A Data Mining Approach in the Analysis of the Weight Space of Multilayer Perceptron Solving Complex Real World Tasks

S. Adam

Dept. Mathematics, University of Patras Artificial Intelligence Research Center (UPAIRC), GR-26110 Patras, Greece and TEI Hpeirou, Arta, Greece

D.A. Karras

Dept. Automation , Chalkis Institute of Technology, Psachna, Evoia GR-34400 and Hellenic Open University, Greece, dakarras@{ieee.org, teihal.gr, usa.net}

M.N Vrahatis

Dept. Mathematics, University of Patras Artificial Intelligence Research Center (UPAIRC), University of Patras, GR-26110 Patras, Greece

Abstract: One of the main reasons for the slow convergence and the suboptimal generalization results of MLP (Multilayer Perceptrons) based on gradient descent training is the lack of a proper initialization of the weights to be adjusted. Even sophisticated learning procedures are not able to compensate for bad initial values of weights, while good initial guess leads to fast convergence and or better generalization capability even with simple gradient-based error minimization techniques. Although initial weight space in MLPs seems so critical there is no study so far of its properties with regards to which regions lead to solutions or failures concerning generalization and convergence in real world problems. There exist only some preliminary studies for toy problems, like XOR. A data mining approach, based on Self Organizing Feature Maps (SOM), is involved in this paper to demonstrate that a complete analysis of the MLP weight space is possible even in the case of complex real world problems. This is the main novelty of this paper. The conclusions drawn from this novel application of SOM algorithm in MLP analysis extend significantly previous preliminary results in the literature. MLP initialization procedures are overviewed along with all conclusions so far drawn in the literature and an extensive experimental study on more representative tasks, using our data mining approach, reveals important initial weight space properties of MLPs, extending previous knowledge and literature results.

Keywords*:* MLP initialization, gradient descent training algorithms, MLP convergence, MLP generalization, Clustering, SOM, data mining

1. Introduction

Weight training in Multilayer Perceptrons (MLPs) is generally formulated as the minimization of an error function, such as the mean square error between the target and actual outputs averaged over all training examples, by iteratively adjusting the connection weights. Most training algorithms, such as back propagation (BP) and conjugate gradient algorithms [9] are based on gradient descent. There have been many successful applications of MLPs trained with gradient descent algorithms in various areas [1], [9], but these MLPs present drawbacks [1], [9], due to their often getting trapped in local minima of the error function and being incapable of finding a global minimum if the error function is multimodal/non-differentiable. A detailed

review of BP and other learning algorithms based on gradient descent can be found in [9]. One of the main factors having impact in the results achieved by MLPs trained with gradient descent procedures, regarding both convergence speed and generalization capability, has been identified to be initialization of weights [1], [9]. In this paper we revisit the problem of weight initialization for neural networks trained with gradient descent based procedures. We verify, experimentally, a number of results reported by several researchers for the XOR-network and we extend these results to a well known problem, the IRIS classification problem. Our approach is based on clustering of the weight vectors after having trained an MLP with the BP procedure. Classification of the weight vectors into clusters is performed using unsupervised clustering of Kohonen's self organizing feature maps, or simply self-organizing maps (SOM). Results of our experiments not only reveal, as it was expected, the basins of attraction for the gradient descent learning algorithm, but also provide significant evidence that no inherent clustering exists for the initial weight space.

2. Problem Statement and Previous Work

Sequential or incremental mode of gradient descent makes the search in the weight space stochastic by nature [9]. Thus, BP training suffers from been very sensitive to initial conditions. In general terms, the choice of the initial weight vector $\boldsymbol{w_0}$ may speed convergence of the learning process towards a global or a local minimum if it happens to be located within the attraction basin of that minimum. Conversely, if $\boldsymbol{w_0}$ starts the search in a relatively flat region of the error surface it will slow down adaptation of the connection weights. Research work of Kolen and Pollack [13] revealed regions of high sensitivity in the weight space, so that, for two very close initial points BP can lead to significantly different trajectories in the weight space resulting in different learning curves. These results provide an alternative justification for the sensitivity of BP procedure to initial weights, along with other learning parameters.

Since the initial formulation of the error BP learning rule, Rumelhart et al. [18] reported weight initialization as a problem of symmetry breaking. They counteracted this problem by choosing small random weights to start with. However this has not proven to be the best strategy for many problems. If the synaptic weights are assigned small values the BP procedure may operate on a very flat area around the origin of the error surface [9]. On the other hand large initial values of the synaptic weights are very likely to drive network's neurons into saturation as reported by Hush et al [10] and Lee et al [15]. It follows that, the proper choice of initialization lies somewhere between these two extreme cases.

In order to avoid the problem of premature saturation Wessels and Barnard [24] state that a good strategy for choosing the magnitudes of the initial connection weights is to start with a small random weighted sum for an arbitrary unit. This can be achieved by setting the initial weights of a unit namely i, to be on the order of $1/\sqrt{f_i}$ where f_i is the number of inputs for unit i.

Fahlman [6] performed studies about random weight initialization techniques for multilayer neural networks. He proposed the use of a uniform distribution over the

interval [-1.0, 1.0], but experimental results showed that the best initialization interval to the problems he dealt with varied in ranges between [-0.5, 0.5] and [-4.0, 4.0].

Sensitivity of BP to initial weights, as well as to other learning parameters, was studied by Kolen and Pollack [13]. Using Monte Carlo simulations on feed forward networks trained with BP to learn the XOR function they discovered that convergence of these networks exhibits a complex fractal-like structure as a function of initial weights.

According to LeCun [14], a good strategy for selecting initial weights is to assume that they are drawn from a uniform distribution with zero mean and a variance equal to the reciprocal of the number of connections of a neuron.

Following an extensive experimental study Schmidhuber and Hochreiter [19] concluded that repeating random initialization, that is, "guessing" the weights, many times results in the fastest way to convergence.

Kim and Ra [11] calculated a lower bound for the initial length of the weight vector of a neuron to be $\sqrt{a/d_{in}}$, where α is the learning rate and din is the neuron's fan-in.

Boers and Kuiper [2] initialize the weights using a uniform distribution over the interval $[-3/\sqrt{d_{in}}, +3/\sqrt{d_{in}}]$ without any mathematical justification.

A simple modification of the widely used random initialization process was proposed by Nguyen and Widrow [17]. The weights connecting the output units to the hidden units are initialized with small random values over the interval [-0.5, 0.5]. The initial weights at the first layer are designed to improve the learning capabilities of the hidden units. Using a scale factor, $\beta = 0.7(q)^{1/p}$, where q is the number of hidden units and p is the number of inputs, the weights are randomly initialized and then scaled by $v = \beta \frac{v}{\|v\|}$ where v is the first layer weight vector.

Random initialization of connection weights seems to be the most widely used method. A number of approaches such as those presented previously provide substantial improvements in BP convergence speed and avoidance of bad local minima. Thimm and Fiesler [21] have compared several random initialization schemes by using a very large number of computer experiments. It appeared that the best initial weight variance is determined by the dataset, but differences for small deviations are not significant and weights in the range ± 0.77 seem to give the best mean performance.

Some authors propose initialization of weights based on clustering techniques. Such initialization methods seem to be more appropriate for use in classification problems. Denoeux and Langelle [4] propose the initialization of the hidden unit weight vectors with normalized vectors selected randomly from the training set. Other approaches within this direction include work done by Weymaere and Martens [25] as well as work of Duch, Adamczak, and Jankowski [5]. Such clustering methods first pre-process the given training vectors to lie into the unit hyper-sphere and, afterwards, following dendrograms or other clustering methods, they find the mean of the normalized data clusters. In the sequel they choose initial connection weights to equal to the centres of these clusters. These research efforts assume a direct mapping

between normalized clusters of input data and clusters of clusters of MLP initial weight space.
Finally, based on the capability of Genetic Algorithms to perform sampling on the entire search space, de Castro et al. [3] propose an Evolutionary Approach to weight initialization. Actually, this capability of GA to perform global exploration of the initial weight space is used to find the best way of initializing the weights. It is thus used to complement the local search performed by the training algorithm, BP, conjugate gradient or other gradient descent procedure.

3. Revisiting the Problem of Effective Weight Initialization – Analysis of the Weight Space

Our approach consists in performing analysis of the weight space after having trained an MLP with the BP procedure for a significant number of weight vectors and for various different sets of training patterns. This approach has already been used by other researchers in the XOR problem, but what is new here is its application to a well known real life problem, the IRIS classification problem. Analysis of the weight space is done using a data clustering and visualization technique. We consider that this approach extends results obtained previously by other researchers. Main considerations of these previous researches are presented hereafter.
Kolen and Pollak [13], stressed the importance of a good choice for the initial set of weights. They showed the existence of chaotic behaviour of the learning dynamics for back propagation due to initial conditions and especially weight initialization. They, also, argued that, when more than one hidden unit is utilized, or when the environment has internal symmetry or is very unconstrained, then there will be multiple attractors.
Similar results were reported by Lisboa and Perantonis [16]. These researchers provide an analytic solution to learning the XOR problem by a neural network illustrating how local minima of the cost function for some training tasks in multilayered networks can be revealed by analysis. Their work also deals with weight initialization and reports a number of results regarding dependence of BP convergence on weight initialization of the neural system. It seems that no general rule can be derived regarding the structure or specific features of the initial weights.
Hamey [7] reconsiders the XOR problem and provides a theoretical study of the error surface for the standard mean square error function. However, he notes the difficulty of having analytic solutions for the general pattern classification case as the study of the error surface is hampered by high dimensionality and because of the difficulty of theoretical analysis.

In light of these results it seems that it is not possible, in general, to provide complete theoretical verification for a number of research results claiming to cope effectively with the problem of weight initialization. This is, partially, due to the fact that an exhaustive study of the error surface and of the learning dynamics is almost unfeasible for the general case of the pattern classification problem. On the other hand it is tempting to examine if the initial weight space possesses some kind of structure

or if it is able to reveal features which may lead to an effective choice of initial weights. To this end, an effective means seems to be the analysis of the weight space of MLPs in different pattern classification problems. This also permits to gain significant evidence on the validity of different results having either a theoretical basis or proven by experiments.
From another point of view, that one of data mining in the weight space of an MLP, this is similar to performing a survey in order to gain insight into the data and determine whether these data are sufficient, by themselves, to justify existing research results on weight initialization. Essential tools for such a survey are data summaries and data visualization. We are interested here in forming clusters of data, since they uncover important characteristics of the input data.

4. Kohonen's Self Organizing Feature Maps, as a Data Mining Tool for the Analysis

Data clustering and visualization of the clusters, in this paper is based on Kohonen's SOM. The SOM is a type of neural network which is based on unsupervised learning. Thus, unlike supervised learning methods, a SOM is able to perform clustering of data without any reference to the class membership of the input data.
Usually, a SOM consists of a regular, one or two-dimensional grid of neurons. Each node on the grid corresponds to a neuron of the SOM and is represented by a weight vector, called a model vector of dimension n, where n is the dimension of the input space. The set of weight vectors is called a codebook. For each unit of the map a number of adjacent neurons are defined and connected to it, according to a neighborhood relation, which defines the topology of the map (rectangular or hexagonal).
Training the map is an iterative process. At each step a sample vector x is randomly chosen from the input data set and distances between x and all the codebook vectors are computed. Distances between codebook vectors and sample data correspond to similarities between input data and units of the SOM. The best matching unit (BMU), i.e. the most similar unit, is the map unit whose weight vector is closest to x. The training algorithm updates the weight vector of the BMU and of those of its neighborhood so as to get these units move closer to the input vector x, i.e. diminish their distance to the sample vector [22]. More details on SOM can be found in [12].
The SOM algorithm performs a mapping from the high dimensional input space onto map units. This mapping preserves topology, in the sense that, relative distances between data points in the input space are preserved by distances between map units. This means that data points lying near each other in the input space will be mapped onto neighboring map units. The SOM can thus serve as a clustering tool of high dimensional data. Compared to standard techniques (k-means, ISODATA, competitive learning etc) SOM not only performs better in terms of effectively clustering input data to unknown clusters but also it is computationally more effective [20], [23]. Other comparisons and studies on the data mining capabilities of SOM can be found in the literature.
We should mention here the use of the SOM Toolbox for SOM training, data visualization, validation and interpretation. SOM Toolbox was developed at Helsinki University of Technology [26]. It is a software package comporting Matlab scripts for

basic initialization, training and validation algorithms of the SOM. SOM Toolbox, also, offers a number of functions for ease and effective SOM visualization.

5. Analyzing the Weight Space for MLPs Trained with BP

We considered two classical benchmarks, the XOR function and the Iris classification problem. The XOR function was studied with a 2-2-1 network while the IRIS classification problem was investigated with two different network architectures, one with 4-10-3 units and another one with 4-5-3 units. For all units the logistic sigmoid was used as an activation function. Experiments for both problems and for different network architectures were carried out according to the following steps:

1. MLPs were trained with the BP learning algorithm and more specifically with the on-line gradient descent procedure provided by Matlab. All experiments were carried out with the same training parameters, that is interval for initial weights [-2.0, 2.0], learning rate 0.9, max number of epochs 30000 and error between target and actual network output less than 0.01.

2. A relatively large number of weight vectors, that is 5000, were chosen from the initial weight space. Weight vectors were randomly sampled in the interval [-2.0, +2.0] using uniform distribution. After training, the set of weight vectors was roughly divided into two distinct subsets, or categories, of weight vectors. One subset was made up from, those weight vectors for which both, training succeeded (the error goal was reached), and generalization performance was good, i.e. less than 20% of previously unseen patterns rejected per class. These vectors are called the ***successful*** weight vectors while those not meeting the above criteria are called the ***failed*** weight vectors and they fall within the second category.

3. For each weight vector w_i^0 considered before training, the MLP was trained with the on-line gradient descent and a weight vector w_i^* after training was obtained. Thus, gradient descent is considered mapping the weight space before training *W* onto the weight space after training *W'*. Given the high dimensionality of these spaces we then used SOMs and projected each one of them on the 2-dimensional space. This approach is graphically depicted in Figure 1.

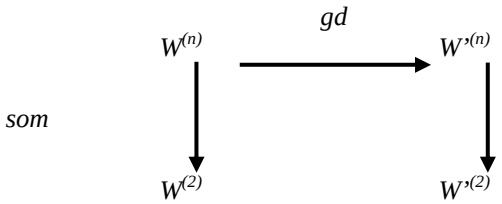


Figure 1.

4. The 2-dimensional projections of *W* and *W'* thus obtained presented the clusters of weight vectors being discovered by the SOM. Visual inspection of the map representing *W'* permitted to draw some interesting qualitative information regarding the basins of attraction for the gradient descent procedure. Activation of the SOM units and visualization of the unified distance matrix (UM) to identify classification of weight vectors into different clusters. Details on these results are presented in the following section.

5. We, finally, used the possibility offered by the SOM Toolbox to identify the weight vectors for which a unit of the SOM is activated to verify density of *W* regarding convergence and generalization. Actually, given a SOM node in a cluster of successful weight vectors we identified one weight vector before training w_i^0 that gave after training a successful weight vector w_i^*. By injecting additive noise, with normal distribution $N(0,\sigma^2)$, on w_i^0, we took a number of weight vectors in the vicinity of w_i^0. Retraining the MLP with the same BP procedure and mapping the weight vectors after training on the SOM we discovered that even for very small variance many of the noisy weight vectors did not behave the same way as w_i^0.

6. Main Results and Discussion

The tool for presenting results and analyzing them is the unified distance matrix (UM). UM represents the organization of the SOM units into groups, as uniform areas on the 2-dimensional grid. These areas correspond to neighboring units that demonstrate similar activation when receiving the same input weight vectors. An example is given in Figure 2.

Result 1. For weight vectors before training the UM did not reveal any particular area and the whole map gives a roughly uniform activity for the units, which is absolutely right given the distribution for the choice of the weight vectors before training; see Figure 3.

Result 2. Clustering of the weight vectors after training, which is performed by the SOM without any class membership information, depicts uniform regions of unit activity corresponding to ***successful*** weight vectors and thin borderline areas for the ***failed*** weight vectors. Actually, the UM demonstrates that, the units, activated for ***successful*** weight vectors, are those associated with uniform areas, that is with clusters, while ***failed*** weight vectors mainly activate units lying between clusters. This is particularly true for the IRIS classification problem and for both network architectures tested, the one using 10 nodes in the hidden layer, Figure 4, a), and b), and the other using 5 nodes in the hidden layer, Figure 4, c) and d).

SOM displays, using the UM, multiple regions for the ***successful*** weight vectors. In accordance to what has been advanced by Kolen and Pollack [13] these regions can be attributed to different basins of attraction for the BP algorithm. This hypothesis still stands if one considers the number of regions corresponding to ***successful*** weight vectors in the case of 4-5-3 network, also used for the IRIS problem Figure 4, c) and d). In this case the number of "***successful weight regions***" is less than the corresponding number in the case of the 4-10-3 network for the IRIS problem.
The above are, also, valid for the XOR function experiments, as presented in Figure 5 a), b). What is worth underlining here is that Figures 5, a) and b) are somewhat complementary, in the sense that Figure 5, a) presents clusters for ***successful*** weight vectors while clusters not affected by these vectors are those activated in Figure 5, b) by the ***failed*** weight vectors.
This is not surprising considering that the SOM does not distinguish between clusters belonging to one class or to the other. It just performs clustering and it is the observer's job to distinguish class membership for individual clusters, Vesanto et al, [23]. Given that the number of ***failed*** weight vectors is 1110, more than 20% of the total number of vectors used, SOM forms clusters for these vectors too and does not arrange them in regions between clusters. Hence, clusters for both successful and failed weight vectors coexist in the same SOM. On the other hand, considering Figures 6, a) and b), one will notice that assignment of weight vectors in clusters is similar to the case of IRIS problem. Actually, these Figures present mapping of the behavior of a XOR network with linear activation function for the output node. This network has significantly fewer ***failed*** weight vectors (111 for our experiment) than the one using logistic sigmoid activation function. The important question in the case of XOR problem is whether clusters of ***failed*** weight vectors correspond to "strange attractors". If the answer is positive a straightforward conclusion is that replacing the non linear activation function with the linear one permitted gradient descent to bypass local minima.
The clusters formed by the SOM correspond to the various minima reached by the gradient descent throughout each experiment. These minima can be global or local as shown by the XOR experiment. In this sense and together with the topology preservation mapping of the SOM it is straightforward to assume that clusters indicated basins of attraction for the dynamics of the learning procedure. As Kolen and Pollack argued there is a dependence of the number of attractors and the number of units in the hidden layer of the network. This is an experimental confirmation that as the number of units increases in the hidden layer the number of basins of attraction increases and therefore the study of the weight space becomes more difficult, see Kolen and Pollack [13].

Result 3. Execution of step 5, described above, for a number of different values of σ^2 demonstrated that even for very small variance many of the noisy weight vectors did not behave the same way as the initial vector $\boldsymbol{w}_i^0$, i.e. they did not result in successful training.

This experiment was done for the following values of σ^2; {0.0001, 0.0025, 0.005, 0.0075, 0.001, 0.0025, 0.005, 0.0075, 0.01, 0.025, 0.05, 0.075, 0.1, 0.25, 0.5, 0.75, 1.0}. For each trial a number of 100 'noisy' weight vectors were produced, and the MLP was retrained. After training the weight vectors for these 100 'noisy' initial vectors were mapped using the existing SOM. The mapping demonstrated that for very small values of σ^2 (approximately less than 0.025) there was no significant change of the behaviour of the 'noisy' weight vectors compared to the initial weight vector. However, for values of σ^2 greater than or equal to 0.025 not all the 'noisy' weight vectors result in successful training. Several between them fail in terms of convergence of the gradient descent and generalization; see Figures 7 a), b), c) and d). This result not only confirms what was reported by Kolen and Pollack [13], but also extends assumptions made by these researchers for the XOR problem to a real life problem such as IRIS.
It should be noted here that in all figures the size of the spots is proportional to the number of the weight vectors hitting the SOM unit.
Despite the fact that the above results are very important they are not of practical consequence. Thus we reconsidered rules of thump, proposed by other researchers as stated in section 'Previous work', above. In order to acquire a better idea on how to deal with these research results we proceeded in a number of experiments using the 4-10-3 MLP for the IRIS problem. During these experiments we used values for the synaptic weight randomly chosen from intervals [-α, +α], with α varying from -6.0 up to +6.0, by a step of 0.20. Though not exhaustive these experiments they cover only that class of research results concerning the size of the sampling interval for the initial weights. The aspect of taking into account the inherent structure of the pattern space, as claimed by other researchers, was consider by executing each experiment at least twice; once with a single set of training patterns for the whole set of initial weight vectors and once by shuffling the input space and training groups of weight vectors with different sets of training patterns. Results of these experiments are stated hereafter.

Result 4. Training seems to be very sensitive to the choice of the training patterns. For the same interval of initial weight vectors and even the same weight vectors, learning curves and subsequent generalization of BP are clearly different.

However, during these trails we did not adopt some specific strategy on how to choose the training patterns and so it remains unclear what characteristic of the input space really biases the learning phase. A possible explanation relies on the inherent structure of the IRIS problem, where two classes are highly correlated. Finally, it seems that a good "strategy" to overcome this problem is to carry out training changing the set of training patterns every 50 or 100 initial weight vectors, these numbers chosen arbitrarily.

Result 5. Training tends to be more successful when the weight vectors are chosen in an interval $\left[-\alpha,+\alpha\right]$ with $\alpha \approx \sigma_p^2$, where σ_p^2 is the maximum standard deviation of the variables of the input pattern space.

While this result is in the same line with some previous research outcomes, it seems that it more accurately reflects a good strategy for weight initialization than previous similar results in the literature. This paper shows that it is not possible to be more specific in the weight initialization range than the above result. More experiments, however, are needed to establish such an outcome.

Result 6. While training seems to be more successful for values of the initial weights within some interval $\left[-\alpha,+\alpha\right]$ as described above, it is very likely for he BP to give a successful; learning curve for even greater values.

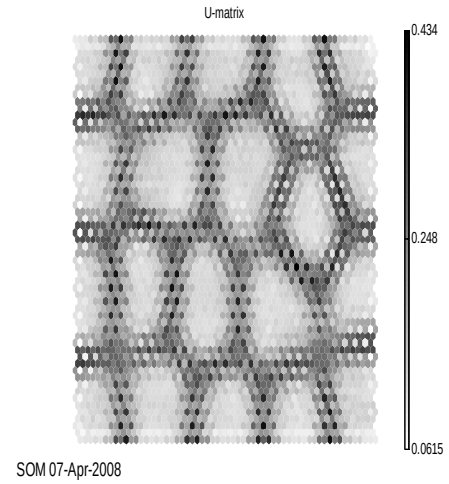


Figure 2. The unified distance matrix for weight vectors after training (Iris network)

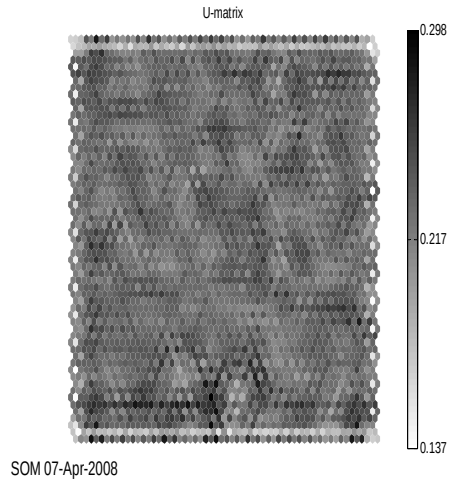
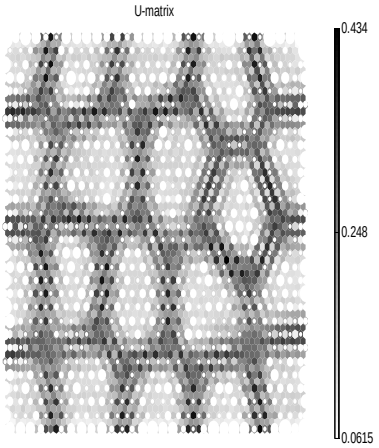
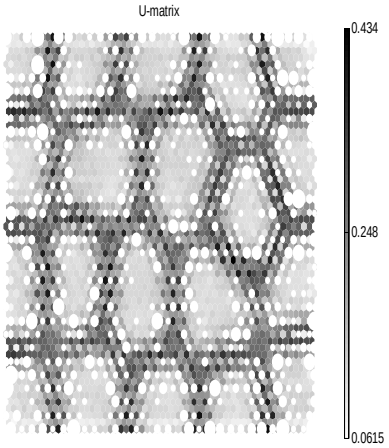


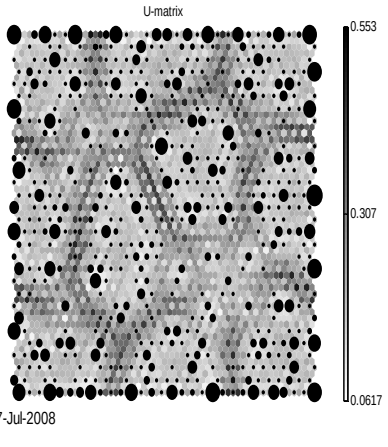
Figure 3. The unified distance matrix for the weight vectors before training (Iris network)



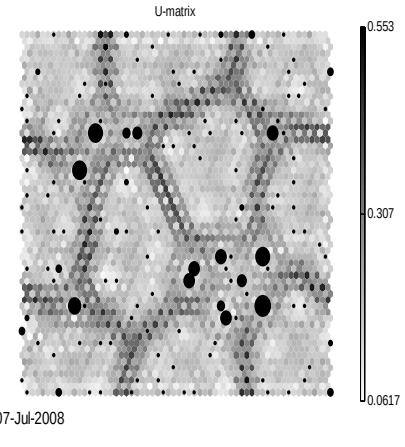
Mapping of weight vectors for the Iris network
Figure 4a). Mapping of the successful weight vectors for the 10 hidden units Iris network



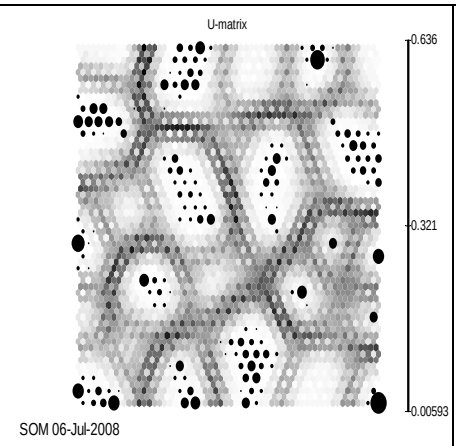
Mapping of weight vectors for the Iris network
Figure 4b) Mapping of the failed weight vectors for the 10 hidden units the Iris network



Mapping of weight vectors for the Iris network
Figure 4c) Mapping of the successful weight vectors for the 5 hidden units Iris network

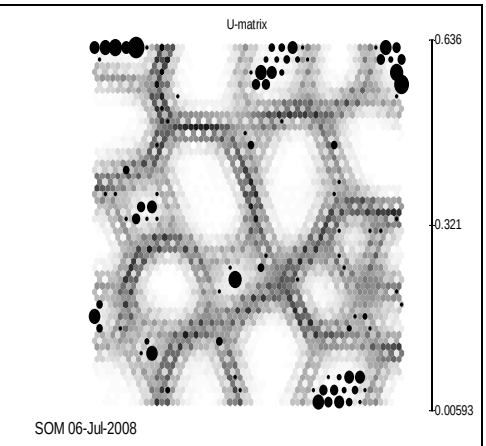


Mapping of weight vectors for the Iris network
Figure 4d) Mapping of the failed weight vectors for the 5 hidden units Iris network



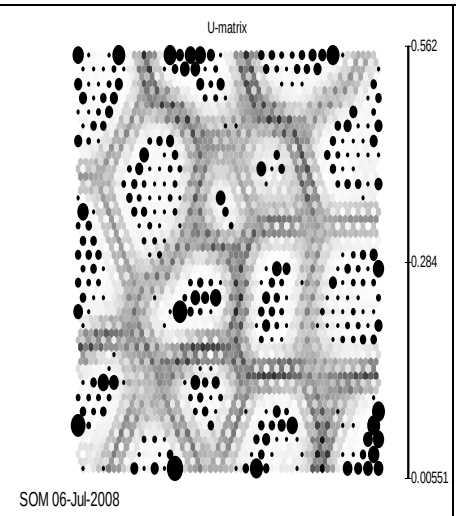
Mapping of weight vectors for the XOR network. Logistic sigmoid activation function used for the output node

Figure 5a) Mapping of the successful weight vectors for XOR network



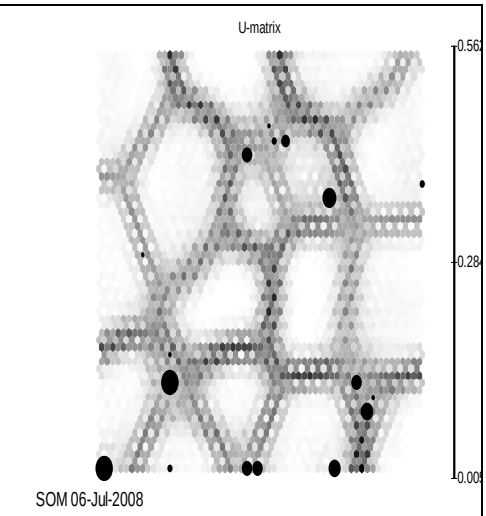
Mapping of weight vectors for the XOR network. Logistic sigmoid activation function used for the output node

Figure 5b) Mapping of the failed weight vectors for XOR network



Mapping of weight vectors for the XOR network. Linear activation function used for the output node

Figure 6a) Mapping of the successful weight vectors for XOR network



Mapping of weight vectors for the XOR network. Linear activation function used for the output node

Figure 6b) Mapping of the failed weight vectors for XOR network

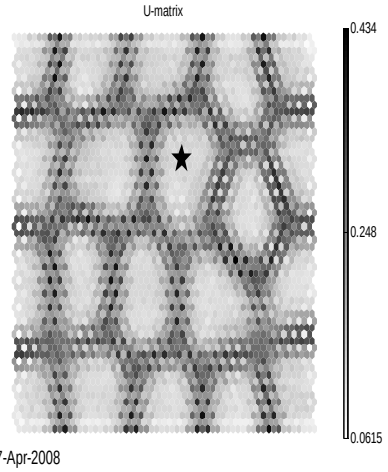


Figure 7a) Mapping of a successful weight vector for the IRIS network.

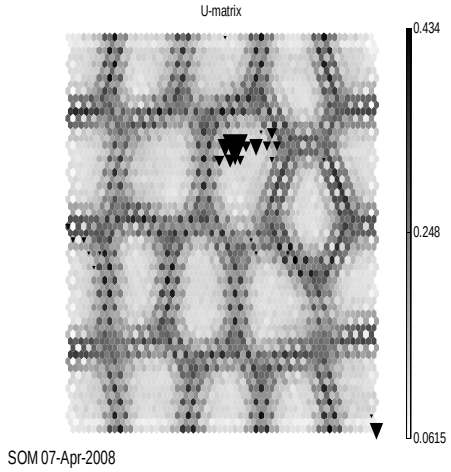


Figure 7b) Mapping of 'noisy' weight vector for the IRIS network. ($\sigma^2 = 0.005$)

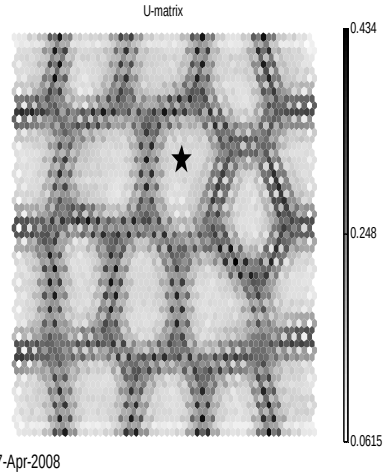


Figure 7c) Mapping of a successful weight vector for the IRIS network.

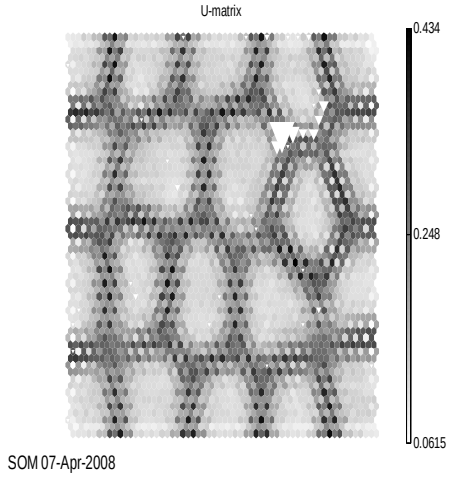


Figure 7d) Mapping of 'noisy' weight vector for the IRIS network. ($\sigma^2 = 0.05$)

7. Conclusions and Future Trends

This paper revisits MLP initialization problem in the case of BP training and extends literature results both in the description of the weight space as well as in the

estimation of a good strategy for selecting weight initialization range. The analysis is performed on a complex classification task, like Iris problem, which is more representative of "real" world problems characteristics than benchmarks used so far in the literature. To this end, a data mining approach, based on Self Organizing Feature Maps (SOM), is involved in this paper. The conclusions drawn from this novel application of SOM algorithm in MLP analysis extend significantly previous preliminary results in the literature. More detailed analysis on real world benchmarks is needed to establish better these results and more elaborate specification of the weight initialization range than the ones of results 5,6 in this study are needed not, however, too "accurate" as in previous studies. Previous studies have been misleading in this aspect not showing that the weight initialization space is not dense in solutions but it follows an almost fractal structure and, therefore, a probabilistic approach is more suitable in order to find out a good strategy for MLP weight initialization.

REFERENCES

[1] Bishop, C., Neural networks for pattern recognition, Clarendon Press, Oxford, 1995.

[2] Boers, E.G.W. & Kuiper, H. Biological Metaphors and the Design of Modular Artificial Neural Networks, Master Thesis, Leiden University, Netherlands, 1992.

[3] de Castro, L. N., Iyoda E. M., Von Zuben F. J., and Gudwin R., Feedforward Neural Network Initialization: an Evolutionary Approach, Proc. Vth Brazilian Symposiuon Neural Networks, 1998

[4] Denoeux. J. and Lengelle, R., Initializing back-propagation networks with prototypes, Neural Networks 6, 1993, pp. 351-363

[5] Duch, W., Adamczak, R., and Jankowski, N., Initialization and optimization of multilayered perceptrons, 3rd Conference on Neural Networks and Their Applications, Kule, Poland 1997.

[6] Fahlman, S.E. "An Empirical Study of Learning Speed in Back-Propagation Networks", Tech. Rep., CMU-CS-88-162, School of Computer Science, Carnegie Mellon University, Pittsburg, PA, September 1988.

[7] Hamey, L., Analysis of the Error Surface of the XOR Network with Two Hidden Units, Proc. 7th Australian Conf. Artificial Neural Networks, pp. 179-183, 1996.

[8] Hassoun, H. M., Fundamentals of Artificial Neural Networks, MIT Press, Cambridge Massachusetts, 1995,

[9] Haykin, S., Neural Networks, A Comprehensive Foundation, Prentice-Hall, 1999.

[10] Hush, D. R., Salas, J. M., and Horne, B., Error surfaces for multi-layer perceptrons, in IJCNN, Seattle, 1991, vol. I, pp. 750-764, IEEE.

[11] Kim, Y.K. & Ra, J.B. "Weight Value Initialization for Improving Training Speed in the Backpropagation Network", Proc. of the IEEE International Joint Conf. on Neural Networks, vol. 3, pp. 2396-2401, 1991.

[12] Kohonen, T., Self-Organization and Associative Memory, Springer-Verlag, 1989.

[13] Kolen, J. F., Pollack J. B., (1991), Back propagation is sensitive to initial conditions, in Advances in Neural Information Processing Systems 3, Denver.

[14] LeCun Y., (1993), Efficient Learning and Second-order Methods, A Tutorial at NIPS 93, Denver.

[15] Lee, Y., Oh, S. H., and Kim, M. W., The effect of initial weights on premature saturation in back-propagation learning, in IJCNN, Seattle, 1991, vol. I, pp. 765-770, IEEE.

[16] Lisboa, P. J. G., and Perantonis, S.J., Complete solution of the local minima in the XOR problem. Network: Computation in Neural Systems, 2, pp. 119–124, 1991.

[17] Nguyen, D. & Widrow, B. "Improving the Learning Speed of Two-Layer Neural Networks by Choosing Initial Values of the Adaptive Weights", in Proc. Int. Joint Conf. N. Net. (IJCNN), Ann Arbor, MI, vol. 3, pp. 21-26, 1990.

[18] Rumelhart, D. E., Hinton, G. E. and Williams, R. J. (1986), Learning internal representations by error propagation, Parallel Distributed Processing, MIT Press, chapter 8, pp. 318-362.

[19] Schmidhuber J., Hochreiter S., Guessing can out perform many long time lag algorithms, Technical Note, IDSIA-19-96.

[20] Snyder, W., Nissman, D. Van den Bout, D. and Bilbro G., Kohonen Networks and clustering: Comparative Performance in Color Clustering, Advances in Neural Information Processing, 1990
[21] Thimm, G. and Fiesler, E., High-Order and Multilayer Perceptron Initialization, IEEE Trans. on Neural Networks, vol. 8, n° 2, pp. 349-359, 1997.
[22] Olli Simula, O., Vesanto, J., Alhoniemi, E., and Hollman J, Analysis and Modeling of Complex Systems Using the Self-Organizing Map, Neuro-Fuzzy Techniques for Intelligent Information Systems, 1999.
[23] Vesanto, J., and Alhoniemi, E., Clustering of the Self-Organizing Map, IEEE Transactions on Neural Networks, vol. 11, no. 3, pp. 586-600, 2000.
[24] Wessels, L. F. A. and Barnard, E., Avoiding false local minima by proper initialization of connections, IEEE Transactions on Neural Networks, 5 (6), pp. 899-905.
[25] Weymaere, N. and Martens, J. P., On the initialization and optimization of multilayer perceptrons, Transactions on Neural Networks 5, 1994, pp. 738-751
[26] Technical Report on SOM Toolbox 2.0, Helsinki University of Technology, April 2000, http://www.cis.hut.fi/projects/somtoolbox/