NETWORK OF EVOLUTIONARY BINARY CLASSIFIERS FOR CLASSIFICATION AND RETRIEVAL IN MACROINVERTEBRATE DATABASES

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

In this paper, we focus on advanced classification and data retrieval schemes that are instrumental when processing large taxonomical image datasets. With large number of classes, classification and an efficient retrieval of a particular benthic macroinvertebrate image within a dataset will surely pose a severe problem. To address this, we propose a novel network of evolutionary binary classifiers, which is scalable, dynamically adaptable and highly accurate for the classification and retrieval of large biological species-image datasets. The classification and retrieval results for the macroinvertebrate test data attain taxonomic accuracy that equals and even surpasses that of an average expert. Our findings are encouraging for aquatic biomonitoring where cost intensity of sample analysis currently poses a bottleneck for routine biomonitoring.

1. INTRODUCTION

Human development within rivers or their watershed can pose an inherent threat to the affected river ecosystem. The monitoring of aquatic macroinvertebrates is particularly efficient in pinpointing the cause-effect structure between slow and subtle *anthropogenically* induced changes and their detrimental consequences in river ecosystems. Currently, the cost-intensity of human expert taxonomic identification of samples is the major obstacle to the implementation of more biomonitoring. Evidence suggests that automated recognition can match human taxa identification accuracy at greatly reduced costs [1], as in [2], [3], [4], and [5] different classifiers (Support Vector Machines, Self Organizing Maps, Artificial Neural Networks) with fixed configurations have been used for automated insect classifications. So far the development of automated identification techniques for freshwater macroinvertebrates has received very little attention [6]. In a recent study [7] on a set of river macroinvertebrates, aerage correct classification of 88.2% and 75.3% have been achieved in training and test sets. The observed levels of taxonomical accuracy match levels of human accuracy for other aquatic taxonomic groups [8]. In follow-up work on the same dataset [9], a significant improvement on the classification accuracy was achieved. The Following an evolutionary radial basis function (RBF) neural network approach the classification error (CE) rate was 7.41% for train and 5.14% for the test set. This high performance was achieved, despite the use of the most basic and primitive geometrical and intensity based features (as detailed in [10]). The key to such success lies in the evolutionary mechanism of the neural network, which searches for both optimal configuration and network parameters simultaneously.

Although these preliminary results are promising, their practical use is limited to fixed (static) datasets with low number of classes, features and dimensions like in [8] and [9], where only 8 classes were used. Using a single classifier invokes the problem of scalability, that is, in the database both the number of feature space dimensions and classes cannot exceed certain limits without rendering the proper training of the classifier(s) infeasible due to the massive size of the input and output layers. This also prevents the use of single classifiers in dynamic databases where new features or classes may emerge or some of the existing ones may disappear over time. For such cases, the learning body should be appropriate for online (incremental) training as well as the offline (batch) training. If an evolutionary approach is chosen, the evolution process should continue during ongoing training sessions from its last stopping point and incorporate new emerging classes/features or new training samples when they become available via user's relevance feedbacks (RFs).

In order to provide an efficient solution to the aforementioned scalability and dynamic adaptability problems whilst maximizing the classification and retrieval performances, in this paper we propose a *Divide and Conquer* type of approach, which is based on a novel framework encapsulating a *network of (evolutionary) binary classifiers* (NBCs). As in [8], we shall use multi-dimensional Particle Swarm Optimization (MD-PSO) as the evolution mechanism –yet this time over the multi-layer perceptrons (MLPs). The evolutionary MLPs were detailed in [12] and in this work we further show the dynamic (incremental) evolutionary process performed over each binary classifier (BC) in an NBC. After certain number of evolutionary runs, the best set of BCs in each NBC, which yields the best generalization capability (the lowest CE in the test/validation set) is then integrated into the search engine of the MUVIS [13] and used for the (dis-) similarity distance computation within the similarity-based queries.

2. NETWORK OF BINARY CLASSIFIER FRAMEWORK

In this work, scalability with respect to a large number of classes and different visual features (descriptors) is the primary aim. To do so, a novel framework encapsulating a *network of binary classifiers* (NBCs) is developed, where NBCs can *evolve* continuously with the ongoing training/evolving sessions (i.e. with user RFs) and the optimum classifier configuration –so far– shall prevail at any time. Each NBC corresponds to a *unique* macroinvertebrate class and shall contain indefinite number of evolutionary binary classifiers (BCs) in the input layer where each BC performs binary classification over an individual feature. Therefore, whenever a new feature is extracted, a new BC can

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

be created, trained and inserted into each NBC of the system using the available RF logs so far, yet keeping the other BCs "as is". On the other hand, whenever an existing feature is removed, the corresponding BC will simply be removed from each NBC in the system. Finally, a "fuser" BC in the output layer shall fuse the binary outputs of all BCs in the input layer and outputs a single binary output, indicating the relevancy of each media item to its class. This makes the system *scalable* to any number of classes since whenever a new class is defined by the human expert, the system can simply create and train a new "corresponding NBC" for this class and thus the overall system dynamically adapts to user demands of macroinvertebrate classes for defining the aquatic organisms in the database. Each BC in an NBC shall in time learn the significance of individual parts (dimensions) of the corresponding feature vector for the discrimination of its class.

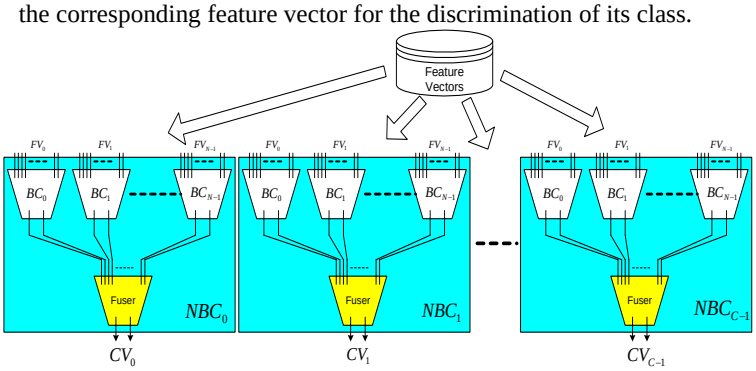

Figure 1: Overview of the proposed classifier framework with *C* classes and *N* features.

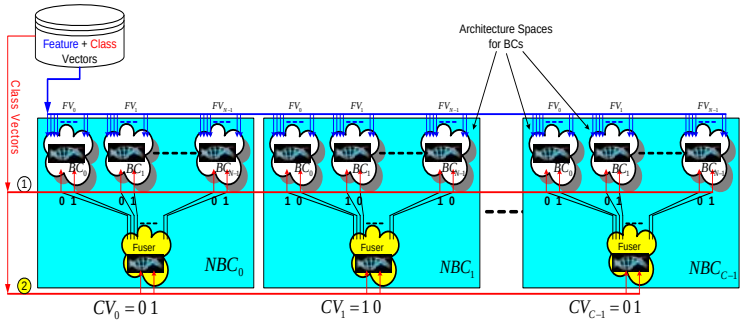

Figure 2: Illustration of a training/evolution session over all BCs' architecture spaces in each NBC.

As shown in Figure 1, the main idea in this approach is to use as large number of classifiers as necessary, so as to "divide" a massive learning problem into many NBC units (and many BCs necessary within) and thus prevent the need of using complex classifiers. The idea is that no more than a few dozens of class items should be learned (discriminated) by a single BC per feature. Since the performance of both training and evolution processes degrades significantly as the complexity rises. A major benefit of our approach to efficient training and evolution process is that architecture space is kept as compact as possible which circumvents unfeasibly large storage and training time requirements. As illustrated in Figure 2, the evolution will be applied over an architecture space (see [12] for details) –not a single fixed configuration in order to find the best (optimal) BC with respect to a given criterion (e.g. training/validation MSE or CE). Note that along with the best classifier, the other classifiers are also subject to evolution and therefore, they are continuously kept trained with the new training set(s). In this ongoing evolution process any configuration can replace the current best one if it surpasses it. Since evolution is basically a search process within an architecture space, besides an evolutionary search method such as MD PSO, another alternative is *exhaustive search*, i.e. a training method such as back-propagation (BP) may be used several times to train each network in the architecture space within which the one with the best classification performance is always kept. In this work for both approaches several runs are conducted and, at each run, a configuration can replace the one currently in the architecture space if it has a better classification performance. In this way, each BC configuration in the architecture space will, therefore, continuously *evolve* to a better state, whilst the best among all at a given time shall be used for classification and retrieval. Once the evolution process is completed for the BCs in the input layer, the best configurations are used to evolve the fuser BC using the composed input vector from their output vectors that are created by the forward propagation of the training set's FVs. In this way the fuser BC shall learn the *significance* of each individual BC (and also its feature) for the discrimination of that particular class. This is in fact a crucial way of applying an efficient *Feature Selection* scheme as some features may be quite discriminative for some classes whereas others may not and the fuser, if properly evolved and trained, can "weight" each BC accordingly. In this way the usage of each feature (and its BC) shall optimally be "fused" according to their discrimination power of each class.

Along with the aforementioned continuous (online) training, a "long term" learning strategy can also be performed where the previous RF logs shall be stored and used for continuous, offline ("idle-time") training of the entire system, in order to improve the overall classification performance.

3. EXPERIMENTAL RESULTS

3.1. Dataset Creation and Feature Extraction

In order to perform comparative evaluations, the same Benthic macroinvertebrate image database as in [7] and [9] is used in this work. This database consists of 1350 images representing 8 different taxonomical groups: *Baetis rhodani*, *Diura nanseni*, *Heptagenia sulphurea*, *Hydropsyche pellucidulla*, *Hydropsyche siltalai*, *Isoperla sp.*, *Rhyacophila nubila* and *Taeniopteryx nebulosa*. Members from the same taxonomical group were imaged by a flatbed scanner, digitized, normalized and eventually each macroinvertebrate in each scan was saved as an individual image.

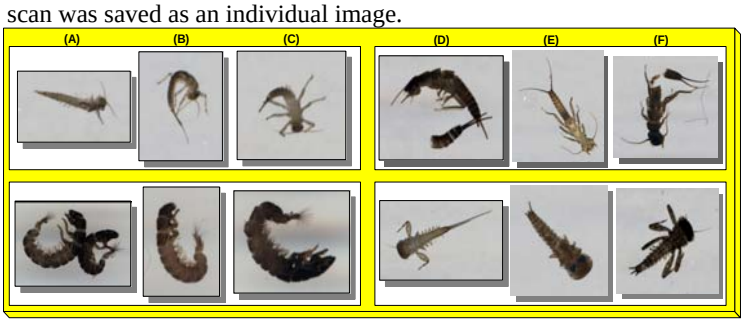

Figure 3: Three samples from *Baetis rhodani* (top: A, B, C), *Diura nanseni* (top: D, E, F), *Hydropsyche pellucidulla* (bottom: A, B, C) and *Heptagenia sulphurea* (bottom: D, E, F) classes.

Three individuals from four taxonomical 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 another. 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 to match the accuracy of expert-level human classification. We used the same feature set as in [9], composed of mainly geometrical and statistical features. 15-D features of each macroinvertebrate image are extracted by using *ImageJ* [10], which is a public domain, Java-based image processing program. 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 [10]. In a pre-processing step, each feature vector is then normalized to have a zero mean and linearly scaled into [-1, 1] interval before being presented to the classifier.

3.2. Classification Results

As in [9], we created 10 distinct train-test partitions each with 650-700 samples, randomly chosen among the dataset samples in order to evaluate the effect of the data partitioning. We partition the 15-D feature vector into two, 8-D and 7-D, sub-features and therefore, each NBC contains 2+1=3 BCs resulting to a collection of NBCs (CNBC) with 8x3=24 BCs. The architecture space for each MLP is defined by the layer range for the minimum and maximum number of layers, $\{L_{\min}, L_{\max}\}$ and two range arrays, $R_{\min} = \{N_I, N^1_{\min}, ..., N^{L_{\max}-1}_{\min}, N_O\}$ and $R_{\max} = \{N_I, N^1_{\max}, ..., N^{L_{\max}-1}_{\max}, N_O\}$, one for minimum and the other for maximum number of neurons allowed for each layer (see [12] for details). The size of both arrays is naturally $L_{\max}+1$ where corresponding entries define the range of the l^{th} hidden layer for all those MLPs having an l^{th} hidden layer. We use *hyperbolic tangent* as the activation function ($\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$) for MLPs. $N_O = 2$ for all BCs and same for all configurations in an architecture space within which *l*-layer MLPs can be defined providing that $L_{\min} \le l \le L_{\max}$. In this work we used $\{L_{\min}, L_{\max}\} = \{1,3\}$ and $R_{\min} - R_{\max} = \{N_I x 4 x 2 x 2\} - \{N_I x 8 x 4 x 2\}$. So there are total of 21 configurations (1 SLP and 20 MLPs) in this architecture space. N_I is set to 8 or 7 for each BC in the input layer and to 4 for the fuser BC. MD PSO [14], is performed with a swarm size, *S*=50, and velocity ranges are empirically set as $V_{\max} = -V_{\min} = X_{\max}/2$, and $VD_{\max} = -VD_{\min} = 10$. Dimension range is determined by the architecture space defined and position range is set as $X_{\max} = -X_{\min} = 2$. In order to simulate the online training, the evolution phase is completed in three sessions each of which contains 10 runs (repetitions): 2 MD PSO evolutions with 500 iterations in each run and, in between, one back-propagation (BP) training session with 250 iterations in each run is performed for each configuration in the architecture space. Due to space limitations, we skip the details of the online training of the BCs.

Table 1 presents the best (minimum) test CE results of the two competing techniques in [9] and the proposed NBC framework over the same dataset partitions. In [9] 2000 iterations for MD PSO clustering (with *S*=200) and 2000 epochs for the following BP operation were performed whereas for the standalone BP training, 10000 epochs were applied for each of the 10 runs.

Table 1: Min. test/validation set CEs per dataset partition

Partitions	*RBF-BP*	*Evol. RBF*	*CNBC*
Par-1	0.0629	0.06	**0.036**
Par-2	0.0843	0.0729	**0.0458**
Par-3	0.0714	0.0529	**0.0458**
Par-4	0.0743	0.0586	**0.0526**
Par-5	0.0829	0.0671	**0.0312***
Par-6	0.0771	0.0671	**0.042**
Par-7	0.0743	0.0514	**0.043**
Par-8	0.0843	0.0614	**0.044**
Par-9	0.757	0.0586	**0.034**
Par-10	0.0757	0.0571	**0.039**

3.3. 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 accomplished by applying a distance metric such as L2 (*Euclidean*) between the feature vectors of the query and each 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 chose the CNBC with the best generalization ability (i.e. the one achieved the overall minimum test CE for partition-5, 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 of the CNBC (8x2=16-D). In order to evaluate the retrieval performances with and without CNBC, we use 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. In [9], ANMRR = 0.0671 and AP = 0.9255 were obtained and with the use of CNBC the retrieval performance has further been improved to the level of ANMRR = 0.05217 and AP = 0.9415. This eventually presents an efficient solution for the accurate retrieval and biomonitoring of the macroinvertebrate specimens. For visual evaluation, Figure 4 presents three typical retrieval results with and without using the proposed NBC framework.

4. CONCLUSIONS

In this paper, a novel NBC framework is introduced to address the problem of cost-intensive manual taxonomic classification and retrieval of macroinvertebrate specimens. This is a *Divide and Conquer* type of approach, which reduces both feature and class vector dimensions significantly to obtain as compact classifiers as possible.

Higher efficiency and accuracy on the evolution/training performance is obtained by compact classifiers rather than complex ones. Note that the competing approaches were computationally significantly more complex than the proposed CNBC construction. Despite of this fact, NBC framework achieved lower test CE results for all partitions. This is due to several reasons. First and the foremost, the high dimensional input FV is divided into two lower dimensional vectors for the two BCs in the input layers of NBCs. Furthermore, instead of a single classifier with 8-D output layer, each BC has only 2-D binary outputs. A reduction in both input and output layers enables the use of compact classifiers which can be evolved and trained better than a single yet more complex classifier as in [9]. Moreover, when trained properly, the fuser BC can correct the erroneous classification of any BC in the input layer, which further increases the classification accuracy.

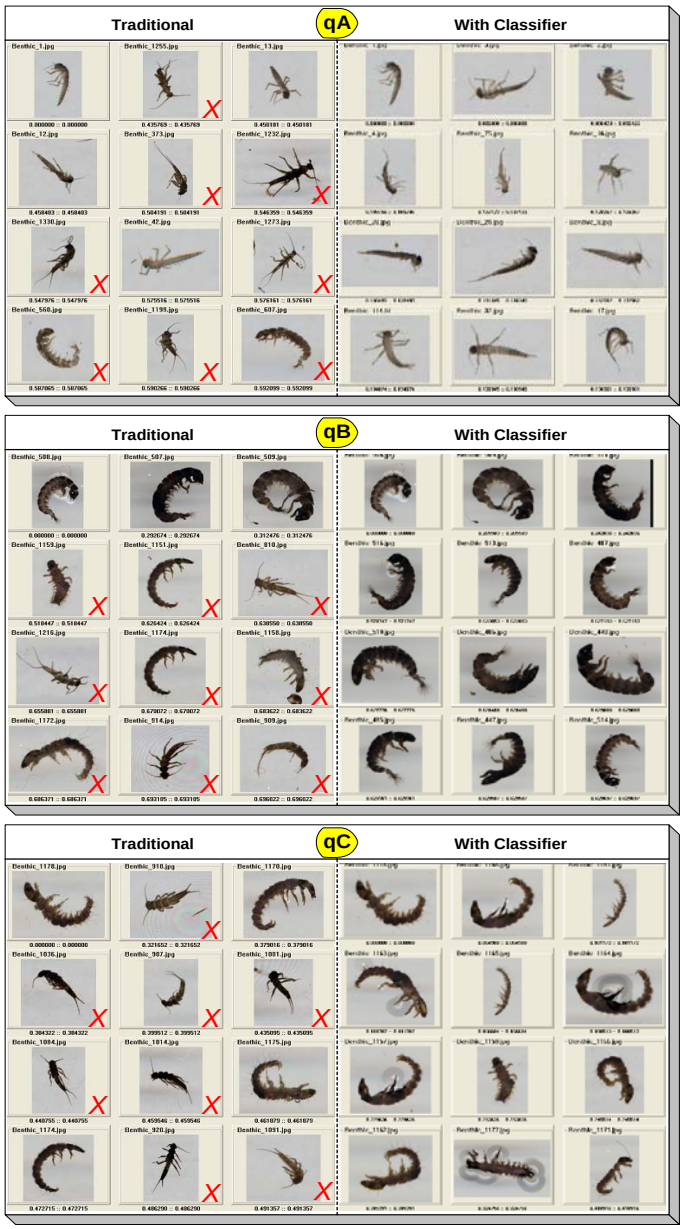

Figure 4: Three sample queries of *Baetis rhodani* (qA), *Hydropsyche pellucidulla* (qB) and *Rhyacophila nubila* (qC) with (right) and without (left) using CNBC. Top-left is the query image. Each irrelevant retrieval is marked with a red '*X*'

Apart from improved classification performance, the main advantage of the proposed framework is that it yields an efficient solution to the problems of scalability and dynamic adaptability by allowing both feature space dimensions and the number of classes in a database to be unlimited and dynamic (incremental). Furthermore, the NBC framework is designed for both online (incremental) and offline (batch) training/evolution operations, which can be repeated by several runs. During each run, any new configuration can replace the current one in the architecture space if it outperforms it . This ensures that the architecture space containing the best configurations is always kept intact and that only the best configuration at any given time is used for classification and retrieval.

REFERENCES

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

[2] T. Arbuckle, S. Schroder, V. Steinhage, D. Wittmann, "Biodiversity informatics in action: identification and monitoring of bee species using ABIS", in Proc. of the 15th Int. Symposium Informatics for Environmental Protection, vol. 1, pp. 425–430, Zurich, 2001.

[3] M.A. O'Neill, I.D. Gauld, K.J. Gaston, P. Weeks, "Daisy: an automated invertebrate identification system using holistic vision techniques", In Proc. of the Inaugural Meeting BioNET INTERNATIONAL Group for Computer-Aided Taxonomy (BIGCAT), pp. 13–22. Egham, 2000.

[4] M. Do, J. Harp, K. Norris, "A test of a pattern recognition system for identification of spiders", Bull. Entomol. Res. 89(3), pp. 217–224 1999.

[5] M-Y. Song, H-J. Hwang, I-S. Kwak, C.W. Ji, Y-N. Oh, B.J. Youn, T-S. Chon, "Self-organizing mapping of benthic macroinvertebrate communities implemented to community assessment and water quality evaluation," Ecological Modeling, vol. 203, pp. 18-25, 2007.

[6] N. Larios, H. Deng, W. Zhang, M. Sarpola, J. Yuen, R. Paasch, A. Moldenke, D.A. Lytle, S.R. Correa, E.N. Mortensen, L.G. Shapiro, T.G. Dietterich "Automated insect identification through concatenated histograms of local appearance features: feature vector generation and region detection for deformable objects," *Machine Vision and Applications*, vol. 19, pp. 105-123, 2008.

[7] 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*, pp. 489-497, 2009.

[8] 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.

[9] S. Kiranyaz, M. Gabbouj, J. Pulkkinen, T. Ince and K. Meissner, "Classification and Retrieval on Macroinvertebrate Image Databases using Evolutionary RBF Neural Networks", Int. Workshop on Advanced Image Technology (IWAIT), Malaysia, Kuala Lumpur, Jan. 2010.

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

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

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

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

[14] 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.