Distance-based Multilayer Perceptrons.

Włodzisław Duch[a], Rafał Adamczak[a] and Geerd H.F. Diercksen[b]
[a]Department of Computer Methods, Nicholas Copernicus University,
Grudziądzka 5, 87-100 Toruń, Poland; e-mail: duch,raad@phys.uni.torun.pl
[b]Max-Planck Institute of Astrophysics, 85740-Garching, Germany,
e-mail: GDiercksen@mpa-garching.mpg.de

Abstract— **Neural network models are presented as special cases of a framework for general Similarity-Based Methods (SBMs). Distance-based multilayer perceptrons (D-MLPs) with non-Euclidean metric functions are described. D-MLPs evaluate similarity to prototypes making the interpretation of the results easier. Renormalization of the input data in the extended feature space brings dramatic changes in the shapes of decision borders. An illustrative example showing these changes is provided.**

I. Introduction

Multilayer perceptrons (MLPs) trained with backpropagation method (BP) are certainly the most popular among all neural techniques [1]. Applied to classification problems MLPs use sigmoidal functions to provide soft hyperplanes dividing the input space into separate regions. MLPs are therefore similar to the statistical discriminant techniques, although combination of soft sigmoids allows for representation of more complex, nonlinear decision borders. This is usually considered to be a strength of the MLP model, although in cases when sharp decision borders are needed it may also become its weakness. For example, classification borders conforming to a simple logical rule $x_1 > 1 \wedge x_2 > 1$ are easily represented by two hyperplanes but there is no way to represent them accurately using soft sigmoidal functions used in MLPs. Increasing the slopes of sigmoidal functions to improve representation of such decision borders leads to problems with learning by backpropagation or any other gradient-based method, since the volume of the input space in which sigmoids change rapidly (and thus gradients are non-zero) is rapidly shrinking. In the limit sigmoidal functions become step-functions but gradient techniques like backpropagation cannot be used to make this transition. As a result for some datasets no change in the learning rule or network architecture will improve the accuracy of neural solutions. A good real-world example is the hypothyroid dataset, for which the best optimized MLPs still give about 1.5% of error [2] while logical rules reduce it to 0.64% (since 3428 cases are provided for testing this is a significant improvement).

An additional problem with MLPs is connected with the interpretation of their classification decisions. Proponents of the logical rule-based machine learning methods consider it to be the biggest drawback of neural networks, limiting their applications in safety-critical fields such as medicine. Similarity-Based Methods (SBMs), for example the k-nearest neighbor (k-NN) method, retrieve the relevant context for each query presented to the classification system, providing some interpretation and estimating probability of different class assignment. Such interpretation is also possible for the Radial Basis Function (RBF) networks using Gaussian or other localized functions, or the Learning Vector Quantization (LVQ) method based on optimization of reference vectors. It may seem that such an interpretation is not possible for MLPs since they belong to the discriminant rather than to memory-based techniques. One way to obtain an interpretation of MLP decisions is to study the transition from nonlinear MLP to the network performing logical operations [3]. Although discriminant methods and prototype methods seem to be quite different in fact the two approaches are deeply connected. A single hyperplane discriminating vectors belonging to two classes may be replaced by two prototypes, one for each class. For N prototypes one can generate $N(N-1)/2$ pair-wise discriminating hyperplanes providing piece-wise linear approximation to the decision borders.

Recently a general framework for Similarity-Based Methods (SBMs) of classification has been presented [4]. The Distance-Based Multilayer Perceptrons (D-MLPs) described in the next section are just one of many models that may be derived from this framework. They improve upon the traditional approach by providing more flexible decision borders and by enabling a prototype-based interpretation of the results. The treatment of symbolic and missing values and the initialization of such networks is also described. The third section contains discussion of the metric functions useful for D-MLPs. To avoid programming the backpropagation training method for each type of distance function a simple transformation of the input data is proposed. The input space is extended and the data renormalized using the chosen distance functions. An illustration of this method on the Iris data is presented for pedagogical purposes in the fourth section. The paper is finished with a short discussion.

II. D-MLP networks as a similarity-based methods.

Although the focus of this paper is on classification the same framework may also be applied to regression and pattern completion problems. The classification problem is stated as follows: given a set of class-labeled training vectors $\{\mathbf{R}^j, \mathbf{C}(\mathbf{R}^j)\}$, $j = 1..N_t$, where $\mathbf{C}(\mathbf{R}^j)$ is the class of $\mathbf{R}^j$, and giv-

en a vector $\mathbf{X}$ of an unknown class, use the information provided in the similarity measure $D(\mathbf{X},\mathbf{R}^j)$ to estimate the probability of classification $p(C_i|\mathbf{X};M)$, where M describes the classification model used (values of all parameters and procedures employed). A general model of an adaptive system used for classification should include at least the following elements:
$M = \{D(\cdot), G(D(\cdot)), k, \{\mathbf{R}^j\}, E[\cdot], K(\cdot))\}$, where
$D(\cdot)$ is a function (usually a distance function), parametrized in various ways, or a table used to compute similarities;
$G(D(\mathbf{X},\mathbf{R}))$ is a weighting function estimating contribution of the reference vector $\mathbf{R}$ to the classification probability;
k is the number of reference vectors taken into account in the neighborhood of $\mathbf{X}$;
$\{\mathbf{R}^j\}$ is the set of reference vectors created from the set of training vectors $\{\mathbf{X}^i\}$ by some procedure;
$E[\cdot]$ is the total cost function optimized during training;
$K(\cdot)$ is a kernel function, scaling the influence of the error, for a given training example, on the total cost function.

An adaptive system may include several such models M_l and an interpolation procedure to select between different models or average results of a committee of models. Various procedures for selection of features, minimization algorithms and architectures used for network computation lead to a large number of similarity-based methods. Here only neural methods based on this framework are considered. The cost function that minimizes risk for overall classification is:

$$E(\{\mathbf{X}\};\mathcal{R},M) = \sum_i \sum_{\mathbf{X}} \mathcal{R}(C_i,C(\mathbf{X})) H\left(p(C_i|\mathbf{X};M), \delta(C_i,C(\mathbf{X}))\right) \quad (1)$$

where $i = 1 \ldots N_c$ runs over all classes, $\mathbf{X}$ over all training vectors, $C(\mathbf{X})$ is the true class of the vector $\mathbf{X}$ and a function $H(\cdot)$ is monotonic and positive, often a quadratic function. The elements of the risk matrix $\mathcal{R}(C_i,C_j)$ are proportional to the risk of assigning the C_i class when the true class is C_j (in the simplest case $\mathcal{R}(C_i,C_j) = 1-\delta_{ij}$), and M specifies all adaptive parameters and variable procedures of the classification model that may affect the cost function. Regularization terms aimed at minimization of the complexity of the classification model are frequently added to the cost function, helping to avoid the overfitting problems. If $H(\cdot)$ is a quadratic function of the $\max_i p(C_i|\mathbf{X}^p;M) - \delta(C_i,C(\mathbf{X}))$ standard mean square error function is recovered.

Neural networks using binary inputs and threshold neurons compute distances in a natural way. If the input signals $\mathbf{X}$ and the weights $\mathbf{W}$ are $(\pm 1 \ldots \pm 1)$ vectors, neuron with N inputs and the threshold θ realizes the following function:

$$\Theta(\sum_i^N W_i X_i - \theta) = \begin{cases} 0 & \text{if } ||\mathbf{W}-\mathbf{X}|| > \theta \\ 1 & \text{if } ||\mathbf{W}-\mathbf{X}|| \le \theta \end{cases} \quad (2)$$

where $||\cdot||$ norm is defined by the Hamming distance. One can interpret the weights of neurons in the first hidden layer as addresses of the reference vectors in the input space and the activity of threshold neuron as activation by inputs falling into a hard sphere of radius θ centered at $\mathbf{W}$. Changing binary into real values and threshold into sigmoidal neurons for inputs normalized to $||\mathbf{X}|| = ||\mathbf{W}|| = \mathbf{1}$ leads to a soft activation of neuron by input vector close to $\mathbf{W}$ on a unit sphere. The Hamming neural network [6] is actually a neural realization of the nearest neighbor method for a single neighbor and binary inputs.

The standard activation of a neuron $\mathbf{W}\cdot\mathbf{X}$ may always be written as:

$$\mathbf{W}\cdot\mathbf{X} = \frac{1}{2}\left(||\mathbf{W}||^2 + ||\mathbf{X}||^2 - ||\mathbf{W}-\mathbf{X}||^2\right) \quad (3)$$

For normalized input vectors $||\mathbf{X}|| = 1$ the transfer functions of MLP neuron is:

$$\sigma(\mathbf{W}\cdot\mathbf{X}+\theta) = \sigma(d_0 - D(\mathbf{W},\mathbf{X})) \quad (4)$$

where $D(\mathbf{W},\mathbf{X})$ is proportional to the square of Euclidean distance between $\mathbf{W}$ and $\mathbf{X}$. Normalization is necessary to avoid the dependence of d_0 on $\mathbf{X}$. This function evaluates the influence of the reference vectors $\mathbf{W}$ on the classification probability $p(C_i|\mathbf{X};\{\mathbf{W},\theta\})$. It plays a role of the weight function $G(D) = \sigma(d_0 - D(\mathbf{W},\mathbf{X}))$, monotonically decreasing, with flat plateau for small distances, reaching the value of 0.5 for $D(\mathbf{W},\mathbf{X}) = d_0$ and approaching zero for larger distances. For normalized $\mathbf{X}$ but arbitrary $\mathbf{W}$ the range of the sigmoid argument lies in the $[\theta - |\mathbf{W}|, \theta + |\mathbf{W}|]$ interval. A unipolar sigmoid has a maximum curvature around ± 2.4, therefore small thresholds and weights mean that the network operates in an almost linear regime. Regularization methods add penalty terms to the error function forcing the weights and thresholds to become small and thus smoothing the network approximation to the training data.

From the SBM point of view in MLP networks, as long as the input data is normalized, sigmoidal functions are used to estimate the influence of weight vectors according to the distance between weight and training vectors. Many such estimations are combined to compute the final output. By changing the distance function in equation (4) from the square of the Euclidean distance to some other distance measures new types of neural networks, called further D-MLP networks, are defined. Another possibility is to write the weighted product in a form:

$$\sigma(\mathbf{W}\cdot\mathbf{X}) = \sigma\left(\frac{1}{4}(||\mathbf{W}+\mathbf{X}||^2 - ||\mathbf{W}-\mathbf{X}||^2)\right) \quad (5)$$

and replace the Euclidean norm by Minkovsky's or other type of norms. Although results are equally interesting only the form (4) has been used below. The D-MLP networks simply replace the square of the Euclidean distance in this equation by some other metric function. The D-MLP network with the nodes computing $\sigma(d_0 - D(\mathbf{W},\mathbf{X}))$ is trained like the standard MLP, using the backpropagation method [1]. Backpropagation procedure requires derivatives of the distance functions, but for Minkovsky and other popular functions they are easily provided.

The network should be initialized taking the centers of the clusters in the extended space as $\mathbf{W}$ and taking $d_0 = D(\mathbf{W},\mathbf{X}^b)$, where $\mathbf{X}^b$ is a vector at the borders of the given cluster (we have tried [7] dendrograms and decision trees but other clusterization methods may also be used for initialization [8]). A parameter which is rarely changed in MLPs is the slope of the sigmoidal function. It defines the area which has an influence on performance of each node. If the slope is too high the area in which the sigmoidal function is not approximately constant is small and only a few training vectors have a chance to influence the gradient-based learning procedures. If it is too low then all functions strongly overlap and there is no possibility to create sharp decision borders. In the standard formulation:

$$\begin{aligned}(\mathbf{W}\cdot\mathbf{X}+\theta)/T &= (\frac{\mathbf{W}}{||\mathbf{W}||}\cdot\mathbf{X}+\frac{\theta}{||\mathbf{W}||})||\mathbf{W}||/T \\ &= (\mathbf{W}'\cdot\mathbf{X}+\theta')/T' \qquad (6)\end{aligned}$$

Thus for normalized $\mathbf{X}$ and $\mathbf{W}'$ increase of the norm of the weights is equivalent to increase of the slope and no special learning for the slopes is needed. A useful variability range of the sigmoid is between its maximum curvature points, which for $T=1$ are between $\Delta(T)=\pm 2.4$. If the variability range is assumed to be 1/10 of the size of the cluster, i.e. $\Delta(T)=\pm d_0/10$ then setting $T\approx d_0/24$ will be appropriate. After such initialization of the network parameters training procedure is usually quite short.

A. Metric functions and input transformation.

In Eq. (4) the parameter d_0 should be treated as an adaptive parameter only if $\mathbf{X}$ is normalized. This may always be done without loss of information if one or more additional components are added to the vector, extending the feature space by at least one dimension. In particular taking $x_r=\sqrt{R^2-||\mathbf{X}||^2}$, where $R\geq \max_X ||X||$, amounts to a projection of the data on a unit semisphere with radius R (more sophisticated projection is described in [7]. If non-Euclidean norm is used the sphere changes its shape. Minkovsky's distance with the scaling factors is:

$$D(\mathbf{A},\mathbf{B};s)^\alpha=\sum_i^N s_i d(\mathbf{A}_i,\mathbf{B}_i)^\alpha \qquad (7)$$

The $d(\cdot)$ function is used to estimate similarity at the feature level and in the simplest case is equal to $|A_i-B_i|$. For large α this metric changes the sphere into a soft cuboid, for $\alpha=1$ it becomes a pyramid and for $\alpha<1$ it has a hypocycloidal shape. Instead of deriving the backpropagation equations for the transfer functions with non-Euclidean distances one may achieve similar result using a standard MLP network with x_r determined by the normalization condition using the desired metric.

The distance function may be heterogeneous, using Minkovsky's metric for numerical features and probabilistic metrics [9] for symbolic features. In memory-based reasoning the Modified Value Difference Metric (MVDM) has gained some popularity [9]. The distance between two N-dimensional vectors A,B with discrete (nominal, symbolic) elements, in a K class problem, is computed using conditional probabilities:

$$D_\alpha(A,B)=\sum_j^N\sum_i^K\left|p(C_i|A_j)-p(C_i|B_j)\right|^\alpha \qquad (8)$$

where $p(C_i|A_j)$ is estimated by calculating the number of times $N_i(A_j)$ the value A_j of the feature j occurred in vectors belonging to class C_i and dividing it by the number of times feature this A_j value occurred for any class. We can also define a "value difference" for each feature j as $d_v(A_j,B_j)=\sum_i^K(p(C_i|A_j)-p(C_i|B_j))$ and compute $D(A,B)$ as a sum of value differences over all features. Metric is defined here via a data-dependent matrix with the number of rows equal to the number of classes and the number of columns equal to the number of all attributes. Generalization for continuos values requires a set of probability density functions $p_{ij}(x)$, with $i=1..K, j=1..N$.

Using VDM type of metrics leads to problems with calculation of gradients, therefore another method is advocated here. The feature space is extended adding enough dimensions to reproduce, using Minkovsky or other metric $D()$, the VDM distances for the training vectors. Let $\mathbf{A}$ stand for the symbolic part of the input vector $\mathbf{X}$. The VDM metric is used only for this symbolic part and allows to calculate $D_{VDM}(\mathbf{A}^i,\mathbf{A}^j)$ distance tables. Numerical representation $\mathbf{B}$ of symbolic vectors $\mathbf{A}$ should preserve the VDM distances. The algorithm proceeds as follows: an arbitrary numerical value $\mathbf{B}^1=(0)$ is taken for the symbolic features of the first vector $\mathbf{A}^1$, and the next vector $\mathbf{A}^j$ is taken. Symbolic values should be replaced by numerical values in such a way that all VDM distances $d_{jk}=D_{VDM}(\mathbf{A}^j,\mathbf{A}^k)$, $k=1..j-1$ are reproduced by the distances $D_{jk}=D(\mathbf{B}^j,\mathbf{B}^k)$ calculated with $\mathbf{B}^k$ vectors. This is possible because VDM metric fulfills the triangle inequality, as any metric function should do. The spheres centered at the previous vectors $\mathbf{B}^k$, with the d_{jk} radiuses, should all cross at least in one point $\mathbf{B}^j$. If this point does not belong to the subspace of the previous $\mathbf{B}^k$ vectors a new feature is added to the numerical vector, and all previous vectors $\mathbf{B}^k$ are extended with $(\mathbf{B},0)$. After the last training vector is included the final numerical vectors $\mathbf{B}$ are appended to the numerical part of the input vectors $\mathbf{X}$ forming a new set of input vectors. The final input space has usually a larger number of dimensions than the original input space, the maximum increase being equal to the number of the training vectors, although sometimes the number of dimensions may even decrease. If the final number of dimensions is for some reason too large those dimensions i for which all features b_i of the $\mathbf{B}$ vectors are small may be dropped and the remaining values rescaled, minimizing the differences $\sum_{j>k} D_{jk}-d_{jk})^2$.

Many other types of metric functions exist [9] and their performance should be empirically verified. Since the architecture of the D-MLP network is completely determined by the initialization procedure and the training is short due to a good starting point various distance functions may be tried on a given problem.

III. Pedagogical illustration

The influence of non-Euclidean distance functions on the decision borders is illustrated here on the classical Iris flowers dataset, containing 50 cases in each of the 3 classes. The flowers are described by 4 measurements (petal and sepal width and length). Two classes, Iris virginica and Iris versicolor, overlap, and therefore a perfect partition of the input space into separate classes is not possible. An optimal solution (from the point of view of generalization) contains 3 errors [10] and may be obtained using only two of the four input features (x_3 and x_4), therefore it is easy to display and only those two features have been left in simulations described below.

A standard MLP solution is obtained with 4 hidden neurons and 3 output neurons. One discriminating plane per class of the smallest and the largest flowers (setosa and virginica) is needed and two planes to separate the vectors of the versicolor class. To increase accuracy and speed up the learning in the final phase of learning only the vectors near the class borders are presented to the network. The selection algorithm loops over all vectors and for a given vector $\mathbf{X}$ finds k (for example $k = 10$) nearest vectors belonging to a different class than $\mathbf{X}$. These vectors are written to the new training file providing a description of the border region. The MLP solution is equivalent to a rule that uses linear combination of the inputs, $a_1x_3 + b_1x_4 + c_1 < 0$ for setosa class, and $a_2x_3 + b_2x_4 + c_2 > 0$ for virginica, and the else condition for versicolor class. This method of training leads to a sharper and more accurate decision borders.

The data has been standardized and rescaled to fit it inside a square with ± 1 corners. An additional feature has been added and the 3-dimensional vectors normalized using various Minkovsky distance measures. The network has been initialized taking the normalized weights that are equal to the centers of the three clusters. In the extended feature space only 3 neurons are necessary. In Figure 1 dramatic changes in the shapes of decision borders for Minkovsky metric are observed. Using squared Euclidean metric in $\sigma(d_0 - D(\mathbf{X},\mathbf{R}))$ transfer functions the standard MLP solution is obtained. Euclidean case corresponds to circular decision borders, the city block metric $\alpha = 1$ gives sharp, romboidal shapes, for large α almost rectangular decision borders are obtained (an approximation using logical rules is in this case straightforward) while for small α a hypocycloidal shapes are created. Since smooth transition between these cases is made α should be treated as an adaptive parameter. For the Iris data the optimal solution (3 errors) has been recovered for all values of $\alpha \geq 0.8$.

IV. Discussion

The similarity based framework accommodates many classification methods, including neural networks. Neural networks of the D-MLP type, using non-Euclidean distance functions, are especially interesting and seem to open many unexplored possibilities. Simple transformation based on normalization of the input data in extended space may completely change the hyperplanar decision borders introducing quite complex shapes. Standard MLP programs may be used for training such networks. The training times are short since a good initialization procedure based on clusterization techniques determines weights, thresholds and the slopes of all neurons. The number of neurons in the network defined in extended space may also decrease, as has been observed in the Iris example. A new method to treat symbolic values and a new training procedure using only the vectors close to the decision borders has been described.

An additional advantage of the approach outlined here is the understanding of what these networks have really learned in terms of the prototypes (weights) and the weighted distances from these prototypes. Moreover, if partial similarity is defined for reduced number of known attributes the same networks may be used for pattern completion tasks. Given a partially known vector $\mathbf{X}$ all sufficiently similar nodes are easily identified, the missing $\mathbf{X}$ values replaced by weights $\mathbf{W}^i$ of these nodes creating several $\mathbf{X}^i$ candidates and the one with the highest classification probability is selected as the complete vector. Such procedure is always defined while calculation of partial activation of neurons does not always make sense. Unknown input features may be obtained by interpolation among several nodes that code similar prototypes. Although only a first step towards the similarity based neural methods has been made here already a number of interesting new models have been introduced. Empirical comparisons of D-MLPs with other classification systems should be reported soon.

Acknowledgments: Support of W.D. and R.A. by the Polish Committee for Scientific Research, grant 8T11F 00308, is gratefully acknowledged.

References

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

[2] W. Schiffman, M. Joost, R. Werner, "Comparison of optimized backpropagation algorithms", Proc. of ESANN'93, Brussels 1993, pp. 97-104

[3] Duch W, Adamczak R, Grźbczewski K, *Extraction of logical rules from backpropagation networks*. Neural Processing Letters 7 (1998) 1-9

[4] W. Duch, Neural minimal distance methods, Proc. 3-rd Conf. on Neural Networks and Their Applications, Kule, Poland, Oct. 14-18, 1997

[5] W. Duch, G.H.F. Diercksen, *Feature Space Mapping as a universal adaptive system*, Comp. Phys. Communic. **87** (1995) 341-371

[6] R.P. Lippmann, An introduction to computing with neural nets, *IEEE Magazine on Acoustics, Signal and Speech Processing* 4 (1987) 4–22; P. Floreen, The convergence of Hamming memory networks, *Trans. Neural Networks* 2 (1991) 449–457

[7] W. Duch, R. Adamczak, N. Jankowski, *Initialization and optimization of multilayer perceptrons*, 3rd Conf. on Neural Networks and Their Applications, Kule, Poland, October 1997, pp. 105-110

[8] P.R. Krishnaiah, L.N. Kanal, eds, Handbook of statistics 2: classification, pattern recognition and reduction of dimensionality (North Holland, Amsterdam 1982)

[9] D.R. Wilson, T.R. Martinez, Improved heterogenous distance functions. J. Artificial Intelligence Research 6 (1997) 1-34

[10] Duch W, Adamczak R, Grźbczewski K, ŕal G, *Hybrid neural-global minimization method of logical rule extraction*, Journal of Advanced Computational Intelligence (in print)

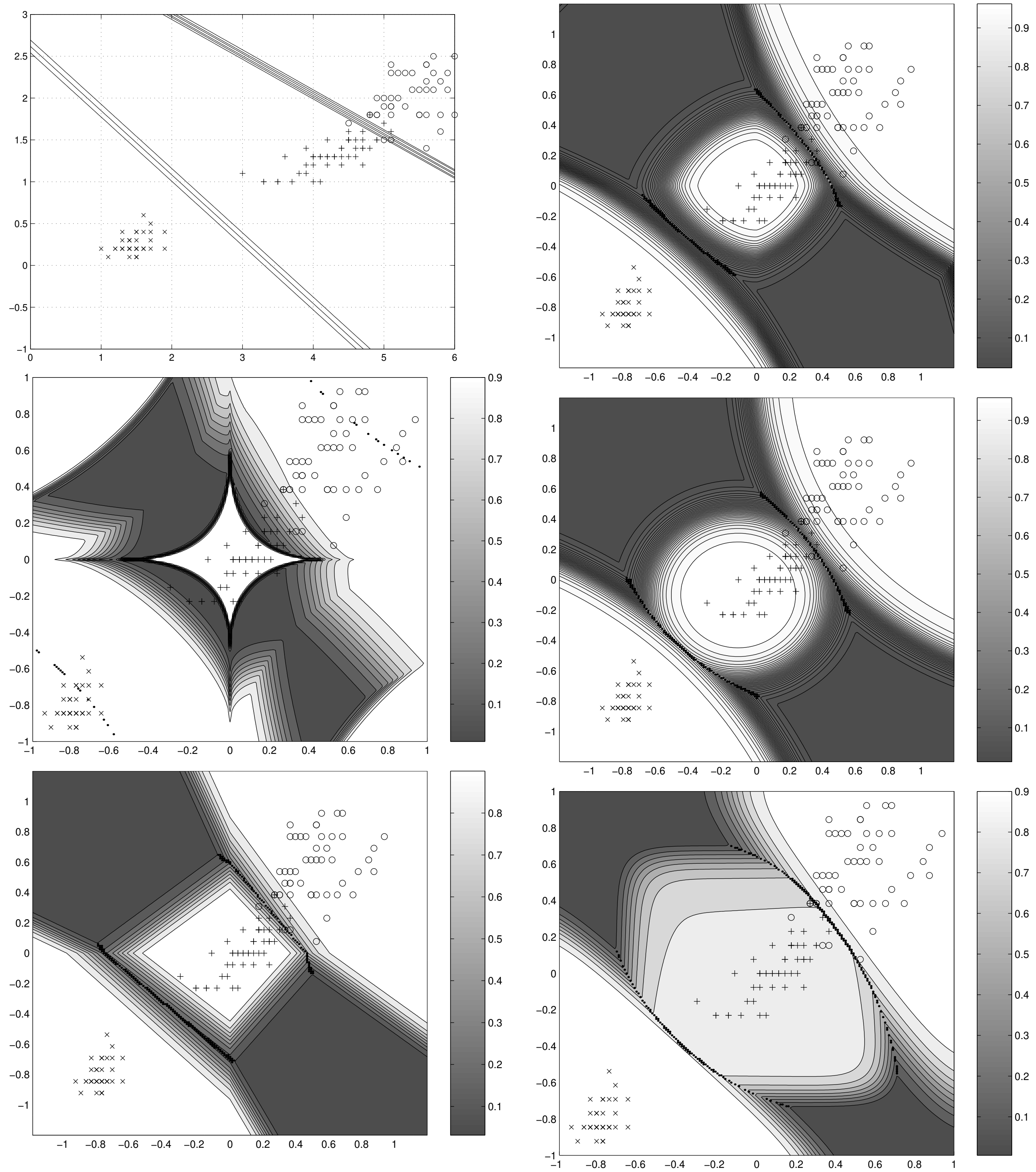

Fig. 1. Shapes of decision borders in the Iris case for standard MLP and D-MLP with Minkovsky metric, $\alpha = 0.5, 1.0, 1.5, 2.0$ and 7.0.