International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:5, No:1, 2011

Feature Weighting and Selection - A Novel Genetic Evolutionary Approach

Serkawt Khola*

***Abstract*—**A feature weighting and selection method is proposed which uses the structure of a weightless neuron and exploits the principles that govern the operation of Genetic Algorithms and Evolution. Features are coded onto chromosomes in a novel way which allows weighting information regarding the features to be directly inferred from the gene values. The proposed method is significant in that it addresses several problems concerned with algorithms for feature selection and weighting as well as providing significant advantages such as speed, simplicity and suitability for real-time systems.

***Keywords*—**Feature weighting, genetic algorithm, pattern recognition, weightless neuron.

I. Introduction

In pattern classification tasks defined via features measured or extracted from objects, one desires the most relevant and least redundant set of features which can best model the underlying data distribution. The necessity of feature selection is then apparent as redundant and/or irrelevant features will often tend to reduce the classification performance [10]. Ideally, only feature subsets with high class separation potential are chosen and other less discriminatory features are either discarded or assigned a reduced weighting. If we are interested in knowing how much attention we should pay to a particular feature for a specific problem, then we need to have information regarding its significance to the given problem at hand. This can be achieved via algorithmic analysis which assigns a significance value for each available feature. This then falls into a subcategory of feature selection termed feature weighting [5].

In this paper I propose a method based on Genetic Algorithms (GA) [6], [11] for feature weighting and selection. In feature selection, GAs fall into the randomized category and have successfully been applied to select features in large feature spaces. Most approaches reported in the literature use the GA, with relevant optimization criteria, to find a *d*-dimensional subspace in a *D*-dimensional pattern feature space such that $d < D$. Siedlecki and Sklansky [14] were the first to introduce the use of GA for direct feature selection. They used GA to find the best subset of features to classify patterns with the K-Nearest Neighbor (KNN) classifier. In their method, a chromosome is coded as a vector of N binary genes, N being the total number of features, where each gene is associated with a feature in the set. Within this scenario, if the *i*th gene value is 1 then the *i*th feature is selected to participate in the classification, otherwise it will have the value of 0, indicating exclusion. The chromosomes are evaluated with a fitness function, which includes classification accuracy and a penalty for the number of selected features. This approach was later viewed as a weighting of the features by 0/1 and expanded to discover the real-valued weights [8], [12]. Given a set of pattern feature vectors in the form of $X = \{x_1, x_2, ..., x_D\}$, the GA produces a transformed vector of features of the form $Y = \{x_1 w_1, x_2 w_2, ..., x_D w_D\}$ where w_i is the weight of feature x_i. This allows a linear scaling of the features and removal of features by assigning weights of zero. Many variations of the Siedlecki and Sklansky approach were later proposed in combination with different classifiers and objective functions [2], [10], [13], [16], [18] and in multiple classifier design [9]. Kalapanidas and Avouris [7] suggested a new way of coding features onto the chromosomes containing only the enabled features. Bhanu and Lin [3] proposed a GA-based feature selection algorithm which uses a multiple-criteria objective function and which automatically adapts its parameters. In [15] the Siedlecki and Sklansky approach was applied to the eigenvectors extracted by Principal Component Analysis

II. The proposed algorithm

The basic structure of the proposed algorithm is derived from some of the works by S. Khola, termed "KEPFA" [19]-[21], which uses the structure of a weightless N-tuple-based neuron [1]. Here, the KEPFA algorithm uses a novel approach for feature weighting and uses a different fitness function.

As with traditional GAs, the algorithm requires all feature values to be represented by binary patterns and combined into a pattern vector $X = \{0,1\}^N$, where, $N = \sum_{i=1}^{D} n_i$, D being the total number of original features and n_i being the total number of binary bits for the *i*th feature of D. In order to keep differences between consecutive numbers, in both real and binary spaces, the same, Gray codes [11] are used to define the base-2 format into which real-valued features are converted, providing the original features were not already coded in any base-2 format.

The proposed method proceeds as follows: Denote the index set for the components of vector X by $\{1,2 \cdots N\}$. A population U of R chromosomes are created, $U = \{u_1, u_2, \ldots u_R\}$,where,

$$u_i = \{g_j \mid g_j \in \{1,2,\ldots N\}, j = \{1,\cdots G_i\}, G_{min} \le G_i \le G_{max}\}$$

The number of genes for a given chromosome is normally

$G_i << N$, where gene g_j represents the index of jth bit in X and $f(g_j) \in X$ is the value of the bit in the pattern X indexed by g_j. Each individual u_i addresses $A = 2^{G_i}$ memory locations according to its gene values, and each memory location can store a set of integer-valued identifiers for classes $\{c_1, c_2 \ldots, c_K\}$, where K is the total number of pattern classes. During learning, the class identifier, stored at the memory location a in chromosome r is given by $a_r(X) = \sum_{n-1}^{j=0} f(g_j) * 2^j$, pertaining to the training pattern X is either set to 1 or incremented, depending on the learning scheme used [1].

III. Fitness function

The effectiveness of the genes coded in each chromosome i is evaluated on unseen patterns using the fitness function f, where,

$$f(i) = p(c_k|x) = p(x|c_k)p(c_k)/p(x) \qquad (1)$$

which is the posterior probability of class k given pattern x.

For a given chromosome, the terms of this probability are obtained from the values stored for each class at the accessed memory address a_j, where $a_j \equiv x$. The class-conditional likelihood estimate $p(x|c_k)$, for accessed memory address a_j given class k, is then obtained as the fraction of the value of the identifier for class c_k and memory address a_j, $n(a_j^{c_k})$, by the total sum of the identifier value in all memory addresses for class c_k.

The number of training patterns for class c_k, $N_{c_k}^{tr}$, is used to calculate the class prior probability for k^{th} class as the fraction of $N_{c_k}^{tr}$ by the total number of training patterns for all classes.

Finally, the unconditional likelihood estimate $p(x)$, is then obtained by:

$$p(x) = \sum_k n(a_j^{c_k}) \Big/ \sum_k \sum_l n(a_l^{c_k}) \qquad (2).$$

After each evaluation, the fittest R^* individuals, defined as all those individuals with above-average fitness, are selected to vote for the bits coded in their gene values. A new population of offspring is created by cross-over of genes between the selected individuals using, for example, roulette wheel selection.

The algorithm is stopped when either the standard deviation of the fitness of the population, averaged over all individuals, is below δ for Θ generations, or the number of GA iterations exceeds a maximum number of generations T, where δ, Θ and T are all user-defined parameters.

The total vote for each bit, after evaluation of each available test pattern and pattern class, is then summed and obtained. The bit weights are finally given by normalising the votes for each bit with respect to the total number of votes.

IV. Comparisons

The proposed algorithm was tested on a dataset of profile correlations extracted from handwritten numerals ‘0’ to ‘9’ [22], in which all features were measured in reals. I used gray codes to represent the features in base-2 and carried out two sets of tests. In test set 1, 10 independent runs were carried out. The results presented in all figures for test set 1 are the averages obtained from all 10 runs. For both test sets I used the first 20 original features of numeral classes ‘0’ and ‘1’. For test set 1, the general theory and functionality of the algorithm was first investigated. I, therefore, used only the first 20 patterns for training and tuning and the following 20 for testing, while for test set 2 I used all available patterns, first 100 for training and tuning and the following 100 for testing, for each class. For all tests, the probability of cross-over and mutation in KEPFA were set to 0.95 and 0.03 respectively, elitism was enabled, and roulette wheel selection with linear fitness scaling [6] and one-point crossover were employed. The values for δ and Θ were set to 0.5 and 5, respectively. As the maximum number of generation I thought that 100 would be a reasonable number to test. The number of genes for all chromosomes in all tests was set to 2 in order to avoid the need of modifying the crossover and mutation operations to deal with different allele sizes. Individuals are taught patterns using the Frequency Weighted Scheme [1], which increments a counter for each training pattern class whenever a training pattern of that class accesses a memory location. The number of individuals was fixed at 50 for all tests.

KEPFA was compared with three classical feature selection methods, Sequential Forward Selection (SFS), Sequential Backward Selection (SBS) and Floating Search Selection (FSS) [17]. In the tests, all three methods selected features in the original D-dimensional real-space. A fourth feature selection method was also used in which feature bits with above average mutual information [4] content were selected after the original features had been transformed from the real-space to the base-2 space.

This is useful to compare with the bit weights obtained by KEPFA. For KEPFA, features with above average weights were selected. Selected features from all methods, both in real-space and in the base-2 space, were also compared with all features, that is, the original *D* features. All evaluations were made with the KNN classifier using 1 nearest neighbour.

V. RESULTS

Figures 1a through 1c show the results obtained for test set 1. Figure 1a shows the average KNN classification performance for the 10 independent runs. In the classification, KEPFA selected the least number of feature bits at 9.14% of the total number of 151 bits, but still achieved 100% classification accuracy. The method using mutual information used 29.14% of the total number of bits to achieve the same performance. All the other classical methods used 5% of the total 20 features and achieved a classification performance of 77.5%, 97.5% and 77.5% for SFS, SBS and FSS, respectively. By using all features for classification, a 100% performance was achieved. Figures 1b and 1c show the percentage of original features and bits selected by the different methods respectively.

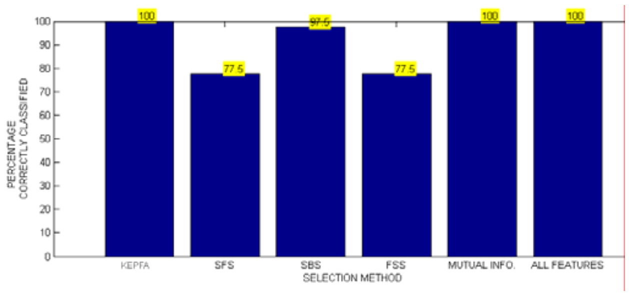


Fig. 1 Classification accuracy using feature subsets selected by different methods, and all available features

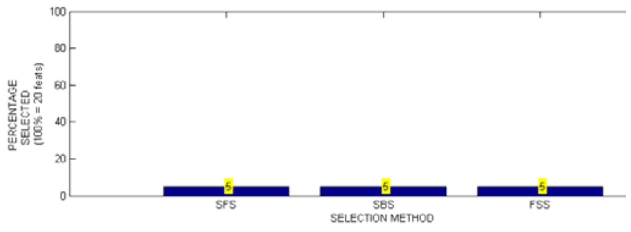


Fig. 2 Percentage of feature subsets selected in the original space using three classical methods

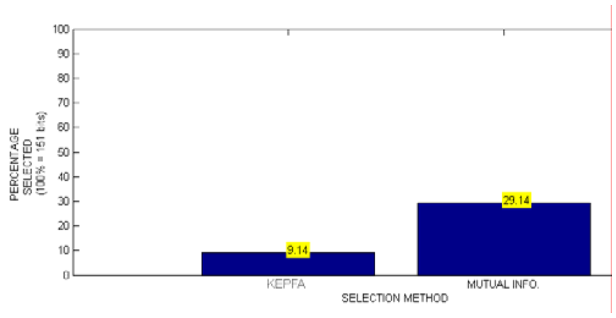


Fig. 3 Percentage of feature subsets selected in the transformed space using three classical methods

Genetic algorithms are often deemed to be slow to find solutions for certain problems which can be found more quickly by other methods. This, however, is expected to be less problematic with the KEPFA approach because the optimisation surface is discrete, since it is defined on probability mass functions. The evolution of the average fitness of the population, for each pattern tested and for each generation, is seen in figure 2a. Figure 2b shows the maximum fitness found in the population for each generation and each pattern tested.

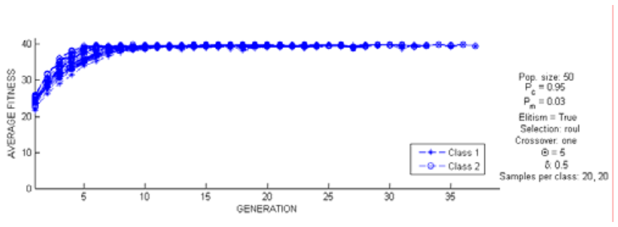


Fig. 4 Average population fitness

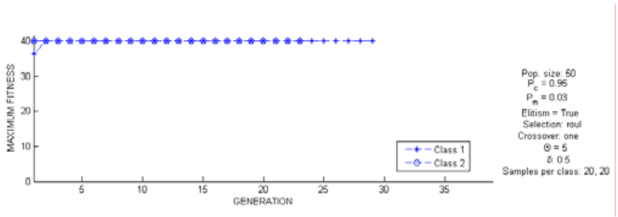


Fig. 5 Maximum fitness achieved in population

The mutual information content of each feature bit calculated over all available patterns and pattern classes is shown in figure 3a. The weighting of each feature bit given by the KEPFA algorithm is seen in figure 3b. It can be seen that feature bits selected by KEPFA are among the feature bits with high mutual information. The difference, however, is the relative importance of the bits. This suggests that KEPFA is also able to find non-linear relations among features and weighs these accordingly to maximise the posterior probability.

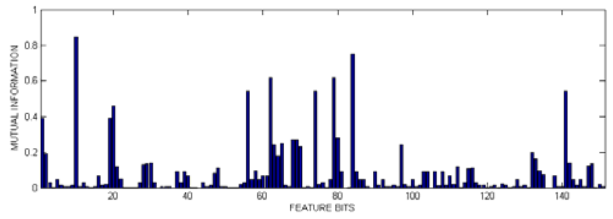


Fig. 6 Mutual information measure for features in the transformed space

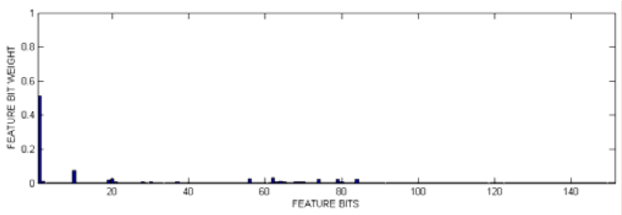


Fig. 7 Feature weights in the transformed space

For the second test set, I made the problem slightly harder by using all available patterns for the two classes used in test set 1.

Figure 4a shows the KNN classification, as for test set 1, for each of the different methods. The percentage of features selected is shown in figures 4b and 4c. This time, the KEPFA algorithm achieved 99.0% accuracy, 1% lower performance than for test set 1, but using only 7.1% of the total 155 feature bits. By contrast, the SFS, SBS and FSS still selected 5% of the total 20 features but achieved only 93.0%, 92.0% and 93.0% classification performance, respectively. The feature selection based on mutual information used 19.35% of the total 155 feature bits and achieved 98% accuracy. By using all features, 99% classification performance was achieved, the same as KEPFA.

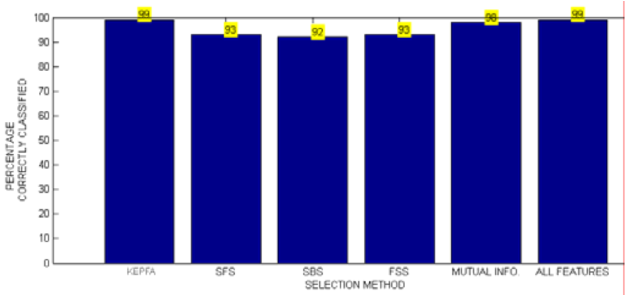


Fig. 8 Classification accuracy using feature subsets selected by different methods, and all available features for test set 2

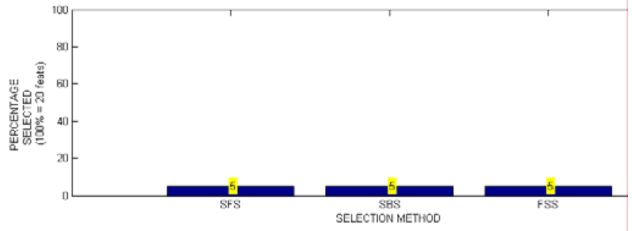


Fig. 9 Percentage of feature subsets selected in the original space using three classical methods for test set 2

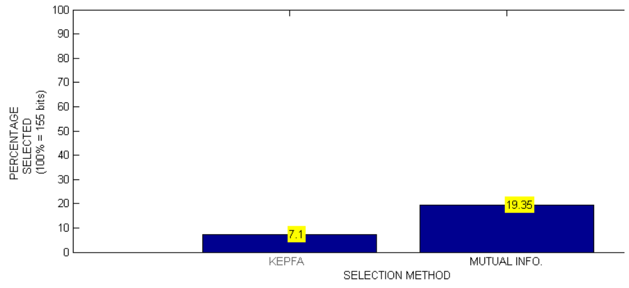
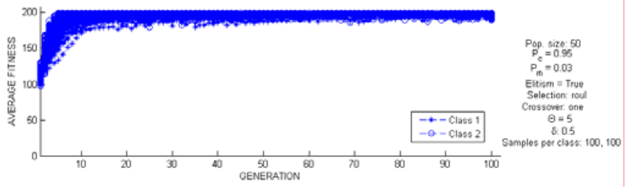


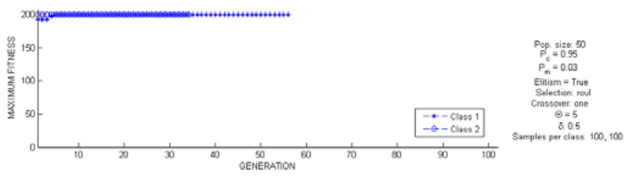
Fig. 10 Percentage of feature subsets selected in the transformed space using three classical methods for test set 2

The convergence of the algorithm for each test pattern is seen in figures 5a and 5b. The number of generations needed to reach a stable state was, similar to test set 1, fairly low. It appears that KEPFA has not converged prematurely, since the performance of the selected features is identical to some of the other methods, or even better.



Pop. size: 50
$P_c = 0.95$
$P_m = 0.03$
Elitism = True
Selection: roul
Crossover: one
$\Theta = 5$
δ: 0.5
Samples per class: 100, 100

Fig. 11 Average population fitness for test set 2



Pop. size: 50
$P_c = 0.95$
$P_m = 0.03$
Elitism = True
Selection: roul
Crossover: one
$\Theta = 5$
δ: 0.5
Samples per class: 100, 100

Fig. 12 Maximum fitness achieved in population for test set 2

The feature bit mutual information and weights obtained by KEPFA are seen in figure 6a and 6b. It can be seen that the weighted feature bits are among the bits with high mutual information, as for test set 1.

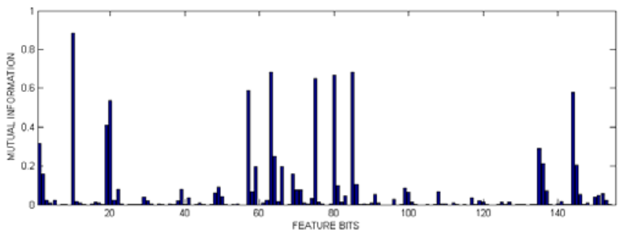


Fig. 13 Mutual information measure for features in the transformed space for test set 2

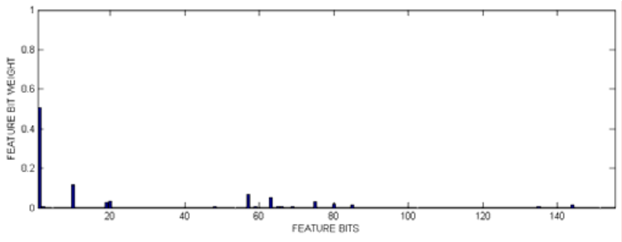


Fig. 14 Feature weights in the transformed space for test set 2

VI. Analysis and Discussion

The results presented show that the KEPFA algorithm is capable of finding important weights of bits when real-valued feature spaces are transformed into binary spaces. It should appear slightly paradoxical to map the original feature space into a larger space as the main objective is to reduce the feature space size. However, as the mapping is from a continuous space into a discrete space, the search space is effectively reduced, also reducing the effect of the curse of dimensionality. This, however, allows the identification of small subspaces in base-2 space that are highly concentrated in information, but non-linearly related to the original real space. The information contained in the real space is theoretically the same as that in the base-2 space, except of course for the loss due to quantisation which can sometimes be regarded as an error term.

The selection of bits based on the extracted weights was evaluated with the KNN classifier. Comparison with other feature selection algorithms showed that the performance of KEPFA was better for both test sets. It should also be mentioned that the three classical feature selection algorithms, SFS, SBS and FSS, were all evaluating feature subsets using the KNN classifier in their selection processes. The final evaluation of the selected features was, hence, biased towards these algorithms, as the final evaluation was carried out using the KNN classifier. KEPFA was, therefore, in a less advantaged position, as the features selected by KEPFA and the final evaluation were made with completely different algorithms.

The evolution of potential solutions shows that KEPFA is capable of carrying out a consistent search and weighting of important feature bits within the pattern space. Furthermore, as the evolution figures show, KEPFA converges after a reasonably low number of generations, indicating its speed and effectiveness. An important aspect of the proposed method is the high level of data compression achievable, since large numbers of features can be expressed in only small subsets of bits. This is a highly desirable aspect for embedded and real-time systems. The compression achieved is expected to be significant though data dependent. Moreover, a further important aspect of the proposed method is that the search spaces for the SFS, SBS and FSS algorithms grow exponentially with the size of the feature space. This limits their application for large spaces due to time constraints. However, the search space of KEPFA only grows linearly allowing search in large spaces.

Acknowledgment

Special thanks to Dr. N. D. Smith for providing useful comments regarding this paper.

References

[1] J. Austin, "RAM-Based Neural Networks, A Short History". In: James Austin (Ed), RAM-Based Neural Networks, Progress in Neural Computing, Vol. 9, World Scientific, London, pp 3-17, 1998.
[2] J. Bala and K. A. De Jong." Using learning to facilitate the evolution of features for recognizing visual concepts". Evolutionary Computation 4(3): 297, 15p, 1996.
[3] B. Bhanu and Y. Lin." Genetic algorithm based feature selection for target detection in SAR images", Image and Vision Computing, Vol. 21, No. 7, pp. 591-608, 2003.
[4] E. R. Jr. Cacciamani." Feature Extraction and Selection using n-tuple logics for recognition of hand-printed alphanumeric characters". PhD thesis, Faculty of the Graduate School of Engineering and Architecture, The Catholic University of America, 1972.
[5] S. Cost and S. Salzberg." A weighted nearest neighbour algorithm for learning with symbolic features". Machine Learning 10(1): 57-78, 1993.
[6] D. Goldberg. "Genetic Algorithms in Search, Optimization and Machine Learning", Addison-Wesley Longman Publishing Co., Inc., 1989
[7] E. Kalapanidas and N. Avouris. "Feature Selection for Air Quality Forecasting: a Genetic Algorithm Approach", AI Communications Journal, 16, (4), pp. 235 – 251, 2003.
[8] J. Kelly and L. Davis. "Hybridizing the Genetic Algorithm and the K Nearest Neighbors Classification AIgorithm". Proceedings of the d th International Conference on Genetic Algorithms and their Applications, 1991.
[9] L. I. Kuncheva and L. C. Jain." Designing classifier fusion systems by genetic algorithms". IEEE Transactions On Evolutionary Computation 4(4): 327-336, 2000.
[10] A. K. Jain and D. Zongker. "Feature Selection: Evaluation, application and small sample performance", IEEE Trans. Pattern Anal. Machine Intell., Vol 19, pp.153-158, 1997.
[11] Z. Michaelwicz. "Genetic algorithms + data structures = evolution programs". (2nd, extended ed.), Springer-Verlag New York, Inc., 1994
[12] W. F. Punch, E. D. Goodman, M. Pei, L. Chia-Shun, P. Hovland, R. Enbody. "Further research on feature selection and classi cation using genetic algorithms". In Proceedings of the Fifth International Conference on Genetic Algorithms, pages 557-564, Palo Alto, CA, USA, 1993.
[13] M. L. Raymer, W. F. Punch, E. D. Goodman, L. A. Kuhn, A. K. Jain. "Dimensionality reduction using genetic algorithms". IEEE Transactions On Evolutionary Computation 4(2): 164-171, 2000.
[14] W. Siedlecki and J. Sklansky. "A Note On Genetic Algorithms For Large-Scale Feature-Selection". Pattern Recognition Letters 10(5): 335-347, 1989.
[15] Z. H. Sun, G. Bebis, R. Miller. "Object detection using feature subset selection", Pattern Recognition 37(11): 2165-2176, 2004.
[16] F. E. H. Tay and L. J. Cao. "A comparative study of saliency analysis and genetic algorithm for feature selection in support vector machines". Intelligent Data Analysis 5(3): 191, 19p, 2001.

[17] S. Theodoridis and K. Koutroumbas."Pattern Recognition". Second Edition, 2003.
[18] J. Yang and V. G. Honavar. "Feature Subset Selection Using a Genetic Algorithm". IEEE Intelligent Systems 13(2): 44-49, 1998.
[19] S. Khola, "System and Method for Information Retrieval", Patent GB1100947.9, Patent Pending, 2011.
[20] S. Khola, "Genetic Evolution using Weighted Sub-spaces and Weightless Neurons for Pattern Recognition", PhD Thesis, University Kent, Canterbury, United Kingdom, 2006.
[21] N. D. Smith, S. Khola and K. Sirlantzis. "Classifier Selection using a Weightless Neuron-Based Genetic Algorithm". Proceedings of the 6th International Conference on Recent Advances in Soft Computing (RASC2006), IEE, ENNS, EUSPLAT, IAPR, Canterbury, Kent, UK. July 10-12, 2006 .
[22] MFEAT, http://www.ics.uci.edu/~mlearn/MLSummary.html Dataset by: Robert P.W. Duin, Department of Applied Physics, Delft University of Technology P.O. Box 5046, 2600 GA Delft, The Netherlands.Email: duin@ph.tn.tudelft.nl http : //www.ph.tn.tudelft.nl/~duin [1] tel +31 15 2786143.

Dr. Serkawt Khola is the Managing Director and Head of R&D at EvoPlexus, Copenhagen, Denmark. He is also a Lecturer at the Computer Science Department, University of Soran, Soran; and a Lecturer at the University of Salahaddin, Erbil (Hawler), Kurdistan Regional Government, Iraq, where he mainly teaches the subjects of Pattern Recognition and Machine Learning, and Artificial Intelligence to Postgraduate and Undergraduate students, besides supervising research projects in these fields. He has been a member of The Institution of Engineering and Technology (IET) since 2001. (e-mail: s.khola@evoplexus.com, s.khola@theiet.org).