specimens in digital microscopy images. *Computer Methods and Programs in Biomedicine*, 200:105923, 01 2021.

[6] Angang Du, Zhaorui Gu, Zhibin Yu, Haiyong Zheng, and Bing Zheng. Plankton image classification using deep convolutional neural networks with second-order features. *Global OCEANS*, pages 1–5, 10 2020.

[7] Tuomas Eerola, Kaisa Kraft, Osku Grönberg, Lasse Lensu, Sanna Suikkanen, Jukka Seppälä, Timo Tamminen, Heikki Kälviäinen, and Heikki Haario. Towards operational phytoplankton recognition with automated high-throughput imaging and compact convolutional neural networks. *Ocean Science Discussions*, pages 1–20, 2020.

[8] Thomas Kerr, James R. Clark, Elaine S. Fileman, Claire E. Widdicombe, and Nicolas Pugeault. Collaborative deep learning models to handle class imbalance in flowcam plankton imagery. *IEEE Access*, 8:170013–170032, 2020.

[9] Michael Kloster, Daniel Langenkämper, Martin Zurowietz, Bánk Beszteri, and Tim W Nattkemper. Deep learning-based diatom taxonomy on virtual slides. *Scientific reports*, 10(1):1–13, 2020.

[10] Mygel Andrei M Martija, Jakov Ivan S Dumbrique, and Prospero C Naval Jr. Underwater gesture recognition using classical computer vision and deep learning techniques. *Journal of Image and Graphics*, 8(1), 2020.

[11] Monika Mathur, Diksha Vasudev, Sonalika Sahoo, Divanshi Jain, and Nidhi Goel. Crosspooled fishnet: transfer learning based fish species classification model. *Multimedia Tools and Applications*, 79(41):31625–31643, 2020.

[12] Vaneeda Allken, Nils Olav Handegard, Shale Rosen, Tiffanie Schreyeck, Thomas Mahiout, and Ketil Malde. Fish species identification using a convolutional neural network trained on synthetic data. *ICES Journal of Marine Science*, 76(1):342–349, 2019.

[13] Kaichang Cheng, Xuemin Cheng, Yuqi Wang, Hongsheng Bi, and Mark Benfield. Enhanced convolutional neural network for plankton identification and enumeration. *PLOS ONE*, 14:e0219570, 07 2019.

[14] B Vikram Deep and Ratnakar Dash. Underwater fish species recognition using deep learning techniques. In *Intl. Conf. on Signal Processing and Integrated Networks (SPIN)*, pages 665–669, 2019.

[15] Jeffrey Ellen, Casey Graff, and Mark Ohman. Improving plankton image classification using context metadata. *Limnology and Oceanography: Methods*, 17, 07 2019.

[16] Arturo Gomez Chavez, Andrea Ranieri, Davide Chiarella, Enrica Zereik, Anja Babić, and Andreas Birk. CADDY underwater stereo-vision dataset for human–robot interaction (HRI) in the context of diver activities. *Journal of Marine Science and Engineering*, 7(1):16, 2019.

[17] Xiangyun Guo, Xuehua Zhao, Yahui Liu, and Daoliang Li. Underwater sea cucumber identification via deep residual networks. *Information Processing in Agriculture*, 6(3):307–315, 2019.

[18] Alessandra Lumini and Loris Nanni. Deep learning and transfer learning features for plankton classification. *Ecological informatics*, 51:33–43, 2019.

[19] Alessandra Lumini and Loris Nanni. Ocean ecosystems plankton classification. In *Recent Advances in Computer Vision*, pages 261–280. Springer, 2019.

[20] Erlend Olsvik, Christian MD Trinh, Kristian Muri Knausgård, Arne Wiklund, Tonje Knutsen Sørdalen, Alf Ring Kleiven, Lei Jiao, and Morten Goodwin. Biometric fish classification of temperate species using convolutional neural network with squeeze-and-excitation. In *IEA-AEI*, pages 89–101. Springer, 2019.

[21] Xinhua Wang, Jihong Ouyang, Dayu Li, and Guang Zhang. Underwater object recognition based on deep encoding-decoding network. *Journal of Ocean University of China*, 18(2):376–382, 2019.

[22] Jing Yang, James P Wilson, and Shalabh Gupta. Diver gesture recognition using deep learning for underwater human-robot interaction. In *MTS/IEEE OCEANS*, pages 1–5, 2019.

[23] Hussein Al-Barazanchi, Abhishek Verma, and Shawn X Wang. Intelligent plankton image classification with deep learning. *Intl. Journal of Computational Vision and Robotics*, 8(6):561–571, 2018.

[24] Erik Bochinski, Ghassen Bacha, Volker Eiselein, Tim JW Walles, Jens C Nejstgaard, and Thomas Sikora. Deep active learning for in situ plankton classification. In *International Conference on Pattern Recognition*, pages 5–15, 2018.

[25] Jinna Cui, Bin Wei, Chao Wang, Zhibin Yu, Haiyong Zheng, Bing Zheng, and Hua Yang. Texture and shape information fusion of convolutional neural network for plankton image classification. In *OCEANS*, pages 1–5, 2018.

[26] Jessica Luo, Jean-Olivier Irisson, Benjamin Graham, Cédric Guigand, Amin Sarafraz, Christopher Mader, and Robert Cowen. Automated

plankton image analysis using convolutional neural networks. *Limnology and oceanography, methods*, 16:814–827, 10 2018.
[27] Lin Meng, Takuma Hirayama, and Shigeru Oyanagi. Underwater-drone with panoramic camera for automatic fish recognition based on deep learning. *IEEE ACCESS*, 6:17880–17886, 2018.
[28] Xin Sun, Junyu Shi, Lipeng Liu, Junyu Dong, Claudia Plant, Xinhua Wang, and Huiyu Zhou. Transferring deep knowledge for object recognition in low-quality underwater videos. *Neurocomputing*, 275:897–908, 2018.
[29] Piotr Szymak. Recognition of underwater objects using deep learning in matlab. In *ICAMCS*, pages 53–535, 2018.
[30] Abdelouahid Ben Tamou, Abdesslam Benzinou, Kamal Nasreddine, and Lahoucine Ballihi. Transfer learning with deep convolutional neural network for underwater live fish recognition. In *IPAS*, pages 204–209, 2018.
[31] Sebastien Villon, David Mouillot, Marc Chaumont, Emily S Darling, Gérard Subsol, Thomas Claverie, and Sébastien Villéger. A deep learning method for accurate and fast identification of coral reef fishes in underwater images. *Ecological Informatics*, 48:238–244, 2018.
[32] ZOOVIS dataset. https://search.dataone.org/view/10.24431_rw1k57s_20210316T191058Z, 2017.
[33] Yolanda Gonzalez-Cid, Antoni Burguera, Francisco Bonin-Font, and Alejandro Matamoros. Machine learning and deep learning strategies to identify posidonia meadows in underwater images. In *OCEANS*, pages 1–5, 2017.
[34] Leilei Jin and Hong Liang. Deep learning for underwater image recognition in small sample size situations. In *OCEANS*, pages 1–4, 2017.
[35] Peihua Li, Jiangtao Xie, Qilong Wang, and Wangmeng Zuo. Is second-order information helpful for large-scale visual recognition? In *IEEE ICCV*, pages 2070–2078, 2017.
[36] Dhruv Rathi, Sushant Jain, and S Indu. Underwater fish species classification using convolutional neural network and deep learning. In *ICAPR*, pages 1–6, 2017.
[37] Yifeng Xu, Yang Zhang, Huigang Wang, and Xing Liu. Underwater image classification using deep convolutional neural networks and data augmentation. In *ICSPCC*, pages 1–5, 2017.
[38] Haiyong Zheng, Ruchen Wang, Zhibin Yu, Nan Wang, Zhaorui Gu, and Bing Zheng. Automatic plankton image classification combining multiple view features via multiple kernel learning. *BMC bioinformatics*, 18(16):570, 2017.
[39] Jialun Dai, Ruchen Wang, Haiyong Zheng, Guangrong Ji, and Xiaoyan Qiao. Zooplanktonet: Deep convolutional network for zooplankton classification. In *OCEANS*, pages 1–6, 2016.
[40] Jialun Dai, Zhibin Yu, Haiyong Zheng, Bing Zheng, and Nan Wang. A hybrid convolutional neural network for plankton classification. In *Asian Conference on Computer Vision*, pages 102–114. Springer, 2016.
[41] Sander Dieleman, Jeffrey Fauw, and Koray Kavukcuoglu. Exploiting cyclic symmetry in convolutional neural networks. 02 2016.
[42] Xiu Li and Zuoying Cui. Deep residual networks for plankton classification. In *OCEANS*, pages 1–4. IEEE, 2016.
[43] Ammar Mahmood, Mohammed Bennamoun, Senjian An, Ferdous Sohel, Farid Boussaid, Renae Hovey, Gary Kendrick, and Robert B Fisher. Coral classification with hybrid feature representations. *ICIP*, pages 519–523, 2016.
[44] Ouyang Py, Hu Hong, and Shi Zhongzhi. Plankton classification with deep convolutional neural networks. In *ITNEC*, pages 132–136, 2016.
[45] Hongwei Qin, Xiu Li, Jian Liang, Yigang Peng, and Changshui Zhang. Deepfish: Accurate underwater live fish recognition with a deep architecture. *Neurocomputing*, 187:49–58, 2016.
[46] Ekaterina Riabchenko et al. Learned vs. engineered features for fine-grained classification of aquatic macroinvertebrates. In *ICPR*, pages 2276–2281, 2016.
[47] Ahmad Salman, Ahsan Jalal, Faisal Shafait, Ajmal Mian, Mark Shortis, James Seager, and Euan Harvey. Fish species classification in unconstrained underwater environments based on deep learning. *Limnology and Oceanography: Methods*, 14(9):570–585, 2016.
[48] Xin Sun, Junyu Shi, Junyu Dong, and Xinhua Wang. Fish recognition from low-resolution underwater images. In *CISP-BMEI*, pages 471–476, 2016.
[49] Fish4Knowledge dataset. http://groups.inf.ed.ac.uk/f4k/, 2015.
[50] PlanktonSet dataset. https://github.com/Planktos/PlanktonSet-1.0, 2015.
[51] Zooscan dataset. https://sites.google.com/view/piqv/, 2015.
[52] Lingqiao Liu, Chunhua Shen, and Anton van den Hengel. The treasure beneath convolutional layers: Cross-convolutional-layer pooling for image classification. In *CVPR*, pages 4749–4757, 2015.
[53] Eric C Orenstein, Oscar Beijbom, Emily E Peacock, and Heidi M Sosik. Whoi-plankton-a large scale fine grained visual recognition benchmark dataset for plankton classification. *arXiv preprint arXiv:1510.00745*, 2015.
[54] Hongwei Qin, Yigang Peng, and Xiu Li. Foreground extraction of underwater videos via sparse and low-rank matrix decomposition. *ICPR Workshop on Computer Vision for Analysis of Underwater Imagery*, pages 65–72, 2014.
[55] Andrew G Howard. Some improvements on deep convolutional neural network based image classification. *arXiv preprint arXiv:1312.5402*, 2013.
[56] Oscar Beijbom, Peter J Edmunds, David I Kline, B Greg Mitchell, and David Kriegman. Automated annotation of coral reef survey images. In *CVPR*, pages 1170–1177, 2012.
[57] Kurt A Kramer. *System for identifying plankton from the sipper instrument platform*. PhD thesis, 2010.
[58] Robert K Cowen and Cedric M Guigand. In situ ichthyoplankton imaging system (ISIIS): system design and preliminary results. *Limnology and Oceanography: Methods*, 6(2):126–132, 2008.
[59] Pavel Pudil, Jana Novovičová, and Josef Kittler. Floating search methods in feature selection. *Pattern recognition letters*, 15(11):1119–1125, 1994.