Symmetry: between indecision and equality of choice.

E.I. Barakova and L. Spaanenburg,
Rijksuniversiteit Groningen, Dept. of Computing Science,
P.O.Box 800, 9700 AV Groningen (The Netherlands)

Abstract. *The training of a neural network is an intricate balance between knowledge, randomness and symmetry. Symmetry can both be beneficial and detrimental to the learning process by respectively equality of choice and indecision. The paper provides a critical review and classification, and offers a constructive procedure to handle problem–indigenous symmetries.*

1 Introduction.

Neural networks learn by extracting knowledge from examples. This can be boosted by inserting existing knowledge in the structure, but once training has started the network should be allowed to evolve under its own dynamics influenced only by the stream of examples. The difference between captured and ideal knowledge can be pictured as an error landscape. This landscape implies the route to be taken while learning: in the presence of hills going down–hill will lead to a better match.

Symmetry and randomness influence the itinerary. The symmetries in the landscape, signified by crests, valleys and flat–regions, presume equally probable itineraries. In other words symmetry gives an equal chance to move in several directions and therefore leads to indecisiveness. On the other hand, randomness is supposed to help the network escape from such a dilemma. It helps the learning algorithm to move away from the current spot in a single direction.

Symmetry can be dominant in the beginning of, but also at specific moments during, learning. Randomness (for instance as stochastic variable in the learning algorithm or as additional noise at the network input, output or parameters [2]) is then required to force the presentation of examples to follow alternative itineraries. When the amount of randomness is not sufficient to balance the effect of symmetry, learning will not be completed: instead of being adapted to ensure the right mapping between input/output data strings, the initial parameters will eventually become zero. If the noise (the randomness) of the system is dominant, learning will also be unsuccessful, because the network will rather learn the noise than the exemplified knowledge. The fundamental issue of learning is therefore the creation of a functional balance between symmetry and randomness directed by the examples (the knowledge).

Having a look at how the neural networks as mostly used in practice are constructed, it can be seen that randomness goes together with symmetry: in Hopfield and Kohonen networks, randomly initialized neurons are connected in a symmetrical lattice; in feedforward networks, permutationally symmetrical topologies are initialized with random parameters, uniformly distributed in a symmetrical range around the origin.

To facilitate an optimal strategy in handling symmetry and randomness in relation to captured knowledge, it is important to know more about them. Though randomness and knowledge presentation are subject of constant interest [2], [13], [19], symmetry has not been elaborated in a systematic manner.

Fig. 1 suggests that the symmetry can be categorized from two different perspectives, concerning either the place (the network or the problem) or the effect (the resulting relief of the error surface) of its appearance. The error landscape displays essentially two meanings to the word “symmetry”: either the scenery itself or the choice in travel direction can be symmetrical. Such different meanings come about when studying the landscape as an interaction between the network structure and the problem. The error landscape results from network topology and value settings, while the travel direction stems from the structure of the problem as presented to the network during learning in request from the learning algorithm.

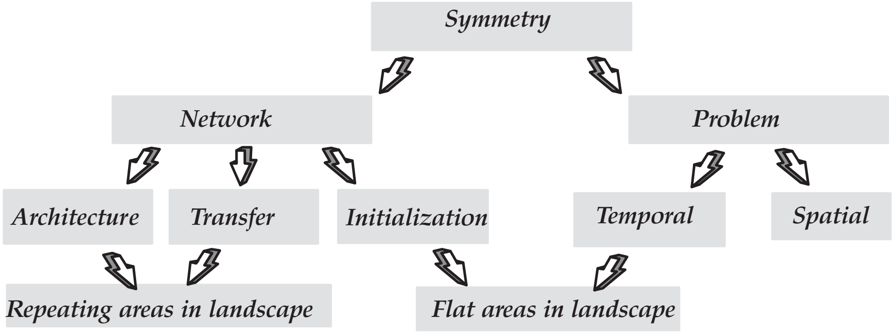

Figure 1: *The different faces of symmetry.*

Examined with an accentuating on the symmetry the interplay between the network (having as a design principles Symmetry and Randomness) and the problem (the Knowledge) is contradictory: it can be helpful when the 3 elements are in the right proportion and can chouse learning problems otherwise. After the critical review of the symmetry breaking attempts, the selective sampling strategy and algorithm will be suggested in the framework of the KRS discoursed model.

The different sides of the problem have been observed by authors with different backgrounds (physicist, computer scientist and engineer) leading to the 3 main theories. The replica method or mean field theory [20] and its alternative on–line learning [15] give the physical prospective, VC dimension [9] and graph theory are used by computer scientists, and Bayesian theory [6] is most close to engineering. Such different views will be cited wherever required.

2 Symmetries in the network.

The symmetries in the network result from the choice of network architecture and transfer function as well as from the specific settings of its initial values (weights and biases, scaling intervals of the example strings). While the former two are responsible for the appearance of repeating areas in the energy landscape, the latter causes flat areas.

First, the symmetry by network architecture and transfer function will be discussed. Because the architecture and transfer of the network are defining its structure, the symmetries which they are predefining can be called structural symmetries. Later on the symmetries caused by the initialization of the network will be summarized. As mentioned before here are not only meant the starting values of weights and biases but also the scaling intervals for the examples. Very often all the initialization intervals are symmetrical around the zero point, which affects the performance symmetry. But a more strong condition is that this intervals are very small, which positions the network initially in the flat area of the error surface [3]. In the next section it will be shown, that problem symmetries cause a similar effect.

2.1 Symmetries in the network structure.

To illustrate the discussion and terminology by application to a concrete network architecture the fully connected multilayer perceptron will be elaborated on. In MLP the structural symmetries are caused by the existence of the so–called *coherent transformations:* transformations, that do not change the network functionality as there are:

Permutation transformation. In the standard feedforward topology, neurons from the same layer are interchangeable (Fig. 2). It is possible to permute all the connections of a neuron (inputs and outputs) with those of another neuron in the same layer without changing the network function.

The following formula represents the response of a multilayer perceptron with one output. If two neurons from the same layer are interchanged, the final output will be the same, because the change affects only the ordering of the elements under the sum sign.

$$f(\boldsymbol{I},\boldsymbol{w}) = \varphi(\sum_{i} w_{oi}\varphi(\sum_{k} w_{ik}\varphi(...\varphi(\sum_{j} w_{lj}I_{i})...)).$$

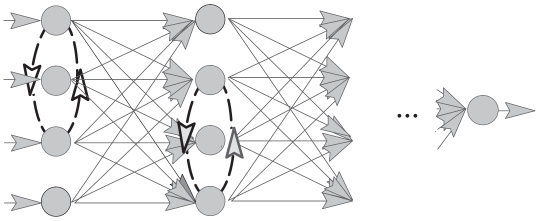

Figure 2: *The neurons in a fully–connected topology are interchangeable within layers.*

Sign transformation. The zero centered sigmoid (Fig. 3a) has odd symmetry, i.e. $\varphi(x) = -\varphi(-x)$. (For the nonsymmetrical sigmoid holds likewise $\varphi(x) = b - \varphi(-x)$.) A transformation, that inverses the sign of all input and output connections of a neuron with odd symmetry, does not change the network transfer function. In [1], it is proved that this symmetry is valid to a wide range of activation functions, i.e. for any infinitely differentiable function σ, satisfying $\sigma(0) = 0,\ \sigma'(0) \neq 0$ and $\sigma''(0) = 0$. [18] extends this analysis to cover any symmetrical function, including even functions such as Gaussian (Fig. 3b).

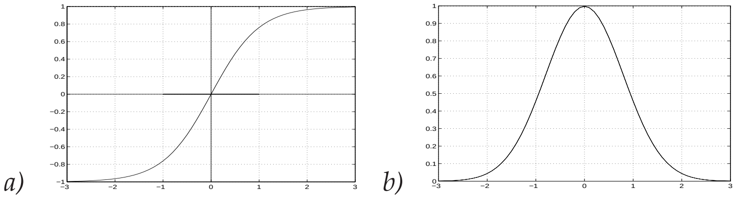

Figure 3: *Transfer functions with (a) odd symmetry $\varphi(x) = -\varphi(-x)$ and b) even symmetry $\varphi(x) = \varphi(-x)$. Both of them does not change the transfer of the complete network.*

The symmetry resulting from these types of transformation have a different impact, depending on the function of the neurons they are affecting. If the neurons are directly interacting with the environment (i.e. the input and output layers), the symmetry will imply the invariance of pattern presentation: Permuting the elements of an input string will not change the output. Because the permutation or sign symmetry at the input makes only sense in the context of reordering the elements of the input string it will be elaborated together with the spatial symmetries in the problem. Symmetries in the hidden layers have an impact on the learning evolution. Before explaining their impact, their variety will be summarized.

It can be shown that every specific neuron contributes a specific network function upon sign transformation for at least two different weight factors. For M hidden units, there will be M such symmetries, and thus any given weight vector will be one of a set of 2^M equivalent weight vectors. Similarly, if the values of all weights and bias of two hidden neurons are interchanged. The resulting network transfer will be unchanged, but, again a new network weight vector is obtained. For M hidden units, there will be $M!$ equivalent weight vectors. In the network these two types of symmetries are combined, and as a result the network has the weight space symmetry factor of $M!2^M$. For feedforward networks with more than one hidden layer the level of symmetry can be obtained by multiplying these factors for every hidden layer. This result is summarized in [8] by the following theorem:

Theorem 1: The set of all equioutput transformations on the weight space W forms a non–Abelian group G of order #G, with $\#G = \prod_{l=2}^{K-1} (M_l!)(2^{M_l})$

where K are the number of hidden layers, and M is the number of neurons in the hidden layer. The authors analyzing this group of symmetries as [1], [8], [16], [18], [30], have concluded that each coherent transformation defines a symmetry in the weight space, as consisting of equivalent parts. Consequently, it is possible to restrict the search of solutions to only one of these parts. By analogy, the error surface is also symmetrical. Even more, if a solution has been found during the learning period then there are more attractors (points with $\nabla E = 0$) on the symmetrical parts of the error surface. An important conclusion, fully proved for backpropagation by [16], is that the learning trajectories are themselves symmetrical. The general conclusions of the referred authors is that in many cases these symmetries are of little practical consequence.

The authors analyzing this group of symmetries as [1], [8], [16], [18], [30], have concluded that each coherent transformation defines a symmetry in the weight space, as consisting of equivalent parts. Consequently, it is possible to restrict the search of solutions to only one of these parts. By analogy, the error surface is also symmetrical. Even more, if a solution has been found during the learning period then there are more attractors on the symmetrical parts of the error surface. An important conclusion, fully proved for backpropagation by [16], is that the learning trajectories are themselves symmetrical. The general conclusions of the referred authors is that in many cases these symmetries are of little practical consequence. We counter that the various symmetries give the cost–function landscape periodicities, multiple minima, flat valleys and flat plateaus and could eventually disturb a satisfactory generalization.

2.2 Symmetries in the initial conditions and network parameters.

There is hardly any literature focussing directly on this subject. However, a lot of recommendations are given on how to increase the random factor, which indirectly destroys the symmetry. In [26] an initialization with small random weights is proposed. More precise borders of the size of intervals for weights selection are given by [14], where it is suggested that the weights and biases should be chosen from the interval $(-2.4/F_i\,,\ +2.4/F_i)$, with F_i for the fan–in of neuron *i*.

Symmetries in the initial conditions are due to the small initial parameter range. The selection interval for the initial settings is usually symmetric, what contributes to the existence of symmetric learning phase in the beginning of the learning process, but to a lesser degree than too small initial values. It is difficult to say in advance when the initial values are too small. A review of initialization techniques is done by [31].

Although the suggested rules are generally giving good results, the presence of additional symmetries brings the learning process in a symmetrical phase. Appearance of symmetries due to the wrong initialization are discussed in [17], but the conclusion of the authors is that wrong initialization can affect only artificial problems, containing high degree of symmetry.

A lot of authors suggest to add noise in the initialization (as well as in the network itself or to the problem) in order to improve the generalization performance. For a recent review of noise injection one is referred to [2]. Increasing the task complexity has a similar effect as increasing the dominance of the random component. Lower symmetries during the learning evolution are observed by [34] when they are introducing a non–zero bias factor in the *committee machine* under consideration. A too small slope of the nodal transfer [34] as well as a low learning rate [33] can cause equivalent effects.

3 Symmetries in the problem.

For the neural network tasks, the problems are described by sets of examples, also called training sets. Then, symmetries in the patterns, or in the training set, will be considered as equivalent terms of the symmetries in the problem.

The problem symmetries can occur in space and time domain. In the former situation, the symmetry may occur within a single pattern; this situation happens primarily in classification networks. Symmetry will then cause that a hidden neuron may be affected in mutually conflicting ways simultaneously. In the latter situation, we focus on the presentation order resulting from the random sampling and/or the time–sequencing; this has a primary appearance approximation and prediction networks. Symmetry will then cause either the poor approximation or the unlearning of previous history.

3.1 Spatial symmetries in patterns.

In [28] the difficulties of first–order Perceptrons and Hopfield networks to recognize mirrored, rotational and translational symmetries are addressed. The authors elaborate on the concept that the *order of a problem* [21] roughly corresponds to the minimum number of input units, that carry any information, relevant to the required output. For example, the Boolean OR function has order 1 because the value of a single input unit contains some information about the output. The Boolean XOR function has order 2 because each unit of the input array, considered in isolation, contains insufficient information about the output value. The generalized XOR or parity function of n binary inputs has order n. Higher–order problems can be solved by adding hidden units. It is suggested, that the XOR is an easy solvable second–order problem if only one hidden unit is added to the network.

Basically [28] attempts to solve the *mirror symmetry problem*: the network must detect which one of the three possible axes of symmetry is present in an *NxN* binary pixel input pattern. They prove that introducing hidden neurons makes symmetry detection an easy task: It is found, that a network with only two hidden neurons can already establish this property, regardless of the size of the input string (Fig. 4a). For every hidden unit the weights are chosen with odd–symmetric values. This means, that if a symmetric pattern is on, both hidden units will receive a summary input zero and their output will be off, because of their negative bias values. The output unit, having a positive bias in this case, will be on.

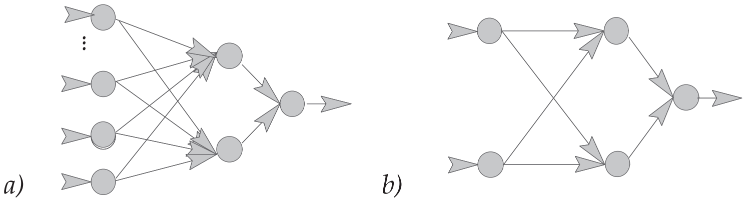

Figure 4: *a) Rumelhart's input pattern symmetry detector b) Blum's XOR network*

Another observation is that the weights on each side of the midpoint of the string are in the ratio 1:2:4, which ensures that each of the eight patterns that can occur sends an unique activation sum to the hidden unit, and therefore there is no pattern on the left, which will balance a non–mirror pattern in the right side.

When a symmetry pattern is on, [28] determines a second–order problem, because single pixels by themselves carry no information about the solution of the problem, but the information can be extracted from pairs of pixels that are related to the mirror symmetries. The authors demonstrate, that the Boltzmann learning algorithm is capable of solving the mirror symmetry problem.

Later on [23] makes a comparative study of the mirror symmetry problem using the Boltzmann machine, mean–field theory and the multilayer perceptron. Mean–field theory offers superior results to the Boltzman machine and slightly better than the multilayer perceptron. The results point to the interesting property that, by using relative entropy, mean–field theory and a multilayer perceptron have an approximately similar performance if they both have a single hidden layer and the size of the input layer is much larger than of the output layer.

The next step is made in [29], where the effects of introducing symmetries to feedforward neural networks are investigated. The author refers again to [21], and particularly to the *group–invariance theorem*. From a practical point of view this suggests a method to simplify training in those cases where the target function is known to be invariant under the action of a particular group of inputs. This aspect of the group invariance theorem is generalized in [29] for multilayer perceptrons. The approach aims to simplify the training task by making the symmetries a priori explicit in the network structure. An example is the XOR network, as shown in Fig. 4b, but with symmetrical weight assignment. This network is invariant to permutation for components of the input vector and so it could be trained to compute the XOR function with 5 parameters and 3 examples instead of the original 9 parameters and 5 examples.

3.2 Temporal symmetries in patterns.

While [26] illustrates that easy and elegant solutions exist for classifying symmetric input strings, [7] searches to approximate symmetric target sets. It proves the existence of a manifold of exact neural solutions for 2–variable Boolean functions. Furthermore there exists a manifold of local minima of the mean–square error E. It is concluded that for linear–separable problems a learning procedure forms stationary points, but these are not local minima and therefore can not cause very serious convergence problems.

The described symmetry problem concerns a classification task with neural nets using a nonsymmetrical activation function. The XOR net in [7] shows, that the null weights configuration can also be an attractor during learning. As a conclusion they propose to follow Rumelhart’s suggestion of setting up the initial weights with small random values, which is likely to prevent failures in the most cases.

In [3] is showing how the specific structure of the training set of a real–life problems have symmetries induced by the “longest run” potential. The longest–run induced symmetries can lead to slow convergence, unpredictable training duration or failure to learn at all.

A popular way to present temporal signals to neural networks is over a tapped delay–line. Design decisions include the size of the delay line and the sampling rate. Inadvertently one may thus introduce temporal symmetries and end up with learning problems. In [10] it has been discussed to monitor this circumstance from the curvature of the input/output plot.

A scarcely researched problem in training temporal patterns, where symmetry can play a misleading role, is the effect of initialization. Different initializations can easily lead to a different behavior: a phenomenon that bears likeness to chaos in system dynamics theory. Resolving symmetry effects for learning can therefore hardly be based on initialization.

4 Breaking the Symmetry.

Statistical Mechanics explains the symmetry in neural networks and problems in a global sense. It relates the microscopic neuron structure to macroscopic neural properties by describing the collective properties of very many interacting elements on basis of individual behavior and mutual interaction. The system properties can be more complex than the mere collection of its elements. This feature is a consequence of *sponta-*

neous symmetry breaking, which means, that a macroscopic system can be in a less symmetrical (more complex) state than the underlying microscopic dynamics.

The error landscape is usually formulated by an *energy function*, as introduced by Hopfield (1982). The term “Energy function” comes from the physical analogy to magnetic systems, but the concept is of much wider applicability. In many fields there is a state function that always decreases during dynamical evolution, or that must be minimized (maximized) to find an optimum state. Its most general name from the theory of dynamic systems is Lyapunov function. Other names in use are Hamiltonian in Statistical Mechanics, cost function (objective function) in Combinatorial Optimization theory, fitness function in Evolution theory.

In general, an energy function exists if the connection strengths are symmetric, i.e. $\varpi_{ij} = \varpi_{ji}$. Thus, symmetry in the connectivity matrix is a necessary condition for the Hopfield networks. Symmetry in the error landscape has its analogy in the temperature charts of the replica method, originating from Spin–Glass theory. Some other networks as the multilevel Kohonen network are developing orientation selective neurons. This orientation selectivity can be modelled by a Spin– Glass Model and thus also be an example for the symmetry breaking process. In [12] techniques are introduced, that enable the application of Statistical Machanics in feedforward neural networks. The output of the hidden neurons have equal absolute values at the beginning of the learning process, because the small initial values are positioning their outputs in the interval close to the zero point of the transfer function. If during learning the network is not able to extract a rule from the data, the hidden neurons do not specialize to a concrete function. This characterizes a symmetrical phase in the learning process, corresponding to a metastable state in the error surface, for instance a plateau. This symmetry should be broken, i.e. the system should reveal its high complexity, which corresponds to a specialization of the hidden layer units, and jump to a lower energy state on the energy landscape. Because initially the symmetry is built into the network, the examples (i.e. knowledge component) together with the noise (the random component) are supposed to break it.

4.1 Student and Teacher.

Statistical Mechanics presumes a straightforward relation between the network, drawn in a symmetrical phase, and the permutation symmetry. In [32], the permutation symmetry is considered in the context of learning evolution: the permutation symmetry makes the learning process to start in the symmetrical phase. Correspondingly, all hidden units have almost equal response. In the language of Statistical Mechanics, every hidden neuron of the *student* (student is any network under learning) imitates with the same degree the *teacher* (the network configuration which gives an exact mapping of the input–output dependence). This permutation symmetry should be broken, i.e. every hidden neuron should specialize, or in other words, should differ in response from the others. This way the network will reveal the higher complexity of the problem, i.e. will go in the state of lesser symmetry.

Moreover the Statistical Mechanics model of feedforward neural networks poses a constraint that induces an additional symmetry in the network structure. The most complicated structure, investigated untill now with the methods of Statistical Mechanics, is a *soft committee machine* [5]. This is a two–layer neural network with adjustable input–hidden, but fixed hidden–output weights. The average learning dynamics of this network is studied in the thermodynamic limit of the infinite input dynamics (the number of neurons is $N \rightarrow \infty$). The biases of the hidden neurons are fixed to zero. This model is said to be quite similar to real–world networks. The constraint that hidden–output weights are fixed on unity has been removed in [25]. There it is shown that the learning dynamics are usually dominated by the input–hidden weights (true mainly for the initial learning of MLPs) and hence the zero bias constraint is severe enough to introduce sufficient symmetries in the network.

In off–line (batch) training the symmetry breaking is connected with the effect of *retarded generalization* [32]: only if the number of examples is large enough, the hidden units can specialize and decrease the generalization error efficiently. Something similar, but not identical happens in on–line learning: initially the hidden units don't specialize which leads to a plateau in the learning curve. If the version space (the space of all weight vectors, consistent with the training set, and not constrained from a specific learning algorithm [11]) of the problem is sufficiently complex, then replica symmetry breaking occurs.

The weak point of the Statistical Mechanics approach is that it can be applied so far only for very simple networks, so–called parity and committee machines [5], [27], [34]. The more complicated networks, as for example 2–layer MLP with learnable analogue output weights are a more difficult task to be investigated by the methods of Statistical Mechanics. On the other hand the two–layer MLP is the simplest network structure in practical use. The results, obtained with simplified structures as a parity and committee machines, give a good general view on what is happening during the symmetrical phase and how it appears.

It should be pointed out, that structures such as parity and committee machines are introducing additional symmetries in the network, because of their fixed and equal hidden–to–output weights and zero biases. In the engineering practice a symmetrical learning phase is not observed often if at all. We are of the opinion that the permutation symmetry is not the reason of bad performance of the learning algorithm. It will rather help for additional symmetries (for example problem symmetries) to keep the network in the symmetrical phase in the beginning of the learning process.

4.2 Selective Sampling.

The perspective considered so far suggests that the learning process is as much dependent on the problem, (i. e. the knowledge) which is being presented as to the object (the network) which is going to process it. The network, implying the two designing principles (Symmetry and Randomness) and the Knowledge we have about the problem interact during the learning process. It can be said that the best learning trajectory ensures the best ratio between these 3 components during the entire learning evolution (Fig. 5).

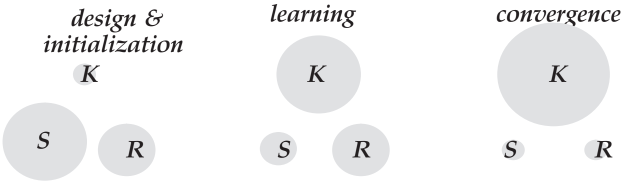

Figure 5: *Evolution of the KRS–system.*

Our critique on the methods of symmetry breaking, made so far are that they are either problem specific, or concern one point (usually the beginning) of the learning process. Alternatively we propose to solve the global learning problems by keeping the proper KRS ratio during the entire learning period. In a sense this idea is borrowed from the *active learning* [24] approach, which presumes that the learning algorithm has control over what part of the problem domain it receives information about. Our suggestion is this control to be based on the instantaneous check of the training set structure and to reorder it if necessary. This way, not only the initial high symmetry will be destroyed, but also the long plateaus of the training error can be shortened and the training time decreased. The other advantage of this proposal is that the problem is not violated by adding extra noise to it.

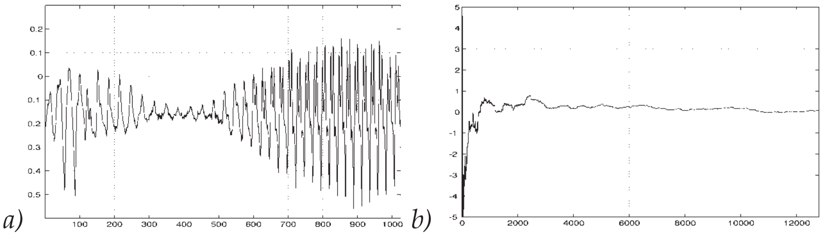

Figure 6: *a) Training set extracted from a real signal possessing the internal conflict property. b) Evolution of the mean value of the direction coefficient for the constructed training sequences.*

The materialization of this idea is based on the observation that unlearning problems show up in the structure of the training set. (Fig. 6). The “longest run” potential of this set shows the same property as the symmetrical function do. A tested algorithm which prevents from the symmetry problems proceeds as:

1. Extract the data set $\boldsymbol{D_n} \equiv \left\{(\boldsymbol{x_i},\boldsymbol{y_i})\right\}_{\boldsymbol{i=1}}^{\boldsymbol{N}}$ from the signal $\boldsymbol{S(x, y)}$ by random sampling.
2. Calculate the direction coefficient mean evolution, for the current training set $\boldsymbol{E_{m_p}} \equiv \left\{(\boldsymbol{x_l},\boldsymbol{y_l})\right\}_{\boldsymbol{i=1}}^{\boldsymbol{pm}}$, (after the current randomization of $\boldsymbol{D_n} \equiv \left\{(\boldsymbol{x_i},\boldsymbol{y_i})\right\}_{\boldsymbol{i=1}}^{\boldsymbol{N}}$).
3. Divide the data set $\boldsymbol{D_n} \equiv \left\{(\boldsymbol{x_i},\boldsymbol{y_i})\right\}_{\boldsymbol{i=1}}^{\boldsymbol{N}}$ on equidistant windows $\boldsymbol{D_{n_m}} \equiv \left\{(\boldsymbol{x_l},\boldsymbol{y_l})\right\}_{\boldsymbol{l=1}}^{\boldsymbol{m}}$.
4. After randomization the training subsets for the first few epochs $\boldsymbol{E_{m_p}} \equiv \left\{(\boldsymbol{x_l},\boldsymbol{y_l})\right\}_{\boldsymbol{i=1}}^{\boldsymbol{pm}}$ are obtained from the data subsets $\boldsymbol{D_{n_m}} \equiv \left\{(\boldsymbol{x_l},\boldsymbol{y_l})\right\}_{\boldsymbol{l=1}}^{\boldsymbol{m}}$.
5. If detected existence of the cancelation, decrease the size of the window. Go to 3.
6. If evolution curves as calculated in 4 stabilize to show absence of cancelation the learning can be left on its internal dynamics. End.

5 Discussion.

We have been reviewing the occurrence and impact of neural networks symmetries from the perspective of the relief they form on the global error surface and thus from the learning evolution point of view. In summary, the symmetries in the network architecture and transfer are forming repeating parts in the error landscape and are of importance only in the beginning of the learning process, before the specialization of the neurons occurs. These symmetries are predictable and common for most networks. Normally they can not harm the learning process. On the other hand, when the problem symmetries are on, together with a high initialization symmetries, the structural equalities can not be destroyed and thus the specialization of the representation neurons or learning is likely to fail or to suffer from too long and unpredictable learning duration.

Up till now, all the reviewed literature concerns either artificial problems (attempts to learn geometrical or algebraic functions, that contain a lot of internal symmetries), or networks of importance rather for theory than for practice (parity machines, committee machines) which introduce an artificial symmetries as well. In [3] it is shown how symmetries can affect real–life problems.

Moreover the suggested strategies for breaking the symmetry have the following weak points: they are often violating the problem to be learned by introducing additional noise – a remedy that can harm the learning quality near the convergence point; the solution they are suggesting is in most cases problem dependent; often they concern only one moment in which the repair should take place.

Considering the training process as an unity of three components (knowledge K, randomness R and symmetry S) and adopting the ideas from the *active learning paradigm* [24] we suggest a method for holding the best KRS ratio during the learning evolution.

One possible algorithm, able to materialize this method, is based on the characteristics, visualized at Fig. 6. The analytical details around the algorithms and its implementation results are beyond the scope of this paper and are to be published elsewhere.

References.

[1] F. Albertini and E. Sontag, “For Neural Networks, Function Determines Form”, *Neural Networks* **6** , pp. 975–990, 1993.

[2] G. An, “The Effects of Adding Noise During Backpropagation Training on a Generalization Performance”, *Neural Computation* **8**, pp. 643–674, 1996.

[3] E. I. Barakova and L. Spaanenburg, “Selective Sampling for Reliable Neural Signal Approximation”, *Proceedings NICROSP’96* (Venice, Italy) pp. 183–193, 1996.

[4] D. Barber, D. Saad and P. Sollich, “Finite size effects in on–line learning of multilayer neural networks”, *Europhysics letters* **34**, pp. 151.

[5] M. Biehl and H. Schwartze, “Learning by Online Gradient Descent”, *Journal of Physics. A: Mathematical and general* **28,** pp. 643–656, 1995.

[6] C. M. Bishop, ”Neural Networks for Pattern Recognition”, *Oxford University Press,* 1995.

[7] E. K. Blum, ”Approximation of Boolean Functions by Sigmoidal Networks: Part I: XOR and Other Two–Variable Functions”, *Neural Computation* **1**, pp. 532–540, 1989.

[8] An Mei Chen, H. Lu, R. Hecht–Nielsen, ”On the geometry of feedforward neural networks error surfaces”, Neural Computation **5**, nr. 6, pp. 910–927, 1993.

[9] D. Chon and G. Tesauro. “How Tight are the Vapnik Chervonenkis Bounds?” *Neural Computation* **4**, nr. 2, pp. 249–269, 1992.

[10] M. Diepenhorst, W.J. Jansen, J.A.G. Nijhuis and L. Spaanenburg, “On the learnability of temporal relations”, to be published, 1997.

[11] E. Gardner, “The Space of Interactions in Neural Networks Models”, *Journal of physics A: Mathematical and general* **21**, pp. 257–270, 1988.

[12] E. Gardner and B. Derida, *Journal of physics A: Mathematical and general* **22**, nr. 12, pp. 1983, 1989.

[13] Y. Grandvalet and S. Canu, ”Comments on Noise Injection into Inputs in Back Propagation Learning”, IEEE Transactions on Systems, Man and Cybernetics **25**, Nr. 4, April 1995.

[14] S. Haykin, *Neural Networks: a comprehensive foundation* (MacMillan), 1994.

[15] T. M. Heskes and B. Kappen, *Physical Review A* **44**, pp. 2718.

[16] F. Jordan and G. Clement, “Using the Symmetries of Multilayered Network To Reduce the Weight Space”, *IEEE ICNN* **2**, pp. 391–396, 1991.

[17] F. Kolen and J.B. Pollack, “Back Propagation is Sensitive to Initial Conditions”, In D.Touretzky, ed.,” *Advances in Neural Information Processing Systems”* **4** (Morgan Kaufmann, San Francisco, CA) 1992.

[18] V. Kurkova and P.Kainen, “Functionally Equivalent Feedforward Neural Networks”, *Neural Computation* ***6***, nr. 3, pp. 553–558.

[19] K. Matsuoka, ”Noise injection into inputs in Backpropagation learning”, *IEEE Transactions on Systems, Man and Cybernetics* **22**, nr. 3, 1992.

[20] M. Mezard, G. Parisi, and M.A. Virasoro, “Spin–Glass Theory and Beyond”, *Lecture notes in Physics* **9** (World Scientific Press, Singapore).

[21] M. Minski and S. Papert, *Perceptrons* (MIT Press, Cambridge) 1969.

[22] R. Monasson and D. O’Kane, “Domains of solutions and replica symmetry breaking in multilayer neural networks”, *Europhysics letters* **27**. nr. 2, pp. 85, July 10 1994.

[23] C. Peterson, “Mean field theory neural networks for feature recognition, content addressable memory and optimization”, *Connectionists Science* **3**, pp. 3–33, 1991.

[24] M. Plutowski and H. White, “Selecting concise Training Sets from Clean Data”, *IEEE Trans. on Neural Networks* **4**, nr. 2, March 1993.

[25] P. Riegler and M. Biehl, “On–line backpropagation in two–layered neural networks.”*Journal of Physics. A: Mathematical and general* **28,** pp. 507–513, 1995.

[26] D. E. Rumelhart, G.E. Hinton, and R.J. Williams, ”Learning internal Representations by Error Propagation”, in *Parallel Distributed Processing* **1** (MIT Press) 1986.

[27] D. Saad and S.A. Solla, “On line learning in soft committee machines”, *Physical Review E* **52**, nr. 4, pp. 4225–4243.

[28] T. J. Sejnowski, P.K. Kienker, and G.E. Hinton, “Learning symmetry groups with hidden units: Beyond the perceptron”, *Physica* **22D**, pp. 260–275, 1986.

[29] J. Shawe–Taylor, “Symmetries and discriminability in feedforward network architectures”, *IEEE Transactions on Neural Networks* **4**, nr. 5, pp. 816–826, 1993.

[30] H. J. Sussmann, ”Uniqueness of the weights for minimal feedforward nets with a given i/o map”, *Neural networks* **5**, pp. 589–593, 1992.

[31] G. Thimm and E. Fiesler, “High Order and Multilayer Perceptron Initialization”, Accepted for publication by *IEEE Transactions on Neural Networks*, 1996.

[32] T. L.Watkin, A. Rau and M. Biehl, “The Statistical Mechanics of Learning a Rule”, *Review of modern Physics* **6**, nr. 2, April 1993.

[33] W. Wiegerinck and T. Heskes, ”How dependencies between successive examples affect on–line learning”, to appear in *Neural Computation*, 1996.

[34] A. West, D. Saad, and I. Nabney, “The learning dynamics of a universal approximator”, *Proceedings NIPS’96*, 1996.