CLASSIFICATION AND RETRIEVAL ON MACROINVERTABRATE IMAGE DATABASES USING EVOLUTIONARY RBF NEURAL NETWORKS

Serkan Kiranyaz*, Moncef Gabbouj*[1], Jenni Pulkkinen* , Turker Ince** and Kristian Meissner***

*Tampere University of Technology, Tampere, Finland
{serkan.kiranyaz, moncef.gabbouj, jenni.pulkkinen}@tut.fi
**Izmir University of Economics, Izmir, Turkey, turker.ince@ieu.edu.tr
***Finnish Environment Institute, Finland, kristian.Meissner@ymparisto.fi

ABSTRACT

Aquatic ecosystems are facing a growing number of human induced changes and threats. Macroinvertebrate biomonitoring is particularly efficient in pinpointing the cause-effect structure between slow and subtle changes and their detrimental consequences in aquatic ecosystems. The greatest obstacle to implementing efficient biomonitoring is currently the cost-intensity of human expert taxonomic identification of samples. While there is evidence that automated recognition techniques can match human taxa identification accuracy at greatly reduced costs, so far the development of automated identification techniques for aquatic organisms has been minimal. In this paper, we focus on advancing classification and data retrieval that are instrumental when processing large macroinvertebrate image datasets. To accomplish this for routine biomonitoring we propose an automated and highly accurate river macroinvertebrate classifier using evolutionary RBF networks. The best classifier, which is trained over a dataset of river macroinvertebrate specimens, is then used in the MUVIS framework for the efficient search and retrieval of particular macroinvertebrate peculiars. Classification and retrieval results present such a delicate accuracy that can match experts' ability for taxonomic identification.

Keywords: Biomonitoring, classification, evolutionary radial basis function networks, Benthic macroinvertebrate.

1. INTRODUCTION

Aquatic ecosystems are facing a growing number of anthropogenic pressures operating at several time and spatial scales (e.g. eutrophication, global warming). Well planned biomonitoring is essential to detect the cause-effect structure between the often subtle pressures and their ecosystem consequences. The resulting growing global need to implement more biomonitoring is apparent but due to the cost-intensity of human expert taxonomic identification of samples, this need cannot currently be met. Automatic and semi-automatic signal and image processing techniques have been successfully applied in similar fields of application to solve such challenges. For instance, automatic image recognition techniques of aquatic phytoplankton have been shown to match human taxa identification accuracy at greatly reduced costs [1]. Despite their obvious potential, the development of automated taxa identification techniques has long been hampered by the reluctance of taxonomic experts to embrace alternative methods of taxa identification. A detailed review on advances in automated taxa identification [2] deemed misconceptions, the lack of vision and the lack of enterprise, more limiting to the development of automated taxa identification than actual practical constraints. Research on automated recognition of aquatic organisms has mainly concentrated on plankton [3], while automated classification and particularly retrieval of freshwater macroinvertebrates has received very little attention [4]. In a recent work [5] on a set of river macroinvertebrates, a mean correct classification of 88.17% and 75.31% have been achieved in training and test sets, which matches the levels of human accuracy for other aquatic taxonomic groups [6].

An earlier work on classification of aquatic organisms has shown that neural networks usually outperform decision trees [7] and other classical statistical techniques [8]. Artificial neural networks (ANNs) have proven to perform complex classification tasks, provided that a proper structure for the network is selected and a suitable training technique is applied to a sufficiently representative set of data. Several researchers have attempted to design ANNs automatically with respect to a particular problem. The earlier attempts fall into two broad categories: constructive and pruning algorithms [9], [10], [11], [12], from which many deficiencies and limitations have been reported [13]. The efforts have then been focused on evolutionary algorithms (EAs) [14] particularly over Genetic Algorithm (GA) [15] and Evolutionary Programming (EP) [16], for both training and evolving ANNs. Most GA based methods have also been found quite inadequate for evolving ANNs mainly due to two major problems: permutation problem and noisy fitness evaluation [17]. Conceptually speaking, Particle Swarm Optimization (PSO), [18], has similar ties with the EA family. Only few researchers have investigated the use of PSO for evolutionary design of ANNs or to be precise, the radial basis function (RBF) networks.

In an earlier work, [19], Multi-Dimensional Particle Swarm Optimization (MD PSO), which re-forms the native structure of swarm particles in such a way that they can make inter-dimensional passes with a dedicated dimensional PSO process, has been introduced. MD PSO eventually negates the necessity of setting a fixed dimension *a priori*, which is a common drawback for the family of swarm optimizers. In this paper, we shall use MD PSO for evolving RBF network, which

[1] This work was supported by the Academy of Finland, project No. 213462 (Finnish Centre of Excellence Program (2006 - 2011)

is then used in the core of the proposed classification system. To achieve a computational cost efficiency, we shall use the same limited dataset as in [5] and apply the most basic and primitive geometrical features as detailed in [20]. In the feature space, MD PSO with fractional global-best formation (FGBF) technique [19] determines the optimal number of *Gaussian* neurons as well as their parameters (centroids and variances) by applying dynamic clustering over the training set. The training of the RBF classifier is then finalized by applying *SuperSAB* enhancement [21] of the Back-propagation (BP) algorithm, only to compute weights and biases (thetas). Since MD PSO is stochastic in nature, in order to maximize the classification accuracy, we evolve several RBF networks and choose the best one, which is then used for (dis-) similarity distance computation within the similarity-based queries of the MUVIS framework [22]. In this way the retrieval performance can be significantly improved, particularly as compared to the traditional query methods.

2. RELATED WORK

2.1 RBF Neural Networks

A popular type of feedforward ANN is the RBF network [23], which has always two layers in addition to the passive input layer: a hidden layer of RBF units and a linear output layer. Only the output layer has connection weights and biases. The activation function of the k^{th} RBF unit is defined as,

$$y_k = \varphi\left(\frac{\|X - \mu_k\|}{\sigma_k^2}\right), \qquad (1)$$

where φ is a radial basis function or, in other words, a strictly positive radially symmetric function, which has a unique maximum at *N*-dimensional center μ_k and whose value drops rapidly close to zero away from the center. σ_k is the width of the peak around the center μ_k. The activation function gets noteworthy values only when the distance between the *N*-dimensional input X and the center μ_k, $\|X - \mu_k\|$, is smaller than the width σ_k. The most commonly used activation function in RBF networks is the *Gaussian* basis function defined as,

where μ_k and σ_k are the mean and standard deviation, respectively, and ||.|| denotes the *Euclidean* norm. More detailed information about RBF networks can be obtained from [23] and [24].

2.2 Dynamic Clustering by MD PSO with FGBF

Instead of operating at a fixed dimension *N*, the MD PSO algorithm is designed to seek both positional and dimensional optima within a dimension range, ($D_{\min} \le N \le D_{\max}$). In order to accomplish this, each particle has two sets of components, each of which has been subjected to two independent and consecutive processes. The first one is a regular positional PSO, i.e. the traditional velocity updates and following positional moves in *N* dimensional search (solution) space. The second one is a dimensional PSO, which allows the particle to navigate through dimensions. Accordingly, each particle keeps track of its last position, velocity and personal best position (*pbest*) in a particular dimension so that when it re-visits the same dimension at a later time, it can perform its regular "positional" fly using this information. The dimensional PSO process of each particle may then move the particle to another dimension where it will remember its positional status and keep "flying" within the positional PSO process in this dimension, and so on. The swarm, on the other hand, keeps track of the *gbest* particles in all dimensions, each of which respectively indicates the best (global) position so far achieved and can thus be used in the regular velocity update equation for that dimension. Similarly the dimensional PSO process of each particle uses its personal best dimension in which the personal best fitness score has so far been achieved. Finally, the swarm keeps track of the global best dimension, *dbest*, among all the personal best dimensions. The *gbest* particle in *dbest* dimension represents the optimum solution (and the optimum dimension).

In a MD PSO process and at time (iteration) *t*, each particle *a* in the swarm, $\xi = \{x_1,..,x_a,..,x_S\}$, is represented by the following characteristics:

$xx_{a,j}^{xd_a(t)}(t)$: j^{th} component (dimension) of the position of particle *a*, in dimension $xd_a(t)$

$vx_{a,j}^{xd_a(t)}(t)$: j^{th} component (dimension) of the velocity of particle *a*, in dimension $xd_a(t)$

$xy_{a,j}^{xd_a(t)}(t)$: j^{th} component (dimension) of the personal best (*pbest*) position of particle *a*, in dimension $xd_a(t)$

gbest(d) : Global best particle index in dimension *d*

$x\hat{y}_j^d(t)$: j^{th} component (dimension) of the global best position of swarm, in dimension *d*

$xd_a(t)$**:** Dimension component of particle *a*

$vd_a(t)$**:** Velocity component of dimension of particle *a*

$xd_a(t)$**:** Personal best dimension component of particle *a*

Figure 1 shows sample MD PSO and *bPSO* particles with index *a*. The *bPSO* particle that is at a (fixed) dimension, *N*=5, contains only positional components whereas MD PSO particle contains both positional and dimensional components respectively. In the figure the dimension range for the MD PSO is given between 2 and 9; therefore the particle contains 8 sets of positional components (one for each dimension). In this example, the current dimension where the particle *a* resides is 2 ($xd_a(t) = 2$) whereas its personal best dimension is 3 ($\tilde{xd}_a(t) = 3$). Therefore, at time *t*, a positional PSO update is first

$$y_k = \exp\left(-\frac{\|X - \mu_k\|^2}{2\sigma_k^2}\right), \qquad (2)$$

performed over the positional elements, $xx_a^2(t)$ and then the particle may move to another dimension by dimensional PSO.

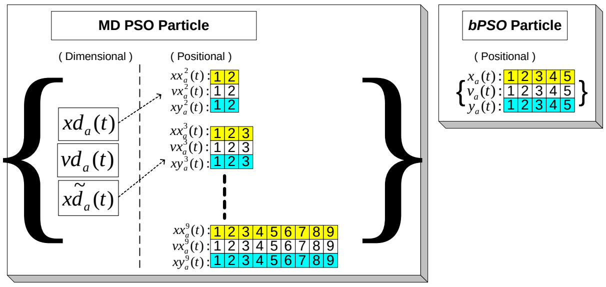

Figure 1: Sample MD PSO (right) vs. *bPSO* (left) particle structures. For MD PSO $[D_{\min} \underset{\sim}{=} 2,\ D_{\max} = 9]$ **and at the current time *t*,** $xd_a(t) = 2$ **and** $xd_a(t) = 3$**. For *bPSO N*=5.**

The clustering problem requires the determination of the solution space dimension (i.e. number of clusters) where in a recent work [19] MD-PSO technique has been successfully used. At time *t*, the particle *a* in the swarm, $\xi=\{x_1,..,x_a,..,x_S\}$, has the positional component formed as, $xx_a^{xd_a(t)}(t)=\{c_{a,1},..,c_{a,j},..,c_{a,xd_a(t)}\} \Rightarrow xx_{a,j}^{xd_a(t)}(t)=c_{a,j}$ meaning that it represents a potential solution (i.e. the cluster centroids) for the $xd_a(t)$ number of clusters whilst jth component being the jth cluster centroid. Apart from the regular limits such as (spatial) velocity, dimensional velocity, $VD_{\max}$ and dimension range $D_{\min} \le xd_a(t) \le D_{\max}$, the N dimensional data space is also limited with some practical spatial range, i.e. $X_{\min} < xx_a^{xd_a(t)}(t) < X_{\max}$. In case this range is exceeded even for a single dimension j, $xx_{a,j}^{xd_a(t)}(t)$, then all positional components of the particle for the respective dimension $xd_a(t)$ are initialized randomly within the range (i.e. refer to step 1.3.1 in MD-PSO pseudo code in [19]) and this further contributes to the overall diversity. In this work as well as in [19], the following validity index is used to obtain computational simplicity with minimal or no parameter dependency,

$$f(xx_a^{xd_a(t)},Z)=\ Q_e(xx_a^{xd_a(t)})(xd_a(t))^{\alpha} \quad where$$

$$Q_e(xx_a^{xd_a(t)})=\frac{1}{xd_a(t)}\sum_{j=1}^{xd_a(t)}\frac{\sum_{\forall z_p \in xx_{a,j}^{xd_a(t)}}\left\|xx_{a,j}^{xd_a(t)}-z_p\right\|}{\left\|xx_a^{xd_a(t)}\right\|} \tag{3}$$

where Q_e is the quantization error (or the average intra-cluster distance) as the *Compactness* term and $(xd_a(t))^{\alpha}$ is the *Separation* term, by simply penalizing higher cluster numbers with an exponential, $\alpha \ge 0$.

On the other hand, (hard) clustering has some constraints. Let $C_j=\{xx_{a,j}^{xd_a(t)}(t)\}=\{c_{a,j}\}$ be the set of data points assigned to a (potential) cluster centroid $xx_{a,j}^{xd_a(t)}(t)$ for a particle *a* at time *t*. The partitions $C_j,\ \forall j \in [1, xd_a(t)]$ should maintain the following constraints:

1) Each data point should be assigned to one cluster set, i.e. $\bigcup_{j=1}^{xd_a(t)} C_j = Z$
2) Each cluster should contain at least one data point, i.e. $C_j \ne \{\phi\},\ \forall j \in [1, xd_a(t)]$
3) Two clusters should have no common data points, i.e. $C_i \cap C_j = \{\phi\},\ i \ne j\ and\ \forall i,j \in [1, xd_a(t)]$

In order to satisfy the 1st and 3rd (hard) clustering constraints, before computing the clustering fitness score via the validity index function in (3), all data points are first assigned to the *closest* centroid. Yet there is no guarantee for the fulfillment of the 2nd constraint since $xx_a^{xd_a(t)}(t)$ is set (updated) by the internal dynamics of the MD-PSO process and hence any dimensional component (i.e. a potential cluster candidate), $xx_{a,j}^{xd_a(t)}(t)$, can be in an abundant position (i.e. no closest data point exists). To avoid this, a high penalty is set for the fitness score of the particle, i.e. $f(xx_a^{xd_a(t)},Z) \approx \infty,\ if\ \{xx_{a,j}^{xd_a(t)}\}=\{\phi\}$ for any *j*.

Further details about MD PSO and dynamic clustering using MD PSO with FGBF are skipped due to space limitations and can be found in [19].

3. CLASSIFICATION AND RETRIEVAL IN MACROINVERTABRATE DATABASES

3.1 Feature Extraction for Classification

Feature extraction and selection depend on the data and classifier used. As such, features can be seen as a part of the classifier system itself. However, such feature selection in case of images has remained empirical science and there are many possible features and enormous amount of their combinations. Examples of classical features are various edge, curve, ridge, blob, and corner based features, shape descriptors such as various moments and Fourier descriptors, simple textural features such as histograms of intensity, gradient [5] and co-occurrence [25]. Current state-of-the-art feature extraction methods includes Local Binary Patterns [26], Gabor packet based methods [27], scale invariant features of SIFT algorithm [28] and various other orientation based features such as in [29]. Also, wavelets and wavelet packets [30], [31] are convenient tools for acoustic and image analysis, compression, and feature extraction.

Among all these possibilities, in order to achieve a low-cost solution and to demonstrate the efficiency of the proposed classifier, we have applied a simple and basic feature extraction technique composed of mainly geometrical and statistical features. 15-D features of each macroinvertebrate image are extracted by using *ImageJ*, which is a public domain, Java-based image processing program developed at the NIH [20]. The following set of 15 features are selected by using *ImageJ*'s built in measurement and analysis functions: pixel value (grayscale) statistics { μ, σ, *Mode, Median, IntDen, Kurtosis, Skewness*} and geometric features {*Area, Perimeter, Width, Height, Ferret, Major, Minor, Circularity*}. The detailed description of these features can be found in [20]. In pre-processing step, each feature vector is then normalized to have a zero mean and linearly scaled into [-1, 1] interval before presented at the input layer of the evolutionary RBF classifier.

3.2 Evolutionary RBF Networks

In an earlier work [32], MD PSO has been successfully used to evolve multi-layer perceptrons (MLPs), that is, the automatic design of the feed-forward ANNs and the search is carried out over all possible network configurations within the specified architecture space. In [32], no assumption was made about the number of (hidden) layers and in fact none of the network properties (e.g. feed-forward or not, differentiable activation function or not, etc.) is an inherent constraint of this scheme. As long as the potential network configurations are transformed into a hash (dimension) table with a proper hash function where indices represent the solution space dimensions of the particles, MD PSO can then seek both positional and dimensional optima in an interleaved PSO process. The optimum dimension found naturally corresponds to a distinct ANN architecture where the network parameters (connections, weights and biases) can be resolved from the positional optimum reached on that dimension.

In the current work, our approach for evolutionary RBF networks is somewhat different since they have a fixed structure, i.e. one hidden layer with only *Gaussian* neurons. So the search for the optimal layer number is not needed whereas determination of the optimal number of *Gaussian* neurons with their correct parameters (centroids and variances) has the utmost importance. Therefore, the dynamic clustering using MD PSO with FGBF is naturally applied to determine the optimal number of clusters (say *N*) and the corresponding (cluster) centroids $\bar{\mu}$. The number of Gaussian neurons and their centroids in the RBF network are then set according to the clustering results. The dynamic clustering is applied to the 15-D feature vectors of the training set of the macroinvertebrate database. Afterwards the variance, σ of each cluster (or Gaussian neuron) can be analytically computed. Note that RBF networks do not have weights assigned between input and hidden (Gaussian) layers –only between hidden and output layers. Therefore, there is no need for an evolutionary algorithm such as GA or PSO to compute the weights and biases, since in this case there is a unique solution or the problem is uni-modal. Hence BP can conveniently be used to compute the remaining network parameters, weights (***w***) and bias (θ) of the each output layer neuron. Figure 2 summarizes the process of evolutionary RBF network design and training.

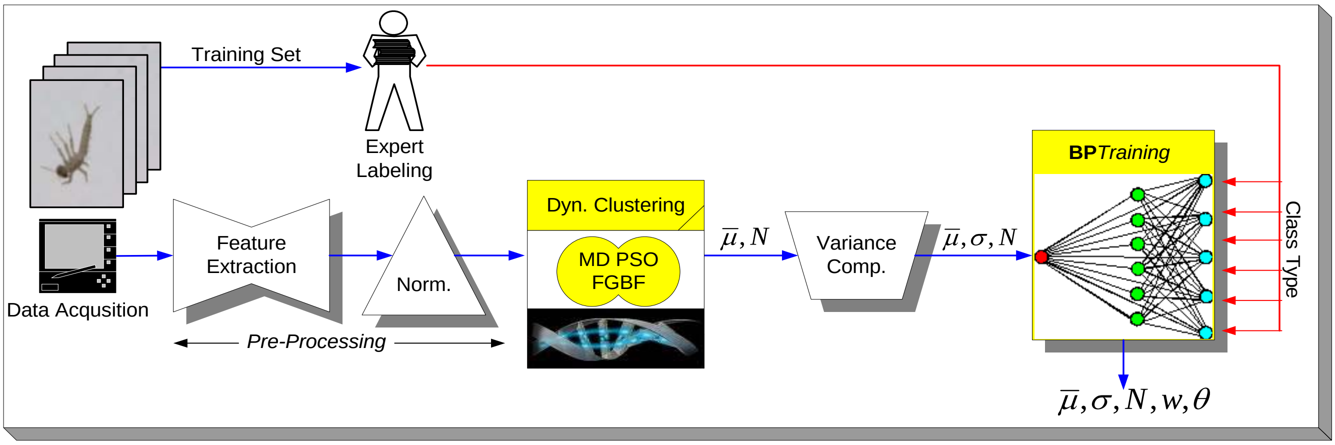

Figure 2: Overview of the evolutionary RBF network design.

4. EXPERIMENTAL RESULTS

The Benthic macroinvertebrate dataset used in this work consists of 1350 images representing 8 different taxonomical groups: *Baetis rhodari*, *Diura nanseni*, *Heptageria Sulphurea*, *Hydropsyche Pellucidulla*, *Hydropsyche Siltalai*, *Iso Perla*, *Rhyacophila Nubila* and *Taeniopteryc Nebulose*. Members from the same taxonomical group were captured by a flatbed scanner, digitized, normalized and eventually each sample in each scan was saved as an individual image.

Three individual samples from two classes are shown in Figure 3 and demonstrate some crucial properties of the data: specimens are semi-rigid so that the actual shape may vary from one sample to other. Furthermore there can be overlapping, repetitions, rotations, scaling and variations in the intensity levels, all of which make the classification problem even more challenging and the need for a powerful classifier is imminent for an accuracy matching the human (expert) classification.

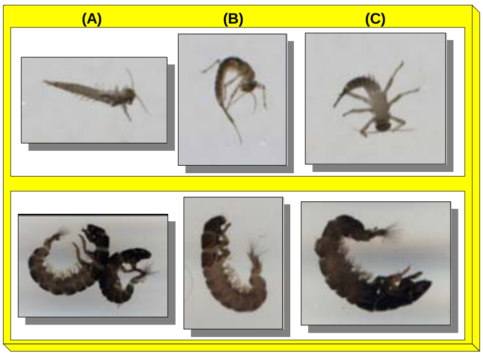

Figure 3: Three samples from *Baetis rhodari* (top) and *Hydropsyche Pellucidulla* (bottom) classes.

4.1 Classification Results

In order to evaluate the effect of the data partitioning, we created 10 distinct train-test partitions each with 650-700 samples, randomly chosen among the dataset samples. The dimensional range for MD PSO (equal to the range for number of Gaussian neurons) is: $D_{\min}=10\leq N\leq D_{\max}=35$. We kept the default parameters for MD PSO [19] with a swarm size, *S*=200, and velocity ranges are empirically set as $V_{\max}=-V_{\min}=X_{\max}/5$, and $VD_{\max}=-VD_{\min}=D_{\max}/5$. The position range is set as $X_{\max}=-X_{\min}=1$. We make the comparative evaluations of the proposed evolutionary RBF networks against the RBF networks trained with the BP method for each partition. Since MD PSO is a stochastic technique in nature, in order to demonstrate its convergence to (near-) optimal solution, we perform 10 runs per partition from which first and second order statistics are computed. We also repeated BP 10 times per partition and per RBF configuration within the aforementioned range since BP is only a training method that cannot search for an optimal configuration. We have applied 2000 iterations for MD PSO clustering and 2000 epochs for the following BP operation whereas for the standalone BP training, we have applied 10000 epochs. The increase and decrease factors for both BP (SuperSAB) operations were set to 1.25 and 0.5, respectively.

As the classification error (CE) statistics on train/test sets presented in Table 1 clearly indicate, evolutionary RBF networks are far superior to the RBFs trained with traditional BP. This is particularly evident between their mean train and test CE statistics. Recall that BP training is applied over all RBF networks within the aforementioned range and still evolutionary RBF networks achieve such an *average* classification performance, which is better than even the *best* performance of the BP-trained RBF networks. This is in fact an expected outcome since BP is just a gradient descent algorithm on the error space, and besides weights and biases, it simultaneously computes 15-D centroids and variance *per Gaussian* neuron. In this case, the error surface obviously becomes quite complex and contains massive amount of deceiving local minima. Therefore, BP most likely gets trapped early into a local minimum, making its classification performance entirely dependent on the *initial* settings. On the other hand, evolutionary RBFs take the advantage of an uni-modal error space in significantly lower dimensions (containing only weights and biases) whilst parameters of the *Gaussian* neurons (their number, centroids and variances) are (near-) *optimally* computed by MD PSO with FGBF, with respect to the clustering validity index function given in Eq. (3). Due to this fact, the deviations (variance) in evolutionary RBF networks are also much lower than the ones with BP-trained. Another observation worth mentioning is that evolutionary RBF networks achieve a better robustness against the variations of the train/test datasets.

Table 1. Train and test classification error statistics for evolutionary and BP-trained RBFs per dataset partition. The best (minimum) statistics are highlighted.

	Train Classification Error						Test Classification Error					
	BP-RBF			Evol. RBF			BP-RBF			Evol. RBF		
	Min.	μ	σ	**Min.**	μ	σ	**Min.**	μ	σ	**Min.**	μ	σ
Par-1	0.0554	0.1054	0.0152	**0.0246**	**0.0395**	**0.086**	0.0629	0.1079	0.017	**0.06**	**0.0684**	**0.0065**
Par-2	0.0446	0.1003	0.0143	**0.0246**	**0.0346**	**0.0061**	0.0843	0.1353	0.0157	**0.0729**	**0.08**	**0.0065**
Par-3	0.0554	0.105	0.0192	**0.0215**	**0.0355**	**0.0112**	0.0714	0.1288	0.0194	**0.0529**	**0.0684**	**0.0095**
Par-4	0.0446	0.1028	0.0145	**0.0277**	**0.0334**	**0.0089**	0.0743	0.1234	0.0149	**0.0586**	**0.0694**	**0.0093**
Par-5	0.0415	0.0954	0.0161	**0.0154**	**0.0309**	**0.0137**	0.0829	0.1329	0.0161	**0.0671**	**0.0791**	**0.0143**
Par-6	0.0477	0.1017	0.0137	**0.0277**	**0.0397**	**0.0126**	0.0771	0.1308	0.014	**0.0671**	**0.0797**	**0.0116**
Par-7	0.0538	0.1136	0.015	**0.0308**	**0.0448**	**0.011**	0.0743	0.1258	**0.0151**	*** 0.0514**	**0.0739**	0.0152
Par-8	0.0462	0.0945	0.0168	**0.02**	**0.0349**	**0.0153**	0.0843	0.1313	**0.017**	**0.0614**	**0.079**	0.018
Par-9	0.0508	0.1122	0.0191	**0.0215**	**0.0337**	**0.0105**	0.757	0.1305	0.02	**0.0586**	**0.0754**	**0.0123**
Par-10	0.0569	0.1001	0.0153	**0.0338**	**0.0385**	**0.0027**	0.0757	0.1145	0.0142	**0.0571**	**0.0653**	**0.0061**

4.2 Retrieval Results

The retrieval process in MUVIS is based on the traditional query by example (QBE) operation. The features of the query item are used for (dis-) similarity measurement among all the features of the visual items in the database. Ranking the database items according to their similarity distances yields the retrieval result. The traditional (dis-) similarity measurement in MUVIS is by applying a distance metric such as L2 (*Euclidean*) between the feature vectors of the query and the (next) database item. So in Benthic macroinvertebrate database, this corresponds to computing *Euclidean* distance between two 15-D feature vectors. In order to obtain the highest retrieval performance, we have chosen the evolutionary RBF classifier with the best generalization ability (i.e. the one achieved the overall minimum test CE for partition-7, indicated with a '*' in Table 1). When the classifier is used, the same (L2) distance metric is now applied to the class vectors at the output layer. In order to evaluate the retrieval performances with and without classifiers, we used average precision (AP) and average normalized modified retrieval rank (ANMRR) measures, both of which are computed querying all (1350) images in the database and within a retrieval window equal to the number of ground truth images, *N(q)* for each query *q*. This henceforth makes the AP identical to average recall and average F1 measures, too.

With the traditional approach (without classifier), we obtain ANMRR = 0.4757 and AP = 0.4912, indicating in fact a quite poor retrieval performance due to the limited discrimination power of the basic descriptors used. With the use of the classifier, the retrieval performance has been improved to the level of ANMRR = 0.0671 and AP = 0.9255. This eventually presents an efficient solution for the accurate retrieval and biomonitoring of the macroinvertebrate specimens. For visual evaluation, Figure 4 presents two typical retrieval results with and without using the proposed classifier.

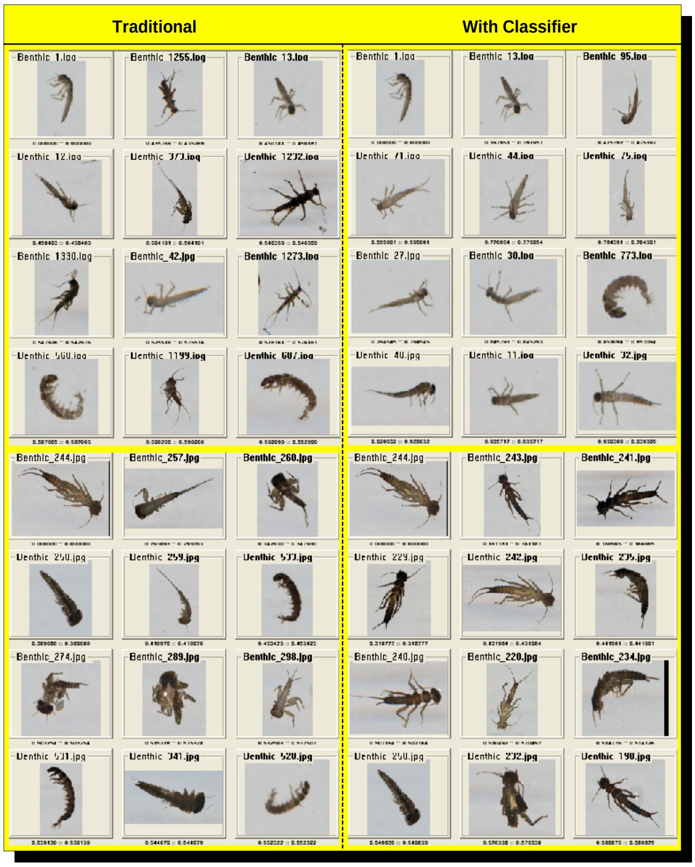

Figure 4: Two sample queries of *Baetis rhodari* and *Diura nanseni* with (right) and without (left) using classifier. Top-left is the query image.

5. CONCLUSIONS

In this paper, we addressed the problem of cost-intensive manual taxonomic classification and retrieval of macroinvertebrate specimens by introducing a novel evolutionary RBF network classifier. With the proper adaptation of the native MD PSO clustering process, the proposed method can evolve to the optimum RBF network within the specified architecture space. Among many alternatives, in order to demonstrate the efficiency of the classifier and to propose a low cost solution, we have intentionally extracted the most basic and simplest features from the Benthic macroinvertebrate images. Both training and test classification results indicate a superior performance with respect to the traditional BP method, i.e. the *average* performance that the proposed evolutionary RBF networks surpasses even the *best* performance obtained by BP training. Even the worst (mean) classification errors for train and test sets, 4.48% and 7.286%, are far superior to the ones reported in a recent work [5], which used the same database with the identical number of classes.

We then used the best classifier in terms of test classification performance, for the purpose of accurate similarity-based retrievals using the MUVIS framework. The retrieval results from the extensive query experiments show that an elegant performance in retrieval accuracy is achieved, particularly when compared to the traditional (without classifier) retrieval methodology.

6. REFERENCES

[1] P.F. Culverhouse, (and 11 others) “Automatic classification of field-collected dinoflagellates by artificial neural network,” *Marine Ecology Progress Series*, vol. 139, pp. 281-287, 1996.

[2] K.J. Gaston, and M.A. O'Neill, “Automated species identification: why not?,” *Philosophical Transactions of the Royal Society of London Series B*, vol. 359, pp. 655-667, 2004.

[3] M. Benfield, (and 14 others) “RAPID: Research on Automated Plankton Identification,” *Oceanography*, vol. 20, no. 2, pp. 172-187, 2007.

[4] N. Larios, H. Deng and W. Zhang, “Automated insect identification through concatenated histograms of local appearance features,” *Machine Vision and Applications*, vol. 19, pp. 105-123, 2008.

[5] V. Tirronen, A. Caponio, T. Haanpää and K. Meissner, “Multiple order gradient feature for macroinvertebrate identification using support vector machines,” *Lecture Notes in Computer Science*, in Press.

[6] P.F. Culverhouse, R. Williams, B. Reguera, V. Herry and S. Gonzales-Gil, ”Do experts make mistakes? A comparison of human and machine identification of dinoflagellates,” *Marine Ecology progress Series*, vol. 247, pp. 17-25, 2003.

[7] Y.P. Ginoris, A.L. Amaral, A. Nicolau, M.A.Z. Coelho and E.C. Ferreira, “Recognition of protozoa and metazoa using image analysis tools, discriminant analysis, neural networks and decision trees,” *Analytica Chimica Acta*, vol. 595, pp. 160-169, 2007.

[8] P.F. Culverhouse, (and 11 others) ”Automatic categorisation of 23 species of dinoflagellate by artificial neural network,” *Marine Ecology Progress Series*, vol. 139, pp.281-287, 1996.

[9] N. Burgess, “A constructive algorithm that converges for real-valued input patterns,” *Int. Journal of Neural Systems*, vol. 5, no. 1, pp. 59-66, 1994.

[10] M. Frean, “The upstart algorithm: A method for constructing and training feedforward neural networks”, *Neural Computation*, vol. 2, no. 2, pp. 198–209, 1990.

[11] Y. LeCun, J. S. Denker, and S. A. Solla, “Optimal brain damage,” *in Advances in Neural Information Processing Systems 2*, pp. 598–605, 1990.

[12] R. Reed, “Pruning algorithms—A survey“, *IEEE Trans. Neural Networks*, vol.4, no.5, pp.740–747, 1993.

[13] P. J. Angeline, G. M. Sauders, and J. B. Pollack, “An evolutionary algorithm that constructs recurrent neural networks”, *in IEEE Trans. Neural Networks*, vol. 5, pp. 54–65, Jan. 1994.

[14] T. Back and H.P. Schwefel, “An overview of evolutionary algorithm for parameter optimization”, *Evolutionary Computation 1*, pp. 1–23, 1993.

[15] D. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Reading, pp. 1-25. MA, 1989.

[16] X. Yao and Y. Liu, “A new evolutionary system for evolving artificial neural networks”, *IEEE Trans. On Neural Networks*, vol.8, no.3, pp. 694–713, May 1997.

[17] S. C. Esquivel and C. A. Coello Coello, “On the Use of Particle Swarm Optimization with Multimodal Functions”, *In IEEE Trans. on Evolutionary Computation*, vol. 2, pp. 1130-1136, 2003.

[18] J. Kennedy, R Eberhart., “Particle swarm optimization”, *in Proc. of IEEE Int. Conf. On Neural Networks*, vol. 4, pp. 1942–1948, Perth, Australia, 1995.

[19] S. Kiranyaz, T. Ince, A. Yildirim and M. Gabbouj, “Fractional Particle Swarm Optimization in Multi-Dimensional Search Space”, *IEEE Transactions on Systems, Man, and Cybernetics – Part B*, in Press, doi:10.1016/j.neunet.2009.05.013, 2009.

[20] ImageJ: public domain Java-based image processing program, [Online]. Available: http://rsbweb.nih.gov/ij/docs/index.html

[21] T. Tollenaere, “SuperSAB: Fast Adaptive Back Propagation with Good Scaling Properties,” Neural Networks, vol. 3, no. 5 pp. 561–573, 1990.

[22] MUVIS [online]. http://muvis.cs.tut.fi

[23] T. Poggio and F. Girosi, “A theory of networks for approximation and learning,” A.I. Memo No. 1140, M.I.T. A.I Lab, 1989.

[24] S. Haykin, “Neural Networks: a Comprehensive Foundation,” Prentice hall, USA, June 1998.

[25] R. M Haralick, K Shanmugam, and I. Dinstein (1973). "Textural Features for Image Classification". *IEEE Transactions on Systems, Man, and Cybernetics* SMC-3 , vol. 6, pp. 610–621, 1973.

[26] T. Ojala M. Pietikainen, T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.24, no.7, pp.971-987, Jul 2002.

[27] S.E. Grigorescu,; N. Petkov, P. Kruizinga, “Comparison of texture features based on Gabor filters,” *IEEE Transactions on Image Processing*,, vol.11, no.10, pp. 1160-1167, Oct 2002.

[28] D.G., Lowe, “Object recognition from local scale-invariant features,” *In Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol.2, pp.1150-1157, 1999.

[29] N. Dalal,; B. Triggs, , "Histograms of oriented gradients for human detection," *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on* , vol.1, no., pp.886-893 vol. 1, 25-25 June 2005.

[30] J. S. Walker, (1999) A Primer on Wavelets and their Scientific Applications, CRC Press, 1999.

[31] D. Alani, A.Averbuch, , and S. Dekel, “Image coding with geometric wavelets” IEEE Trans. on Image Processing, 16(1), 69-77, 2007.

[32] S. Kiranyaz, T. Ince, A. Yildirim and M. Gabbouj, “Evolutionary Artificial Neural Networks by Multi-Dimensional Particle Swarm Optimization,” Neural Networks, in Press, doi:10.1016/j.neunet.2009.05.013, 2009.