Neural Computing and Applications

Robust Classification with Reject Option Using the Self-Organizing Map

--Manuscript Draft--

Manuscript Number:	NCA-3945R1
Full Title:	Robust Classification with Reject Option Using the Self-Organizing Map
Article Type:	Original Article
Keywords:	Self-Organizing Maps; Reject Option; Robust Classification; Prototype-based Classifiers; Neuron Labeling
Corresponding Author:	Ajalmar Rêgo da Rocha Neto Federal Institute of Ceara Fortaleza, BRAZIL
Corresponding Author Secondary Information:	
Corresponding Author's Institution:	Federal Institute of Ceara
Corresponding Author's Secondary Institution:	
First Author:	Ricardo Sousa, PhD
First Author Secondary Information:	
Order of Authors:	Ricardo Sousa, PhD
	Ajalmar Rêgo da Rocha Neto
	Jaime S. Cardoso, PhD
	Guilherme A. Barreto, PhD
Order of Authors Secondary Information:	
Response to Reviewers:	Response to Reviewers Manuscript: No. #NCA-3945 PAPER TITLE: Robust Classification with Reject Option Using the Self-Organizing Map June 17, 2014 First of all, we would like to thank the editor and the reviewers for their valuable, detailed thought-provoking comments. In the next lines, we try to address very closely all the suggestions, corrections and recommendations provided by each reviewer, aiming at answering them, hopefully in the most satisfactory way. The authors ANSWERS TO REVIEWERS REVIEWER 1 (R1) R1.1) “(...) which input vectors should be in the training data set should be at least addressed in the introduction (also know as instance or prototype selection).” Answer R1.1: DONE! We dedicated an entire new paragrah to this issue in the revised version. The 3rd paragraph of the introductory section addresses instance/prototype selection and discuss the similarities and differences with reject option. As a consequence, several new references have been included accordingly. Thank you very

much for the recommendation because it certainly enriched the content of the paper considerably.

R1.2) “Section 6 should include an independent subsection for each dataset analysed. Plot for synthetic 2-dimensional function could be included as well.”.

Answer R1.2: DONE! We have included a subsection to describe the datasets and a plot for the syntheticl dataset.

R1.3) “I think it would be interesting to add to the tables the time required to obtain each classification in order to see which method is faster (sometimes is not only about to get the best result but to get a solution on time).”

Answer R1.3: This is a very pertinent point that we would like to address with care, since it maybe was not clearly expressed on the manuscript. Our proposal of “Robust Classification with Reject Option Using the Self-Organizing Map” has the goal to suggest a new method for improved performance and recognition reliability with SOMs, and not on improving the training or running (testing) times of SOMs reject option algorithms. Truth must be said that training times of reject option algorithms will be significantly higher independently of their standard approaches (either with MLP, SOM or LVQ). For instance, for the ROSOM-1C it will take slightly longer times due to the extra parameter (wR) that has to be tuned during the cross-validation part. Not to say, that ROSOM-2C takes even more time given the twice number of classifiers that it has to train: It requires to train two classifiers for a binary problem, and 2*K classifiers (when using a One-Versus-All approach) for a multiclass (K-class) problem.

Having said that, it should now be clear that no matter the improvements made on the algorithms, the training times of reject option algorithms will be inexorably higher than their standard approaches. Regarding the running times, the rationale is analogous. For the SOM1C the running time is the same as the standard approach since neuron labelling is conducted during the training phase. A testing instance is labelled accordingly since the quantity P(Ck | wj, x) express the probability of an instance that falls within the Voronoi cell of neuron j to belong to class Ck. The running time of ROSOM-2C will be higher than the baseline for obvious reasons.

In conclusion, training or running times of SOMs reject option algorithms leading to major efforts in improving and proposing robust classifiers for reliable recognition tasks.

R1.4) “In order to improve tables, bold font could be used to identify easily the best results. Furthermore, the dataset could be added as a first row and then the caption could be used to provide more information about the data displayed in the table.”

Answer R1.4: DONE! We have highlighted the best results in boldface in both Tables 1 and 2. In Table 2, for each value of the reject rate r, we highlighted the performances of the two best results since they are statistically equivalent. Note, however, a full analysis of the results must be conducted with the help of Figure 4. We have included this information in the captions of Tables 2a and 2b.

REVIEWER 2 (R2)

R2.1) “Post-labelling is still the main method to turn SOM into a classifier. Labelling methods and further kernelising techniques can help improve the performance by significant margins - see: Lau, Yin and Hubbard, "Kernel self-organising maps for classification", Neurocomputing, 69(16-18), pp.2033-2040 (2006).”

Answer R2.1: We agree with the reviewer. Inthis regard, we now include this reference in the paper and also included in the 4th paragraph of the 2nd page of the Introduction the following sentence:

“Teuvo Kohonen, the proponent of the SOM himself, developed the first application of the SOM as a supervised pattern classifier in his neural phonetic typewriter [36], but several strategies for this purpose have been devised since then, with the post-training labelling of SOM prototypes being the most common one. Furthermore, kernelising techniques can be used to improve the performance of the SOM as a classifier

	considerably (see [40], for example). R2.2) “Standard SOM does not approximate probability densities. This could be the reason why the proposed methods do not seem to perform as well as expected (as the toolbox was used).” Answer R2.2: We agree with the reviewer that the SOM does not approximate probability densities accurately, but it does approximate at some degree. SOM-based density estimation can be carried out with satisfactory degree of accuracy through the use of Gaussian mixture modelling. In fact, we have done that for the ROSOM-1C. In the revised version we discussed this approach in the 2nd paragraph of Subsection 5.1.1 and cited suitable methods for doing that using the SOM, such as the SOMN (self-organizing mixture network, [68]) and the SO-RKDE (self-organizing reduced kernel density estimation, [1]). In the simulations, we decided to use the SO-RKDE for its simplicity, but we could have equally used the SOMN. In what concern the performances of the proposed ROSOM-C1 and ROSOM-C2 classifiers, we do not agree that they did not perform as well as expected. On the contrary, during the development of the research reported in the paper, we had no expectations at all. We decided to build SOM-based classifiers and endow them with reject option class because our group had previous successful experiences with the SOM in real-world applications, both in industrial environments and technology-oriented companies. Given this previous experiences with the SOM, this neural network came to be a natural choice to be evaluated as a classfier with reject option mechanisms. At the end, we got very surprised that the proposed SOM-based classifiers achieved results that could rivalize with those produced by standard supervised classifiers, including LVQ-, MLP- and SVM-based classifiers (see the answer to the next comment of this reviewer). For us, this was a remarkable result to be shared with others. R2.3) “Comparison should also include standard SOM with various post-labelling methods, as well as SVM if possible, to benchmark the performance of the proposed methods.” Answer R2.3: Indeed, we carried out experiments with several neuron labelling methods (at least, the ones evaluated in reference [43]) and choose two of them. We did not find basically statistically significant difference among the several neuron labelling methods, but we agree with the reviewer that this issue could be evaluated more deeply. We decided not to include all these experiments in the paper to not overload it with not relevant results. At the end, the post-labelling method was chosen because it is probably the most common method. The self-supervised method was chosen because historically it was used by Kohonen himself in reference [36]. In what concern the comparison with SVM-based classifiers, we included a complete set of new experiments with an SVM classifier endowed with an embedded reject option strategy as proposed by Fumera and Roli in reference [21]. The obtained results are discussed in Section 6 and compared with those achieved by the proposed SOM-based classifiers.

Response to Reviewers

Manuscript: No. #NCA-3945

PAPER TITLE: Robust Classification with Reject Option Using the Self-Organizing Map

---

June 17, 2014

First of all, we would like to thank the editor and the reviewers for their valuable, detailed thought-provoking comments. In the next lines, we try to address very closely all the suggestions, corrections and recommendations provided by each reviewer, aiming at answering them, hopefully in the most satisfactory way.

The authors

ANSWERS TO REVIEWERS

REVIEWER 1 (R1)

R1.1) **"(...) which input vectors should be in the training data set should be at least addressed in the introduction (also know as instance or prototype selection)."**

Answer R1.1: DONE! We dedicated an entire new paragrah to this issue in the revised version. The 3rd paragraph of the introductory section addresses instance/prototype selection and discuss the similarities and differences with reject option. As a consequence, several new references have been included accordingly. Thank you very much for the recommendation because it certainly enriched the content of the paper considerably.

R1.2) **"Section 6 should include an independent subsection for each dataset analysed. Plot for synthetic 2-dimensional function could be included as well."**.

Answer R1.2: DONE! We have included a subsection to describe the datasets and a plot for the syntheticI dataset.

R1.3) **"I think it would be interesting to add to the tables the time required to obtain each classification in order to see which method is faster (sometimes is not only about to get the best result but to get a solution on time)."**

Answer R1.3: This is a very pertinent point that we would like to address with care, since it maybe was not clearly expressed on the manuscript. Our proposal of "Robust Classification with Reject Option Using the Self-Organizing Map" has the goal to suggest a new method for improved performance and recognition reliability with SOMs, and not on improving the training or running (testing) times of SOMs reject option algorithms. Truth must be said that training times of reject option algorithms will be significantly higher independently of their standard approaches (either with MLP, SOM or LVQ). For instance, for the ROSOM-1C it will take slightly longer times due to the extra parameter (w_R) that has to be tuned during the cross-validation part. Not to say, that ROSOM-2C takes even more time given the twice number of classifiers that it has to train: It requires to train two classifiers for a binary problem, and 2*K classifiers (when using a One-Versus-All approach) for a multiclass (K-class) problem.

Having said that, it should now be clear that no matter the improvements made on the algorithms, the training times of reject option algorithms will be inexorably higher than their standard approaches. Regarding the running times, the rationale is analogous. For the SOM1C the running time is the same as the standard approach since neuron labelling is conducted during the training phase. A testing instance is labelled accordingly since the quantity $P(C_k \mid \mathbf{w}_j, \mathbf{x})$ express the probability of an instance that falls within the Voronoi cell of neuron j to belong to class C_k. The running time of ROSOM-2C will be higher than the baseline for obvious reasons.

In conclusion, training or running times of SOMs reject option algorithms leading to major efforts in improving and proposing robust classifiers for reliable recognition tasks.

R1.4) **"In order to improve tables, bold font could be used to identify easily the best results. Furthermore, the dataset could be added as a first row and then the caption could be used to provide more information about the data displayed in the table."**

Answer R1.4: DONE! We have highlighted the best results in boldface in both Tables 1 and 2. In Table 2, for each value of the reject rate r, we highlighted the performances of the two best results since they are statistically equivalent. Note, however, a full analysis of the results must be

conducted with the help of Figure 4. We have included this information in the captions of Tables 2a and 2b.

REVIEWER 2 (R2)

R2.1) **"Post-labelling is still the main method to turn SOM into a classifier. Labelling methods and further kernelising techniques can help improve the performance by significant margins - see: Lau, Yin and Hubbard, "Kernel self-organising maps for classification", Neurocomputing, 69(16-18), pp.2033-2040 (2006)."**

Answer R2.1: We agree with the reviewer. Inthis regard, we now include this reference in the paper and also included in the 4th paragraph of the 2nd page of the Introduction the following sentence:

"Teuvo Kohonen, the proponent of the SOM himself, developed the first application of the SOM as a supervised pattern classifier in his neural phonetic typewriter [36], but several strategies for this purpose have been devised since then, with the post-training labelling of SOM prototypes being the most common one. Furthermore, kernelising techniques can be used to improve the performance of the SOM as a classifier considerably (see [40], for example).

R2.2) **"Standard SOM does not approximate probability densities. This could be the reason why the proposed methods do not seem to perform as well as expected (as the toolbox was used)."**

Answer R2.2: We agree with the reviewer that the SOM does not approximate probability densities accurately, but it does approximate at some degree. SOM-based density estimation can be carried out with satisfactory degree of accuracy through the use of Gaussian mixture modelling. In fact, we have done that for the ROSOM-1C. In the revised version we discussed this approach in the 2nd paragraph of Subsection 5.1.1 and cited suitable methods for doing that using the SOM, such as the SOMN (self-organizing mixture network, [68]) and the SO-RKDE (self-organizing reduced kernel density estimation, [1]). In the simulations, we decided to use the SO-RKDE for its simplicity, but we could have equally used the SOMN.

In what concern the performances of the proposed ROSOM-C1 and ROSOM-C2 classifiers, we do not agree that they did not perform as well as expected. On the contrary, during the development of the research reported in the paper, we had no expectations at all. We decided to build SOM-based classifiers and endow them with reject option class because our group had previous successful experiences with the SOM in real-world applications, both in industrial environments and technology-oriented companies. Given this previous experiences with the SOM, this neural network came to be a natural choice to be evaluated as a classfier with reject option mechanisms. At the end, we got very surprised that the proposed SOM-based classifiers achieved results that could rivalize with those produced by standard supervised classifiers, including LVQ-, MLP- and SVM-based classifiers (see the answer to the next comment of this reviewer). For us, this was a remarkable result to be shared with others.

R2.3) **"Comparison should also include standard SOM with various post-labelling methods, as well as SVM if possible, to benchmark the performance of the proposed methods."**

Answer R2.3: Indeed, we carried out experiments with several neuron labelling methods (at least, the ones evaluated in reference [43]) and choose two of them. We did not find basically statistically significant difference among the several neuron labelling methods, but we agree with the reviewer that this issue could be evaluated more deeply. We decided not to include all these experiments in the paper to not overload it with not relevant results. At the end, the post-labelling method was chosen because it is probably the most common method. The self-supervised method was chosen because historically it was used by Kohonen himself in reference [36].

In what concern the comparison with SVM-based classifiers, we included a complete set of new experiments with an SVM classifier endowed with an embedded reject option strategy as proposed by Fumera and Roli in reference [21]. The obtained results are discussed in Section 6 and compared with those achieved by the proposed SOM-based classifiers.

Noname manuscript No.
(will be inserted by the editor)

Robust Classification with Reject Option Using the Self-Organizing Map

Ricardo Sousa* · **Ajalmar R. da Rocha Neto*** · **Jaime S. Cardoso** · **Guilherme A. Barreto**

the date of receipt and acceptance should be inserted later

Abstract Reject option is a technique used to improve classifier's reliability in decision support systems. It consists in withholding the automatic classification of an item, if the decision is considered not sufficiently reliable. The rejected item is then handled by a different classifier or by a human expert. The vast majority of the works on this issue has been concerned with the developement of reject option mechanisms to be used by supervised learning architectures (e.g., MLP, LVQ or SVM). In this paper, however, we aim at proposing alternatives to this view which are based on the Self-Organizing Map (SOM), originally an unsupervised learning scheme, but that has also been successfully used in the design of prototype-based classifiers. The basic hypothesis we defend is that it is possible to design SOM-based classifiers endowed with reject option mechanisms whose performances are comparable to or better than those achieved by standard supervised classifiers. For this purpose, we carried out a comprehensively evaluation of the proposed SOM-based classifiers on two synthetic and three real-world data sets. The obtained results suggest that the proposed SOM-based classifiers consistently outperform standard supervised classifiers.

Ricardo Sousa*
INEB – Instituto de Engenharia Biomédica, Universidade do Porto,
Porto, Portugal,
E-mail: contact@rsousa.org

Ajalmar R. da Rocha Neto*
Programa de Pós-Graduação em Engenharia de Telecomunicações, Instituto Federal do Ceará (IFCE),
Fortaleza, Ceará, Brasil,
E-mail: ajalmar@ifce.edu.br
*First two authors contributed equally to this manuscript.

Jaime S. Cardoso
INESC Porto, Faculdade de Engenharia da Universidade do Porto,
Porto, Portugal,
E-mail: jaime.cardoso@inescporto.pt

Guilherme A. Barreto
Departamento de Engenharia de Teleinformática, Universidade Federal do Ceará (UFC),
Fortaleza, Ceará, Brasil,
E-mail: gbarreto@ufc.br

Keywords Self-Organizing Maps, Reject Option, Robust Classification, Prototype-based Classifiers, Neuron Labeling

1 Introduction

The field of machine learning has been evolving at a very fast pace, being mostly motivated and pushed forward by increasingly challenging real world applications. For instance, in credit scoring modeling, models are developed to determine how likely applicants are to default with their repayments. Previous repayment history is used to determine whether a customer should be classified into a 'good' or a 'bad' category [58]. Prediction of insurance companies' insolvency has arisen as an important problem in the field of financial research, due to the necessity of protecting the general public whilst minimizing the costs associated to this problem [58]. In medicine, the last decades have witnessed the development of advanced diagnostic systems as alternative, complementary or a first opinion in many applications [3].

Notwithstanding, real world problems still pose challenges which may not be solvable satisfactorily by the existing learning methodologies used by automatic decision support systems [22, 27, 29], leading to many incorrect predictions. This is particularly true for conventional learning systems (e.g. neural networks), in which the number of possible outputs is equal to the number of class labels. For instance, in a binary classification task, the possible outputs are encoded as good (normal) or bad (abnormal) categories. However, there are situations in which the decision should be postponed, giving the support system the opportunity to identify critical items for posterior revision, instead of trying to automatically classify every and each item. In such cases, the system automates only those decisions which can be reliably predicted, letting the critical ones for a human expert to analyze. Therefore, the development of binary classifiers with a third output class, usually called the *reject class*, is attractive. This approach is known as classification with reject option [11, 16, 31] or soft decision making [33]. Roughly speaking, reject option comprises a set of techniques aiming at improving the classification reliability in decision support systems, being originally formalized in the context of statistical pattern recognition in [11], under the minimum risk theory. Basically, it consists in withholding the automatic classification of an item, if the decision is considered not sufficiently reliable. Rejected patterns can then be handled by a different classifier, or manually by a human. Implementation of reject option strategies requires finding a trade-off between the achievable reduction of the cost due to classification errors, and the cost of handling rejections (which are application-dependent).

Reject option can be seen as an alternative to data subset selection strategies widely known as *instance selection*, *prototype selection* or yet, in a broader framework, *active learning* [19, 24, 28, 30, 45]. Roughly speaking, instance/prototype selection mechanisms aim at selecting a suitable subset of input patterns that are included in the training set of the classifier. By *suitable subset* we mean the smallest set of representative training patterns that eventually lead to the design of accurate classifiers. Thus, the ultimate goal of an instance selection mechanism besides reducing data storage costs is to increase the accuracy of the classifier by making it more insensitive to noisy/redundant/ambiguous patterns. Similar goal is pursued by reject option mechanisms, but the way they are implemented differs considerably. While instance selection mechanisms are commonly applied *before* training the classifier, rejection option thresholds are usually computed *after* training the classifier[1].

[1] There are reject option strategies which are executed *during* the training of the classifier. These are known as embedded reject option mechanisms. See Section 2 for more detail.

It is worth mentioning, however, that rejection option strategies has been recently used for the purpose of instance selection [55]. This can be done by eliminating from the those training instances which are rejected by the classifier.

Despite its potential advantage, the problem of classification with a reject option has been tackled only occasionally in machine learning literature, using supervised learning methods, such as the MLP, LVQ and SVM classifiers.

Historically, modifications of supervised neural network classifiers in order to include reject option date back to the first half of the 1990s. The works of Vasconcelos *et al.* [64,65] and Cordella *et al.* [13] pioneered in proposing reject option strategies specifically for the MLP network. More or less at the same time, Cordella *et al.* [12] developed a reject option strategy to be used by the LVQ network. Later, De Stefano *et al.* [15] generalized the works in [12, 13] by introducing a general framework for endowing supervised neural classifiers with reject option mechanisms. They successfully tested their approach in MLP, LVQ and RBF classifiers. In recent years, contributions focusing in a specific neural learning method, such as the LVQ [57] and the MLP [23], can still be found, but several works have aimed at the development of reject option mechanisms for SVM-like kernel classifiers [2,6,21,54,69].

In parallel, comparative studies involving several classification paradigms and different rejection option strategies have been carried out by many authors. For example, Fumera *et al.* [20] performed computer experiments for text categorization with three kinds of classifiers commonly used in the literature (i.e. *k*-NN, MLP and SVM). They concluded that the reject option can indeed significantly improve the performance of a text categorisation system, at the expense of a reasonably small rate of rejected decisions for each category. Tortorella [59] presented an optimal reject rule for binary classifiers based on the Receiver Operating Characteristic (ROC) curve. The rule is optimal since it maximizes a classification utility function, defined on the basis of classification and error costs particular for the application at hand. Santos-Pereira and Pires [50] established a connection between Tortorella's approach, which is based on ROC curves, and a generalization of Chow's optimal rejection rule. Finally, Lotte *et al.* [41] evaluated pattern rejection strategies for self-paced Brain-Computer Interfaces. Best results were achieved using the reject option and nonlinear classifiers, such as a Gaussian SVM, a fuzzy inference system or an RBF network.

A common feature of all the aforementioned works on reject option is that they have been implemented using supervised classifiers. As a feasible alternative, we advocate the use of classifiers built from the Self-Organizing Map (SOM) [37]. The SOM is originally an unsupervised learning algorithm, but it has been successfully applied to supervised pattern classification tasks (see [40, 43, 53, 56, 60] and references therein). Teuvo Kohonen, the proponent of the SOM himself, developed the first application of the SOM as a supervised pattern classifier in his neural phonetic typewriter [36], but several strategies for this purpose have been devised since then, with the post-training labelling of SOM prototypes being probably the most common one. Furthermore, kernelising techniques can be used to improve the performance of the SOM as a classifier considerably (see [40], for example).

It is worth mentioning that SOM-based classifiers belong to the class of prototype-based classifiers, to which also belong LVQ [5, 51] and ARTMAP classifiers [9]. Such classifiers possess two desirable properties which are hard to find in standard MLP and SVM classifiers. Firstly, due to the local nature of prototype-based classifiers, interpretation of the decisions in terms of local explanatory rules associated to each prototype is facilitated. Secondly, prototype-based classifiers are easily endowed with adaptive strategies for adding and deleting prototypes to fit the current data distribution, a valuable property specially in evolving, nonstationary environments.

Despite LVQ-based classifiers with reject option being available since a while [12, 57], the same is not true for SOM-based classifiers. Would SOM-based classifiers with reject option perform better than the LVQ-based counterpart? Further, in a broader perspective, would SOM-based classifiers with reject option perform better than its supervised counterparts?

Bearing these thought-provoking questions in mind, in this paper we develop two novel strategies to design SOM-based classifiers with reject option, and compare their performances with those of the MLP, LVQ and SVM classifiers also endowed with reject option mechanisms. For this purpose, we promote a comprehensively evaluation of all these classifiers on two synthetic and three real-world data sets. It is worth emphasizing that the strategies proposed in this work are robust with respect to their capacity of controling the confidence to label a new input vector as belonging to a certain class.

The remainder of this paper is organized as follows. Fundamental concepts regarding the reject option are described in Section 2, followed by a brief overview of the SOM in Section 3. In Section 5 our proposals are delineated to incorporate reject option in SOMs and a thoroughly experimentation is described in Section 6. Finally, in Section 7 conclusions are drawn.

2 Basics of Classification with Reject Option

As mentioned before, in possession of a "complex" dataset (e.g. from a medical diagnosis problem), every classifier is bound to misclassify some data samples. Depending on the costs of the errors, misclassification can lead to very poor classifier's performance. Therefore, techniques where the classifier can abstain from providing a decision by delegating it to a human expert (or to another classifier) is very appealing. In the following, we limit the discussion of reject option strategies to the binary classification problem. For that, we assume that the problem (and hence, the data) involves only two classes, say $\{\mathcal{C}_{-1}, \mathcal{C}_{+1}\}$, but the classifier must be able to output a third one, the reject class $\{\mathcal{C}_{-1}, \mathcal{C}_{\text{Reject}}, \mathcal{C}_{+1}\}$).

Assuming that the input information is represented by an n-dimensional real vector $\mathbf{x} = [x_1 \ x_2 \ \cdots \ x_n]^T \in \mathbb{R}^n$, the design of classifiers with reject option can be systematized in three different approaches for the binary problem:

1. **Method 1**: It involves the design of a single, standard binary classifier. If the classifier provides some approximation to the a posteriori class probabilities, $\mathbb{P}(\mathcal{C}_k|\mathbf{x})$, $k = 1, 2, ..., K$, then a pattern is rejected if the largest value among the K posterior probabilities is lower than a given threshold, say β ($0 \leq \beta \leq 1$) [21,54]. More formally, according to Chow [11] one holds a decision if

$$\max_k [\mathbb{P}(\mathcal{C}_k|\mathbf{x})] < \beta, \tag{1}$$

or, equivalently,

$$\max_k [\mathbb{P}(\mathbf{x}|\mathcal{C}_k)\mathbb{P}(\mathcal{C}_k)] < \beta, \tag{2}$$

where $\mathbb{P}(\mathcal{C}_k)$ is the a priori probability distribution of the k-th class and $\mathbb{P}(\mathbf{x}|\mathcal{C}_k)$ is the conditional probability density for the pattern $\mathbf{x}$ given the k-th class. If the classifier does not provide probabilistic outputs, then a rejection threshold targeted to the particular classifier's output should be used [33]. In this case, reject the classification of $\mathbf{x}$ if

$$\max_k \{o_k\} < \beta, \tag{3}$$

where o_k is the k-th output of the classifier, $k = 1,2...,K$. For the binary classification problem, we have $K = 2$.
For this method, the classifier is trained as usual (i.e. without referring to an explicit rejection class); but rather, the rejection region is determined *after* the training phase, heuristically or based on the optimization of some post-training criterion that weighs the trade-off between the costs of misclassification and rejection.

2. **Method 2**: The design of two, *independent*, classifiers. A first classifier is trained to output $\mathcal{C}_{-1}$ only when the probability of $\mathcal{C}_{-1}$ is high and a second classifier trained to output $\mathcal{C}_{+1}$ only when the probability of $\mathcal{C}_{+1}$ is high. When both classifiers agree on the decision, the corresponding class is outputted. Otherwise, in case of disagreement, the reject class is the chosen one. The intuitive idea behind this approach is that if both classifiers have high levels of confidence in their decisions then the aggregated decision should be correct in case of agreement. In case of disagreement, the aggregated decision is prone to be unreliable and hence rejection would be preferable [54].
3. **Method 3**: The design of a single classifier with embedded reject option; that is, the classifier is trained following optimality criteria that automatically take into account the costs of misclassification and rejection in their loss functions, leading to the design of algorithms specifically built for this kind of problem [6, 21, 54].

Later in this paper, we will introduce two SOM-based (and similarly two LVQ-based) strategies that instantiate the classification with reject option paradigms described above as Methods 1 and 2.

3 The Self-Organizing Map

The Self-Organizing Map (SOM) [34, 37] is one of the most popular neural network architectures. It belongs to the category of unsupervised competitive learning algorithms and it is usually designed to build an ordered representation of spatial proximity among vectors of an unlabeled data set. The SOM has been widely applied to pattern recognition and classification tasks, such as clustering, vector quantization, data compression and data visualization. In these applications, the weight vectors are called *prototypes* or *centroids* of clusters of input vectors, being obtained usually through a process of learning.

The neurons in the SOM are put together in an output layer, $\mathcal{A}$, in one-, two- or even three-dimensional arrays. Each neuron $j \in \mathcal{A}$, $j = 1,2,\ldots,q$, has a weight vector $\mathbf{w}_j \in \mathbb{R}^d$ with the same dimension of the input vector $\mathbf{x} \in \mathbb{R}^d$. The network weights are trained according to a competitive-cooperative learning scheme in which the weight vectors of a winning neuron (also called, the best-matching unit - BMU) and its neighbors in the output array are updated after the presentation of an input vector. Roughly speaking, the functioning of this type of learning algorithm is based on the concept of *winning neuron*, defined as the neuron whose weight vector is the closest to the current input vector.

Using Euclidean distance, the simplest strategy to find the winning neuron, $i(n)$, is given by:

$$i(n) = \arg\min_{\forall j} \|\mathbf{x}(n) - \mathbf{w}_j(n)\| \tag{4}$$

where $\mathbf{x}(n) \in \mathbb{R}^d$ denotes the current input vector, $\mathbf{w}_j(n) \in \mathbb{R}^d$ is the weight vector of neuron j, and n denotes the current iteration. Accordingly, the weight vectors are adjusted by the following iterative equation:

$$\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \eta(n)h(j,i;n)[\mathbf{x}(n) - \mathbf{w}_j(n)], \tag{5}$$

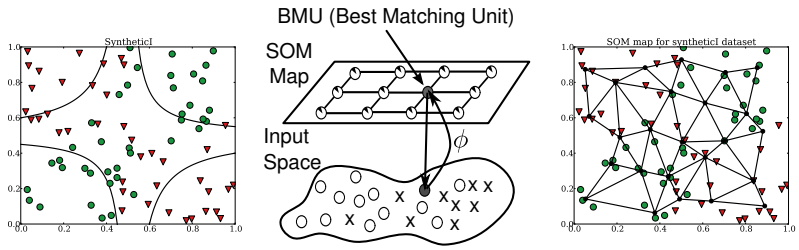


Fig. 1: Example of a SOM as a compact, topology-preserving, representation of a synthetic dataset (left figure). A mapping (ϕ) is learned in order to reflect the input data distribution (center figure). The distribution of the weight vectors of the SOM in the input space, where neighboring prototypes in the output grid are shown connected in the input space (right figure).

where $h(j,i;n)$ is a Gaussian function which control the degree of change imposed to the weight vectors of those neurons in the neighborhood of the winning neuron:

$$h(j,i;n) = \exp\left(-\frac{\|\mathbf{r}_j(n)-\mathbf{r}_i(n)\|^2}{\sigma^2(n)}\right) \tag{6}$$

where $\sigma(n)$ defines the radius of the neighborhood function, $\mathbf{r}_j(n)$ and $\mathbf{r}_i(n)$ are, respectively, the coordinates of neurons j and i in the array. The learning rate, $0 < \eta(n) < 1$, should decrease gradually with time to guarantee convergence of the weight vectors to stable states. In this paper, we use $\eta(n) = \eta_0(\eta_T/\eta_0)^{(n/T)}$, where η_0 and η_T are the initial and final values of $\eta(n)$, respectively. The variable $\sigma(n)$ should also decrease with time similarly to the learning rate $\eta(n)$.

The SOM has several features which make it a valuable tool in data mining applications [46]. For instance, the use of a neighborhood function imposes an order to the weight vectors, so that, at the end of the training phase, input vectors that are close in the input space are mapped onto the same winning neuron or onto winning neurons that are close in the output array. This is the so-called *topology-preserving property* of the SOM, which has been particularly useful for data visualization purposes [17].

Once the SOM converges, the set of ordered weight vectors summarizes important statistical characteristics of the input (see Fig. 1). The SOM should reflect variations in the statistics of the input distribution [47]: regions in the input space $\mathcal{X}$ from which a sample $\mathbf{x}$ are drawn with a high probability of occurrence are mapped onto larger domains of the output space A, and therefore with better resolution than regions in $\mathcal{X}$ from which sample vectors are drawn with a low probability of occurrence. For the interested reader, further information about the SOM and applications can be found in [63] and [67].

3.1 SOM for Supervised Classification

In order to use the SOM for supervised classification, modifications are necessary in its original learning algorithm. There are many ways to do that (see [40, 43] and references therein), but in the present paper we will resort to two well-known strategies.

Strategy 1 - The first strategy involves a post-training neuron labeling. It consists firstly in training the SOM in the usual unsupervised way until convergence of the weights. Once training is finished, one has to present the whole training data once again to the SOM in

order to find the winning neuron for each pattern vector. A given neuron can be selected the winner for pattern vectors belonging to different classes. However, among all the patterns a given neuron was selected the winner, the number of exemplars of a given class usually is higher that the number of exemplars of other classes. Hence, a class label is assigned to a neuron on a majority voting basis, i.e. a neuron receives the label of the class with the highest number of exemplars.

Two undesirable situations may occur: (*i*) ambiguity or (*ii*) dead neurons. Ambiguity occur when the frequency of the class labels of the patterns mapped to a given neuron are equivalent. Dead neurons are those never selected as winners for any of the input patterns. In these cases, the neuron could be pruned (i.e. disregarded) from the map, or even be tagged with a "rejection class" label.

For the case in which these neurons receive the "rejection class" label, whenever any of them is selected as winner for a new incoming pattern, this pattern is then rejected. Despite its simplicity, this approach does not give the user the freedom of searching for an acceptable degree of rejection for a given problem by means of the specification of the rejection cost ω_r, an approach introduced by Chow [11]. The rejection cost impacts directly on the ratio between the total number of rejected patterns and the number of misclassified patterns. A classifier with high ω_r tends to reject just a few patterns (in other words, it is costly to reject a pattern); thus increasing its misclassification rate. A classifier with low ω_r tends to reject a high number of patterns, thus decreasing its misclassification rate[2].

In this paper, we extend Strategy 1 in order to allow the SOM network to handle pattern classification problems with reject option. For this purpose, we follow a more systematic and principled approach based on Chow's concept of rejection cost [11], instead of simply tagging ambiguous or dead neurons with "rejection class" labels.

Strategy 2 - The second strategy, usually called the *self-supervised SOM* training scheme, is the one used by Kohonen for the neural phonetic typewriter [36]. According to this strategy, the SOM is made supervised by adding class information to each input pattern vector. Specifically, the input vectors $\mathbf{x}(n)$ are now formed of two parts, $\mathbf{x}_p(n)$ and $\mathbf{x}_l(n)$, where $\mathbf{x}_p(n)$ is the pattern vector itself, while $\mathbf{x}_l(n)$ is the corresponding class label of $\mathbf{x}_p(n)$. During training, these vectors are concatenated to build augmented vectors $\mathbf{x}(n) = [\mathbf{x}_p(n)\, \mathbf{x}_l(n)]^T$ which are used as inputs to the SOM. The corresponding augmented weight vectors, $\mathbf{w}_j(n) = [\mathbf{w}_j^p(n)\ \ \mathbf{w}_j^l(n)]^T$, are adjusted as in the usual SOM training procedure.

Usually, the label vector $\mathbf{x}_l(n)$ is represented as a unit-length binary vector; that is, only one of its components is set to "1", while the others are set to "0". The index of the "1" position indicates the class of the pattern vector $\mathbf{x}_p(n)$. For example, if three classes are available, then three label vectors are possible: one for the first class ([1 0 0]), one for the second class ([0 1 0]) and one for the third class ([0 0 1]).

For the classification of an unknown pattern $\mathbf{x}(n)$, the $\mathbf{x}_l(n)$ part is not considered, i.e. only its $\mathbf{x}_p$ part is compared with the corresponding part of the weight vectors. However, the class label of the unknown pattern vector is decided on the basis of the $\mathbf{w}_i^l(n)$ part of the winning weight vector $\mathbf{w}_i(n)$. The index of the component of $\mathbf{w}_i^l(n)$ with largest value defines the class label of the unknown pattern vector $\mathbf{x}_p$.

[2] At the limit, a classifier with a very low ω_r would classify only the so-called "easy patterns".

4 The Learning Vector Quantization (LVQ)

Introduced by Kohonen [35, 39], LVQ comprises a class of competitive learning algorithms for nearest prototype classification (NPC) [45, 52, 61]. In NPC, the discriminant functions are parametrized using a set of prototype vectors for each class, and classification is based on the distance between a data point and the class to which its closest prototype belongs to. Often an Euclidean distance measure is used, but other measures including divergences can be equally used [66]. Being a popular approach to pattern classification, several versions of LVQ algorithms are available in the literature. For recent advances on LVQ theory, the interested reader is referred to [5, 51].

In LVQ classifiers, a set of q prototype vectors, $\{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_q\}$, $\mathbf{w}_j \in \mathbb{R}^p$, are initialized with data vectors randomly sampled from different classes and inherit their labels. Given an input pattern $\mathbf{x}(n) \in \mathbb{R}^p$ at the n-th learning iteration, competition between neurons is implemented using Equation (4) in order to find the winning neuron j^* at time step n. Then, the weight vector associated with the winning neuron $j^*(n)$ is updated as follows

$$\mathbf{w}_{j^*}(n+1) = \mathbf{w}_{j^*}(n) + s(n)\eta(n)[\mathbf{x}(n) - \mathbf{w}_{j^*}(n)], \tag{7}$$

where $\eta(n)$ is the learning rate. We set $s(n) = 1$, if the label of the winning neuron $j^*(n)$ is the same as the one of the input pattern; otherwise, we set $s(n) = -1$.

Once the training phase is finished, the LVQ network follows the nearest neighbor rule for pattern classification purposes, i.e. the class label for any new input pattern is the label associated with its nearest weight vector found by Equation (4).

5 Incorporating Reject Option into the SOM: Two Proposals

Before proceeding with the description of the two proposals, it is worth exposing the main reasons that led to the choice of the SOM for supervised classification with rejection option instead of other prototype-based classifiers. Firstly, it has been verified that the use of a neighborhood function makes the SOM less sensitive to weight initialization [38] and accelerates its convergence [14] when compared with other prototype-based classifiers, such as the LVQ. Once trained, one can also take advantage of the SOM's density matching and topology-preserving properties to extract rules from a trained SOM network [42] in order to permit further analysis of the results towards better decision making.

In particular, the density matching and topology-preserving properties will be used by both proposals to be described in order to estimate $\mathbb{P}(\mathbf{x}|\mathcal{C}_k)$ (or $\mathbb{P}(\mathcal{C}_k|\mathbf{x})$) using the distribution of SOM's weight vectors. An optimal threshold value has to be determined in order to re-tag some of the weight vectors with the rejection class label. In this paper we will also discuss briefly techniques to obtaining suitable estimates of the likelihood function $\mathbb{P}(\mathbf{x}|\mathcal{C}_k)$ or the posterior probability $\mathbb{P}(\mathbf{x}|\mathcal{C}_k)$.

The first proposal will be referred to as the *ROSOM-1C* methodology, since it requires only one SOM network, trained in the usual unsupervised way. The second proposal consists in training two SOMs, one is trained to become specialized on the class of negative examples, say, class $\mathcal{C}_{-1}$, while the other is trained to become specialized on the class of positive examples, say, class $\mathcal{C}_{+1}$. The decision to reject a given pattern will be determined based on the combination of results provided by the outputs of each map. This approach will be referred as the *ROSOM-2C* methodology along the remainder of the paper.

As a final remark, it is worth mentioning that the design methodologies of the ROSOM-1C and ROSOM-2C classifiers are general enough in the sense that they can be used to

develop pattern classifiers with reject option using, in principle, any topology-preserving prototype-based neural networks, such as the Growing Neural Gas (GNG) [18] and the Parameterless SOM (PLSOM) [4] algorithms.

5.1 SOM with Reject Option Using One Classifier

Initially, the ROSOM-1C requires post-training neuron labeling via Strategy 1, as described in Subsection 3.1. Additional steps are included in order to change the labels of some neurons to *rejection class*. The main idea behind the proposal of the ROSOM-1C approach relies exactly on developing formal techniques to assign the rejection class label to a given neuron. In greater detail, the design of the ROSOM-1C requires the following steps.

STEP 1 - For a given data set, a number of training realizations are carried out using a single SOM network in order to find the best number of neurons and suitable map dimensions. For this purpose, the conventional unsupervised SOM training is adopted.

STEP 2 - Present the training data once again and label the prototypes $\mathbf{w}_j$, $j = 1,...,q$, according to the mode of the class labels of the patterns mapped to them. No weight adjustments are carried out at this step.

STEP 3 - Based on the SOM's ability to approximate the input data density, we approximate $\mathbb{P}(\mathbf{x}|\mathcal{C}_k)$ with $\mathbb{P}(\mathbf{w}_j|\mathcal{C}_k,\mathbf{x})$, for $j = 1,...,q$ and $k = 1,...,K$. In Subsection 5.1.1, we describe two techniques to compute $\mathbb{P}(\mathbf{w}_j|\mathcal{C}_k,\mathbf{x})$ based on standard statistical techniques, namely, Parzen Windows and Gaussian Mixture Models.

STEP 4: Finding an optimum value for the rejection threshold β requires the minimization of the empirical risk as proposed in [11]:

$$\widehat{R} = \omega_r R + E \tag{8}$$

where R and E are, respectively, the ratio of rejected and misclassified patterns (computed using validation data), while ω_r is the rejection cost (whose value must be specified in advance by the user). It is worth recalling that a low (high) ω_r leads to the induction of a classifier that rejects many (few) patterns, thus increasing (decreasing) its recognition rate.

The searching procedure is described as follows.

STEP 4.1 - For a given rejection cost ω_r, vary β from an initial value β_i to a final value β_f in fixed increments $\Delta\beta$. Typical values are: $\beta_i = 0.55$, $\beta_f = 1.00$ and $\Delta\beta = 0.05$.

STEP 4.2 - Then, for each value of β, do

(*i*) Compute $R(\beta) = \dfrac{\text{number of rejected patterns}}{\text{total number of patterns}}$

(*ii*) Compute $E(\beta) = \dfrac{\text{number of misclassified patterns}}{\text{total number of patterns}}$

(*iii*) Compute $\widehat{R}(\beta)$ as in Equation (8).

STEP 4.3 - Select the optimum rejection threshold (β_o) according to the following rule:

$$\beta_o = \arg\min_{\beta}\{\widehat{R}(\beta)\}. \tag{9}$$

STEP 5: Re-label the prototypes using the following rule:

$$\begin{aligned} \text{IF} \quad & \max_k \{\mathbb{P}(\mathcal{C}_k)\mathbb{P}(\mathbf{w}_i|\mathcal{C}_k,\mathbf{x})\} < \beta \\ \text{THEN} \quad & \text{change class}(\mathbf{w}_i) \text{ to } \textit{Rejection Class}, \\ \text{ELSE} \quad & \text{keep class}(\mathbf{w}_i) \text{ as determined in STEP 2.} \end{aligned} \tag{10}$$

Once the prototypes have been re-labeled, the following decision rule is used for classifying new incoming patterns:

$$\begin{aligned} & \text{IF } \mathbf{w}_i \text{ is the winning prototype for pattern } \mathbf{x}(n), \\ & \text{THEN reject } \mathbf{x}(n) \text{ if class}(\mathbf{w}_i) = \textit{Rejection Class}, \\ & \text{ELSE class}(\mathbf{x}(n)) \leftarrow \text{class}(\mathbf{w}_i). \end{aligned} \tag{11}$$

5.1.1 On the Estimation of $\mathbb{P}(\boldsymbol{w}_j|\mathcal{C}_k,\boldsymbol{x})$

The first approach to be used to compute SOM-based estimates of $\mathbb{P}(\mathbf{w}_j|\mathcal{C}_k)$ is through the Parzen windows nonparametric method. The estimation is usually performed by some kernel function, usually a Gaussian, averaged by the number of points belonging to a given class:

$$\mathbb{P}(\mathbf{w}_j|\mathcal{C}_k,\mathbf{x}) = \frac{1}{N_k}\sum_{i=1}^{N_k} \frac{1}{h^d(2\pi)^{\frac{d}{2}}|\mathbf{C}_k|^{\frac{1}{2}}} \exp\left(-\frac{Q(\mathbf{x}_i^{(k)},\mathbf{w}_j)}{2h^2}\right) \tag{12}$$

with $Q(\mathbf{x}_i^{(k)},\mathbf{w}_j) = (\mathbf{x}_i^{(k)}-\mathbf{w}_j)^T\mathbf{C}_k^{-1}(\mathbf{x}_i^{(k)}-\mathbf{w}_j)$, where h is the width of the Gaussian window, $\mathbf{x}_i^{(k)}$ is the i-th pattern of the k-th class, $\mathbf{C}_k$ is the covariance matrix estimated from the training instances of the k-th class (i.e. $\mathcal{C}_k$), N_k the number of elements of the k-th class and d is the dimension of $\mathbf{x}_i^{(k)}$ and $\mathbf{w}_j$.

Another approach to estimate $\mathbb{P}(\mathbf{w}_j|\mathcal{C}_k,\mathbf{x})$ based on the distribution of SOM prototypes is via Gaussian mixture modelling (GMM) principles [1,32,48,52,62,68]. It is well known that the SOM itself provides only a rough approximation of the data density. Better approximations can be obtained using, for example, the self-organizing mixture network (SOMN) [68] or the self-organizing reduced kernel density estimation (SO-RKDE) method [1]. The basic idea behind the SO-RKDE model consists in using the SOM prototypes as kernel centers. Priors and conditional densities for unit j are estimated using the data samples from the Voronoi cell of unit i and also from its neighboring cells. The neighborhood function is used to get a weighted contribution of data from the neighboring units. In the simulations, a Gaussian neighborhood kernel was used. Probabilistic interpretation for the outputs of the SOM units can then be generated by a Gaussian mixture model based on the SO-RKDE method. In this paper, we use the SO-RKDE model implemented in the SOM toolbox[3].

5.1.2 Neuron Re-Labeling Based on Gini Index

For the application of the decision rule in (10), one has to store all the values of the posterior probabilities estimates $\mathbb{P}(\mathcal{C}_k|\mathbf{w}_j,\mathbf{x}) \propto \mathbb{P}(\mathcal{C}_k)\mathbb{P}(\mathbf{w}_j|\mathcal{C}_k,\mathbf{x})$ for each neuron j. The quantity $\mathbb{P}(\mathcal{C}_k|\mathbf{w}_j,\mathbf{x})$ express the probability of an instance that has fallen within the Voronoi cell of neuron j to belong to class $\mathcal{C}_k$. By means of concepts borrowed from information theory, it

[3] Available for download at `http://www.cis.hut.fi/somtoolbox/`.

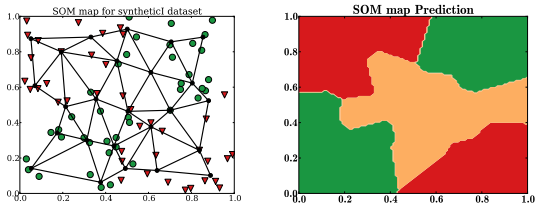


Fig. 2: On the left-hand figure it is shown a trained ROSOM-1C classifier using the Gini coefficient approach for a synthetic dataset. The right-hand figure depicts a class prediction results for a given testing data, where the red and green colors denote the decision classes and beige the reject decisions.

is possible to merge all the probabilities $\mathbb{P}(\mathcal{C}_k|\mathbf{w}_j,\mathbf{x})$, $k=1,...,K$, associated with a given neuron, into a single quantity to be called *cell impurity*.

Roughly, the impurity of neuron (or cell) j is a measure of the entropy of the class labels of the patterns mapped to this neuron. If the entropy is high, the distribution of class labels is more or less uniform (i.e. no class label dominates over the others). If the entropy is low, one class label clearly dominates over the others. In order to quantify the inequality of class labels distribution within a neuron, one can resort to the Gini coefficient [25, 26]. In the present context, this measurement is given by

$$G_j = 1 - \sum_{k=1}^{K} \mathbb{P}^2(\mathcal{C}_k|\mathbf{w}_j,\mathbf{x}), \quad j=1,...,q \tag{13}$$

where $\mathbb{P}(\mathcal{C}_k|\mathbf{w}_j,\mathbf{x})$ can be, for simplicity, computed as the frequency of instances within the Voronoi cell belonging to the class $\mathcal{C}_k$. Ideally, the desirable situation is to have always low values for the Gini coefficient, indicating predominance of a certain class label within neuron j. Neurons located at the borders of decision regions usually have high Gini coefficients, indicating higher entropy in the frequency of class labels within those neurons and, hence, a lower confidence in labeling them with a specific class label.

Using the Gini coefficient measure, the decision rule in (10) is now written as the following decision rule:

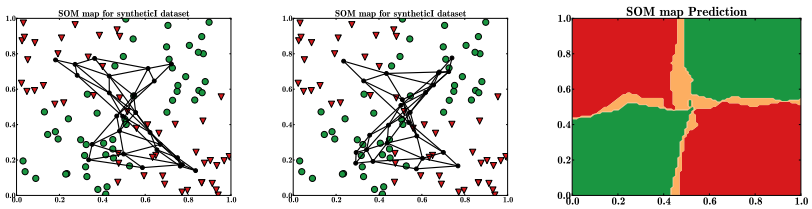
$$\begin{aligned} \text{IF} \quad & G_i > \beta \\ \text{THEN} \quad & \text{reject } \mathbf{x}(n), \\ \text{ELSE} \quad & \text{class}(\mathbf{x}(n)) = \text{class}(\mathbf{w}_i). \end{aligned} \tag{14}$$

where i is the index of the winning neuron for the current input pattern $\mathbf{x}(n)$.

$$\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \begin{cases} \eta(n)h(i,j;n)[\mathbf{x}(n)-\mathbf{w}_j(n)]\omega_r, & \text{if class}(\mathbf{x}(n)) = \mathcal{C}_{+1} \\ \eta(n)h(i,j;n)[\mathbf{x}(n)-\mathbf{w}_j(n)](1-\omega_r), & \text{if class}(\mathbf{x}(n)) = \mathcal{C}_{-1}. \end{cases} \tag{15}$$

$$\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \begin{cases} \eta(n)h(i,j;n)[\mathbf{x}(n)-\mathbf{w}_j(n)](1-\omega_r), & \text{if class}(\mathbf{x}(n)) = \mathcal{C}_{+1} \\ \eta(n)h(i,j;n)[\mathbf{x}(n)-\mathbf{w}_j(n)]\omega_r, & \text{if class}(\mathbf{x}(n)) = \mathcal{C}_{-1}. \end{cases} \tag{16}$$

Fig. 2 shows the results of a ROSOM-1C classifier for synthetic dataset (see Section 6) using the Gini coefficient approach. Each neuron has been initially trained and labeled, respectively, according to Steps 1 and 2 of the design procedure. Once the optimum rejection threshold has been determined, decision for rejection are made based on (14).



(a) (b) (c)

Fig. 3: Figures on the left (a) and center (b) present the trained SOM-1 and SOM-2 networks, respectively; (c) if both agree on the outcome a decision is emitted (green or red). Otherwise, instances are rejected (beige).

5.2 SOM with Reject Option Using Two Classifiers

In comparison to the ROSOM-1C, the individual SOM networks that comprise the ROSOM-2C have an extra feature: the ability to control the preference for patterns of a given class by the inclusion of cost parameter ω_r into the learning rules of the individual networks. In other words, one individual network is trained to become specialized, say, on class $\mathcal{C}_{-1}$, while the other is trained to become specialized on class $\mathcal{C}_{+1}$.

By allowing one of the networks to have preference for (i.e. to be biased toward) the patterns of class C_{+1}, while the other has preference for the patterns of class C_{-1}, makes the decision rule of ROSOM-2C more reliable. More reliable in the sense that a pattern is classified only when the outputs of both network coincides, otherwise the pattern is rejected.

The design of the ROSOM-2C requires the following steps.

STEP 1 - Choose a rejection cost ω_r.

STEP 2 - Train two SOM networks following the self-supervised SOM training scheme describe in Subsection 3.1.

STEP 2.1 - Train the first SOM network, henceforth named SOM-1 classifier, to become specialized on the class $\mathcal{C}_{-1}$. For that, we replace the standard SOM learning rule with Equation (15).

STEP 2.2 - Train the second SOM network, henceforth named SOM-2 classifier, to become specialized on the class $\mathcal{C}_{+1}$. For that, we replace the standard SOM learning rule with Equation (16).

STEPS 3, 4 and 5 - The same as the ones described for the ROSOM-1C classifier. The Gini coefficient approach can also be used to re-label the prototypes of the ROSOM-2C classifier.

Once the ROSOM-2C is trained, a new incoming pattern $\mathbf{x}(n)$ can be classified or rejected by the application of the following procedure:

- Find the winning prototype $\mathbf{w}_{i_1}$ for $\mathbf{x}(n)$ in SOM-1.
- Find the winning prototype $\mathbf{w}_{i_2}$ for $\mathbf{x}(n)$ in SOM-2

$$\begin{aligned} \text{IF} \quad & \text{class}(\mathbf{w}_{i_1}) = \text{class}(\mathbf{w}_{i_2}), \\ \text{THEN} \quad & \text{class}(\mathbf{x}(n)) \leftarrow \text{class}(\mathbf{w}_{i_1}), \\ \text{ELSE} \quad & \text{reject } \mathbf{x}(n). \end{aligned} \tag{17}$$

According to the ROSOM-2C decision rule in (17), three situations may occur:

Situation 1 - When the labels of the prototypes $\mathbf{w}_{i_1}$ and $\mathbf{w}_{i_2}$ match and are equal to one of the class labels (i.e. C_{+1} or C_{-1}), the pattern is classified *with confidence* as belonging to that class.

Situation 2 - When the labels of the prototypes $\mathbf{w}_{i_1}$ and $\mathbf{w}_{i_2}$ match and are equal to 'Rejection Class', the pattern is rejected *with confidence*.

Situation 3 - When the labels of the prototypes $\mathbf{w}_{i_1}$ and $\mathbf{w}_{i_2}$ do not match (i.e. in case of doubt), the pattern is also rejected.

From the exposed, it is useful to think of the ROSOM-2C as a committee of two specialized classifiers, one biased toward class C_{+1}, the other biased toward class C_{-1}. When the two classifiers agree in their decisions it means that the pattern can be classified with confidence, including the possibility of being rejected with higher confidence when the outputs of the two individual classifiers agree in rejecting the pattern. When they disagree in their decision, a more conservative approach is to also reject the new incoming pattern.

Fig. 3 illustrates the decision regions found produced by a ROSOM-2C classifier for a synthetic dataset (details are given in Section 6). In Fig. 3 (left) the individual network is trained to have preference for patterns of the class 'red', while in Fig. 3 (center) the individual network is trained to have preference for patterns of the class 'green'[4].

A final remark is necessary here. Extension of the ROSOM-2C approach to multiclass problems is straightforward. For this, one should adopt a One-Against-One strategy, which is commonly used to extend SVM binary classifiers to multiclass problem. In this case the algorithm would be the following: For K classes, construct $K(K-1)/2$ ROSOM-2C classifiers. Each classifier discriminates between two classes. A new incoming pattern is assigned using each classifier in turn and a majority vote taken. In case of ambiguity of the majority vote, with no clear decision for some patterns, the pattern is rejected.

5.3 LVQ with Reject Option Using One and Two Classifiers

Similarly to SOM-like strategies for classification with reject option, one can build LVQ-like classifiers with reject option based on either **Method 1** or **Method 2**. In order to do that it is only necessary to follow those steps which were presented in Subsection (5.1) or (5.2) but training LVQ classifiers as described in Section (4) instead of SOM network. The resulting LVQ-based classifiers thus designed will be referred to as ROLVQ-1C and ROLVQ-2C, respectively.

6 Computer Simulations and Discussion

In tis section we report the results of a comprehensive performance comparison among the proposed ROSOM-1C and ROSOM-2C classifiers and their supervised counterparts, which are based on the MLP, LVQ and SVM classifiers. The performance of the classification methods were assessed over five datasets which are described in the next subsections. The first two were synthetically generated; the remainder datasets includes real-world data.

[4] Note that there are more prototypes over patterns of the class 'red' in Fig. 3 (left) than in Fig. 3 (center)

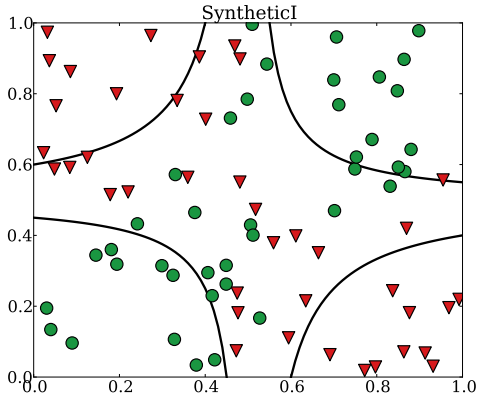


Fig. 4: Plot with samples from the two classes of the `syntheticI` dataset.

6.1 Synthetic Datasets

As in [8], for the first synthetic dataset (`syntheticI`), we began by generating 400 points $\mathbf{x} = [x_1\ x_2]^T$ in the unit square $[0,1] \times [0,1] \subset \mathbb{R}^2$ following a uniform distribution. Then, we assigned to each example $\mathbf{x}$ a class $y \in \{-1,+1\}$ corresponding to

$$y = \begin{cases} t, & t \neq 0 \\ +1, & t = 0 \wedge \varepsilon_2 < \alpha \\ -1, & t = 0 \wedge \varepsilon_2 > \alpha \end{cases},$$

where $t = \min_{r \in \{-1,0,+1\}} \{r : b_{r-1} < \alpha + \varepsilon_1 < b_r\}$, $\alpha = 10(x_1 - 0.5)(x_2 - 0.5)$, $\varepsilon_1 \sim N(0, 0.125^2)$, $\varepsilon_2 \sim \text{Uniform}(b_{-1}, b_0)$ and $(b_{-2}, b_{-1}, b_0, b_1) = (-\infty; -0.5; 0.25; +\infty)$. This distribution creates two uniformly distributed plateaus and a transition zone of linearly decreasing probability, delimited by hyperbolic boundaries (see Fig. 4).

A second synthetic dataset of 400 points—`syntheticII`—was generated from two Gaussian in $\mathbb{R}^2$: $\mathbf{y}_{-1} \sim N\left(\begin{bmatrix}-2\\-2\end{bmatrix}, \begin{bmatrix}9 & 0\\0 & 9\end{bmatrix}\right) + \varepsilon$ and $\mathbf{y}_{+1} \sim N\left(\begin{bmatrix}+2\\+2\end{bmatrix}, \begin{bmatrix}25 & 0\\0 & 25\end{bmatrix}\right) + \varepsilon$ corresponding to classes $\{-1,+1\}$ respectively, where ε follows a uniform distribution in $[0.025, 0.25]$.

6.2 Real-World Datasets

The first real-world dataset is a subset of `letter` problem, publicly available on the UCI machine learning repository, which is composed of $20,000$ instances with 16 features describing the 26 capital letters. Each instance is mainly defined by statistical moments and edge counts. In our experiments we used a subset of the whole dataset comprehending only the discrimination of the letter A versus the letter H.

The second real-world data set represents the discrimination of normal subjects from those with a pathology on the vertebral column. This database, also publicly available on the UCI machine learning repository, contains information about 310 patients obtained from

sagittal panoramic radiographies of the vertebral column described by 6 different biomechanical features. 100 patients were volunteers without any pathology (normal patients). The remaining data is from patients eventually operated due to disc hernia (60 patients) or spondylolisthesis (150 patients), comprising of 210 abnormal patients. For this study, we merge the groups of subjects with disk hernia and spondylolisthesis into a single pathology class. See [49] for more detail on this data set.

The last real dataset, encompassing 1144 observations, expresses the aesthetic evaluation of Breast Cancer Conservative Treatment [7, 44]. For each patient submitted to BCCT, 30 measurements were recorded, capturing visible skin alterations or changes in breast volume or shape. In this work we used only 4 measures as identified in [44] as the most relevant ones. The aesthetic outcome of the treatment for each and every patient was classified in one of the four categories: Excellent$\succ$Good$\succ$Fair$\succ$Poor. For the experimental work with binary models, the multiclass problem was transformed into a binary one, by aggregating Excellent and Good in one class, and the Fair and Poor cases in another class.

Experimental results are provided for all the ANN-based models previously discussed and the embedded rejection option approach for SVMs introduced by Fumera and Roli [21]. All classifiers were evaluated on synthetic and real-world datasets. We thank G. Fumera for providing the source code (in C/C++) of his method. Note that Fumera and Roli's method is for SVMs only and the provided implementation works only with linear kernels. We used the SOM toolbox for implementing the ROSOM-1C and ROSOM-2C classifiers and the Matlab^{TM}Neural Networks toolbox for MLP-based classifiers. For fair performance comparison, we have instantiated the same rejection option strategies used for the SOM-based classifiers into the MLP-based classifiers, giving rise to the MLP-1C and MLP-2C classifiers. Since we have trained the MLP-based classifiers to estimate the posterior probabilities, decisions for the MLP-1C are obtained simply through the application of the rule in 1. For the MLP-2C, each individual network penalizes differently the misclassifications according to the same costs as presented for the ROSOM-2C classifier.

For the SOM-based classifiers a two-dimensional map was used in the experiments with a hexagonal neighborhood structure and a Gaussian neighborhood function. For determining the best parameterization, we conducted a 5-fold cross validation in order to find the best number of neurons and the initial radius size for the neighborhood function. Our search considered a squared map spanning 5×5 to 25×25 neurons. The learning phase stopped after 200 epochs.

For the MLP-based classifiers, an exhaustive search over the number of hidden neurons, ranging from 5 to 20 neurons, was carried out for a single hidden layer network, with a single output neuron, and logistic sigmoid as activation function for all neurons. We defined a maximum number of 15 epochs as the stopping criterion in order to avoid overfitting [10]. The resilient back-propagation (RPROP) training algorithm was used.

It is important to point out that, in the absence of further insights about the problem at our disposal (other than the data itself), we cannot select only one value for ω_r, since its selection is intrinsically application-dependent. Thus, we started by running the classifiers spanning three values for ω_r in Equation (8): 0.04, 0.24 and 0.44[5]. As mentioned the ω_r value is directly related to how many patterns an expert is willing to reject. For high values of ω_r each pattern will have high rejection costs and, in consequence, we will eventually have a low number of rejected patterns. To assess the stability of the proposed approaches the experiments were repeated 50 times by averaging the results. Moreover, since in Fumera method only linear kernels were implemented, we extended the data sets with second order

[5] Values of ω_r higher than 0.5 are equivalent to random guesses.

`syntheticI` (training set size = 60%)

	$\omega_r = 0.44$		$\omega_r = 0.24$		$\omega_r = 0.04$	
Method name	Rej.	Perf.	Rej.	Perf.	Rej.	Perf.
(ROSOM-1C) Parzen	**0.13**	**0.90**	**0.28**	**0.96**	**0.49**	**0.99**
(ROSOM-1C) Gini	0.08	0.87	0.25	0.94	0.47	0.98
(ROSOM-1C) GMM	0.06	0.83	0.15	0.85	0.99	1.00
(ROLVQ-1C) Parzen	0.33	0.87	0.36	0.88	0.52	0.94
(MLP-1C)	0.29	0.91	0.40	0.96	0.56	0.99
(SVM) Fumera-Roli	0.22	0.61	0.67	0.86	0.95	0.99

(a) Performance for `syntheticI` dataset with 60% of training data using the ROSOM-1C (and corresponding variants), ROLVQ-1C, MLP-1C and SVM classifiers. Items in bold correspond to the best global results according to A-R curve (see Fig. 5a).

`syntheticI` (training set size = 80%)

	$\omega_r = 0.44$		$\omega_r = 0.24$		$\omega_r = 0.04$	
Method name	Rej.	Perf.	Rej.	Perf.	Rej.	Perf.
(ROSOM-1C) Parzen	**0.11**	**0.91**	**0.27**	**0.96**	**0.48**	**0.99**
(ROSOM-1C) Gini	0.07	0.87	0.28	0.94	0.51	0.98
(ROSOM-1C) GMM	0.07	0.83	0.18	0.87	0.95	1.00
(MLP-1C)	0.26	0.92	0.39	0.96	0.57	0.99
(SVM) Fumera-Roli	0.31	0.67	0.67	0.86	0.92	0.97

(b) Performance for `syntheticI` dataset with 80% of training data using the ROSOM-1C (and corresponding variants), MLP-1C and SVM classifiers. Items in bold correspond to the best global results according to A-R curve (see Fig. 5c).

Table 1: Performances achieved for `syntheticI` dataset using the ROSOM-1C, ROLVQ-1C, MLP-1C and SVM classifiers. We recommend the analysis of these tables with the assistance of Fig. 5.

terms $x_i x_j$ when evaluating this method. In this extended space, the optimal solutions for the synthetic data sets are indeed linear.

Table 1 and Table 2 illustrate the implications of an incorrect choice of the ω_r value. As an example, in Table 2 for the MLP-2C classifier (the same argument applies for the ROSOM-2C) we can have three times more patterns rejected with subtle improvements on the performance when selecting $\omega_r = 0.24$ instead of $\omega_r = 0.44$.

By analyzing Table 1 we observe that the performances the the proposed ROSOM-1C/Parzen are much better than those achieved by the MLP-1C, for all values of ω_r. In Table Table 2 the results follow the same pattern, with the proposed ROSOM-2C/Parzen and ROSOM-2C/Gini performing much better than the MLP-2C classifier, for $\omega_r = 0.24$ and 0.04. Only when the cost of rejecting a pattern is high (i.e. for $\omega_r = 0.44$), the performance of the MLP-2C class becomes equivalent that of the ROSOM-2C/GMM.

What follows next is a set of figures that allows a better understanding of the performances through the Accuracy-Reject (A-R) curve, whose major advantage resides on the straightforward interpretation of the results over the rejection costs presented by the A-R curve. In Fig. 5 to Fig. 9 we present the experimental results for each of the aforementioned data sets. In each plot the results of the proposed approaches compared to the MLP-, LVQ- and SVM-based counterparts are presented. Each point break in the curves corresponds to a given ω_r value: 0.04, 0.24 and 0.44.

By analyzing the performance on an A-R curve one can easily read the performance achieved by a given method and how much it was rejected for a given ω_r: the highest the curve, the better the performance is. For example, for the A-R curves shown in Fig. 5a, the

`syntheticI` (training set size = 60%)

	$\omega_r = 0.44$		$\omega_r = 0.24$		$\omega_r = 0.04$	
Method name	Rej.	Perf.	Rej.	Perf.	Rej.	Perf.
(ROSOM-2C) Parzen	0.07	0.88	**0.12**	**0.90**	**0.30**	**0.96**
(ROSOM-2C) Gini	0.04	0.88	**0.13**	**0.91**	**0.32**	**0.96**
(ROSOM-2C) GMM	**0.07**	**0.89**	0.17	0.92	0.44	0.97
(ROLVQ-2C) Parzen	0.01	0.78	0.06	0.79	0.12	0.81
(MLP-2C)	**0.09**	**0.90**	0.30	0.96	0.66	0.99
(SVM) Fumera-Roli	0.22	0.61	0.67	0.86	0.95	0.99

(a) Performance for `syntheticI` dataset with 60% of training data using the ROSOM-2C, ROLVQ-2C, MLP-2C and SVM classifiers. The two results in boldface for each ω_r are the best ones and they are statistically equivalent. The ROLVQ-2C and the Fumera-Roli SVM attained very poor results. For a better analysis of the results consider Fig. 5b.

`syntheticI` (training set size = 80%)

	$\omega_r = 0.44$		$\omega_r = 0.24$		$\omega_r = 0.04$	
	Rej.	Perf.	Rej.	Perf.	Rej.	Perf.
(ROSOM-2C) Parzen	**0.08**	**0.89**	**0.13**	**0.91**	**0.33**	**0.97**
(ROSOM-2C) Gini	0.04	0.88	**0.13**	**0.91**	**0.32**	**0.96**
(ROSOM-2C) GMM	0.07	0.88	**0.15**	**0.91**	0.43	0.98
(MLP-2C)	**0.10**	**0.91**	0.30	0.95	0.64	1.00
(SVM) Fumera-Roli	0.31	0.67	0.67	0.86	0.92	0.97

(b) Performance for `syntheticI` dataset with 80% of training data using the ROSOM-2C, MLP-2C and SVM classifiers. The two results in boldface for each ω_r are the best ones and they are statistically equivalent. For a better analysis of the results consider Fig. 5d.

Table 2: Performances achieved for `syntheticI` dataset using the ROSOM-2C, ROLVQ-2C, MLP-2C and SVM classifiers.

ROSOM-1C using the Parzen and Gini coefficient approaches achieved the best overall results. Note that for a reject rate of 0.2 (red vertical line) these classifiers achieved accuracies higher than 0.90; in other words, by rejecting 20% of the patterns, the accuracies of these classifiers go higher than 90% for the `SyntheticI` dataset, both performing much better than the MLP-1C classifier. In Fig. 5b, we can see that the performances of all ROSOM-2C variants and the MLP-2C were equivalent.

We also carry out some simulations to evaluate the performances of LVQ-based classifiers with reject option. For the sake of fairness, we used the same strategies devised for the SOM-based classifiers. Thus, the resulting classifiers were named ROLVQ-1C (for Method 1) and ROLVQ-2C (for Method 2). Fig. 5a and Fig. 5b for the LVQ variants with reject option show that they underperform the SOM-based counterparts on the synthetic dataset. The reasons for this behaviour are multi-fold. First, the parameterization for the LVQ is considerable higher than the SOMs where an incorrect setting could lead to sub-optimal results. Even if we set them carefully, other fine-tuning steps may be required for a better fitting of the data. Our proposals changed significantly the behaviour of the classical SOMs, where, in certain extent, class information was incorporated in the algorithms as described in Section 5. LVQs, on the other hand, were devised to account with data labelling and thus it was not needed to delve further algorithmic schemes to incorporate the reject option on the LVQs besides the a posteriori labelling as conducted with the SOMs.

Note, that we do not argue that LVQ are generally inferior to SOMs counterparts. Further tuning and experimentation could lead to more competitive results. However, this falls off

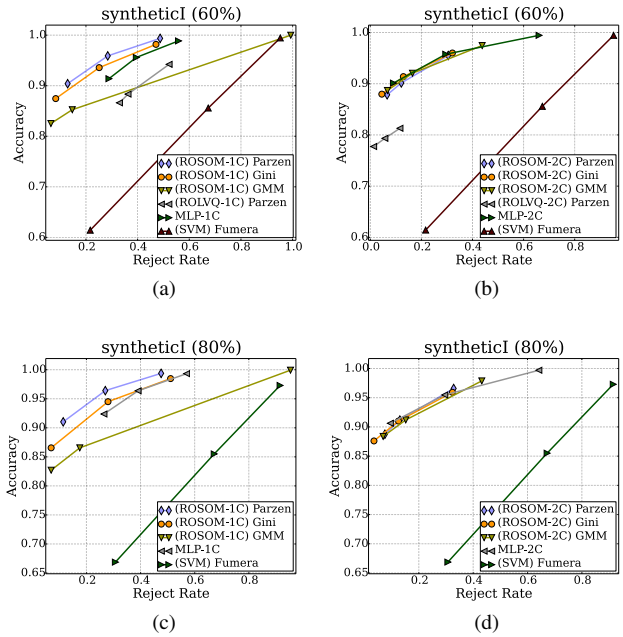


Fig. 5: The A-R curves for the `SyntheticI` dataset using 60% and 80% of training data.

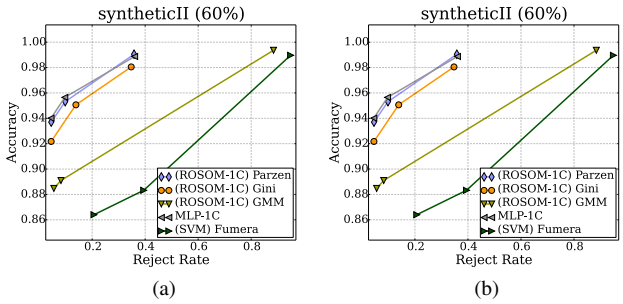
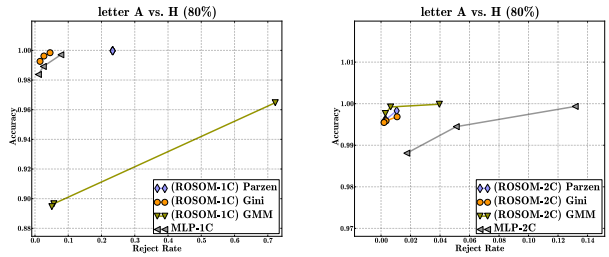


Fig. 6: The A-R curves for the `SyntheticII` dataset using 60% of training data.

of the scope of this work and may be considered in future works. Summing up, based on the aforementioned rationale and obtained results we did not considered the LVQ variants with reject option in the remainder of the experiments.

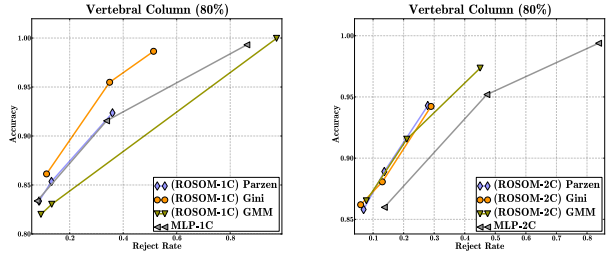
For the `SyntheticII` dataset, the A-R curves in Fig. 6a reveal that the ROSOM-1C/Parzen and the MLP-1C performed equivalently, followed closely by the ROSOM-1C/Gini. The A-R curves in Fig. 6b show that the best performance was achieved by the MLP-2C, while all the ROSOM-2C variants achieved equivalent performance.

For the `Letter AH` dataset, the A-R curves in Fig. 7a reveal that the best performance was achieved by the ROSOM-1C/Gini, followed closely by the MLP-1C. Both classifiers



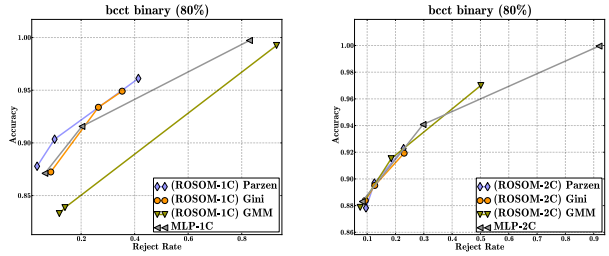
(a) (b)

Fig. 7: The A-R curves for the `Letter AH` dataset using 80% of training data.



(a) (b)

Fig. 8: The A-R curves for the `Vertebral Column` dataset using 80% of training data.



(a) (b)

Fig. 9: The A-R curves for the `BCCT` dataset using 80% of training data.

achieve very high accuracy rates, rejecting less than 5% of the patterns. The A-R curves in Fig. 7b show that all the ROSOM-2C variants performed better than the MLP-2C.

For the `Vertebral Column` dataset, the A-R curves in Fig. 8a indicate that the ROSOM-1C/Gini achieved the best overall performance. The A-R curves in Fig. 8b show that all the ROSOM-2C variants performed better than the MLP-2C.

Finally, for the `BCCT` dataset, the A-R curves in Fig. 9a reveal that the best performance was achieved by the ROSOM-1C/Parzen. For a small range of reject rate values (around

0.3) the performances of the ROSOM-1C/Parzen and the ROSOM-1C/Gini overlap. The A-R curves in Fig. 9b show that all ROSOM-2C variants and the MLP-2C performed equivalently.

It is worth mentioning that to verify that the performances of the SOM-based and MLP-based classifiers are equivalent is *not* a bad thing for the SOM-based classifiers. On the contrary, it is an issue worth emphasizing. Let us recall that the SOM is being adapted to work as a supervised classifier, since it is originally an unsupervised learning algorithm. But even so, the proposed SOM-based approaches achieved very competitive results in comparison with the MLP-based counterparts.

For all datasets the ROSOM-1C/GMM achieved in average the worst results. However, the ROSOM-2C/GMM achieved competitive results in comparison with the other approaches based on two classifiers. Such behavior can be partly explained by the fact that the proposed modified learning rules in (15) and (16) provide additional improvement over the raw estimates of the posterior probabilities in the performances of the ROSOM-2C classifier.

As a general conclusion, although neither the Parzen windows nor the Gini coefficient approaches outperformed one another over all datasets, Parzen and Gini attained better performances than the MLP-based counterparts. For instance, on the vertebral column dataset—see Fig. 8a—, one can achieve a performance of more than 85% rejecting less than 20% for both the ROSOM-1C and ROSOM-2C approaches.

7 Conclusions

Reject option comprises a set of techniques aiming at improving the classification reliability in decision support systems. However, the problem of classification with a reject option has been tackled only occasionally in machine learning literature, in most cases using supervised learning methods, such as the SVM, LVQ and MLP classifiers. In this paper we presented two SOM-based pattern classifiers that incorporate the rejection option class and compared their performances with MLP-, SVM-, and LVQ-based counterparts. To the best of our knowledge, this is the first time such approach is developed for the SOM or similar neural networks.

The first proposal, called ROSOM-1C, requires a single SOM trained in the usual unsupervised way. The second proposed classifier, called ROSOM-2C, requires two SOMs which are trained in the self-supervised learning scheme. Both proposals require either the estimation of the likelihood function $\mathbb{P}(\mathbf{x}|\mathcal{C}_k)$ or the posterior probability $\mathbb{P}(\mathcal{C}_k|\mathbf{x})$) using the distribution of SOM's weight vectors. An optimal threshold value has to be determined in order to re-tag some of the weight vectors with the rejection class label.

Estimates of the likelihood function $\mathbb{P}(\mathbf{x}|\mathcal{C}_k)$ were approximated by estimates of $\mathbb{P}(\mathbf{w}_j|\mathcal{C}_k,\mathbf{x})$, which can be obtained via Parzen Window or via Gaussian mixture models. When the proposed classifiers use the Gini coefficient approach, estimates of the posterior probability $\mathbb{P}(\mathcal{C}_k|\mathbf{x})$ were approximated by estimates of $\mathbb{P}(\mathcal{C}_k|\mathbf{w}_j,\mathbf{x})$, which can be computed by the frequency of instances within the Voronoi cell of neuron j belonging to class $\mathcal{C}_k$.

For the ROSOM-2C, in particular, the SOM learning rules were modified by the introduction of the rejection cost as a weight. The goal is to train one of the SOMs to become specialized on the class $\mathcal{C}_{-1}$, while the other is trained to become specialized on the class $\mathcal{C}_{+1}$. The decision to accept or reject a given pattern is determined based on the combination of results provided by the outputs of each map.

We carried out a comprehensively evaluation of the performances of the proposed SOM-based classifiers on two synthetic and three real-world data sets. The simulations have indi-

cated that the proposed approaches achieved results that are equivalent to or even better than those obtained by the standard supervised classifiers. In other words, the proposed SOM-based classifiers with reject option are competitive in terms of performance with standard supervised classifiers. The simulations also show that the proposed classifiers are very robust in terms of confidence in decision making process, since the proposed SOM-based classifiers can achieve very high accuracies (i.e. higher than 95%), rejecting fewer patterns than the standard supervised classifiers (see e.g Tables 1 and 2).

Currently, we are evaluating the proposed approaches for the classification of dynamic data, such as time series, and also developing a SOM-based classifier with reject option for nonstationary scenarios.

Acknowledgements

This work was partially supported through Program CNPq/Universidade do Porto/590008/2009-9 and conducted when Ricardo Sousa was in internship at Universidade Federal do Ceará (UFC), Brazil. This work was also partially funded by Fundação para a Ciência e a Tecnologia (FCT) - Portugal through project PTDC/SAU-ENB/114951/2009 and by FEDER funds through the Programa Operacional Factores de Competitividade - COMPETE in the framework of the project PEst-C/SAU/LA0002/2013. The authors also thank Fundação Núcleo de Tecnologia Industrial do Ceará (NUTEC) for providing the laboratorial infrastructure for the execution of the research activities reported in this paper.

References

1. Alhoniemi, E., Himberg, J., Vesanto, J.: Probabilistic measures for responses of self-organizing map units. In: Proceedings of the International ICSC Congress on Computational Intelligence Methods and Applications (CIMA'99, pp. 286–290. ICSC Academic Press (1999)
2. Bartlett, P.L., Wegkamp, M.H.: Classification with a reject option using a Hinge loss. Journal Machine Learning Research **9**, 1823–1840 (2008)
3. Bellazzi, R., Abu-Hanna, A.: Artificial intelligence in medicine AIME'07. Artificial Intelligence in Medicine **46**(1), 1–3 (2009)
4. Berglund, E., Sitte, J.: Parameterless self-organizing map algorithm. IEEE Transactions on Neural Networks **17**(2), 305–316 (2006)
5. Biehl, M., Ghosh, A., Hammer, B.: Dynamics and generalization ability of LVQ algorithms. Journal of Machine Learning Research **8**(Feb), 323–360 (2007)
6. Bounsiar, A., Beauseroy, P., Grall-Maës, E.: General solution and learning method for binary classification with performance constraints. Pattern Recognition Letters **29**(10), 1455–1465 (2008)
7. Cardoso, J.S., Cardoso, M.J.: Towards an intelligent medical system for the aesthetic evaluation of breast cancer conservative treatment. Artificial Intelligence in Medicine **40**, 115–126 (2007)
8. Cardoso, J.S., da Costa, J.F.P.: Learning to classify ordinal data: the data replication method. Journal of Machine Learning Research **8**, 1393–1429 (2007)
9. Carpenter, G.A., Grossberg, S., Markuzon, N., Reynolds, J.H., Rosen, D.B.: Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. IEEE Transactions on Neural Networks **3**(5), 698–713 (1992)
10. Caruana, R., Lawrence, S., Giles, C.L.: Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In: Proceedings of the 2000 Neural Information Processing Systems Conference (NIPS'00), pp. 402–408 (2000)
11. Chow, C.: On optimum recognition error and reject tradeoff. IEEE Transactions on Information Theory **16**(1), 41–46 (1970)
12. Cordella, L., De Stefano, C., Sansone, C., Vento, M.: An adaptive reject option for LVQ classifiers. In: Image Analysis and Processing, vol. LNCS 974/1995, pp. 68–73. Springer (1995)
13. Cordella, L., De Stefano, C., Tortorella, F., Vento, M.: A method for improving classification reliability of multilayer perceptrons. IEEE Transactions on Neural Networks **6**(5), 1140–1147 (1995)

14. de Bodt, E., Cottrell, M., Letremy, P., Verleysen, M.: On the use of self-organizing maps to accelerate vector quantization. Neurocomputing **56**, 187–203 (2004)
15. De Stefano, C., Sansone, C., Vento, M.: To reject or not to reject: That is the question - an answer in case of neural classifiers. IEEE Transactions on Systems, Man and Cybernetics - Part C: Applications and Reviews **30**(1), 574–585 (2000)
16. El-Yaniv, R., Wiener, Y.: On the foundations of noise-free selective classification. Journal of Machine Learning Research **11**, 1605–1641 (2010)
17. Flexer, A.: On the use of self-organizing maps for clustering and visualization. Intelligent Data Analysis **5**(5), 373–384 (2001)
18. Fritzke, B.: A growing neural gas network learns topologies. In: Advances in Neural Information Processing Systems 7, pp. 625–632. MIT Press, Cambridge, MA (1995)
19. Fu, Y., Zhu, X., Li, B.: A survey on instance selection for active learning. Knowledge and Information Systems **35**(2), 249–283 (2013)
20. Fumera, G., Pillai, I., Roli, F.: Classification with reject option in text categorisation systems. In: Proceedings of the 12th International Conference on Image Analysis and Processing (ICIAP'2003), pp. 582–587. IEEE Computer Society (2003)
21. Fumera, G., Roli, F.: Support vector machines with embedded reject option. In: Proceedings of the 1st International Workshop on Pattern Recognition with Support Vector Machines (SVM'2002), pp. 68–82. Springer (2002)
22. Gama, J., de Carvalho, A.C.: Machine learning. In: Machine Learning: Concepts, Methodologies, Tools and Applications, pp. 13–22. IGI-Global (2012)
23. Gasca, A.E., na, T.S.S., Sánchez, G.J.S., Velasquéz, G.V., Rendón, L.E., Abundez, B.I.M., Valdovinos, R.R.M., Cruz, R.R.: A rejection option for the multilayer perceptron using hyperplanes. In: Proceedings of the 10th International Conference on Adaptive and Natural Computing Algorithms (ICANNGA'2011), vol. LNCS 6593/2011, pp. 51–60. Springer (2011)
24. Geebelen, D., Suykens, J., Vandewalle, J.: Reducing the number of support vectors of SVM classifiers using the smoothed separable case approximation. IEEE Transactions on Neural Networks and Learning Systems **23**(4), 682 –688 (2012)
25. Giles, D.: Calculating a standard error for the gini coefficient: Some further results. Oxford Bulletin of Economics and Statistics **66**(3), 124–126 (2004)
26. Gini, C.: Measurement of inequality of incomes. The Economic Journal **31**(121), 124–126 (1921)
27. Goldszmidt, M., Cohen, I., Fox, A., Zhang, S.: Three research challenges at the intersection of machine learning, statistical induction, and systems. In: Proceedings of the 10th conference on Hot Topics in Operating Systems (HOTOS'05), vol. 10, pp. 1–6 (2005)
28. Guillen, A., Herrera, L.J., Rubio, G., Pomares, H., Lendasse, A., Rojas, I.: New method for instance or prototype selection using mutual information in time series prediction. Neurocomputing **73**(10–12), 2030–2038 (2010)
29. Han, J., Gao, J.: Research challenges for data mining in science and engineering. In: H. Kargupta, J. Han, P.S. Yu, R. Motwani, V. Kumar (eds.) Next Generation of Data Mining, pp. 1–18. Chapman & Hall / CRC Press (2009)
30. Hasenjäger, M., Ritter, H.: Active learning with local models. Neural Processing Letters **7**(2), 107–117 (1998)
31. Herbei, R., Wegkamp, M.H.: Classification with reject option. The Canadian Journal of Statistics **34**(4), 709–721 (2006)
32. Holmström, L., Hämäläinen, A.: The self-organizing reduced kernel density estimator. In: Proceedings of the 1993 IEEE International Conference on Neural Networks (ICNN'93), pp. 417–421 (1993)
33. Ishibuchi, H., Nii, M.: Neural networks for soft decision making. Fuzzy Sets and Systems **34**(115), 121–140 (2000)
34. Kohonen, T.: Self-organized formation of topologically correct feature maps. Biological Cybernetics **43**(1), 59–69 (1982)
35. Kohonen, T.: An introduction to neural computing. Neural Networks **1**(1), 3–16 (1988)
36. Kohonen, T.: The 'neural' phonetic typewriter. Computer **21**(3), 11–22 (1988)
37. Kohonen, T.: The self-organizing map. Proceedings of the IEEE **78**(9), 1464–1480 (1990)
38. Kohonen, T.: Self-Organizing Maps, 3rd edn. Springer (2001)
39. Kohonen, T.: Learning vector quantization. In: M.A. Arbib (ed.) The Handbook of Brain Theory, Neural Networks, 2nd edn., pp. 631–635. MIT Press (2003)
40. Lau, K.W., Yin, H., Hubbard, S.: Kernel self-organising maps for classification. Neurocomputing **69**, 2033–2040 (2006)
41. Lotte, F., Mouchère, H., Lécuyer, A.: Pattern rejection strategies for the design of self-paced EEG-based brain-computer interfaces. In: Proceedings of the 19th International Conference on Pattern Recognition (ICPR'2008), pp. 1–5 (2008)

42. Malone, J., McGarry, K., Wermter, S., Bowerman, C.: Data mining using rule extraction from Kohonen self-organising maps. Neural Computing and Applications **15**, 9–17 (2005)
43. Mattos, C.L.C., Barreto, G.A.: ARTIE and MUSCLE models: building ensemble classifiers from fuzzy ART and SOM networks. Neural Computing & Applications **22**(1), 49–61 (2013)
44. Oliveira, H.P., Magalhaes, A., Cardoso, M.J., Cardoso, J.S.: An accurate and interpretable model for BCCT.core. In: Proceedings of the 32nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pp. 6158–6161 (2010)
45. Pedreira, C.E.: Learning vector quantization with training data selection. IEEE Transactions on Pattern Analysis and Machine Intelligence **28**(1), 157–162 (2006)
46. Peng, H., Zhu, S.: Handling of incomplete data sets using ICA and SOM in data mining. Neural Computing & Applications **16**(2), 167–172 (2007)
47. Ritter, H.: Asymptotic level density for a class of vector quantization processes. IEEE Transactions on Neural Networks **2**(1), 173–175 (1991)
48. Riveiro, M., Johansson, F., Falkman, G., Ziemke, T.: Supporting maritime situation awareness using self organizing maps and gaussian mixture models. In: Proceedings of the 2008 Conference on 10th Scandinavian Conference on Artificial Intelligence (SCAI'08), pp. 84–91. IOS Press (2008)
49. Rocha-Neto, A.R., Sousa, R., Cardoso, J.S., Barreto, G.A.: Diagnostic of pathology on the vertebral column with embedded reject option. In: Proceedings of the 5th Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA'2011), vol. LNCS-6669, pp. 588–595 (2011)
50. Santos-Pereira, C.M., Pires, A.M.: On optimal reject rules and ROC curves. Pattern Recognition Letters **26**(7), 943–952 (2005)
51. Schleif, F.M., Villmann, T., Hammer, B., Schneider, P.: Efficient kernelized prototype based classification. International Journal of Neural Systems **21**(6), 443–57 (2011)
52. Seo, S., Obermayer, K.: Soft learning vector quantization. Neural Computation **15**, 1589–1604 (2002)
53. Sim, S.F., Sági-Kiss, V.: Multiple self-organising maps (mSOMs) for simultaneous classification and prediction: Illustrated by spoilage in apples using volatile organic profiles. Chemometrics and Intelligent Laboratory Systems **109**(1), 57–64 (2011)
54. Sousa, R., Mora, B., Cardoso, J.S.: An ordinal data method for the classification with reject option. In: Proceedings of the International Conference on Machine Learning and Applications (ICMLA'09), pp. 746–750 (2009)
55. Sousa, R., Rocha Neto, A.R., Barreto, G.A., Cardoso, J.S., Coimbra, M.T.: Reject option paradigm for the reduction of support vectors. In: Proceedings of the 22th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN'2014), pp. 1–6 (2014)
56. Souza Júnior, A.H., Barreto, G.A., Varela, A.T.: A speech recognition system for embedded applications using the SOM and TS-SOM networks. In: J.I. Mwasiagi (ed.) Self-Organizing Maps - Applications and Novel Algorithm Design, pp. 97–108. InTech Open (2011)
57. Suutala, J., Pirttikangas, S., Riekki, J., Röning, J.: Reject-optional LVQ-based two-level classifier to improve reliability in footstep identification. In: Pervasive Computing, pp. 182–187. Springer (2004)
58. Thomas, L.C., Edelman, D.B., Crook, J.N.: Credit Scoring and Its Applications, 1st edn. SIAM (2002)
59. Tortorella, F.: A ROC-based reject rule for dichotomizers. Pattern Recognition Letters **26**(2), 167–180 (2005)
60. Turky, A.M., Ahmad, M.S.: The use of SOM for fingerprint classification. In: IEEE International Conference on Information Retrieval & Knowledge Management (CAMP'2010), pp. 287–290 (2010)
61. Umer, M.F., Khiyal, M.S.H.: Classification of textual documents using learning vector quantization. Information Technology Journal **6**, 154–159 (2007)
62. Utsugi, A.: Density estimation by mixture models with smoothing priors. Neural Computation **10**, 2115–2135 (1998)
63. van Hulle, M.: Self-organizing maps. In: G. Rozenberg, T. Baeck, J. Kok (eds.) Handbook of Natural Computing: Theory, Experiments, and Applications, pp. 1–45. Springer-Verlag (2010)
64. Vasconcelos, G.C., Fairhurst, M.C., Bisset, D.L.: Enhanced reliability of multilayer perceptron networks through controlled pattern rejection. Electronics Letters **29**(3), 261–263 (1993)
65. Vasconcelos, G.C., Fairhurst, M.C., Bisset, D.L.: Investigating feedforward neural networks with respect to the rejection of spurious patterns. Pattern Recognition Letters **16**(2), 207–212 (1995)
66. Villmann, T., Haase, S.: Divergence-based vector quantization. Neural Computation **23**(5), 1343–1392 (2011)
67. Yin, H.: The self-organizing maps: Background, theories, extensions and applications. In: J. Fulcher, L.C. Jain (eds.) Computational Intelligence: A Compendium, *Studies in Computational Intelligence*, vol. 115, pp. 715–762. Springer-Verlag (2008)
68. Yin, H., Allinson, N.M.: Self-organizing mixture networks for probability density estimation. IEEE Transactions on Neural Networks **12**(2), 405–411 (2001)
69. Zidelmal, Z., Amirou, A., Belouchrani, A.: Heartbeat classification using support vector machines (SVMs) with an embedded reject option. International Journal of Pattern Recognition and Artificial Intelligence **26**(1), 1250,001–1–1250,001–17 (2012)