FACTA UNIVERSITATIS
Series: **Automatic Control and Robotics** Vol. 20, Nº 1, 2021, pp. 1 - 19
https://doi.org/10.22190/FUACR201118001M

Regular Paper

SPECIES IDENTIFICATION FOR AQUATIC BIOMONITORING USING DEEP RESIDUAL CNN AND TRANSFER LEARNING

UDC ((004.032.26+004.6):504/502)

Aleksandar Milosavljević[1], Đurađ Milošević[2], Bratislav Predić[1]

[1]University of Niš, Faculty of Electronic Engineering,
Department of Computer Science, Republic of Serbia
[2]University of Niš, Faculty of Sciences and Mathematics,
Department of Biology and Ecology, Republic of Serbia

Abstract. *Aquatic insects and other benthic macroinvertebrates are mostly used as bioindicators of the ecological status of freshwaters. However, an expensive and time-consuming process of species identification represents one of the key obstacles for reliable biomonitoring of aquatic ecosystems. In this paper, we propose a deep learning (DL) based method for species identification that we evaluated on several available public datasets (FIN-Benthic, STONEFLY9, and EPT29) along with our Chironomidae dataset (CHIRO10). The proposed method relies on three DL techniques used to improve robustness when training is done on a relatively small dataset: transfer learning, data augmentation, and feature dropout. We applied transfer learning by employing a ResNet-50 deep convolutional neural network (CNN) pretrained on ImageNet 2012 dataset. The results show significant improvement compared to original FIN-Benthic, STONEFLY9, and EPT29 contributions and confirm that there is a considerable gain in accuracy when there are multiple images per specimen.*

Key words: *Convolutional neural networks, image classification, transfer learning, data augmentation, biomonitoring*

1. INTRODUCTION

The diversity of genes, species, and ecosystems are declining globally faster than at any time in human history [1]. The aquatic ecosystems are showing among the highest rates of decline with the alarming acceleration of biodiversity loss. That poses a need for the development of proper and cost-effective tools for monitoring aquatic biota diversity.

Received November 18, 2020 / Accepted February 11, 2021
Corresponding author: Aleksandar Milosavljević
University of Niš, Faculty of Electronic Engineering, Aleksandra Medvedeva 14, 18000 Niš, Republic of Serbia
E-mail: aleksandar.milosavljevic@elfak.ni.ac.rs

A traditional morphological identification approach in biomonitoring requires ‘as high as possible’ taxonomic resolution in data [2]. However, the identification of macroinvertebrate taxa based on morphological features can be problematic, as the number of misidentified taxa increases with taxonomic resolution. In addition, morphological identification is a time-consuming process, which is not cost-effective for routine monitoring and requires taxonomic expertise for many aquatic biota groups (e.g. non-biting midge) [3], [4]. All such drawbacks of traditional biomonitoring have created a need for developing alternative approaches in macroinvertebrate sample processing. Recent advances in computer vision, and especially image classification, which have been introduced using convolutional neural networks (CNN) and deep learning (DL), have paved the way for reliable automation of the identification processes.

Image classification in computer vision is a problem where, based on a set of images with known categories, a model is built that can predict a category, with certain accuracy, for some new images. The task itself is not easy due to the different variations present in the images. Typically, a data-driven approach is used to solve this problem. Instead of trying to describe each of the classes that need to be identified, many examples are used for each of the classes to train a particular model (classifier) to be able to identify them. In order to evaluate the quality of the trained model when classifying new samples, a certain part of the starting set is put aside and used for testing purposes.

The traditional approach to the problem of image classification relied on ‘hand-crafting’ various image feature extractors that were later used to train the classifier [5]–[8]. Although artificial neural networks (ANN) have been used before [9], major advances in this area have been made with the introduction of CNNs [10]. CNNs combine three architectural ideas that help them achieve a certain level of invariance to translation and distortion: local receptive fields, shared weights, and spatial subsampling [11]. CNNs came into the focus of interest in 2012 after a network called AlexNet [12] won in the ImageNet Large Scale Visual Recognition Challenge [13] (hereafter abbreviated as ImageNet). AlexNet had a top-5 error of 15.3%, which was 10.8% better than the second-placed solution. This result was possible due to the application of Graphical Processing Units (GPU) for the training, which is considered a turning point for the development of DL. Over the next few years, both deep CNNs and the results they achieved in the ImageNet competition undergo huge improvement. The 2013 winner was ZFNet [14] with a top-5 error of 14.8%. The importance of this solution is predominantly in the developed visualization technique by mapping learned filters into images. The coming year brought huge improvement and two significant solutions. VGGNet [15] achieved a top-5 error of 7.3% while promoting simplicity and depth. The winner of 2014, with a top-5 error of 6.7%, is GoogLeNet [16], which, in addition to a significantly better result, also had 12x fewer parameters than AlexNet. The following year, ResNet [17] recorded a top-5 error of 3.6%, which was almost twice as good as the previous year. The ResNet architecture is inspired by the philosophy of VGGNet with the introduction of a residual learning approach to facilitate the training of deep networks. The network that won in 2015 is a 152-layer variation of ResNet. In the proposed method for species identification, we used the ResNet-50 variation of this architecture as an encoder, i.e., a network that extracts features from input images.

The importance of ImageNet competition goes beyond its original goal which involves the classification of images into 1,000 categories. Since few real applications can match the ImageNet set by the number of samples (about 1,200,000 training samples), the idea is to use the networks previously trained on the ImageNet dataset as general feature extractors or a starting point for further training. The approach is called transfer learning [18], [19] and is one of the key elements of the proposed solution.

The rest of the paper is organized as follows. Section 2 presents related work, while Section 3 provides an extensive description of the datasets we used and the preprocessing we performed. In Section 4 we describe the proposed method, including details specific to the implementation. Details about conducted experiments and achieved results, along with the corresponding discussion are given in Section 5. Finally, Section 6 concludes the paper and suggests topics for future research.

2. RELATED WORK

The use of deep CNNs represents the current state-of-the-art for image classification tasks in general. This approach has become very popular in the domain of biology and medicine [20]–[24] where the aim is to replace human experts in the diagnostic process. Transfer learning, in the form of ImageNet pretrained feature extractors, also proves to be very useful in the field [21] despite the very different nature of the images being processed. When it comes to image-based species identification, according to Wäldchen and Mäder's review [25], deep learning-based approaches represent current state-of-the-art and pretty much revolutionized the domain.

The problem of automated image-based taxonomic identification of benthic macroinvertebrates is addressed in [26]–[31]. Lytle *et al.* [26] developed one of the first systems to address this problem. Their BugID system used the Scale Invariant Feature Transform (SIFT) descriptors [32] in combination with Random Forests (RF) classifier that was previously described in [33]. When tested on images from 9 larval stonefly taxa (STONEFLY9 dataset [26], [34]), BugID correctly identified 94.5% of specimens.

Larios *et al.* [27] used a stacking classification approach that combines the results from multiple classifiers, having the benefit of allowing each classifier to handle a different feature space. As base feature extractors they used Histogram of Oriented Gradients (HOG), Beam Angle Statistics (BAS), and SIFT. For testing purposes, they created a dataset which contains 4722 images with 29 aquatic insect species, belonging to the 3 most common orders used to assess the aquatic ecosystem health: Ephemeroptera (mayflies), Plecoptera (stoneflies), and Trichoptera (caddisflies). The dataset is known as EPT29. The best result of 88.06% was obtained using spatial-pyramid kernel support vector machines (SVM) applied to stacked 3-feature-types combination. All the experiments were performed with a stratified 3-fold cross-validation setup.

Another 'pre-deep learning' approach to address the problem of cost-intensive manual taxonomic classification and retrieval of macroinvertebrate specimens was introduced by Kiranyaz *et al.* [28]. The benthic macroinvertebrate image dataset used in this work consists of 1350 images representing 8 different taxonomic groups. For feature extraction, they applied ImageJ, a public domain Java-based image processing software, that produced a 15-D feature vector of each macroinvertebrate image from the dataset. Obtained feature vectors were used for training different classifiers: SVM, Bayesian Classifiers (BC), and two ANN models: Multi-Layer Perceptron (MLP) and Radial Basis Function Network (RBFN). To evaluate the effect of the data partitioning, they randomly selected 10 different training and test partitions each with 650 training and 700 test samples. SVM classifiers showed slightly better performance than BCs with classification errors (CEs) ranging from 4.86–7.86%

compared to 5.71–7.86% achieved by BCs. MLP outperformed all other approaches with recorded best CE of 3.57% on the test set. RBFNs, on the other hand, showed the worst results.

Joutsijoki *et al.* [29] used the same dataset and feature extraction methodology as in [28] to examine the suitability of ANNs for automated taxa identification of macroinvertebrates. They experimented with different training algorithms of MLP, Probabilistic Neural Network (PNN), and RBFN. The best classification accuracies they achieved are 95.3% for MLP, 92.8% for PNN, and 95.7% for RBFN. It is important to mention that these results were achieved using a different training methodology, i.e. 80% (1080) of images were used for training, 10% (135) for validation, and 10% (135) for testing. The reported accuracies are mean values calculated after training 100 models on different data splits.

Raitoharju *et al.* [30] made publicly available benchmark datasets (FIN-Benthic) that enable the evaluation of classification methods for distinguishing visually similar categories of aquatic macroinvertebrate taxa. There are three overlapping datasets with 64, 29, and 9 categories. The number of images per category ranges from 7 to 577. They also proposed a methodology for taking photos of the specimens using two cameras from two different directions. This helps in the identification process since it is possible to combine predictions for two images to determine the species of the specimen. Another important property of the datasets is that they have 10 explicit splits in train (50%), validation (20%), and test (30%) subsets. This removes ambiguity and helps with comparing results. The experiments they conducted, for the first time, included direct use of CNNs, i.e. AlexNet [12] in particular. They achieved mean classification accuracies of 75.74% on Dataset 1, 81.04% on Dataset 2, and 90.14% on Dataset 3. Besides direct use of CNNs, the paper also shows comparative results when previously trained AlexNet is used to extract features (output of ‘fc7’ layer, 4096-D feature vector) and more traditional classification methods are applied. However, end-to-end training proves slightly better than the best alternatives like SVMs.

A paper by Milošević *et al.* [31] represents our contribution to the field of automated species identification of non-biting midges (Diptera: Chironomidae). The research involved the development of a chironomid image dataset of 10 morphologically similar species from the same genus or subfamilies, comprising 1846 images (one per specimen). We used the same deep CNN model based on pretrained ResNet-50 encoder. The difference between the approach of the present paper and that of our previous paper is that the latter used larger images (512×512 pixels) and a single data split (80% for training, 20% for validation). In our previous paper, we presented the results of three experiments with datasets on chironomid species, genera, and subfamilies levels. The final models achieved an accuracy of 99.465% for species-level identification, while at the genus and subfamily level all images were correctly assigned (100% accuracy).

3. Data

The basis for our research was our chironomid dataset comprised of 1846 images of 10 chironomid species [31] (will be further referred to as CHIRO10 dataset). Species selection included morphologically similar (same genus), moderately different (different genera), and highly different species (different subfamilies). The images show chironomid larvae head-capsules photographed from the ventral point of view which is most informative in terms of species-specific morphological characteristics. The images were obtained using Leica MZ16A stereomicroscope with 150x magnification and Leica DFC320 Digital Camera system.

The size of the original images is 2088×1550 pixels. In our previous experiments [31] we used input images of size 512×512 pixels, where we added a black margin on the top and bottom of the image to make it square size. Since we decided to unify all experiments in this paper to use images of 256×256 pixels, to keep most of the relevant data in the image, we applied horizontal cropping to square size of 1550×1550 pixels. The idea was to keep the chironomid head-capsule in the center of the cropped image, so we used the following preprocessing pipeline (see Fig. 1): blur image using median filter, convert to grayscale, remove vignette effect, normalize, threshold, find connected components, find bounding box of the biggest component, calculate crop bounding box, and crop image. The final step in the dataset preparation was to resize all images to a dimension suitable to be used as CNN input, which in our case was 256×256 pixels. An illustration of the CHIRO10 dataset with the number of images and 6 random samples per species is shown in Fig. 2. Please notice that the number of images per species is around 200, except for *Polypedilum laetum* where we only have 79 images.

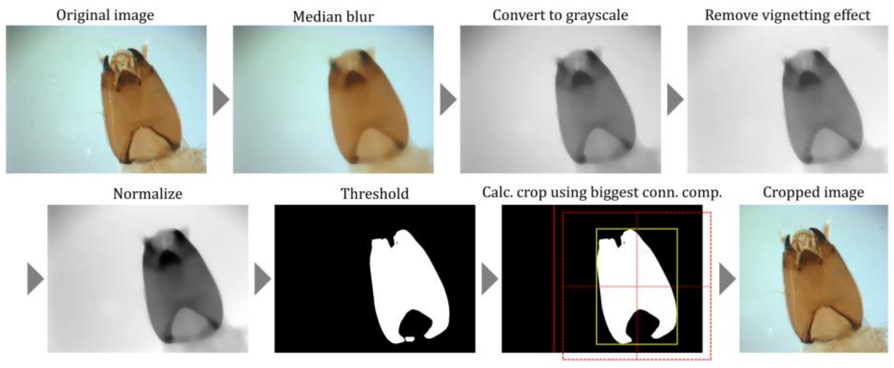

Fig. 1 Illustration of preprocessing steps applied to CHIRO10 images

The division of the dataset was made according to the method used in [30] (FIN-Benthic), i.e. 50% of the images are used for training, 20% for validation, while remaining 30% are used for testing and results reporting. To reduce ambiguity related to the selection of samples into one of these three subsets, 10 different partitions were made and explicitly saved as a tabular text file whose format resembles the one used in [30]. The partitions were made by rotating pattern (*train*, *test*, *train*, *test*, *train*, *val*, *train*, *test*, *train*, *val*) one element to the left for each new specimen. Table 1 contains a sample from the dataset definition file that illustrates the structure of the file and shows how 10 partitions are specified rotating the previous pattern. The pattern has 5 *train*, 2 *val*, and 3 *test* entries which roughly results in the desired split for each of the 10 partitions. Please notice that *im_num* parameter is always 1 since our dataset has one image per specimen.

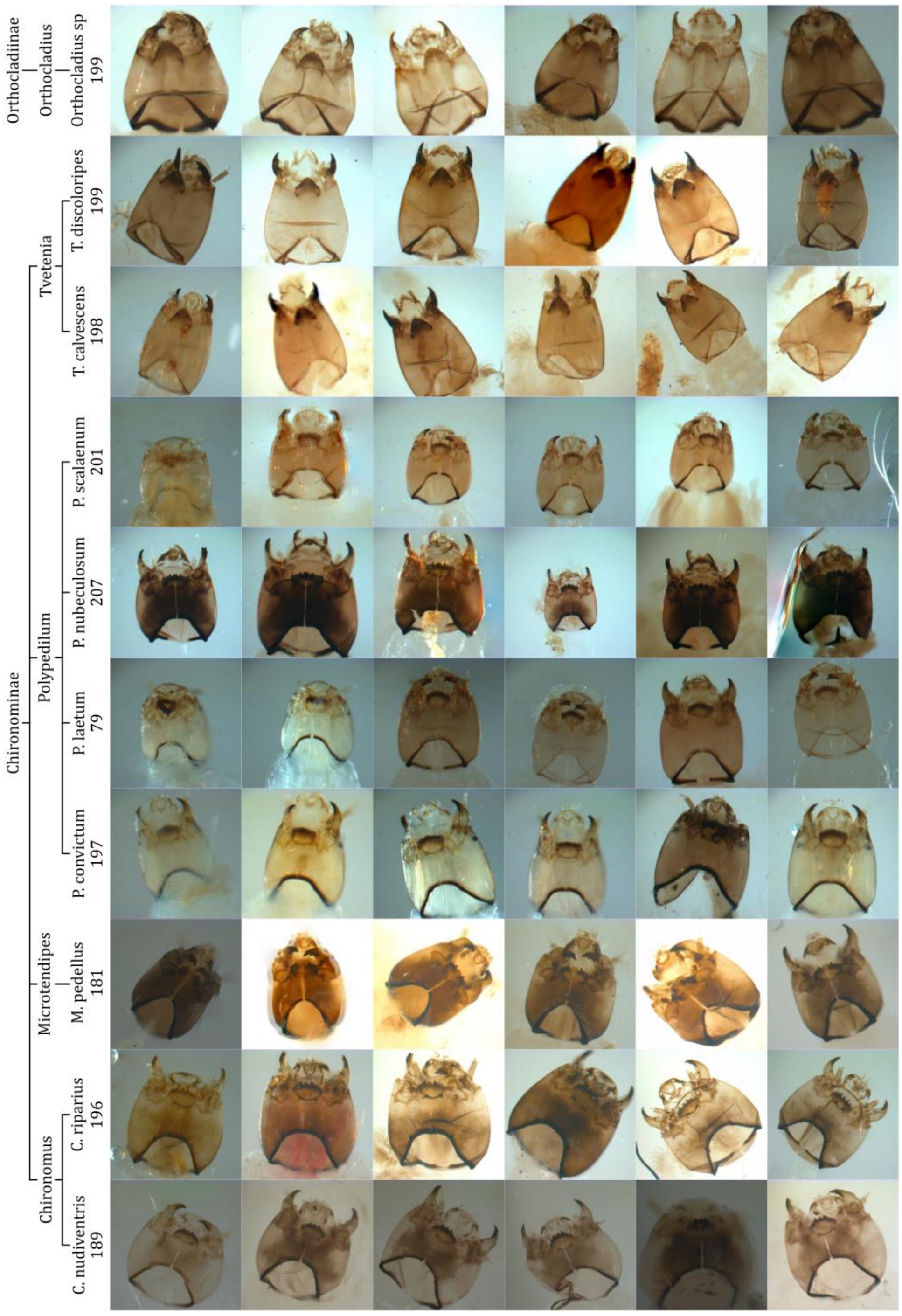

Fig. 2 Illustration of the CHIRO10 dataset

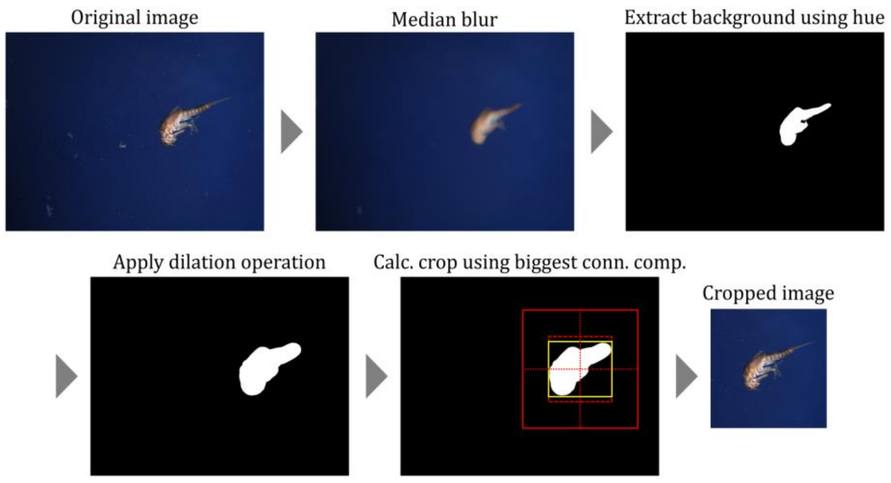
Original image
Median blur
Extract background using hue
Apply dilation operation
Calc. crop using biggest conn. comp.
Cropped image

Fig. 3 Illustration of preprocessing steps applied to STONEFLY9 and EPT29 images

Table 1 Sample from dataset definition file showing how 10 partitions are specified

10_data_partitions										im_num	im_path	cls_name	cls_id
train	test	train	test	train	val	train	test	train	val	1	Orthocladius_sp\\100.jpg	Orthocladius_sp	4
test	train	test	train	val	train	test	train	val	train	1	Orthocladius_sp\\101.jpg	Orthocladius_sp	4
train	test	train	val	train	test	train	val	train	test	1	Orthocladius_sp\\102.jpg	Orthocladius_sp	4
test	train	val	train	test	train	val	train	test	train	1	Orthocladius_sp\\103.jpg	Orthocladius_sp	4
train	val	train	test	train	val	train	test	train	test	1	Orthocladius_sp\\104.jpg	Orthocladius_sp	4
val	train	test	train	val	train	test	train	test	train	1	Orthocladius_sp\\105.jpg	Orthocladius_sp	4
train	test	train	val	train	test	train	test	train	val	1	Orthocladius_sp\\106.jpg	Orthocladius_sp	4
test	train	val	train	test	train	test	train	val	train	1	Orthocladius_sp\\107.jpg	Orthocladius_sp	4
train	val	train	test	train	test	train	val	train	test	1	Orthocladius_sp\\108.jpg	Orthocladius_sp	4
val	train	test	train	test	train	val	train	test	train	1	Orthocladius_sp\\109.jpg	Orthocladius_sp	4

Table 2 Details of the datasets used in this study

Dataset name	Subsets	Categories	Specimens	Images	Images per specimen	Images per category
	Set 1	10				79–207 (~186)
CHIRO10	Set 2	5	1846	1846	1	199–684 (~369)
	Set 3	2				199–1647 (~923)
	Set 1	64	7705	15074	1–2 (~2)	7–577 (~235)
FIN-Benthic	Set 2	29	6038	11832	1–2 (~2)	230–577 (~408)
	Set 3	9	1692	3272	1–2 (~2)	322–395 (~363)
STONEFLY9	–	9	774	3845	1–5 (~5)	119–532 (~427)
EPT29	–	28	1608	4794	1–4 (~3)	27–366 (~171)

By applying the described method, we created three dataset definition files: on species (Set 1), genus (Set 2), and subfamily (Set 3) levels. The only differences in these files are *cls_name* and *cls_id* values which reflect a different grouping of specimens.

In addition to the results obtained on CHIRO10 datasets, we also applied our method to previously mentioned, publicly available datasets: FIN-Benthic [30], STONEFLY9 [26], [34], and EPT29 [27]. Since the FIN-Benthic dataset had already prepared, i.e. cropped and resized images (256×256), so as 10 predefined partitions, there was no additional effort to apply our method to it. On the other hand, images from the other two datasets needed to be preprocessed first. Again we applied a preprocessing pipeline that used the following steps (see Fig. 3): blur image using median filter, convert color from RGB to HSV model, extract background (blue) using hue values range, invert mask and apply dilation morphological operation, find connected components, calculate bounding box of the biggest component, enlarge and fit bounding box to an image, and finally crop and resize image to 256×256 pixels. Both STONEFLY9 and EPT29 originally used 3-fold cross-validation, but to unify the method across all datasets we created 10 partitions following the same split rule of 50% for training, 20% for validation, and 30% for testing. To do so, we applied the same method previously used for the CHIRO10 dataset. One additional note about the downloaded EPT29 dataset is that it had only 28 categories, while the total number of images was 4799. During image preprocessing we discovered that 5 images did not have a subject in it, so we removed them from the resized set, ending with a total of 4794 images. Table 2 shows a summary of the datasets used in this study.

4. METHOD

Our method of image-based species identification relies on DL, i.e. on using deep residual CNN and end-to-end learning. To enable robustness when training a model with a limited number of sample images, we applied the following techniques when designing the classifier:

1. Transfer learning
2. Dropout
3. Data augmentation

Transfer learning is one of the key techniques that allows using of deep CNN to solve problems with a relatively small dataset. It relies on using a network pretrained on some large dataset such as ImageNet 2012. Training the network on the ImageNet dataset ensures that it will build a hierarchy of different features that can be found in photos in general and can be used to classify new photos. To enable such transfer, it is necessary to replace the top of the network, which is responsible for the classification, and train the new one using features extracted by the deep CNN (see Fig. 4). Typically, the part of a CNN responsible for feature extraction is called an encoder. Depending on the nature of the dataset, this approach may be sufficient. However, in our case, the input does not represent something that is typically found in the photographs that are used for the initial training, so it was necessary to carry out two-phase training. After training the classifier in the first phase, the whole network was fine-tuned in the second phase. Fine-tuning is nothing more than end-to-end training of the entire network. The term ‘fine’ is used to indicate that a very small learning rate is used to preserve initial network parameters.

Another technique that was used to improve classifier robustness is called dropout [35]. The appropriate layer was added after the encoder so that in each training step, a certain percentage (in our case 50%) of the extracted features is discarded. This is done only in the training phase, while during evaluation all outputs are considered, but scaling is performed for the appropriate dropout rate. The scaling is applied to adjust the average output value to the one we had during the training. The effect achieved by applying dropout is that the classifier learns to rely on many different features when ‘making the decision’. The dropout technique prevents overfitting and tends to provide better results on the validation set.

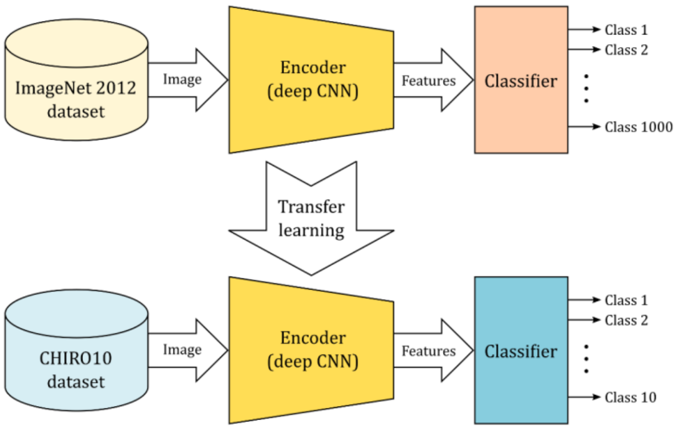

Fig. 4 Illustration of transfer learning

Training a neural network to classify images requires determining features that can be easily classified into a certain category. This process requires a large number of training samples to build a hierarchy of features and to produce a classifier resistant to the different variations that may appear in the images. When we have a relatively small amount of training data, and we use a high capacity model, the model can easily ‘remember’ all the training samples, and thus give a poor result on the validation set. A typical technique used to avoid such behavior is called data augmentation. Data augmentation involves applying random transformations over the input images so that in each training cycle the network is presented with something it has not ‘seen’ before. Depending on the nature of the dataset, typical transformations include flipping, rotation, translation, scaling, skewing, changing brightness and contrast, etc.

4.1. Network architecture

Following the previously described principles, we adopted the architecture based on the ResNet-50 [13] encoder that is depicted in Fig. 5. The choice of the ResNet-50 network, pretrained on ImageNet 2012 dataset, to be used as the encoder was influenced by several factors: good results on ImageNet 2012 dataset, network size in terms of the number of parameters, memory usage during training that allows bigger batches, reasonable training time, and model availability in the software environment used in the implementation. In general, ResNet architecture is very often a good initial choice because of its good ratio between representation capacity and training time.

ResNet-50 encoder has 23,564,800 parameters, while the output is 2048-dimensional feature vector. The outputs are obtained by applying global average pooling on the last feature map. For input images of 256×256 pixels, this map dimension is 8×8×2048. The good property of ResNets and global average pooling is that the size of the output feature vector does not depend on the size of the input image, which implies we can train the network with images of a certain size and later evaluate it using images of different size.

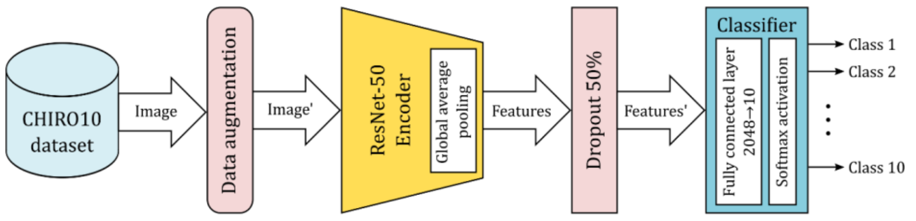

Fig. 5 Schematic representation of the proposed architecture based on ResNet-50 encoder

The resulting 2048-dimensional feature vector is subject to a 50% dropout. That means, in the training phase, half of it, i.e. 1024 randomly selected elements are set to zero. The feature vector modified this way is fed to a fully connected layer of *N* neurons where each output corresponds to a single class (*N*=10 for CHIRO10). The layer has *N**(2048+1) parameters that are trained in the first phase. A softmax activation function (1) is applied to the outputs, giving this layer the role of a classifier of features that are provided by the ResNet-50 encoder. The outputs of the softmax function represent the probabilities that a sample belongs to one of the classes. In other words, the individual outputs have values from the interval (0, 1) and the sum of all outputs is equal to 1.

$$S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_i}}. \tag{1}$$

4.2. Implementation and training

To implement the proposed architecture, we relied on the Python programming language and the Keras library. Keras is a high-level library that defines a simplified interface for implementing deep neural networks, and in our case, it relies on the TensorFlow library as a backend engine.

Within the *applications* module, Keras has several deep CNN architectures pretrained on the ImageNet 2012 dataset. Instantiating *ResNet50V2* class with appropriate parameters provided the encoder for our model. The rest of the model included one *Dropout* layer and one *Dense* layer with *softmax* activation. Since in the first phase of the training weights of the ResNet-50 encoder should remain frozen, we needed to set the *trainable* attribute to *False* for all convolution layers in the encoder.

The created model was compiled using the *RMSprop* [36] optimization algorithm (*optimizers* module), *sparse_categorical_crossentropy* loss (*losses* module), while *sparse_categorical_accuracy* (*metrics* module) was used as the metric for accuracy. This optimization algorithm was chosen because of its fast convergence and relatively low memory footprint, while the loss and the corresponding accuracy functions are the standard choices for classification problems.

For data augmentation, we relied on Keras built-in *ImageDataGenerator* class, which is located in the *preprocessing* module, submodule *image*. When constructing a corresponding image generator, we specified parameters for the various transformations that will be randomly applied. Data augmentation included horizontal and vertical image flipping, rotation up to ±90°, translation up to ±15% in both directions, shear up to ±10%, and scaling up to ±20% (see Fig. 6). Data augmentation is only applied to the training set, while for validation purposes we used unmodified images.

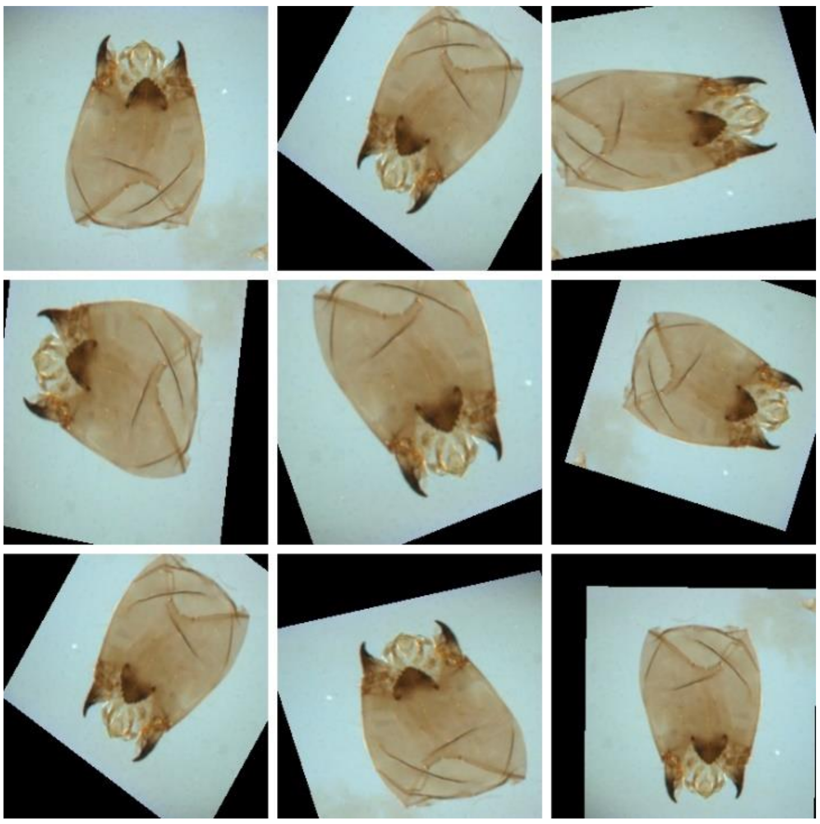

Fig. 6 Illustration of the used data augmentation

Since we used an image generator for data augmentation, training was done using the *fit_generator* method. Additional control of the training process in Keras is possible using *callback* objects. The corresponding classes are located in the *callbacks* module and we used the following ones: *LearningRateScheduler*, *EarlyStopping*, *ModelCheckpoint*, and *CSVLogger*.

LearningRateScheduler is used to specify an arbitrary function for calculating the learning rate depending on the current training epoch. We used this callback to implement the so-called cosine annealing [37]. In cosine annealing, the learning rate decreases following cosine function from some initial value to some minimum value during a certain number of epochs (period). In our implementation, with each new period, the initial learning rate was decreased by factor of 0.7. We used the initial value of 10^{-3} for the first phase of the training and 10^{-5} for the fine-tuning. The minimum value was 0.01 times the initial value. The appropriate schedule is shown in Fig. 7.

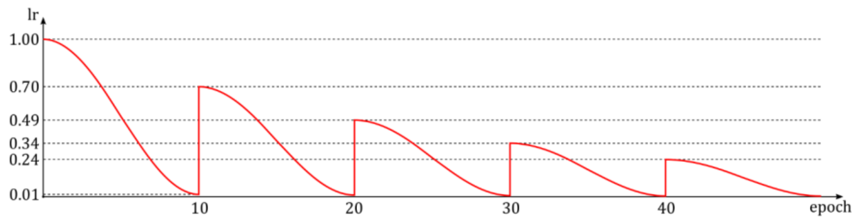

Fig. 7 Cosine annealing schedule used for controlling the learning rate during the training process

EarlyStopping callback, as the name suggests, is used to stop the training process if there is no progress on a given parameter in some number of epochs. In our case, 30 epochs were used and accuracy on the validation set was monitored. *CSVLogger* callback is used to record loss and accuracy on the training and validation sets during the training process. Finally, *ModelCheckpoint* was used to save the current best model in terms of accuracy achieved on the validation set.

The fine-tuning was done in an almost identical way, with a few minor modifications. After loading the model obtained from the first training phase, training was re-enabled for all the layers of the ResNet-50 encoder (set the *trainable* attribute to *True*). The second change affected the initial learning rate that was reduced to 10^{-5} to avoid significant weight changes in the encoder. The results we obtained are presented in the next section.

5. Experiments

For the evaluation of the proposed method, for each dataset, we trained 10 models, one for each partition. The results are obtained by evaluating each of the trained models on the corresponding test set that contains 30% of the specimens from the dataset. We would like to emphasize that images from the test set are not used in the training process, so the obtained results should fairly represent the model's expected performance.

Since all the datasets, except CHIRO10, contain more than one image per specimen, as a basic metric for comparison we calculated specimen-level accuracy. To compare how different methods of acquiring multiple images per specimen influence the results, we also calculated image-based accuracy. Specimen-level accuracy is calculated by averaging the predictions, i.e. probability distributions, obtained for each of the specimen images, and then assigning the most probable class. For example, if we have 3 classes and 2 images for the specimen, and if the model predicts (**0.6**, 0.1, 0.3) and (0.1, 0.4, **0.5**), the average would be (0.35, 0.25, **0.4**), so the specimen would be assigned to the third class.

Table 3 presents a summary of the evaluation done on test sets where the specimen-level accuracy is shown, for all the datasets and partitions, including mean and standard deviation values. In Fig. 8–12 corresponding normalized confusion matrices for CHIRO10 Set 1, FIN-Benthic Set 3 and Set 2, STONEFLY9, and EPT29 datasets are depicted. The matrices show mean specimen classification accuracies per class, so as standard deviation. Finally, Table 4 shows a comparison of the obtained results with the results reported in the original contributions (FIN-Benthic [30], STONEFLY9 [26], EPT29 [27]), and with the results obtained when image-level accuracy is calculated.

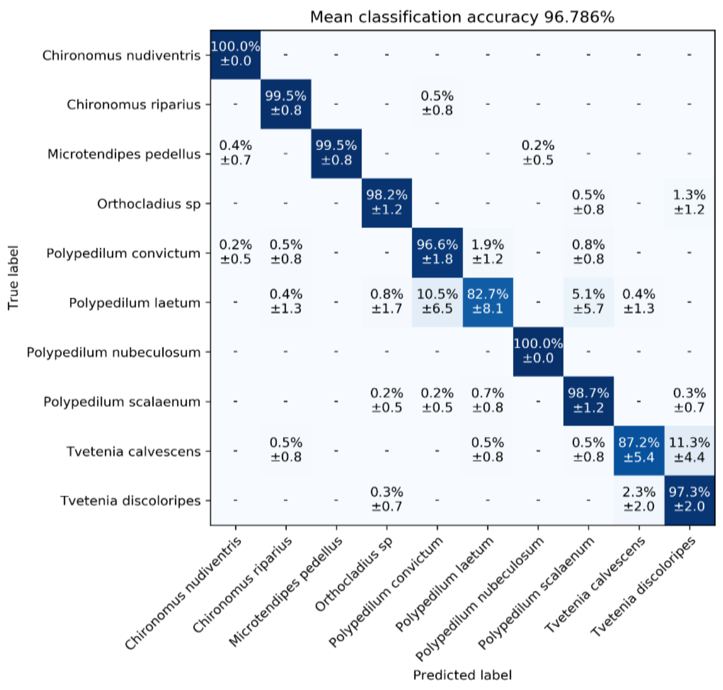

Fig. 8 Confusion matrix for CHIRO10 Set 1

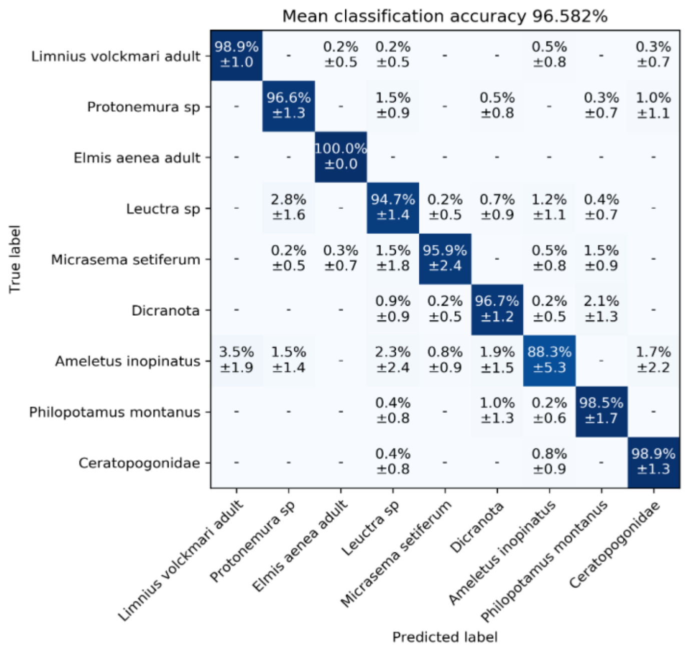

Fig. 9 Confusion matrix for FIN-Benthic Set 3

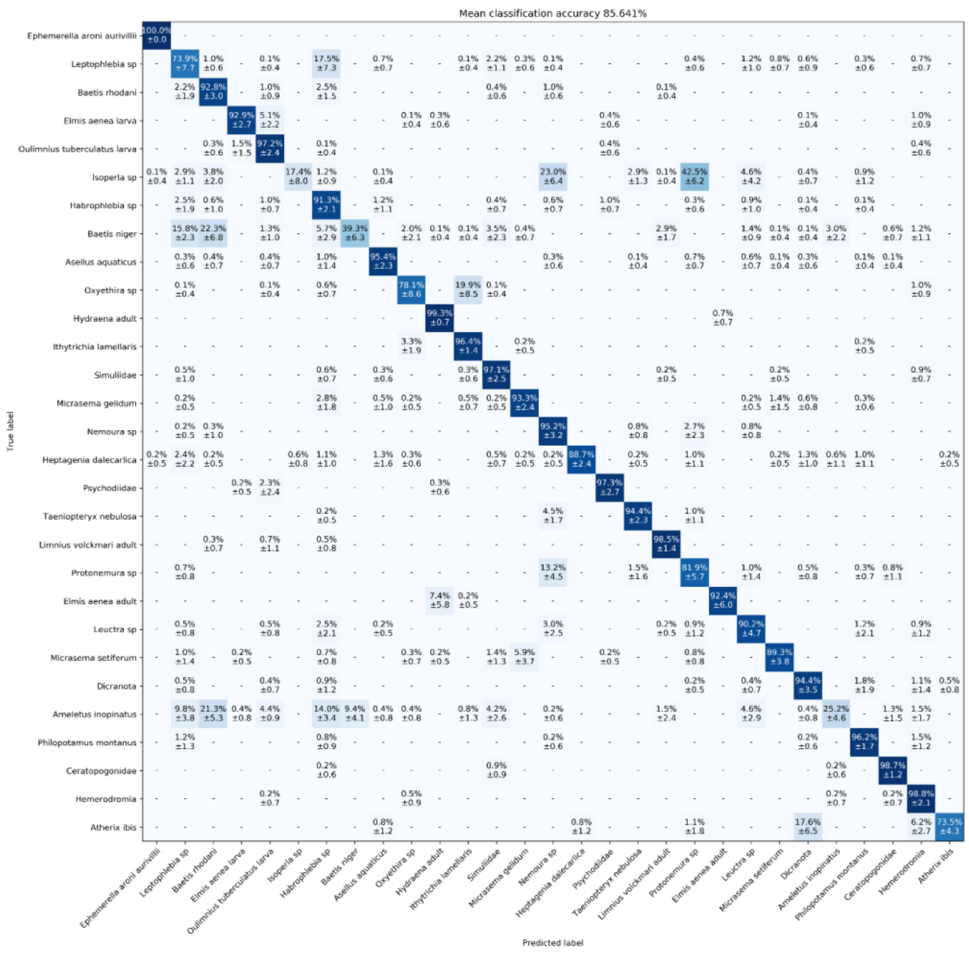

Fig. 10 Confusion matrix for FIN-Benthic Set 2

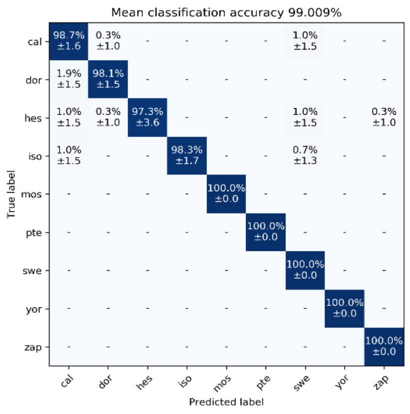

Fig. 11 Confusion matrix for the STONEFLY9 dataset

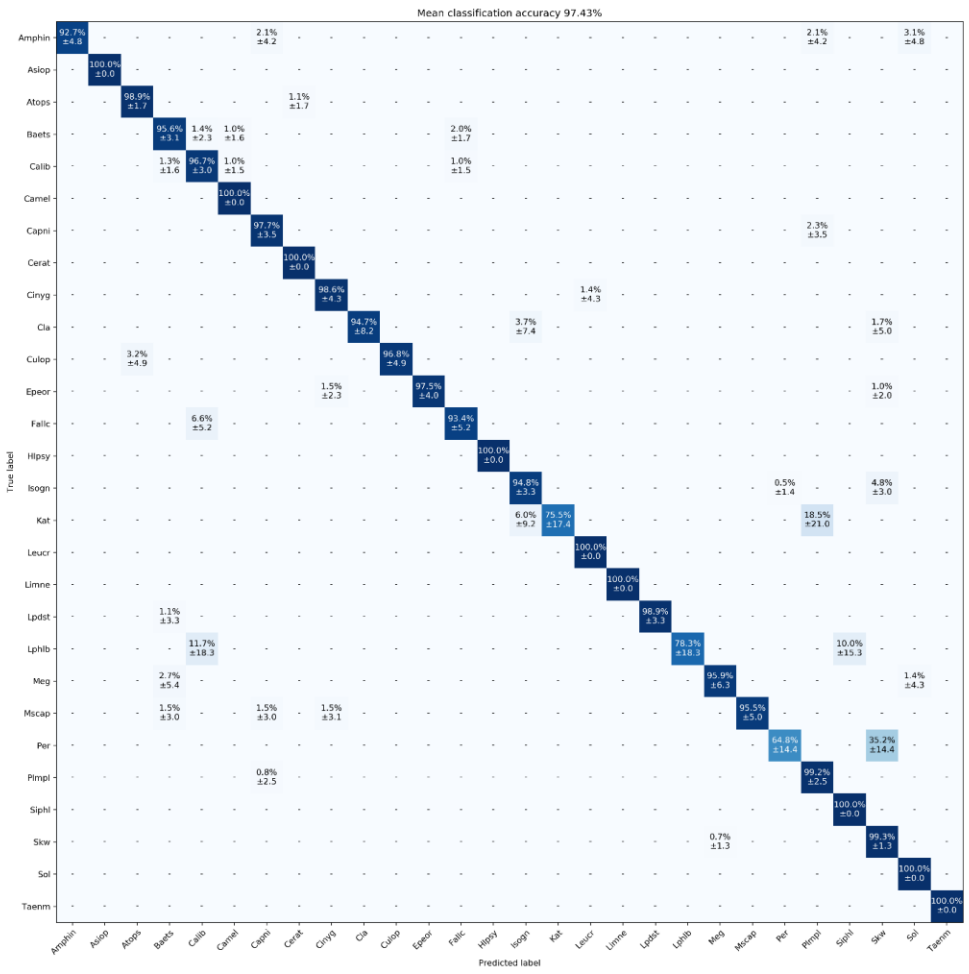

Fig. 12 Confusion matrix for the EPT29 dataset

5.1. Analysis and discussion

The proposed method showed significant improvement in species identification accuracy on all external datasets that range between 4.55% and 9.56%. The only worse results were on CHIRO10 datasets compared to our results presented in [31]. The reason for that is a different evaluation strategy that involved much fewer images for training (50%) than in the previous paper (80%). The evaluation based on an independent test set, instead of the validation set, also has a negative impact on the accuracy. If we look at the confusion matrix shown in Fig. 8, the *Polypedilum laetum* class has the lowest accuracy of 82.7±8.1%. The reason for that is that all other classes have ~200 images, while this one has only 79. There are just not enough images for proper training. *Tvetenia calvescens* took penultimate place with an accuracy of 87.2±5.4%, but this time most of the misclassified specimens (11.3±4.4%) went to the *Tvetenia discoloripes* class which is very similar (the same genus). To determine the impact of the smaller image collection that is used compared to our previous paper, we conducted two additional experiments where we trained 10 models per dataset joining train and test images, while results were evaluated on the remaining 20% of validation images. In the second experiment, we used an image size of 384×384 pixels that corresponds to the level of details we previously had on 512×512 images without cropping. Then we conducted the third experiment using images of 384×384 pixels with standard 50–20–30% split. The detailed results of these experiments are shown in Table 5. The mean accuracies for the second experiment (80–20% split, 384×384 pixels) are 99.13% for Set 1, 99.84% for Set 2, and 99.86% for Set 3, which closely resembles the previously reported results. The third experiment recorded an increase in accuracy for 0.88% on Set 1, 0.20% on Set 2 and 0.25% on Set 3 compared to the results presented in Table 3. If we compare per-class accuracies for *Polypedilum laetum* and *Tvetenia calvescens*, we can observe that in the first case it remains nearly the same (82.6±7.8), while in the second case it is significantly improved (93.7±4.7%). This result was expected, since the number of images for *Polypedilum laetum* stayed the same, while the increased level of details (384×384 image size) helped better distinguish similar species.

Table 3 Specimen classification accuracy

Dataset name	CHIRO10			FIN-Benthic			STONEFLY9	EPT29
Subset / Part.	Set 1	Set 2	Set 3	Set 1	Set 2	Set 3	–	–
1	97.11	98.56	99.10	80.73	86.68	97.66	98.71	97.72
2	95.67	98.92	99.46	80.44	85.53	96.88	99.57	97.93
3	96.75	99.46	99.82	80.31	85.14	95.70	97.84	97.51
4	98.01	99.64	99.46	81.42	85.64	96.68	99.57	96.68
5	96.21	98.74	99.46	80.01	87.01	96.29	98.71	97.72
6	97.11	99.46	98.56	82.02	85.31	95.70	99.57	97.10
7	97.11	98.92	99.64	80.99	85.53	97.46	98.28	96.89
8	95.85	99.28	99.64	81.33	85.53	96.48	99.14	96.89
9	96.93	99.46	99.10	82.61	84.59	95.90	99.14	97.93
10	97.11	99.28	99.10	80.14	85.47	97.07	99.57	97.93
Mean	96.79	99.17	99.33	81.00	85.64	96.58	99.01	97.43
Std	0.70	0.36	0.37	0.85	0.70	0.70	0.61	0.49

Table 4 Comparison of the obtained results with original contributions

Dataset name	FIN-Benthic			STONEFLY9	EPT29
Subset	Set 1	Set 2	Set 3	–	–
Original contributions	75.74	81.04	90.14	94.50	88.06
Our results	81.00	85.64	96.58	99.01	97.43
Increase	+5.29	+4.60	+6.44	+4.55	+9.56
Our results per image	76.59	81.19	93.63	97.69	95.37
Increase per specimen	+4.41	+4.45	+2.95	+1.32	+2.06

Table 5 Comparison of classification accuracies on CHIRO10 datasets

Configuration	CHIRO10 80–20%, 256×256			CHIRO10 80–20%, 384×384			CHIRO10 50–20–30%, 384×384		
Subset / Part.	Set 1	Set 2	Set 3	Set 1	Set 2	Set 3	Set 1	Set 2	Set 3
1	97.83	100	100	98.92	100	100	97.65	99.64	99.64
2	98.64	100	100	99.19	100	100	96.39	99.46	99.46
3	98.65	99.73	99.73	99.73	99.46	100	96.75	99.64	99.82
4	97.30	99.73	99.73	98.65	99.46	99.73	98.73	99.46	99.82
5	97.56	99.73	100	98.92	100	100	98.74	98.74	99.46
6	98.37	99.46	100	99.19	100	99.73	97.65	99.64	99.28
7	98.37	99.73	99.46	99.46	100	99.73	97.47	99.46	100
8	98.10	99.73	100	98.92	100	100	97.47	99.10	99.64
9	97.83	98.92	100	99.19	99.73	100	98.38	99.10	99.10
10	98.37	99.46	99.73	99.19	99.73	99.46	97.47	99.46	99.64
Mean	98.10	99.65	99.86	99.13	99.84	99.86	97.67	99.37	99.58
Std	0.46	0.31	0.19	0.31	0.23	0.19	0.77	0.30	0.27

When it comes to the other datasets used in this research, the same rule regarding the number of images per class applies. For example, if we look at Fig. 12 where the confusion matrix for the EPT29 dataset is shown, three lowest accuracies correspond to the classes with the least image count: *Per* 64.8% – 17 specimens / 51 images, *Kat* 75.5% – 16 / 48, and *Lphlb* 78.3% – 17 / 27.

The impact of specimen-level classification, compared to image-level classification, showed an increase of accuracy between 1.32% and 4.45%. It appears that FIN-Benthic's method of taking two images per specimen from two different directions has a better impact on specimen-level accuracy. When compared to corresponding STONEFLY9 and EPT29 datasets, that use 5 and 3 images per specimen but taken from the same direction, FIN-Benthic Set 3 and Set 2 show more than two times bigger increase of specimen-level accuracy.

Finally, the quality, i.e. level of details present in the images also plays an important role. The FIN-Benthic Set 2 has 29 classes, 6038 specimens, and 11832 images. On the other hand, the EPT29 dataset, has 28 classes, 1608 specimens, and 4794 images. These two datasets are comparable by the number of classes, while the number of specimens and images goes in favor of FIN-Benthic. The method of taking two images per specimen using two cameras also favors FIN-Benthic. Nevertheless, the accuracy for FIN-Benthic Set 2 is 85.64%, while on EPT29 we achieved staggering 97.43%. The reason for this, in our opinion, is a much lower level of details present in the FIN-Benthic images compared to the other sets. Interestingly, this EPT29 result also shows that more classes do not necessarily mean much lower accuracy.

6. CONCLUSIONS

In this paper, we proposed and evaluated an approach for species identification for aquatic biomonitoring. The proposed method relies on three DL techniques used to improve robustness when training is done on a relatively small dataset: transfer learning, data augmentation, and feature dropout. We applied transfer learning by using ResNet-50 deep CNN pretrained on ImageNet 2012 dataset. For the evaluation, we used our CHIRO10 dataset, along with several available public datasets (FIN-Benthic, STONEFLY9, and EPT29).

To be able to compare results for different datasets, we tried to unify the training process by using images of 256×256 pixels, splitting the data the same way, and using 10 partitions to effectively measure the mean and standard deviation. The results showed significant improvement compared to the original contributions, confirmed that there is a considerable gain when multiple images per specimen are used, showed that image quality plays an important role in the overall accuracy, and also showed that the number of samples per class must be carefully selected.

Future research might go in several directions. Trying to determine what is the optimal number of specimens that each class should have in order to build a classifier that outperforms human capability. The second direction may concern demining the best pretrained encoder network suitable to the problem, and possibly using some sophisticated method instead of a brute force. Finally, the third direction would be to examine how an ensemble of different models would perform on these datasets.

Acknowledgement: *This study was supported by the Ministry of Education, Science and Technological Development, Republic of Serbia, as part of the projects III-43007 and III-47003.*

REFERENCES

[1] IPBES, "Summary for policymakers of the global assessment report on biodiversity and ecosystem services of the Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services. S. Díaz, J. Settele, E. S. Brondizio E.S., H. T. Ngo, M. Guèze, J. Aga," Bonn, 2019.

[2] P. F. M. Verdonschot, "Evaluation of the use of Water Framework Directive typology descriptors, reference sites and spatial scale in macroinvertebrate stream typology," *Hydrobiologia*, vol. 566, no. 1, pp. 39–58, Aug. 2006.

[3] G. W. Hopkins and R. P. Freckleton, "Declines in the numbers of amateur and professional taxonomists: Implications for conservation," *Anim. Conserv.*, vol. 5, no. 3, pp. 245–249, Aug. 2002.

[4] F. C. Jones, "Taxonomic sufficiency: The influence of taxonomic resolution on freshwater bioassessments using benthic macroinvertebrates," *Environ. Rev.*, vol. 16, no. NA, pp. 45–69, Dec. 2008.

[5] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual Categorization with Bags of Keypoints," in *Workshop on statistical learning in computer vision, ECCV*, 2004.

[6] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural Features for Image Classification," *IEEE Trans. Syst. Man. Cybern.*, vol. SMC-3, no. 6, pp. 610–621, Nov. 1973.

[7] P. J. D. Weeks, M. A. O'Neill, K. J. Gaston, and I. D. Gauld, "Species-identification of wasps using principal component associative memories," *Image Vis. Comput.*, vol. 17, no. 12, pp. 861–866, 1999.

[8] J. A. Jose and C. S. Kumar, "Genus and Species-Level Classification of Wrasse Fishes Using Multidomain Features and Extreme Learning Machine Classifier," *Int. J. Pattern Recognit. Artif. Intell.*, Mar. 2020.

[9] I. Kanellopoulos and G. G. Wilkinson, "Strategies and best practice for neural network image classification," *Int. J. Remote Sens.*, vol. 18, no. 4, pp. 711–725, Mar. 1997.

[10] Y. LeCun *et al.*, "Handwritten digit recognition with a back-propagation network," *papers.nips.cc*, pp. 396–404, 1990.

[11] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series.MIT Press, Cambridge," *Handb. brain theory neural networks*, vol. 3361, no. 10, 1995.

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, vol. 2, pp. 1097–1105.

[13] O. Russakovsky *et al.*, "ImageNet Large Scale Visual Recognition Challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[14] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Lecture Notes in*

Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2014, vol. 8689 LNCS, no. PART 1, pp. 818–833.

[15] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 2015.

[16] C. Szegedy *et al.*, “Going deeper with convolutions,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, vol. 07-12-June, pp. 1–9.

[17] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, vol. 2016-Decem, pp. 770–778.

[18] L. Shao, F. Zhu, and X. Li, “Transfer learning for visual categorization: A survey,” *IEEE Trans. Neural Networks Learn. Syst.*, vol. 26, no. 5, pp. 1019–1034, 2015.

[19] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?,” in *Advances in Neural Information Processing Systems*, 2014, vol. 4, no. January, pp. 3320–3328.

[20] C. Affonso, A. L. D. Rossi, F. H. A. Vieira, and A. C. P. de L. F. de Carvalho, “Deep learning for biological image classification,” *Expert Syst. Appl.*, vol. 85, pp. 114–122, Nov. 2017.

[21] H. Hermessi, O. Mourali, and E. Zagrouba, “Deep feature learning for soft tissue sarcoma classification in MR images via transfer learning,” *Expert Syst. Appl.*, vol. 120, pp. 116–127, Apr. 2019.

[22] F. F. Ting, Y. J. Tan, and K. S. Sim, “Convolutional neural network improvement for breast cancer classification,” *Expert Syst. Appl.*, vol. 120, pp. 103–115, Apr. 2019.

[23] M. Jiang, L. Cheng, F. Qin, L. Du, and M. Zhang, “White Blood Cells Classification with Deep Convolutional Neural Networks,” *Int. J. Pattern Recognit. Artif. Intell.*, vol. 32, no. 9, Sep. 2018.

[24] S. Liu, F. Y. Shih, G. Russell, K. Russell, and H. Phan, “Classification of Ecological Data by Deep Learning,” *Int. J. Pattern Recognit. Artif. Intell.*, May 2020.

[25] J. Wäldchen and P. Mäder, “Machine learning for image based species identification,” *Methods Ecol. Evol.*, vol. 9, no. 11, pp. 2216–2225, Nov. 2018.

[26] D. A. Lytle *et al.*, “Automated processing and identification of benthic invertebrate samples,” *J. North Am. Benthol. Soc.*, vol. 29, no. 3, pp. 867–874, 2010.

[27] N. Larios *et al.*, “Stacked spatial-pyramid kernel: An object-class recognition method to combine scores from random trees,” in *2011 IEEE Workshop on Applications of Computer Vision, WACV 2011*, 2011, pp. 329–335.

[28] S. Kiranyaz *et al.*, “Classification and retrieval on macroinvertebrate image databases,” *Comput. Biol. Med.*, vol. 41, no. 7, pp. 463–472, 2011.

[29] H. Joutsijoki *et al.*, “Evaluating the performance of artificial neural networks for the classification of freshwater benthic macroinvertebrates,” *Ecol. Inform.*, vol. 20, pp. 1–12, Mar. 2014.

[30] J. Raitoharju *et al.*, “Benchmark database for fine-grained image classification of benthic macroinvertebrates,” *Image Vis. Comput.*, vol. 78, pp. 73–83, Oct. 2018.

[31] D. Milošević *et al.*, “Application of deep learning in aquatic bioassessment: Towards automated identification of non-biting midges,” *Sci. Total Environ.*, vol. 711, p. 135160, Apr. 2020.

[32] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[33] N. Larios *et al.*, “Automated insect identification through concatenated histograms of local appearance features: Feature vector generation and region detection for deformable objects,” *Mach. Vis. Appl.*, vol. 19, no. 2, pp. 105–123, Mar. 2008.

[34] G. Martinez-Munoz *et al.*, “Dictionary-free categorization of very similar objects via stacked evidence trees,” 2010, pp. 549–556.

[35] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, “Improving neural networks by preventing co-adaptation of feature detectors,” *arxiv.org*, 2012.

[36] G. Hinton, N. Srivastava, and K. Swersky, “Neural Networks for Machine Learning, Lecture 6a Overview of mini-batch gradient descent,” 2012. [Online]. Available: http://www.cs.toronto.edu/~hinton/coursera/lecture6/lec6.pdf. [Accessed: 24-Jan-2020].

[37] J. Jordan, “Setting the learning rate of your neural network,” 2018. [Online]. Available: https://www.jeremyjordan.me/nn-learning-rate/. [Accessed: 24-Jan-2020].