Distance-based large margin classifier suitable for integrated circuit implementation

L.C.B. Torres, C.L. Castro, F. Coelho, F. Sill Torres and A.P. Braga✉

A new learning method for classification problems that is suitable for integrated circuit implementation is presented. The method, which outperforms current approaches in many data sets, is based on a structural description of the learning set represented by a planar graph. The final classification function is composed of a hierarchical mixture of local experts, which yields a large margin classifier for the whole learning set. Since it is based only on distance calculations, on-chip learning can also be executed. The method is also appropriate for online and incremental learning, since model parameters are obtained directly from the data set, without need of user interaction for learning.

Introduction: Machine learning design usually requires user interaction to set model structure and learning parameters, and may also involve the implementation of complex optimisation algorithms. In addition, learning from data involves trading-off bias and variance [1, 2], which may also require setting regularisation parameters and performing cross-validation on data. These have been regarded as major limitations for the implementation of autonomous learning machines such as artificial neural networks (ANNs) on the integrated circuit (IC) level [3]. In the 1990s and early 2000s, ANN accelerator boards and ICs appeared on the market motivated mainly by the slow convergence rates of learning algorithms. Although ETANN [4], Intel's neural chip launched in 1989, had adaptive synapses, a host computer was needed to run the optimisation algorithms and to provide the user interface. In recent years, a new interest in embedded learning systems has been motivated by the rise of the Internet of Things and by the high throughput of data collection systems. IBM has recently launched a new high-performance neural chip [5], which is a further step towards next generation online adaptive systems.

The old limitations, however, still exist, since current high-performance learning algorithms, which treat learning as a trade-off problem, demand user interaction and complex computations, which are not feasible to be directly implemented in ICs. In this Letter, we present a new learning algorithm that does not depend on user-defined parameters to deal with the bias and variance dilemma [2]. The method takes into consideration only the structure of the data set in order to minimise the training set error and maximise the separation margin between classes. Since only distance computations are required, the new method is particularly suitable for implementation in application specific ICs (ASICs) and field-programmable gate arrays (FPGAs).

Data set structure and support vectors: Large margin classifiers became popular in the past two decades after the description of support vector machines (SVMs) [2]. In SVMs, margin maximisation is accomplished in feature space after kernel mapping and linearisation occurs. The resulting quadratic programming problem yielded by minimising the square norm of the weights ($||\boldsymbol{w}||^2 = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{w}$) subject to the error function constraint ($J_{\mathrm{e}}(\boldsymbol{w}) \leq \epsilon$) has a single minimum, subject to pre-established kernel and regularisation parameters [2].

In SVM's learning, support vectors (SVs) are an outcome of optimisation. However, the existence of SV is regarded in this Letter as a property of the problem, inherent to margin definition. According to this perspective, the maximum margin solution can be obtained by exploring the statistical properties and geometry of the learning set. This is supported by the demonstration that the SVM's solution can be approximated geometrically by finding the maximum margin separator in relation to the convex hulls of the two classes [6].

The distance-based large margin classifier presented next is based on a planar graph model of the data set, the Gabriel graph [7]. The resulting discrimination function is also supported by margin vectors, the structural SVs (SSVs), which are analogous to the SVM's SVs, but are obtained here directly from the structure of the data set. As will be shown next that only distance calculations are required to obtain the graph, the SSV and the resulting classifier.

Structural information extracted from a planar graph: Structural information is obtained from the corresponding Gabriel graph (G_{G}) [7], which depends only on pattern-to-pattern distances within the learning set. Considering the data set $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)|i = 1, \ldots, N\}$, with $y_i \in \{+1, -1\}$ and $\boldsymbol{x}_i \in \mathbb{R}^n$, the graph G_{G} of $\mathcal{D}$ with vertices $V = \{\boldsymbol{x}_i \in \mathcal{D}|i = 1, \ldots, N\}$ has edge E with vertices $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ if and only if $\delta(\boldsymbol{x}_i, \boldsymbol{x}_j)^2 \leq \left[\delta(\boldsymbol{x}_i, \boldsymbol{x}_k)^2 + \delta(\boldsymbol{x}_j, \boldsymbol{x}_k)^2\right]$, $\forall \boldsymbol{x}_k \in V$ and $i \neq j \neq k$, where $\delta(\cdot, \cdot)$ is the square Euclidean distance operator.

Support edges (SEs) are those edges with vertices $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ that belong to different classes, so they are located in the separation margin between classes. All those vertices $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ that form a SE are selected as SSVs. A local hyperplane H_{l} in the middle point of the SE corresponds to a maximum margin classifier in relation to $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$.

Combination of large margin classifiers: Local hyperplanes H_{l} are not expected to separate all the N patterns in $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)|i = 1, \ldots, N\}$, since they are based on a single SE. However, the combination of all hyperplanes yields a large margin classifier that is based on the spatial information of all patterns in $\mathcal{D}$, as shown in the example of Fig. 1.

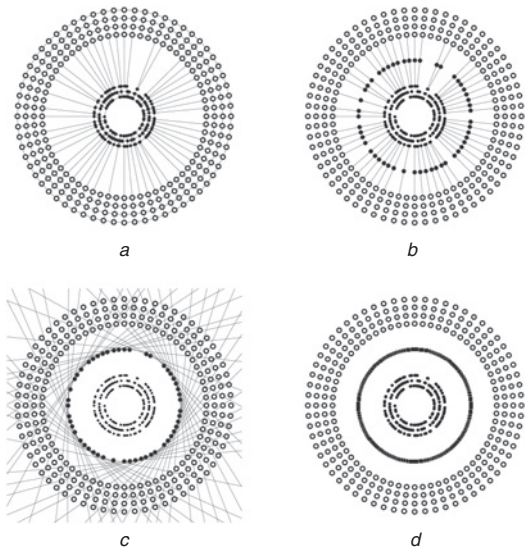

Fig. 1 *Combination of large margin classifiers yielded by SEs*
a G_{G} with original data set
b SEs with middle points
c Local hyperplanes
d Final classifier

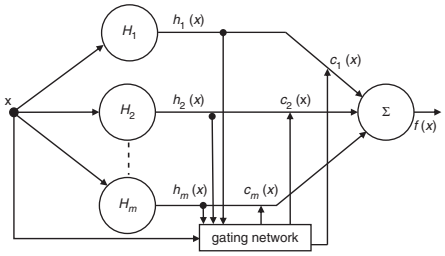

Fig. 2 *SEs yielding local experts of hierarchical mixture model*

Fig. 1*a* shows the original data set, formed by two concentric classes, with the corresponding G_{G}. Fig. 1*b* shows the data set with the obtained SEs and their middle points represented as dots. Fig. 1*c* shows all the classifiers corresponding to all middle points of all SEs and Fig. 1*d* the original data set and a continuous line indicating the projection in the plane of the separation surface due to the final combined classifier. In addition to separating the two classes, the resulting classifier has also maximised the separation margin, as can be seen in Fig. 1. As will be shown next, what differs the present method from previous ones is that its parameters are unique and can be obtained directly from the data set structure.

Hierarchical mixture of experts: The final classifier results from hierarchical mixture of experts [2] with a gating network that weights the relevance of each classifier (expert) in the mixture for a given input

pattern $\boldsymbol{x}$. The first layer of the mixture corresponds to the local experts, the classification functions $h_1(\boldsymbol{x}),\ \ldots,\ h_m(\boldsymbol{x})$, yielded by all m individual hyperplanes $\{H_1,\ \ldots,\ H_m\}$. As shown schematically in Fig. 2, the responses of the local experts are weighted by the gating network. Weighting parameters $c_l(\boldsymbol{x})$ are also obtained according to distance calculations, as will be shown next.

Let H_l be a local hyperplane yielded by SE with vertices $(\boldsymbol{x}_i,\ \boldsymbol{x}_j)$ such that $y_i = -1$ and $y_j = +1$. The classification outcome due to hyperplane H_l for an arbitrary pattern $\boldsymbol{x}$ is given by $h_l(\boldsymbol{x}) = \text{sign}(\boldsymbol{x}^{\text{T}}\boldsymbol{w}_l - b_l)$, where $\text{sign}(\cdot)$ is the sign function and $\boldsymbol{w}_l = (\boldsymbol{x}_i - \boldsymbol{x}_j)$, since it corresponds to the hyperplane that is perpendicular to the straight line connecting $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. Similarly, the bias term b_l is obtained by the expression $b_l = [(1/2)(\boldsymbol{x}_i + \boldsymbol{x}_j)]\boldsymbol{w}_l^{\text{T}}$. The middle point of each SE is also obtained directly as $p_l = (\boldsymbol{x}_i + \boldsymbol{x}_j)/2$. Therefore, the parameters of the local experts depend only on the vertices $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ and are then obtained by direct calculation. The weighting parameter $c_l(\boldsymbol{x})$ of H_l for an arbitrary pattern $\boldsymbol{x}$ is calculated according to (1)

$$c_l(\boldsymbol{x}) = \exp - \left(\frac{\{\max[\delta(\boldsymbol{x},\ p_k)]\}^2}{d(\boldsymbol{x},\ p_l)} \right), \quad \forall k = 1,\ \ldots,\ m \qquad (1)$$

A normalisation is also imposed so that $\sum_{l=1}^{m} c_l(\boldsymbol{x}) = 1$. The final classification is obtained as follows: $f(\boldsymbol{x}) = \text{sign}\left[\sum_{l=1}^{m} h_l(\boldsymbol{x})\ c_l(\boldsymbol{x})\right]$. The coefficients $c_l(\boldsymbol{x})$ of (1), which are combined with $h_l(\boldsymbol{x})$, yield larger values to those hyperplanes that are closer to $\boldsymbol{x}$. In practice, for circuit-level implementation, the final classification can be estimated solely by the nearest hyperplane.

Results: Experiments were performed with 13 real-world data sets from the UCI repository [8] and two gene expression problems: 'Golub' [9] and 'BcrHess' [10]. All these data sets have passed through the following preprocessing steps: noise filtering, removal of missing values and attribute rescaling to the range $\{-1,\ 1\}$. To ensure statistical relevance, the experiment was replicated using ten-fold cross-validation. The average performances (AUC) and standard deviations calculated after ten trials are listed in Table 1, along with some characteristics of the data sets in the last column. N and N_d correspond to the total amount of patterns and attributes, respectively.

Table 1: Results: average AUC and standard deviation. Best results are represented in bold.

Data set	New method	SVM-RBF	SVM-poly	N/N_d
Australian Cr.	0.85 ± 0.04	0.86 ± 0.04	**0.87 ± 0.04**	690/14
Banknote Auth.	0.98 ± 0.03	**1 ± 0**	**1 ± 0**	1372/4
BcrHess	**0.81 ± 0.12**	0.76 ± 0.11	0.77 ± 0.15	133/30
B. Cancer W.P.	0.96 ± 0.03	**0.97 ± 0.01**	0.96 ± 0.03	683/9
Climate M.S.C.	**0.84 ± 0.07**	0.53 ± 0.06	0.72 ± 0.11	540/18
Fertility	**0.59 ± 0.26**	0.5 ± 0	0.5 ± 0	100/9
German Cr.	0.67 ± 0.04	0.66 ± 0.07	**0.68 ± 0.05**	1000/24
Golub	0.77 ± 0.17	**0.8 ± 0.16**	0.78 ± 0.17	72/50
Haberman's S.	**0.57 ± 0.09**	0.52 ± 0.06	0.5 ± 0.02	306/3
ILPD	**0.56 ± 0.09**	0.49 ± 0.02	0.5 ± 0	579/10
liver disorders	0.61 ± 0.1	0.67 ± 0.05	**0.72 ± 0.07**	345/6
P. ind. diabetes	**0.72 ± 0.04**	0.71 ± 0.05	0.71 ± 0.07	768/8
Parkinsons	**0.9 ± 0.15**	0.77 ± 0.11	0.81 ± 0.12	195/22
Sonar. M against R.	**0.88 ± 0.08**	0.84 ± 0.09	0.87 ± 0.08	208/60
Stalog heart	0.8 ± 0.08	**0.83 ± 0.07**	**0.83 ± 0.07**	270/13
Average rank	1.87	2.23	1.90	

The SVMs method was used to compare our method. Two configurations of SVM Kernel were tested. First, SVM-RBF with radial basis function, and second SVM-poly with polynomial function. Kernel and regularisation parameters for SVM-RBF and SVM-poly were set via a grid search with ten-fold cross-validation using the packages *Kernlab* and *Caret* available for the language *R* (http://www.R-project.org/).

The Friedman's test [11] was applied assuming the null hypothesis H_0 that all algorithms are equivalent. In our experiment, the statistic F_F was distributed according to the F-distribution with two degrees of freedom for the numerator and 28 for the denominator. Considering that the critical value $F(2,\ 28)$ is 3.34 for $\alpha = 0.05$, the bottom row of Table 1 shows the average ranks $[R(\mathcal{L})]$ achieved by all algorithms. The corresponding value of F_F is 0.6, and thus since $F_F < F(2,\ 28)$, the hypothesis H_0 cannot be rejected. It is possible to conclude that our approach has outperformed the other methods for the data sets tested, according to the Friedman's test.

Conclusions: An autonomous learning approach is described in this Letter, which is based on the principle that SEs embody a maximum margin classifier in its middle point, which also minimises the error in relation to its vertices. Individual classifiers are then combined in order to obtain a global large margin classifier that also minimises the error of the learning set. SEs are unique for a given data set and can be obtained directly from the structural information extracted from a planar graph, which is described uniquely by the pairwise distances within the learning set. The resulting classifier does neither require any parameter to be set in advance nor user interaction to balance bias and variance trade-off. In addition, mainly distance calculations are involved, which makes the classifier particularly suitable for circuit-level implementation.

The basic data structure requires an $N \times (n+1)$ matrix for storing the learning set and another $m \times 2n$ matrix containing the parameters of the classifiers. This structure as well as the calculations for distance estimation during learning and classification are perfectly applicable for implementation in ASICs and FPGAs. As demonstrated in numerous works, both yield outstanding performance in speed, power and area for matrix operations with floating point numbers [12]. In addition, the presented method is well suited for autonomous and incremental learning, since the classifier can be updated without great impact on the current graph and hardware structures. In fact, a new learned pattern can be directly aggregated to the current data set without the need to recalculate the whole graph. It will only affect the current separation function if it results in a new SE, which can be simply added to the current set of SEs without disrupting the others.

Acknowledgments: The authors thank the CNPq and the CAPES for support.

Submitted: *12 May 2015* E-first: *30 October 2015*
doi: 10.1049/el.2015.1644

L.C.B. Torres, C.L. Castro, F. Coelho, F. Sill Torres and A.P. Braga (*Department of Electronic Engineering, Federal University of Minas Gerais, Belo Horizonte, Brazil*)

✉ E-mail: apbraga@ufmg.br

References

1 Costa, M.A., Braga, A.P., and de Menezes, B.R.: 'Convergence analysis of sliding mode trajectories in multi-objective neural networks learning', *Neural Netw.*, 2012, **33**, pp. 21–31
2 Haykin, S.: 'Neural networks: a comprehensive foundation' (Prentice-Hall, New Jersey, 1999, 2nd edn)
3 He, M., Klein, J.O., and Belhaire, E.: 'Design and electrical simulation of on-chip neural learning based on nanocomponents', *Electron. Lett.*, 2008, **44**, (9), pp. 575–576
4 Holler, M., Tam, S., Castro, H., *et al.*: 'An electrically trainable artificial neural network (ETANN) with 10240 "floating gate" synapses', *IEEE Neural Netw., IJCNN*, 1989, pp. 191–196
5 Hsu, J.: 'IBM's new brain [News]', *IEEE Spectrum*, 2014, **51**, (10), pp. 17–19
6 Bennett, K.P., and Bredensteiner, E.J.: 'Duality and geometry in SVM classifiers'. Proc. ICML, Standford, CA, USA, July 2000, pp. 57–64
7 Zhang, W., and Irwin, K.: 'A study of the relationship between support vector machine and Gabriel graph', *IEEE Neural Netw., IJCNN*, 2002, **1**, pp. 239–244
8 'UCI machine learning repository', 2013. [Online]. Available at http://www.archive.ics.uci.edu/ml
9 Golub, T.R., Slonim, D.K., Tamayo, P., *et al.*: 'Molecular classification of cancer: class discovery and class prediction by gene expression monitoring', *Science*, 1999, **286**, (5439), pp. 531–537
10 Hess, K.R., Anderson, K., Symmans, W.F., *et al.*: 'Pharmacogenomic predictor of sensitivity to preoperative chemotherapy with paclitaxel and fluorouracil, doxorubicin, and cyclophosphamide in breast cancer', *J. Clin. Oncol.*, 2006, **24**, (26), pp. 4236–4244
11 Demšar, J.: 'Statistical comparisons of classifiers over multiple data sets'. JMLR, 2006, vol. 7, pp. 1–30
12 Dou, Y., Vassiliadis, S., Kuzmanov, G., *et al.*: '64-bit floating-point FPGA matrix multiplication'. ACM Int. Symp. on Field-programmable Gate Arrays, Monterey, CA, USA, February 2005, pp. 86–95