Deep Learning and Transfer Learning Features for Plankton Classification

Alessandra Lumini[1], Loris Nanni[2]

[1] Department of Computer Science and Engineering, University of Bologna, via Pavese 50, 47521, Cesena (FC), Italy.

[2] Department of Information Engineering, University of Padua, via Gradenigo 6/B, 35131 Padova, Italy.

Abstract: Plankton are the most fundamental components of ocean ecosystems, due to their function at many levels of the oceans food chain. Studying and monitoring plankton distribution is vital for global climate and environment protection. Currently, much research is concentrated on the automated recognition of plankton and several imaging-based technologies have been developed for collecting plankton images continuously using underwater image sensors. In this paper, we present a study about an automated plankton recognition system, which is based on the fusion of different deep learning methods. In this work we study both the fine tuning of several deep learned models and transfer learning from the same models with the aim of exploiting their diversity in designing an ensemble of classifiers, we deal with: *(i)* the ability of fine-tuning pre-trained CNN for plankton classification, *(ii)* the possibility of using pre-trained CNN for transfer learning, *(iii)* the possibility of coupling pre-processing to CNN in order to improve their feature extraction capability. The combination of such different descriptors/methods in a heterogeneous ensemble grants a substantial performance improvement with respect to other state-of-the-art approaches based on feature selection and classification. The experimental evaluation on three large publicly available datasets demonstrates high classification accuracy and f-measure of our ensemble with respect to other classifiers on the same datasets. One of the main contributions of this work is a collection of classification models and a wide experimental evaluation to report performance of both single CNN and ensemble of CNNs in different available plankton datasets with a given testing protocol. Moreover, we show how to combine different CNN in order to improve the performance. To encourage future comparisons the MATLAB source code is available in the GitHub repository: https://github.com/LorisNanni.

Keywords— Convolutional Neural Network, Transfer learning, Plankton Classification.

1. Introduction

Studying marine plankton, i.e. organisms that drift freely in the water, is critical to assessing the health of the world's oceans, since plankton is very sensitive to environment variations, therefore their distribution is a suitable indicator for climatic events, such as global warming. Plankton microorganisms form the basis of the food web, play an important function at many levels of the oceans food chain, link the atmosphere to the deep ocean, and regulate global-scale biogeochemical cycles.

Studying and monitoring plankton distribution is vital for global climate and environment protection: plankton is the main oxygen producer (for about 90% of all oxygen), so low level of plankton is dangerous for the ocean ecosystem; on the other hand, its excessive abundance can also result in a devastating effects due to production of toxins [1].

Researchers are increasingly replacing sampling techniques in favor of specially engineered in situ digital imaging systems that produce very large data sets (i.e. [2][3][4]). Unfortunately the task of monitoring plankton distribution is very challenging due to microscopic size of the organism forming plankton and the dynamic nature of the ocean, moreover manual annotation of plankton images is expensive, time consuming, and error prone.

Many researchers have explored automated methods for performing plankton classification based on computer vision techniques. Automatic plankton classification [1][5][6][7] is a difficult task, due to three main reasons: *(i)* plankton images are often acquired at low resolution which make object classification hard even by a human expert; *(ii)* plankton images include a wide range of phylogenetic species which offer specific challenges to taxonomy, *(iii)* the classes are unbalanced among the same and different datasets and data drifts can income between training and test sets.

Recent works in this area are mainly based on two different classes of approaches: *(i)* handcrafted approaches [1][7], where feature extraction is based on hand-crafted descriptors (such as SIFT and LBP) and classification is often done with Support Vector Machines or Random Forests, and *(ii)* deep learning approaches [5][6], that exploit Convolutional Neural Networks (CNN) [8] for image classification.

One of the first noticeably approach based on handcrafted features is the method of Tang et al. [9], which combines invariant moment features and Fourier boundary descriptors with grayscale morphological granulometries. Another texture descriptor, the co-occurrence matrices, is used by Hu and Davis [10] to train a Support Vector Machine (SVM) as the classifier. Li et al. [11] use a set of shape descriptors concatenated together and reduced to lower dimensionality by Principal Component Analysis (PCA), classification is performed by an ensemble of pairwise binary classifiers based on nonparametric discriminant analysis technique. Ellen et al. [12] suggest the use of feature selection and test several different classifiers: their ensemble is a combination of gradient boosted random forest classifier and SVM. Chang et al. [13] use SURF descriptors and LPB texture operator coupled with PCA for dimensionality-reduction. However their method is evaluated on a small dataset with few classes. More recently Zheng et al. [7] propose a system based on the combination of features via multiple kernel learning classifier. Images are first pre-processed in order to enhance their quality (the pre-processing operations depend on the dataset, to fit to different acquisition devices), then a large set of descriptors (i.e. geometric features, texture features and some local features) are extracted and reduced to an optimal subset by feature selection (optimized in each dataset). This method is validated on the same datasets selected for this work with very valuable performance, therefore it is a baseline for our results. Bi et al. [14] developed a semi-automated approach to analyze plankton data from images acquired in turbid estuarine waters: they customized a segmentation procedure to locate plankton objects within each image and used handcrafted features for classification.

The most recent trend in Plankton classification is based on deep learning [15]. CNN [16] are multi-layered neural networks inspired by the human visual perception mechanism which incorporates spatial context and weight sharing between pixels and are able to learn the optimal image features and classification weights for a given classification task. A great advantage of CNN vs handcrafted approaches is that since CNN adopt e□ective representations of the original image, they require a very small amount of pre-processing.

In 2015 the National Data Science Bowl[1] has been held to classify the images of plankton, which has been win by an ensemble of over 40 convolutional neural networks. The main innovation brought by Dieleman and his team was some layers designed to increase the network robustness to cyclic variation [17].

Py et al. [18] use the same large dataset for training a CNN inspired to GoogleNet and improved with an inception module with a convolutional layer for distortion minimization and maximization of image information extraction. Lee et al. [5] use a larger dataset including more than 3 million images: their solution is based on a simpler net inspired by the pre-trained model CIFAR10, and the main aim of their study is to overcome the class-imbalance problem by performing transfer learning with pre-training the CNN on class-normalized data. Another CNN is proposed by Dai et al. [6]: they suggest an ad-hoc model, inspired by AlexNet and VGGNet and named ZooplanktoNet, which is made by 11 layers and makes use of data augmentation to overcome the overfitting in their small dataset. A solution including preprocessing is proposed by Dai et al. [19]: their Hybrid CNN is a 3-channel network which takes as input the original image, a two preprocessed version of it (a global feature image which describe the appearance of plankton and omit the internal texture and a local feature image obtained by edge detection). The whole network structure is composed of three AlexNet networks that share the final fully-connected layer. A similar idea is exploited by Cui et al. [20] which propose a AlexNet CNN trained combining different inputs obtained by image pre-processing (i.e. Gaussian filtering). In our experiments we test a similar idea of fusion at feature level, but obtaining lower performance than our fusion at score level. Bochinski et al. [21] propose a Deep Active Learning which requires only a limited amount of labelled images, while the most part of training set is made of automatically labelled images. The performance are encouraging, even if obtained on a simplified classification problem consisting of only four classes. Cui et al. proposed a transfer learning approach starting from model trained on several datasets. They extract features from 2 different CNNs and show that their performance are better than other state-of-the-art handcrafted descriptors [22] (similarly to our baseline method in [23]).

In this work we study both deep learned approaches and transfer learning with the aim of exploiting their diversity in designing an ensemble of classifiers, we deal with: *(i)* the ability of fine-tuning pre-trained CNN for plankton classification, *(ii)* the possibility of using pre-trained CNN for transfer learning, *(iii)* the possibility of coupling pre-processing to CNN in order to improve their feature extraction capability. The combination of such different descriptors/methods in a heterogeneous ensemble grants a substantial performance improvement with respect to other state-of-the-art approaches based on feature selection and classification.

Our ensembles is validated using three well-known plankton datasets and compared with other approaches proposed in the literature. Despite of the complexity of the method, one of the main benefit of the proposed system is to work well out-of-the-box in different problems, requiring few parameter tuning without specific optimization for each datasets. The experimental results reported in section 3 show that the proposed system obtains state-of-the-art performance in all the tested datasets.

[1] https://www.kaggle.com/c/datasciencebowl

The paper is organized as follows. The proposed approach is detailed in Section 2, where both deep learning networks and transfer learning approaches are discussed. In Section 3, we describe the three datasets used for experiments, the testing protocols and discuss the experimental results. The conclusion is given in Section 4.

2. Methods

In this work the deep learned methods are based on fine-tuning of some well-known CNN architectures according to some different training strategies (one and two round training, preprocessing before training). The trained CNNs have been used both for classification purposes and as feature extractors to extract features used to train a general purpose classifier.
In this section we describe the CNN architecture used for deep learning, the pre-processing approaches tested as pre-filters before training CNNs and the transfer learning approaches used to extract features from the CNNs.

2.1 *CNN models*

Convolutional Neural Networks (CNNs) are a category of neural networks designed for image recognition and classification. CNNs have been successful applied in several image classification task as identifying faces, objects recognition, pedestrian and traffic signs recognition. CNNs are designed to work similarly to the human brain in visually perceiving the world: they include layers of "neurons" that exclusively respond to neurons in their direct environment. A CNN is a feed-forward neural network including a repeated concatenation of some classes of layers [24]:

- Convolutional layers (CONV), whose aim is to extract features from the input image, by convolving input to filters that apply to small squares of input data and preserve the spatial relationship between pixels. The CONV layers substitute conventional handcrafted feature extractors.
- Activation layers (ACT), which are used to introduce nonlinearity to the system; the most used activation function is ReLU, an element wise operation which replaces all negative pixel by zero. Other (less used) ACT layers are sigmoid or tanh.
- Pool layers (POOL), also called subsampling or down sampling, reduce the dimensionality of each feature map but retain the most important information with the purpose of making the input representations smaller and more manageable, decreasing the size of the data to be inferred and the risk of over-fitting, making the network invariant to small transformations, distortions and translations in the input image. The most used pooling functions are max, average and sum.
- Fully-connected layers (FC) are layers which connect every neuron in the previous layer to every neuron on the next layer. The purpose of a FC layer is to use the features extracted from the previous layers for classifying the input image into various classes based on the training dataset.
- Classification layers (CLASS) perform the final classification. They can be implemented as general purpose classifiers (i.e. SVM), but generally a SoftMax function is used which takes a vector of arbitrary real-valued scores and squashes it to a vector of values between zero and one that sum to one.

The CNN are trained using backpropagation: first all filters and parameters are initialized with random values, then using training images as input, the CNN performs a forward propagation step and finds the output probabilities (and the classification errors) for each class, finally backpropagation is used to calculate the gradients of the error with respect to all weights in the network and use gradient descent to update all filter and parameter values to minimize the output error. The main problem is that often the amount of training images is not enough to adjust all the parameters without incurring in overfitting: in such a case transfer learning and fine tuning are used, i.e. the solutions consisting in using a pre-trained network where only the parameters of the last classifications levels need to be inferred from scratch using the training set. In this work we use 9 different pre-trained model modified in order to fit the new classification problem (i.e. changing the last FC and CLASS layers to fit the number of plankton classes, without freezing the weights of the previous layers) and "fine-tuned" with the training set of the current problem. In our experiments, we test and combine the following different CNN architectures available in the MATLAB Deep Learning Toolbox; all the models, which are pre-trained on a large dataset of objects (the ImageNet database[2]) are "fine-tuned" on the current problem:

- AlexNet [25]. AlexNet was the winner of the ImageNet ILSVRC challenge in 2012. Its architecture includes five CONV layers followed by three FC layers, with some max-POOL layers in the middle. A ReLu is applied to each convolutional and fully connected layer to enable faster training. The input layer of AlexNet accepts images of 227×227 pixels.
- GoogleNet [26]. GoogleNet was the winner of the ImageNet ILSVRC challenge in 2014. It evolves AlexNet including a new "Inception" module (INC), that is a subnetwork consisting of parallel convolutional filters whose outputs are concatenated, that makes the network deeper but strongly reduced the number of parameters to be inferred. GoogleNet is composed by 22 layers that require training (27 counting also POOL layers), but has fewer parameters than AlexNet. The input layer of GoogleNet accepts images of 224×224 pixels.
- InceptionV3 [27]. InceptionV3 is a variant of GoogleNet based on the factorization of 7x7 convolutions into 2 or 3 consecutive layers of 3×3 convolutions. The input layer of InceptionV3 accepts images of 299×299 pixels.
- VGGNet [28]. VGGNet was the network placed second in ILSVRC 2014. It is a very deep network with respect to GoogleNet: it includes 16 or more CONV/FC layers. Each CONV layer uses small 3×3 convolution filters and a POOL layer is placed between each group of 2 or 3 CONV layers. In our experiments we consider two of the best-performing VGG models (i.e. VGG-16 and VGG-19), with 16 and 19 weight layers, that are available as pretrained models. The input layers of VGG-16 and VGG-19 accept images of 224×224 pixels.
- ResNet [29]. ResNet was the winner of ILSVRC 2015. It is a network about 20 times deeper than AlexNet and 8 times deeper than VGGNet. ResNet introduces a new kind of layer, named residual (RES), which a kind of "network-in-network" architecture. Another novelty is the use of global average pooling layers instead of FC layers at the end of the network. Thanks to these tricks ResNet is deeper than VGGNet, using a smaller model size. The input layer of ResNet accepts images of 224×224 pixels. In this work we use ResNet50 (a 50 layer Residual Network) and ResNet101 (a deeper variant of ResNet50).
- DenseNet [30]. DenseNet in an evolution of ResNet which connects each layer to every other layer in a feed-forward fashion, thus increasing the number of connections from the number of layers L to L(L+1)/2. DenseNet, which has been trained on ImageNet, obtain significant improvements over other state-of-the-art models at the cost of an increased computation requirement. The input layer of DenseNet accepts images of 224×224 pixels
- SqueezeNet [31]. SqueezeNet is a very compact model made of only 18 layers and able of gain performance similar to AlexNet with with 50x fewer parameters. The input layer of SqueezeNet accepts images of 227×227 pixels, the final classification layer is not preceded by a fully connected layer differently from most other models.

Three different fine-tuning approaches have been evaluated for each of the pretrained models:

- One round tuning (1R): this is the standard fine tuning approach, based on performing retraining of the net starting from the weights obtained in the pre-trained CNN and using the training set of the target problem for training. Some works reports different strategies for fixing or weighting different network layers during the tuning, in this work we use the same learning rate in all layers, therefore allowing training also in the first layers.
- Two rounds tuning (2R): a first round of tuning is performed training CNN using an external dataset (described in section 3) including plankton images from classes not incorporated in the datasets used for evaluation; the second round tuning is performed as the above cited One round tuning, but starting from the weights obtained from the first round instead of the publicly available pre-trained models. The rationale behind this method is to use the first round in order to adjust the weights of the networks (in particular the first levels) in order deal with plankton images (which are quite different from the images used for scene/object classification used to create the pre-trained models), then the second round is used to adjust the classification weights.
- Pre-processing tuning (PR): this approach uses preprocessed images to feed the networks, instead of the original ones. In this work we evaluate different pre-processing approaches strategy (i.e. gradient [32], orientation [32], LBP [33], LTP [33], LPQ [33], wavelet [34]) in order to study the responses of the networks varying the input images. The rationale behind this method is to use a preprocessing strategy in order to highlight the features that can make the plankton classes more discernible. In the experimental section only the most significant results are reported. Even if this is a task similar to that performed by first layers of a CNN, we argue that using experience acquired in many years of study for handcrafted approaches can be positively exploited in this way.

[2] http://www.image-net.org

The training procedures is performed setting the maximum number of epochs for training to 30 and using a mini-batch with 16 to 128 observations at each iteration (depending on the net)[3] and fixing the learning rate to 0.001. Unlike most of works published in the literature, we do not use data augmentation for fine tuning, since it not granted sensible performance improvements in our experiments.

2.2 *Image Preprocessing*

Among several different methods used as a pre-preprocessing step before feeding CNNs, the following four approaches have been gained significant results: gradient, orientation, LBP and LTP. Gradient and orientations are related to the basic idea of characterizing an image by the distribution of local intensity gradients and edge directions.
“Gradient” preprocessing is obtained simply calculating the gradient image, obtained as the module of the gradient evaluated at each pixel (figure 1.b)
“Orient” preprocessing is the orientation at each pixel discretized over 0-π (figure 1.c)
LBP is a 3D image obtained as a multi-scale LBP transformation [35], which combines into the three color channel: the original image and the two images preprocessed by the LBP operator with parameters R=1 and R=2 (figure 1.d)
LTP is a variation of the approach in [36] which proposes a novel transformation to map a LBP image to a 3-channel space, designed to be invariant to monotonic photometric transformations. In this work we use LTP encoding ([37]) instead of LBP, thus obtaining 2 output images named LTP+,LTP- (figures 1.e, 1.f).

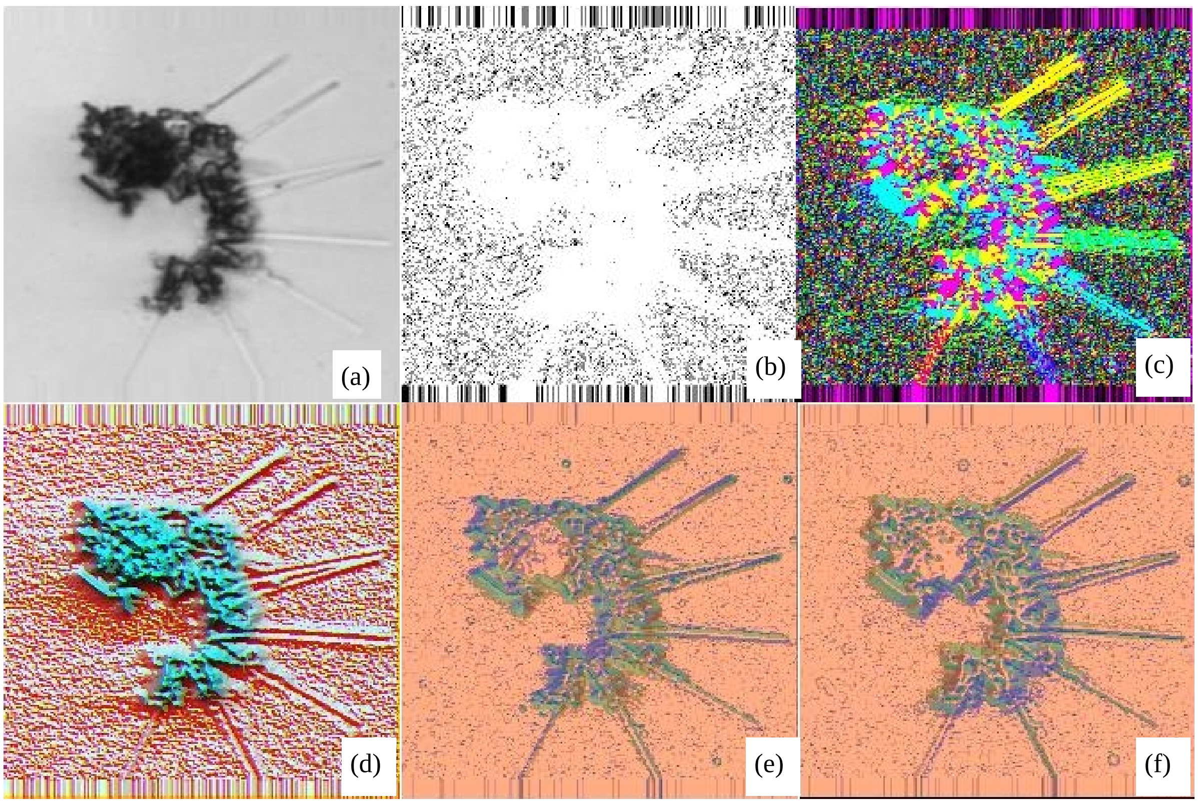


Fig. 1. A sample image (a) and its preprocessed version: Gradient (b), Orient (c), LBP(d), LTP+(e), LTP-(f)

[3] AlexNet, Vgg16, Vgg19, GoogleNet: 128; ResNet50: 32; ResNet101, Inceptionv3: 16

2.3 Transfer learning

Transfer learning is a machine learning approach where a CNN trained for a task is reused as the starting point for a model on a second task. Many different research studies have been recently published about transfer learning [38], in this work we perform a feature-based transfer learning, where the output of a given layer of a pretrained network is used as a descriptor for the target task. Most of exiting approaches [39][40] performs a manual selection of the source for transfer learning: usually the output of the last fully-connected layers are used for feature extraction, but also internal layers have proved to be effective [39]. In this work we propose to use a feature selection approach to automatically select the best layers to be used for transfer learning. The feature selection is performed according to the well-known Sequential Floating Forward Selection (SFFS) [41]. SFFS is a feature selection method that successively deletes features from an initial set in order to obtain a smaller set of optimal features. This algorithm provides a heuristic for determining the best order to transverse the feature subset space. In this work the feature set is the set of layers: the responses given from each layer of a fine-tuned CNN for a given image is treated as a descriptor and used to feed a different classifier, the scores from the classifiers trained using such descriptors are selected by SFFS. Each layer gives a different descriptor which is used to train a Support Vector Machine (SVM) [42]. In order to avoid the curse of dimensionality problem the dimension of the descriptor is reduced to 5000 using Discrete Cosine Transform (DCT) [43] if larger.

The objective function of the SFFS selection is the classification performance obtained by the ensemble of the selected SVMs (fusion by the sum rule): using this method we automatically select 3 to 5 layers for each CNN. Since SFFS requires a training phase in order to select the best layers to be used as feature extractors, we used a leave-one-out dataset testing protocol (i.e. the objective function is optimized separately on two out three datasets and the selected descriptors are used in the other one). The list of selected layers for each of the tested CNN in each dataset is reported in Table 1. Unfortunately the optimization based on maximizing accuracy seems to generate overfitting in the layer selection, leading to a solution which is non optimal in the testing set.

In the experiments we compare the performance of this automatic selection approach for transfer learning (named SFFS) with other strategies: i.e. the selection of the last fully connected layer (1FC), the selection of all the fully connected layers (aFC) and the manual selection of the layers suggested by [39] (Man).

Table 1. Layers used in each model for the Feature Transfer learning ensemble. Notice that all the layers of each net are considered in the numeration (e.g. layer 15 of AlexNet is relu5 layer). The result for SqueezeNet are not reported due to low performance and the lack of the 1FC layer.

	Selection Strategy					
Model	SFFS (WHOI)	SFFS (ZooScan)	SFFS(Kaggle)	1FC	aFC	Man
AlexNet	9 11 13 16 23	8 13 16 20	7 9 13 16 23	23	17 20 23	15 17 20 23 24
GoogleNet	54 61 69 74 142	69 70 74 75 142	61 68 70 74 142	142	142	127 142 143
InceptionV3	53 68 79 112 312	59 63 208 256 314	50 59 62 63 314	314	314	216 314 315
VGG16	22 26 29 32 36	21 29 31 32 39	26 29 32 36 39	39	36 39	29 33 36 37 39 40
VGG19	28 32 33 35 38	32 34 38 40 41	28 32 33 35 38	45	42 45	31 39 42 45
ResNet50	128 133 136 139 142	87 97 105 126 140	128 133 139 140 142	175	175	124 175 176
ResNet101	107 116 117 119 345	45 48 61 115 345	61 104 115 119 345	345	345	241 345 346
DenseNet	205 356 486 692 707	222 456 620 685 706	228 301 569 707	707	707	707 699 656

3. Experiments

In this section we first describe details of training and testing datasets used in our experiments, next we show experimental results of the methods based on deep features and CNNs. Our experiments have been carried out on the same three Plankton datasets used by [7][4].

- WHOI is a dataset of cells and other particles captured by Imaging FlowCytobot from Woods Hole Harbor water and can be accessed as supplemental material[5] of [44]. The dataset contains 6600 manually categorized images stored in tiff format and split between training and testing sets of equal size. The images belongs to 22 categories with equal number of samples for each category (150 training samples and 150 test samples). In our experiments, we used the same division in training and testing proposed by the authors.
- ZooScan is a dataset of 3771 images collected from the Bay of Villefranche-sur-mer using the Zooscan teconlogy [4][6]. The images belong to 20 categories with different number of samples for each category (from 28 to 427 samples per class). Most categories are zooplankton, other species of Medusae, and eggs of zooplankton; the remaining categories are non-zooplankton and images with bad focus. In order to compare results with [7] in our experiments we used 2-fold cross validation on this dataset. Since this dataset contains artifacts (see Fig. 2 second row), all the images have been automatically cropped in order to remove artefacts before classification.
- Kaggle is a subset of the dataset collected in the Straits of Florida using ISIIS [3] and used for the National Data Science Bowl of 2015 competition. The original dataset contains images from 121 categories from which the authors of [7] selected the 38 categories having more than 100 samples in each, for a total of 14374 images. The distribution among classes is not uniform but varies from a minimum of 108 to a maximum of 1979 samples per class. According to [7] in our experiments we used 5-fold cross validation on this dataset.

In order to perform a 2-rounds training we used a further training dataset, obtained by fusing the images from the dataset used for the National Data Science Bowl and not included in the Kaggle dataset (15962 images from 83 classes) and the dataset "Esmeraldo" (11005 samples, 13 classes) obtained from the Zooscan [4] site[7]

Fig. 2 shows some sample images from the three datasets (downloaded from the Github repository of [7][8]).

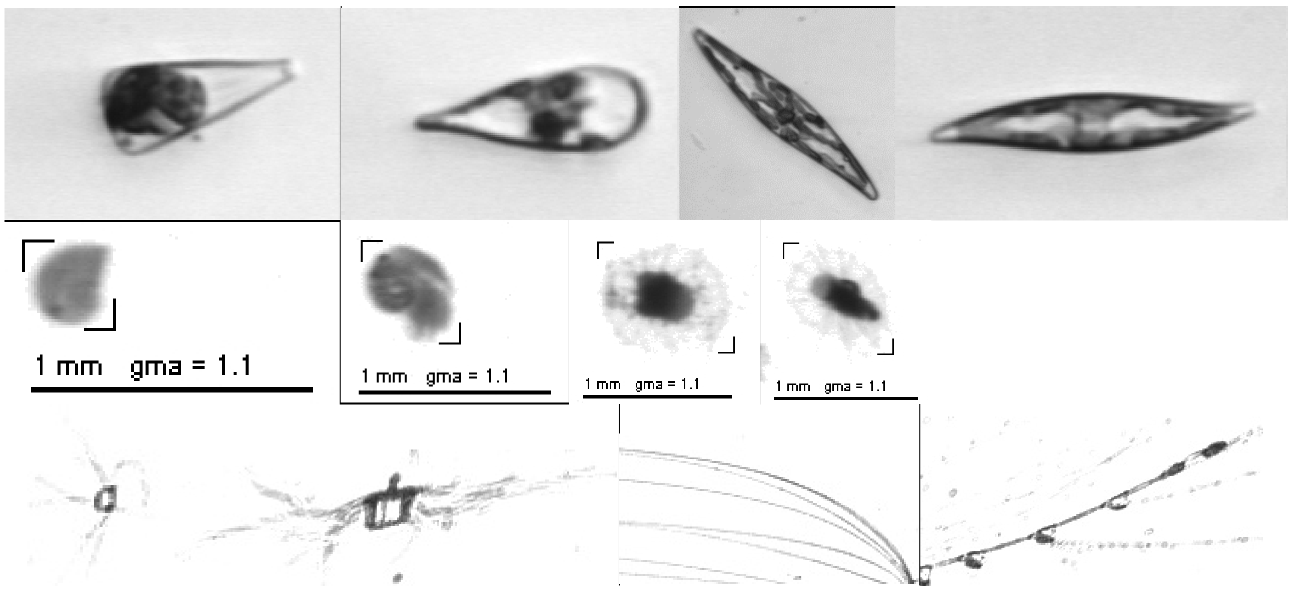


Fig. 2. In each row 4 sample images from different classes (2 images per class) of the three datasets: WHOI (Licmophora, Pleurosigma), ZooScan (Mollusq_limacina, Radiolaria), Kaggle (hydromedusae_aglaura, jellies_tentacles)

[4] Available from https://github.com/zhenglab/PlanktonMKL/tree/master/Dataset

[5] https://aslopubs.onlinelibrary.wiley.com/doi/abs/10.4319/lom.2007.5.204

[6] http://www.zooscan.obs-vlfr.fr/

[7] http://www.zooscan.obs-vlfr.fr/article.php3?id_article=115 (training) + http://www.zooscan.obs-vlfr.fr/article.php3?id_article=117 (test)

[8] https://github.com/zhenglab/PlanktonMKL/tree/master/Dataset

According to some early work (e.g. [45]), one of the main problems in plankton classification is the so-called dataset drift, that occurs when the distribution of a class changes between training and test sets. This phenomenon is related to the variation of testing conditions over time: in our experiments in order to imitate the dataset drift condition, the class distribution has not been maintained when splitting dataset between training and testing. Moreover our experiments, differently from those reported in [7] have been carried out without ad hoc parameter optimization and with no ad-hoc preprocessing for each dataset.

3.1 *Evaluation*

The evaluation of the proposed approaches and the comparison with the methods proposed in the literature is performed according to the most used performance indicators in the plankton recognition problem. Among the several performance indicators proposed for multi-label classification problems, the most used measures are the following: F-measure, accuracy and AUC. To deal with multi-classes each performance indicator is evaluated as the two-class value (one-vs-all) averaged on the number of the classes. Given C confusion matrices M_c, i.e. 2×2 tables including the number of samples true positive samples (TP_c), the number of true negatives (TN_c), the number of false positives (FP_c) and false negatives (FN_c) for each class $c \in [1..C]$, the following indicators can be derived as:

- Recall is the true positive rate, $R_c = \frac{TP_c}{TP_c + FN_c}$ $R = \frac{1}{C}\sum_c R_c$
- Precision is the positive predictive value (1- error rate), $P_c = \frac{TP_c}{TP_c + FP_c}$ $P = \frac{1}{C}\sum_c P_c$
- F-Measure is the harmonic mean of precision and recall, $F_c = 2\frac{P_c \cdot R_c}{P_c + R_c}$ $F = \frac{1}{C}\sum_c F_c$
- Accuracy is the ratio between the number of true positive samples and the total number of samples.
- AUC: the area under ROC curve is a performance indicator for 2-class problems, which can be interpreted as the probability that the classifier will assign a higher score to a randomly picked positive sample than to a randomly picked negative one. AUC is calculated as the area under the ROC curve, a graphical plot of the sensitivity of a binary classifier versus false positives (1-specificity), as its discrimination threshold is varied. In this multiclass problem, the average value of one-versus-all AUC is used [46].

3.2 *Results*

The first experiment is related to the One round fine-tuning of several CNN models: in Table 2 in the results obtained using the deep learning approaches are evaluated, by comparing the performance of seven fine-tuned CNN on the three datasets. Since CNN require input images with fixed dimensions we evaluate 2 different strategies for resizing: *SqR*, the image is padded to square size and resized to the CNN input size, *Pad*, if the size of the image is lower than the input size, the image is padded to the input size, otherwise it is padded to square size and reduced to the input size. The difference is that the second strategy does not perform rescaling for small images.

In Table 2 the performance of the seven CNN fine tuned (1R) for plankton classification are reported; moreover the last two lines includes some ensemble obtained by the fusion of scores of the above approaches:

- Fus_RS: is the sum rule among the eight networks (all except the low performance SqueezeNet) trained using the same Resize Strategy.
- Fus_1R: is the sum rule among the two Fus_RS fusions, i.e. $Fus\_1R = Fus\_RS_{SqR} + Fus\_RS_{Pad}$

From results in Table 2 the best resizing strategy is SqR; Pad performs better only in the Zooscan dataset, maybe due to varying resolution of the input images. Since SqR is the best strategy in 2 out 3 datasets we use SqR strategy in the following. In the comparison among models, DenseNet is the single best model: it obtains the best results in all the datasets.

Table 2. F-measure obtained from CNN (1R training).

F-measure (1R)	Dataset	WHOI		ZooScan		Kaggle	
	Resize Strategy	SqR	Pad	SqR	Pad	SqR	Pad
Model	AlexNet	0.923	0.900	0.804	0.825	0.872	0.835
	GoogleNet	0.935	0.931	0.836	0.841	0.890	0.869
	InceptionV3	0.947	0.939	0.843	0.856	0.904	0.869
	VGG16	0.940	0.936	0.847	0.863	0.890	0.881
	VGG19	0.939	0.937	0.840	0.848	0.890	0.873
	ResNet50	0.939	0.929	0.847	0.834	0.898	0.871
	ResNet101	0.938	0.944	0.848	0.825	0.904	0.887
	DenseNet	0.949	0.945	0.878	0.851	0.912	0.887
	SqueezeNet	0.908	0.907	0.782	0.756	0.858	0.837
Ensemble	Fus_RS	0.952	0.952	0.879	0.886	0.923	0.910
	Fus_1R	0.952		0.890		0.923	

The second experiment is related to the 2 round tuning procedure (2R) using the same models involved in the previous experiment: in Table 3 the classification performance obtained with 2 round tuning (2R) is reported, showing a behavior similar to the previous approach. In our opinion the amount of samples in the three benchmark dataset is not so small to suggest the use of a preliminary training on external dataset, therefore 1R training is enough in this classification problem. Anyway the use of a preliminary training (2R) allows to create classifiers diverse from 1R and their fusion can significantly improve the performance in this classification problem.

The third experiment is about the use of preprocessing before classification by CNN. In Table 4 the classification performance obtained by several preprocessing strategies (PR) are evaluated: for a sake of space only the best 2 among the 4 pre-processing strategies evaluated are reported, i.e. Gradient (Gra), which computes the image gradient, and Orientation (Ori), which computes the orientation image. The fusion results reported in the last row include all the preprocessing strategies, anyway this choice does not grant a performance improvement with respect to the ensemble including only Gra and Ori.

Table 3. F-measure obtained with the same models above using 2 round tuning.

F-measure (2R)	Dataset	WHOI	ZooScan	Kaggle
Model	AlexNet	0.920	0.839	0.880
	GoogleNet	0.940	0.854	0.894
	InceptionV3	0.944	0.849	0.909
	VGG16	0.929	0.840	0.887
	VGG19	0.930	0.831	0.871
	ResNet50	0.932	0.863	0.903
	ResNet101	0.938	0.869	0.904
	DenseNet	0.947	0.882	0.914
	SqeezeNet	0.901	0.759	0.858
Ensemble	Fus_2R	0.951	0.892	0.926
	Fus_2R + Fus_1R	0.953	0.897	0.926

Table 4. F-measure obtained from pre-processing Tuning (Gradient and Orientation)

F-measure (PR)	Dataset	WHOI		ZooScan		Kaggle	
	Preprocessing	Gra	Ori	Gra	Ori	Gra	Ori
Method	AlexNet	0.900	0.894	0.754	0.778	0.855	0.847
	GoogleNet	0.930	0.925	0.804	0.807	0.867	0.871
	InceptionV3	0.943	0.934	0.831	0.816	0.889	0.891
	VGG16	0.938	0.939	0.814	0.837	0.880	0.879
	VGG19	0.939	0.942	0.820	0.832	0.871	0.886
	ResNet50	0.925	0.926	0.801	0.826	0.879	0.875
	ResNet101	0.930	0.935	0.831	0.820	0.885	0.888
	DenseNet	0.938	0.928	0.850	0.827	0.908	0.903
	SqeezeNet	0.888	0.897	0.761	0.738	0.847	0.840
Ensemble	Fus_PR = Gra+Ori	0.951		0.869		0.916	
	Fus_PR_All=Gra+Ori+LBP+LTP2	0.950		0.874		0.918	
	Fus_PR + Fus_2R + Fus_1R	0.953		0.896		0.926	

Table 5. F-measure obtained from transfer learning from base SqR networks (1R)

F-measure (TL)	Dataset	WHOI				ZooScan				Kaggle			
	Selection Strategy	1FC	aFC	Man	SFFS	1FC	aFC	Man	SFFS	1FC	aFC	Man	SFFS
Model	AlexNet	0.919	0.925	0.923	0.927	---	0.822	0.794	0.812	0.859	0.884	0.872	0.882
	GoogleNet	0.932	0.932	0.937	0.940	0.797	0.797	0.785	0.800	0.882	0.882	0.879	0.892
	InceptionV3	0.914	0.914	0.947	0.930	0.835	0.835	0.804	0.806	---	---	0.892	0.854
	VGG16	0.940	0.943	0.941	0.941	---	0.829	0.805	0.825	0.880	0.888	0.880	0.877
	VGG19	0.929	0.941	0.939	0.942	---	0.830	0.814	0.820	0.872	0.890	0.880	0.909
	ResNet50	0.921	0.921	0.938	0.936	---	---	0.799	0.813	0.863	0.863	0.883	0.880
	ResNet101	0.933	0.933	0.941	0.943	---	---	0.793	0.810	0.890	0.891	0.890	0.872
	DenseNet	0.939	0.941	0.945	0.945	0.870	0.875	0.880	0.865	0.900	0.905	0.909	0.893
Ensemble	Fus_TL	0.948	0.949	0.952	0.950	0.833	0.870	0.875	0.875	0.910	0.920	0.921	0.905

The fourth experiment is about transfer learning: in Table 5 the classification performance obtained with the transfer learning methods are reported. The models fine-tuned using 1R training are used for transfer learning. In some few cases, denoted by "---", the F-measure is not reported since the SVM classifier did not converge to a result, denoting that the features selected are useless for this problem. This behavior happens mainly for the 1FC method in the Zooscan dataset, meaning that extracting features from the last FC layer (which contains only 20 features) is not beneficial for this dataset. From the results in Table 5 it is clear that the best selection approach is Man: it outperforms all the other selection approaches. It is interesting to note that the manual selection of CNN layers performed in Man allows a performance improvement with respect to aFC, while the automatic selection by SFFS decreases the performance, probably due to overfitting. Even if the above performance is lower than those reported for CNN, this results is significant since the sets of features extracted using transfer learning outperform SVM classification based on handcrafted features (FUS_Hand [39]).

Finally, in Table 6 the comparison among the ensemble proposed in this work and other results published in the literature on the same datasets are reported. The following approaches from the literature are considered:

- FUS_Hand [39] is an ensemble of four methods based on handcrafted descriptors;
- Baseline the version of [7] of the method based on a total of 263 handcrafted features reduced by feature selection [44] (optimized per dataset) and classified by SVM (with C and γ optimized by grid search in each dataset).
- Gaussian SVM [7] is a handcrafted approach based on a SVM classifier.
- MKL [7] is the best approach proposed in [7] and based on the same handcrafted features used above and combined via multiple kernel learning classifiers.

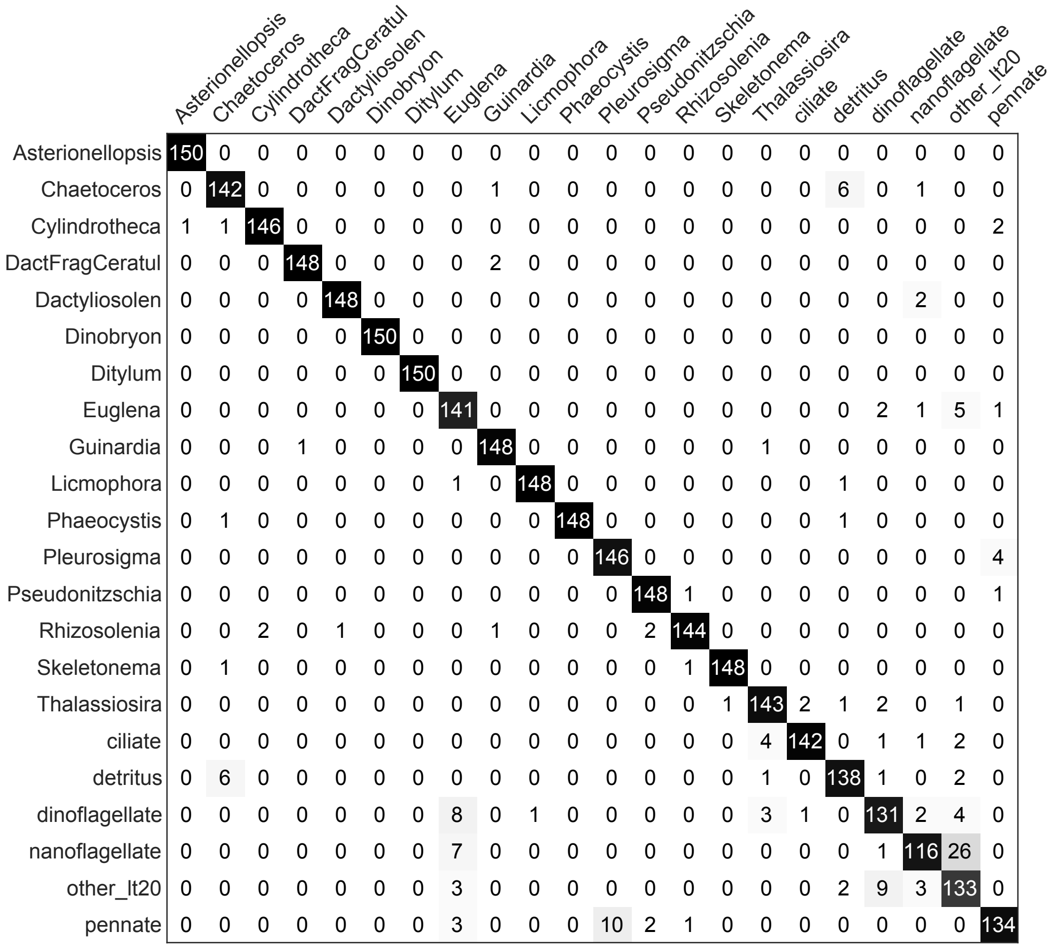
The results from different experiments are combined together in order to obtain a method that maximize the performance. This confirms the idea of diversity: if we train multiple CNN learners based on different training strategies, they can have decorrelated errors and their predictions can be summed to improve performance. The last column of Table 6 shows the Rank of the average rank, which is obtained by ranking methods for each dataset, averaging the results and ranking again the approaches.

The first ranked method is the proposed ensemble Fus_2R + Fus_1R. The results show than our ensembles work very well with respect to the other state-of-the art approaches, and since they do not require an ad hoc parameter optimization per dataset, they can be considered as "methods on the shelf" for practitioner who approach for the first time this application.

If we analyze a single model, the fusion (here not reported for a sake of space) of different resizing (SqR+Pad) or training approaches (1R+2R+PR) gains a performance improvement with respect to a single CNN; anyway if we consider the fusion of all the 7 models considered in this work, the fusion of different approaches does not significantly improve the performance, maybe because there is not enough independence among classifiers. Since the fusion with Transfer learning methods (Fus_TL) or Preprocessing approaches (Fus_PR) does not reach a sensible performance improvement with respect to Fus_2R+Fus_1R we select this method as our best approach and we report in the following Figs. 4, 5, 6, the confusion matrices of Fus_2R+Fus_1R in the three tested datasets (for future comparisons also accuracy and AUC are reported in the figure captions).

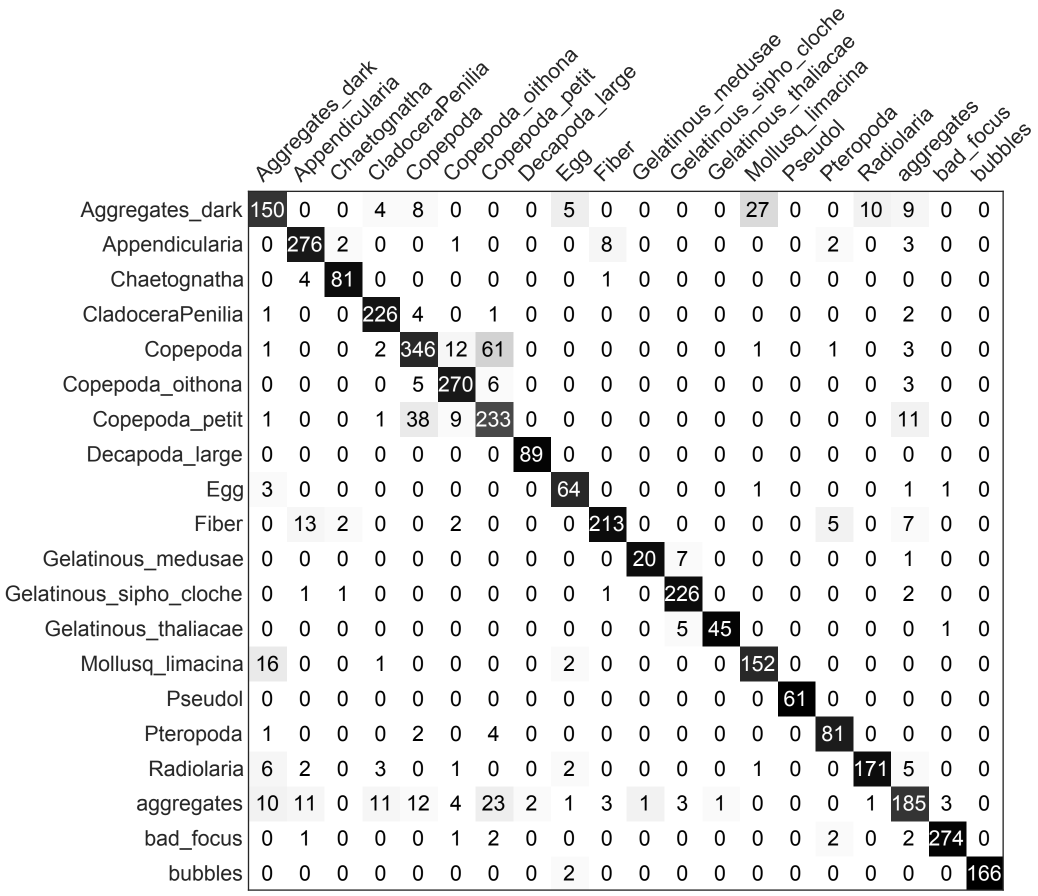
Table 6. Comparison among several ensembles and state-of-the-art methods

F-measure	Dataset	WHOI	ZooScan	Kaggle	Rank of the AVG Rank
Proposed approaches	Fus_1R	0.952	0.890	0.923	5
	Fus_2R	0.951	0.892	0.926	4
	Fus_PR	0.951	0.869	0.916	7
	Fus_TL(Man)	0.952	0.875	0.921	6
	Fus_2R + Fus_1R	0.953	0.897	0.926	1
	Fus_PR + Fus_2R + Fus_1R	0.953	0.896	0.926	2
	Fus Hand _TL(Man) + Fus_2R + Fus_1R	0.952	0.897	0.926	3
Methods from the literature	FUS_Hand [23]	0.903	0.843	0.849	9
	Baseline [7]	0.883	0.821	0.769	11
	Gaussian SVM [7]	0.896	0.861	0.830	10
	MKL (3 kernels) [7]	0.900	0.894	0.846	8



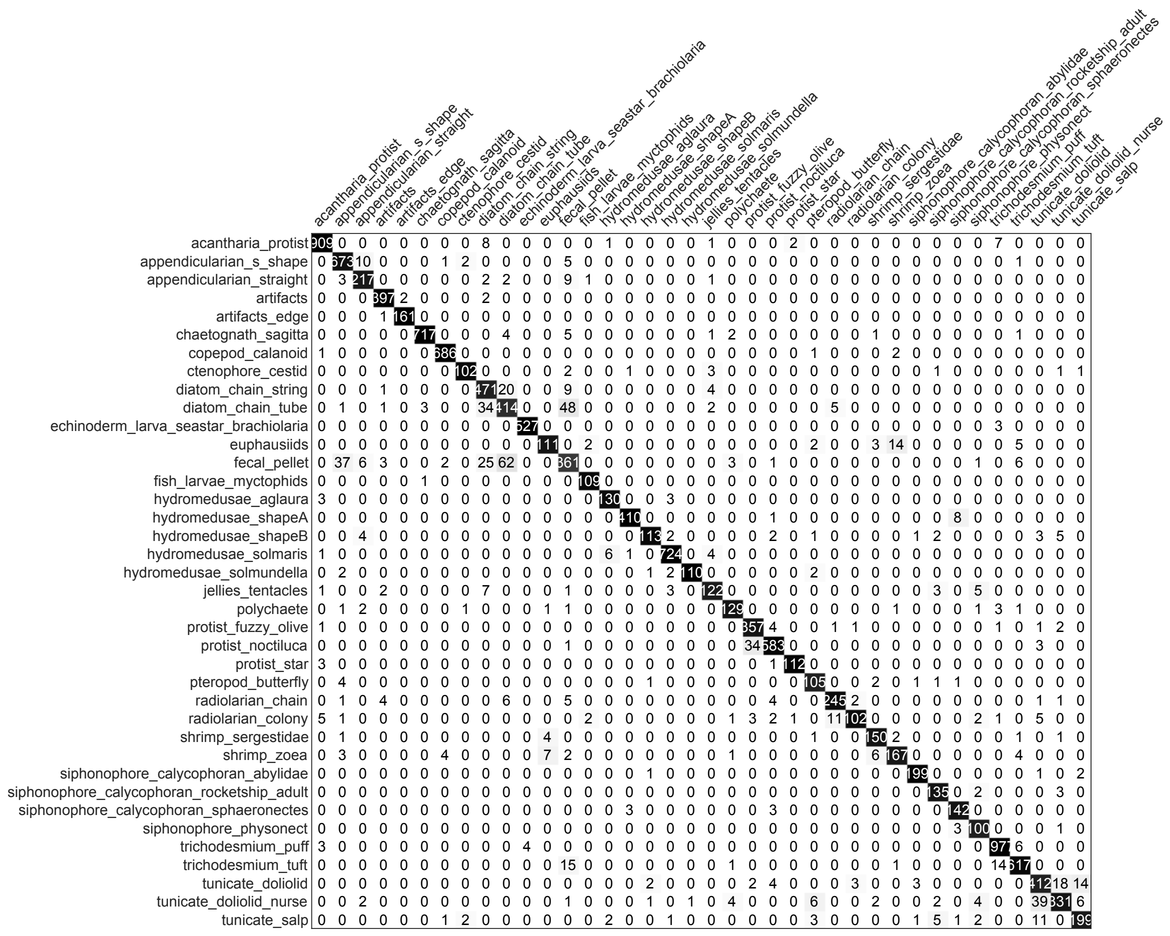
	Asterionellopsis	Chaetoceros	Cylindrotheca	DactFragCeratul	Dactyliosolen	Dinobryon	Ditylum	Euglena	Guinardia	Licmophora	Phaeocystis	Pleurosigma	Pseudonitzschia	Rhizosolenia	Skeletonema	Thalassiosira	ciliate	detritus	dinoflagellate	nanoflagellate	other_lt20	pennate
Asterionellopsis	150	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Chaetoceros	0	142	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	6	0	1	0	0
Cylindrotheca	1	1	146	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
DactFragCeratul	0	0	0	148	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0
Dactyliosolen	0	0	0	0	148	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0
Dinobryon	0	0	0	0	0	150	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Ditylum	0	0	0	0	0	0	150	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Euglena	0	0	0	0	0	0	0	141	0	0	0	0	0	0	0	0	0	0	2	1	5	1
Guinardia	0	0	0	1	0	0	0	0	148	0	0	0	0	0	0	1	0	0	0	0	0	0
Licmophora	0	0	0	0	0	0	0	1	0	148	0	0	0	0	0	0	0	1	0	0	0	0
Phaeocystis	0	1	0	0	0	0	0	0	0	0	148	0	0	0	0	0	0	1	0	0	0	0
Pleurosigma	0	0	0	0	0	0	0	0	0	0	0	146	0	0	0	0	0	0	0	0	0	4
Pseudonitzschia	0	0	0	0	0	0	0	0	0	0	0	0	148	1	0	0	0	0	0	0	0	1
Rhizosolenia	0	0	2	0	1	0	0	0	1	0	0	0	2	144	0	0	0	0	0	0	0	0
Skeletonema	0	1	0	0	0	0	0	0	0	0	0	0	0	1	148	0	0	0	0	0	0	0
Thalassiosira	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	143	2	1	2	0	1	0
ciliate	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	142	0	1	1	2	0
detritus	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	138	1	0	2	0
dinoflagellate	0	0	0	0	0	0	0	8	0	1	0	0	0	0	0	3	1	0	131	2	4	0
nanoflagellate	0	0	0	0	0	0	0	7	0	0	0	0	0	0	0	0	0	0	1	116	26	0
other_lt20	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	2	9	3	133	0
pennate	0	0	0	0	0	0	0	3	0	0	0	10	2	1	0	0	0	0	0	0	0	134

Fig. 3. Confusion matrix of Fus_2R+Fus_1R in the WHOI dataset. The accuracy is 0.9527, AUC = 0.9989.



	Aggregates_dark	Appendicularia	Chaetognatha	CladoceraPenilia	Copepoda	Copepoda_oithona	Copepoda_petit	Decapoda_large	Egg	Fiber	Gelatinous_medusae	Gelatinous_sipho_cloche	Gelatinous_thaliacae	Mollusq_limacina	Pseudol	Pteropoda	Radiolaria	aggregates	bad_focus	bubbles
Aggregates_dark	150	0	0	4	8	0	0	0	5	0	0	0	0	27	0	0	10	9	0	0
Appendicularia	0	276	2	0	0	1	0	0	0	8	0	0	0	0	0	2	0	3	0	0
Chaetognatha	0	4	81	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
CladoceraPenilia	1	0	0	226	4	0	1	0	0	0	0	0	0	0	0	0	0	2	0	0
Copepoda	1	0	0	2	346	12	61	0	0	0	0	0	0	1	0	1	0	3	0	0
Copepoda_oithona	0	0	0	0	5	270	6	0	0	0	0	0	0	0	0	0	0	3	0	0
Copepoda_petit	1	0	0	1	38	9	233	0	0	0	0	0	0	0	0	0	0	11	0	0
Decapoda_large	0	0	0	0	0	0	0	89	0	0	0	0	0	0	0	0	0	0	0	0
Egg	3	0	0	0	0	0	0	0	64	0	0	0	0	1	0	0	0	1	1	0
Fiber	0	13	2	0	0	2	0	0	0	213	0	0	0	0	0	5	0	7	0	0
Gelatinous_medusae	0	0	0	0	0	0	0	0	0	0	20	7	0	0	0	0	0	1	0	0
Gelatinous_sipho_cloche	0	1	1	0	0	0	0	0	0	1	0	226	0	0	0	0	0	2	0	0
Gelatinous_thaliacae	0	0	0	0	0	0	0	0	0	0	0	5	45	0	0	0	0	0	1	0
Mollusq_limacina	16	0	0	1	0	0	0	0	2	0	0	0	0	152	0	0	0	0	0	0
Pseudol	0	0	0	0	0	0	0	0	0	0	0	0	0	0	61	0	0	0	0	0
Pteropoda	1	0	0	0	2	0	4	0	0	0	0	0	0	0	0	81	0	0	0	0
Radiolaria	6	2	0	3	0	1	0	0	2	0	0	0	0	1	0	0	171	5	0	0
aggregates	10	11	0	11	12	4	23	2	1	3	1	3	1	0	0	0	1	185	3	0
bad_focus	0	1	0	0	0	1	2	0	0	0	0	0	0	0	0	2	0	2	274	0
bubbles	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	166

Fig. 4. Confusion matrix of Fus_2R+Fus_1R in the ZooScan dataset. The accuracy is 0.8826, AUC= 0.9952.



	acantharia_protist	appendicularian_s_shape	appendicularian_straight	artifacts	artifacts_edge	chaetognath_sagitta	copepod_calanoid	ctenophore_cestid	diatom_chain_string	diatom_chain_tube	echinoderm_larva_seastar_brachiolaria	euphausiids	fecal_pellet	fish_larvae_myctophids	hydromedusae_aglaura	hydromedusae_shapeA	hydromedusae_shapeB	hydromedusae_solmaris	hydromedusae_solmundella	jellies_tentacles	polychaete	protist_fuzzy_olive	protist_noctiluca	protist_star	pteropod_butterfly	radiolarian_chain	radiolarian_colony	shrimp_sergestidae	shrimp_zoea	siphonophore_calycophoran_abylidae	siphonophore_calycophoran_rocketship_adult	siphonophore_calycophoran_sphaeronectes	siphonophore_physonect	trichodesmium_puff	trichodesmium_tuft	tunicate_doliolid	tunicate_doliolid_nurse	tunicate_salp
acantharia_protist	909	0	0	0	0	0	0	0	8	0	0	0	0	0	1	0	0	0	0	1	0	0	0	2	0	0	0	0	0	0	0	0	0	7	0	0	0	0
appendicularian_s_shape	0	673	10	0	0	0	1	2	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
appendicularian_straight	0	3	217	0	0	0	0	0	2	2	0	0	9	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
artifacts	0	0	0	397	2	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
artifacts_edge	0	0	0	1	161	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
chaetognath_sagitta	0	0	0	0	0	717	0	0	0	4	0	0	5	0	0	0	0	0	0	1	2	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0
copepod_calanoid	1	0	0	0	0	0	686	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	2	0	0	0	0	0	0	0	0	0
ctenophore_cestid	0	0	0	0	0	0	0	102	0	0	0	0	2	0	0	1	0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1
diatom_chain_string	0	0	0	1	0	0	0	0	471	20	0	0	9	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
diatom_chain_tube	0	1	0	1	0	3	0	0	34	414	0	0	48	0	0	0	0	0	0	2	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0
echinoderm_larva_seastar_brachiolaria	0	0	0	0	0	0	0	0	0	0	527	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0
euphausiids	0	0	0	0	0	0	0	0	0	0	0	111	0	2	0	0	0	0	0	0	0	0	0	0	2	0	0	3	14	0	0	0	0	0	5	0	0	0
fecal_pellet	0	37	6	3	0	0	2	0	25	62	0	0	361	0	0	0	0	0	0	0	3	0	1	0	0	0	0	0	0	0	0	0	1	0	6	0	0	0
fish_larvae_myctophids	0	0	0	0	0	1	0	0	0	0	0	0	0	109	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
hydromedusae_aglaura	3	0	0	0	0	0	0	0	0	0	0	0	0	0	130	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
hydromedusae_shapeA	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	410	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0
hydromedusae_shapeB	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	113	2	0	0	0	0	2	0	1	0	0	0	0	1	2	0	0	0	0	3	5	0
hydromedusae_solmaris	1	0	0	0	0	0	0	0	0	0	0	0	0	0	6	1	0	724	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
hydromedusae_solmundella	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	110	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0
jellies_tentacles	1	0	0	2	0	0	0	0	7	0	0	0	1	0	0	0	0	3	0	122	0	0	0	0	0	0	0	0	0	0	3	0	5	0	0	0	0	0
polychaete	0	1	2	0	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	0	129	0	0	0	0	0	0	0	1	0	0	0	1	3	1	0	0	0
protist_fuzzy_olive	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	357	4	0	0	1	1	0	0	0	0	0	0	1	0	1	2	0
protist_noctiluca	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	34	583	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0
protist_star	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	112	0	0	0	0	0	0	0	0	0	0	0	0	0	0
pteropod_butterfly	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	105	0	0	2	0	1	1	1	0	0	0	0	0	0
radiolarian_chain	0	1	0	4	0	0	0	0	0	6	0	0	5	0	0	0	0	0	0	0	0	0	4	0	0	245	2	0	0	0	0	0	0	0	0	1	1	0
radiolarian_colony	5	1	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	1	3	2	1	0	11	102	0	0	0	0	0	2	1	0	5	0	0
shrimp_sergestidae	0	1	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	150	2	0	0	0	0	0	1	0	1	0
shrimp_zoea	0	3	0	0	0	0	4	0	0	0	0	7	2	0	0	0	0	0	0	0	1	0	0	0	0	0	0	6	167	0	0	0	0	0	4	0	0	0
siphonophore_calycophoran_abylidae	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	199	0	0	0	0	0	1	0	2
siphonophore_calycophoran_rocketship_adult	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	135	0	2	0	0	0	3	0
siphonophore_calycophoran_sphaeronectes	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	142	0	0	0	0	0	0
siphonophore_physonect	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	100	0	0	0	1	0
trichodesmium_puff	3	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	977	6	0	0	0
trichodesmium_tuft	0	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	14	617	0	0	0
tunicate_doliolid	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	2	4	0	0	0	3	0	0	3	0	0	0	0	0	412	18	14
tunicate_doliolid_nurse	0	0	2	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	4	0	0	0	6	0	0	2	0	0	2	0	4	0	0	39	331	6
tunicate_salp	0	0	0	0	0	0	1	2	0	0	0	0	0	0	2	0	0	1	0	0	0	0	0	0	3	0	0	0	0	1	5	1	2	0	0	11	0	199

Fig. 5. Confusion matrix of Fus_2R+Fus_1R in the Kaggle dataset. The accuracy is 0.9413, AUC=0.9983.

The analysis of the confusion matrices can be useful to find out the classes which are more difficult to distinguish: *nanoflagellate* vs. *other_lt20* for the WHOI dataset, *copepoda* and *copepod_petit* in the ZooScan dataset and *fecal_pellet* vs. *diatom_chain_tube* in the Kaggle dataset. Some images from the above cited couple of classes are shown in Fig. 6 (2 images from each dataset): they have a very low extra-class variation, therefore they are difficult to distinguish even by a human expert. In our opinion it is highly difficult to improve the system in order to avoid these misclassification errors. Anyway the performance limit is not already reached, since there are some samples misclassified by the system, that can be visually distinguished by a human expert: in Fig. 7 we report two of these cases taken from the Kaggle dataset.

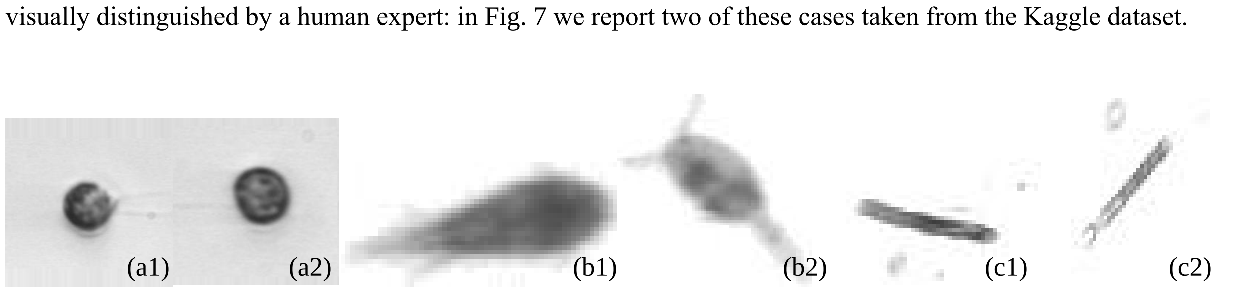


Fig. 6. Misclassified samples hard to distinguish also for a human expert: (a1-a2) WHOI dataset *nanoflagellate* vs. *other_lt20* (b1-b2) ZooScan dataset *copepoda* vs. *copepod_petit* (c1-c2) Kaggle dataset *fecal_pellet* vs. *diatom_chain_tube*

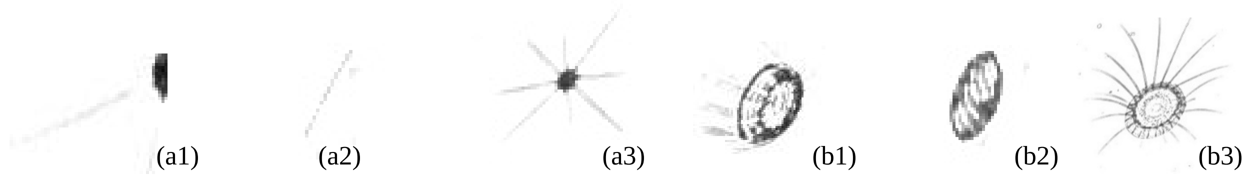


Fig. 7. Misclassified samples due to low image quality in the Keggle dataset: (a1) is *diatom_chain_string* sample incorrectly classified as *acantharia_protist,* maybe due to cropping; (a2) and (a3) are two correct samples from *acantharia_protist* and *diatom_chain_string*, respectively. (b1) is a *hydromedusae_solmaris* sample incorrectly classified as a *tunicate_doliolid*; (b2) and (b3) are two correct samples from *tunicate_doliolid* and *hydromedusae_solmaris*, respectively.

4. Conclusions

In this paper, we study both deep learned approaches and transfer learning with the aim of exploiting their diversity for designing an ensemble of classifiers. Our system is based on the fine-tuning of different CNN architectures trained using different strategies, which fused together in a final ensemble gains higher performance than the single networks. In this work we evaluate different CNN models, different training methods and approaches for transfer learning by automated selection of the CNN to be used for feature extraction. Our experiments show that the best model for this classification problem is DenseNet, moreover combination of such different descriptors/methods in a heterogeneous ensemble grants a substantial performance improvement with respect to other state-of-the-art approaches based on feature selection and classification.

Our ensembles are validated using three well-known plankton datasets and compared with other approaches proposed in the literature. Despite of the complexity of the method, one of the main benefits of the proposed system is to work well out-of-the-box in different problems, requiring few parameters tuning without specific optimization for each dataset. The experimental results show that the proposed system obtains state-of-the-art performance in all the tested datasets.

As a future work, we plan to evaluate different layer selection strategies for transfer learning, the evaluation of other preprocessing methods and the study of methods based on diversity for classifier selection.

To reproduce the experiments reported in this paper and for future comparisons, the MATLAB code of all the ensembles will be available in the GitHub repository: https://github.com/LorisNanni.

Acknowledgments We would like to acknowledge the support that NVIDIA provided us through the GPU Grant Program. We used a donated TitanX GPU to train CNNs used in this work.

References

[1] F. Zhao, F. Lin, H.S. Seah, Binary SIPPER plankton image classification using random subspace, Neurocomputing. 73 (2010) 1853–1860. doi:10.1016/j.neucom.2009.12.033.

[2] R.J. Olson, H.M. Sosik, A submersible imaging-in-flow instrument to analyze nano-and microplankton: Imaging FlowCytobot, Limnol. Oceanogr. Methods. 5 (2007) 195–203. doi:10.4319/lom.2007.5.195.

[3] R.K. Cowen, C.M. Guigand, In situ ichthyoplankton imaging system (ISIIS): System design and preliminary results, Limnol. Oceanogr. Methods. 6 (2008) 126–132. doi:10.4319/lom.2008.6.126.

[4] G. Gorsky, M.D. Ohman, M. Picheral, S. Gasparini, L. Stemmann, J.B. Romagnan, et al., Digital zooplankton image analysis using the ZooScan integrated system, J. Plankton Res. 32 (2010) 285–303. doi:10.1093/plankt/fbp124.

[5] H. Lee, M. Park, J. Kim, Plankton classification on imbalanced large scale database via convolutional neural networks with transfer learning, in: Proc. - Int. Conf. Image Process. ICIP, 2016: pp. 3713–3717. doi:10.1109/ICIP.2016.7533053.

[6] J. Dai, R. Wang, H. Zheng, G. Ji, X. Qiao, ZooplanktoNet: Deep convolutional network for zooplankton classification, in: Ocean. 2016 - Shanghai, 2016. doi:10.1109/OCEANSAP.2016.7485680.

[7] H. Zheng, R. Wang, Z. Yu, N. Wang, Z. Gu, B. Zheng, Automatic plankton image classification combining multiple view features via multiple kernel learning, BMC Bioinformatics. 18 (2017) 1–18. doi:10.1186/s12859-017-1954-8.

[8] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, et al., Recent advances in convolutional neural networks, Pattern Recognit. 77 (2018) 354–377. doi:10.1016/J.PATCOG.2017.10.013.

[9] X. Tang, W. Kenneth Stewart, L. Vincent, H. Huang, M. Marra, S.M. Gallager, et al., Automatic Plankton Image Recognition, Artif. Intell. Rev. 12 (1998) 177–199. doi:10.1023/A:1006517211724.

[10] Q. Hu, C. Davis, Automatic plankton image recognition with co-occurrence matrices and Support Vector Machine, Mar. Ecol. Prog. Ser. 295 (2005) 21–31. doi:10.3354/meps295021.

[11] Z. Li, F. Zhao, J. Liu, Y. Qiao, Pairwise Nonparametric Discriminant Analysis for Binary Plankton Image Recognition, IEEE J. Ocean. Eng. 39 (2014) 695–701. doi:10.1109/JOE.2013.2280035.

[12] J. Ellen, H. Li, M.D. Ohman, Quantifying California current plankton samples with efficient machine learning techniques, in: Ocean. 2015 - MTS/IEEE Washingt., 2015: pp. 1–9. doi:10.23919/OCEANS.2015.7404607.

[13] L. Chang, R. Wang, H. Zheng, J. Dai, B. Zheng, Phytoplankton feature extraction from microscopic images based on SURF-PCA, in: Ocean. 2016 - Shanghai, 2016: pp. 1–4. doi:10.1109/OCEANSAP.2016.7485699.

[14] H. Bi, Z. Guo, M.C. Benfield, C. Fan, M. Ford, S. Shahrestani, et al., A Semi-Automated Image Analysis Procedure for In Situ Plankton Imaging Systems, PLoS One. 10 (2015) 1–17. doi:10.1371/journal.pone.0127121.

[15] M. Moniruzzaman, S.M.S. Islam, M. Bennamoun, P. Lavery, Deep learning on underwater marine object detection: A survey, in: Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), 2017: pp. 150–160. doi:10.1007/978-3-319-70353-4_13.

[16] J. Schmidhuber, Deep learning in neural networks: An overview, Neural Networks. 61 (2015) 85–117. doi:10.1016/J.NEUNET.2014.09.003.

[17] S. Dieleman, J. De Fauw, K. Kavukcuoglu, Exploiting Cyclic Symmetry in Convolutional Neural Networks, CoRR. abs/1602.02660 (2016). http://arxiv.org/abs/1602.02660.

[18] O. Py, H. Hong, S. Zhongzhi, Plankton classification with deep convolutional neural networks, in: 2016 IEEE Inf. Technol. Networking, Electron. Autom. Control Conf., 2016: pp. 132–136. doi:10.1109/ITNEC.2016.7560334.

[19] J. Dai, Z. Yu, H. Zheng, B. Zheng, N. Wang, A Hybrid Convolutional Neural Network for Plankton Classification, in: C.-S. Chen, J. Lu, K.-K. Ma (Eds.), Comput. Vis. -- ACCV 2016 Work., Springer International Publishing, Cham, 2017: pp. 102–114.

[20] J. Cui, B. Wei, C. Wang, Z. Yu, H. Zheng, B. Zheng, et al., Texture and Shape Information Fusion of Convolutional Neural Network for Plankton Image Classification, in: 2018 Ocean. - MTS/IEEE Kobe Techno-Oceans, 2018: pp. 1–5. doi:10.1109/OCEANSKOBE.2018.8559156.

[21] E. Bochinski, G. Bacha, V. Eiselein, T.J.W. Walles, J.C. Nejstgaard, T. Sikora, Deep Active Learning for In Situ Plankton Classification, in: Z. Zhang, D. Suter, Y. Tian, A. Branzan Albu, N. Sidère, H. Jair Escalante (Eds.), Pattern Recognit. Inf. Forensics, Springer International Publishing, Cham, 2019: pp. 5–15.

[22] F.C.M. Rodrigues, N.S.T. Hirata, A.A. Abello, L.T.D. La Cruz, R.M. Lopes, R.H. Jr., Evaluation of Transfer Learning Scenarios in Plankton Image Classification, in: Proc. 13th Int. Jt. Conf. Comput. Vision, Imaging Comput. Graph. Theory Appl. - Vol. 5 VISAPP, SciTePress, 2018: pp. 359–366. doi:10.5220/0006626703590366.

[23] L. Nanni, A. Lumini, Ocean Ecosystems Plankton Classification, in: M. Hassaballah, K.M. Hosny (Eds.), Recent Adv. Comput. Vis. Theor. Appl., Springer, 2018.

[24] A. Goodfellow, Ian, Bengio, Yoshua, Courville, Deep Learning, MIT Press. (2016). http://www.deeplearningbook.org/.

[25] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, Adv. Neural Inf. Process. Syst. (2012) 1–9. doi:http://dx.doi.org/10.1016/j.protcy.2014.09.007.

[26] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, et al., Going deeper with convolutions, in: Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., 2015: pp. 1–9. doi:10.1109/CVPR.2015.7298594.

[27] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the Inception Architecture for Computer

Vision, in: 2016 IEEE Conf. Comput. Vis. Pattern Recognit., 2016: pp. 2818–2826. doi:10.1109/CVPR.2016.308.

[28] K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, Int. Conf. Learn. Represent. (2015) 1–14. doi:10.1016/j.infsof.2008.09.005.

[29] K. He, X. Zhang, S. Ren, J. Sun, Deep Residual Learning for Image Recognition, in: 2016 IEEE Conf. Comput. Vis. Pattern Recognit., 2016: pp. 770–778. doi:10.1109/CVPR.2016.90.

[30] G. Huang, Z. Liu, L. Van Der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017, 2017. doi:10.1109/CVPR.2017.243.

[31] F.N. Iandola, M.W. Moskewicz, K. Ashraf, S. Han, W.J. Dally, K. Keutzer, SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size, ArXiv. (2016). doi:10.1007/978-3-319-24553-9.

[32] N.S. Vu, Exploring patterns of gradient orientations and magnitudes for face recognition, IEEE Trans. Inf. Forensics Secur. 8 (2013) 295–304. doi:10.1109/TIFS.2012.2224866.

[33] L. Nanni, A. Lumini, S. Brahnam, Local binary patterns variants as texture descriptors for medical image analysis, Artif. Intell. Med. 49 (2010) 117–125.

[34] L. Nanni, A. Lumini, Wavelet selection for disease classification by DNA microarray data, Expert Syst. Appl. 38 (2011) 990–995.

[35] C.-H. Chan, J. Kittler, K. Messer, Multi-scale local binary pattern histograms for face recognition, in: Int. Conf. Biometrics, 2007: pp. 809–818.

[36] G. Levi, T. Hassner, Emotion Recognition in the Wild via Convolutional Neural Networks and Mapped Binary Patterns, in: Proc. 2015 ACM Int. Conf. Multimodal Interact., ACM, New York, NY, USA, 2015: pp. 503–510. doi:10.1145/2818346.2830587.

[37] X. Tan, B. Triggs, Enhanced local texture feature sets for face recognition under difficult lighting conditions, IEEE Trans. Image Process. 19 (2010) 1635–1650. doi:10.1109/TIP.2010.2042645.

[38] M.J. Afridi, A. Ross, E.M. Shapiro, On automated source selection for transfer learning in convolutional neural networks, Pattern Recognit. 73 (2018) 65–75. doi:10.1016/J.PATCOG.2017.07.019.

[39] L. Nanni, S. Ghidoni, S. Brahnam, Handcrafted vs. non-handcrafted features for computer vision classification, Pattern Recognit. 71 (2017) 158–172. doi:10.1016/j.patcog.2017.05.025.

[40] S. Tulsiani, J. Carreira, J. Malik, Pose induction for novel object categories, in: Proc. IEEE Int. Conf. Comput. Vis., 2015. doi:10.1109/ICCV.2015.16.

[41] P. Pudil, J. Novovičová, J. Kittler, Floating search methods in feature selection, Pattern Recognit. Lett. (1994). doi:10.1016/0167-8655(94)90127-9.

[42] C.C.J.C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Min. Knowl. Discov. 2 (1998) 121–167. doi:10.1023/A:1009715923555.

[43] R.O. Duda, P.E. Hart, D.G. Stork, Pattern classification, John Wiley & Sons, 2012.

[44] H.M. Sosik, R.J. Olson, Automated taxonomic classification of phytoplankton sampled with imaging-in-flow cytometry, Limnol. Oceanogr. Methods. 5 (2007) 204–216. doi:10.4319/lom.2007.5.204.

[45] P. Gonzalez, E. Alvarez, J. Diez, A. Lopez-Urrutia, J.J. del Coz, Validation methods for plankton image classification systems, Limnol. Oceanogr. Methods. 15 (2017) 221–237. doi:10.1002/lom3.10151.

[46] T.C.W. Landgrebe, R.P.W. Duin, Approximating the multiclass ROC by pairwise analysis, Pattern Recognit. Lett. 28 (2007) 1747–1758. doi:10.1016/j.patrec.2007.05.001.