An Incremental On-line Classifier for Imbalanced, Incomplete, and Noisy Data

Marko Tscherepanow and **Sören Riechers**[1]

Abstract. Incremental on-line learning is a research topic gaining increasing interest in the machine learning community. Such learning methods are highly adaptive, not restricted to distinct training and application phases, and applicable to large volumes of data. In this paper, we present a novel classifier based on the unsupervised topology-learning TopoART neural network. We demonstrate that this classifier is capable of fast incremental on-line learning and achieves excellent results on standard datasets. We further show that it can successfully process imbalanced, incomplete, and noisy data. Due to these properties, we consider it a promising component for constructing artificial agents operating in real-world environments.

1 Introduction

The development of artificial agents with cognitive capabilities as they are found in humans and animals is still an unsolved problem. While biological agents can manage uncertain ever-changing environments with complex interdependencies, artificial agents are often limited to very specific, extremely simplified, and unchanging problems.

One possibility to improve the performance of current artificial systems is the usage of incremental learning mechanisms (e.g, [1], [18], and [19]). In contrast to traditional machine learning approaches based on distinct training and application phases, incremental approaches have to cope with additional difficulties – the most important being the *stability-plasticity dilemma* [15]: while plasticity is required in order to learn anything new, stability ensures that already acquired knowledge does not get lost in an uncontrolled way.

Sensor data obtained in natural environments often exhibit characteristics which further impede learning. In particular, their distributions can be non-stationary, noisy, and imbalanced. In addition, individual input vectors may be incomplete, for instance, due to different sensor latencies.

In this paper, we present an incremental classifier (see Section 4) based on the unsupervised TopoART neural network [26] (see Section 3). It is capable of stable and plastic incremental on-line learning and can cope with noisy, imbalanced, and incomplete data. These properties are shown using synthetic datasets (see Section 3) and real-world datasets from the UCI machine learning repository [12] (see Section 5).

2 Related Work

Adaptive Resonance Theory (ART) neural networks constitute an early approach to unsupervised incremental on-line learning. They incrementally learn a set of templates called categories. Some well-known ART variants are Fuzzy ART [5] and Gaussian ART [29]. While Fuzzy ART is capable of stable incremental learning using hyperrectangular categories but is prone to noise, the categories of Gaussian ART are Gaussians, which diminishes its sensitivity to noise but impairs the stability of learnt representations.

Regarding the formed representations, Gaussian ART is strongly related to on-line kernel density estimation (oKDE) [20]: oKDE incrementally estimates a Gaussian mixture model representing a given data distribution. Depending on an adjustable parameter, the estimated distribution is stable to a certain degree.

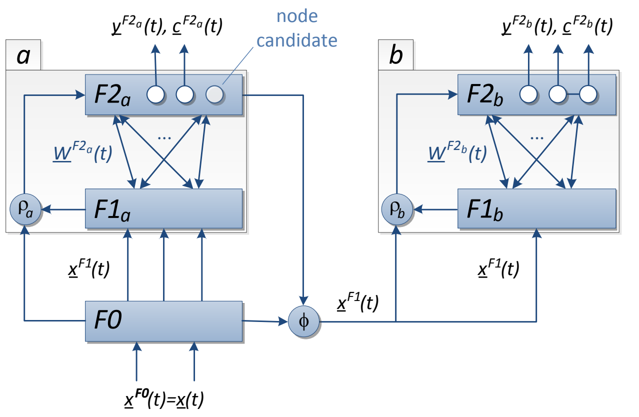
Incremental topology-learning neural networks, such as Growing Neural Gas [13], constitute an alternative approach to unsupervised on-line learning. Some of these networks, e.g., the Self-Organising Incremental Neural Network (SOINN)[14] and Incremental Growing Neural Gas (IGNG) [21], alleviate the problems resulting from the stability-plasticity dilemma. However, they rely on neurons representing prototype vectors. During learning, any shift of these prototype vectors in the input space inevitably causes some loss of information.

TopoART [26] has been proposed as a neural network combining properties from ART and topology-learning neural networks. As its architecture and representations are based on Fuzzy ART [5], each neuron (also called node) represents a hyperrectangular region of the input space, which can only grow during learning. As a result, once an input vector has been enclosed by a category, it will stay inside. In addition, TopoART inherited the insensitivity to noise from SOINN [14], which is a major improvement in comparison to Fuzzy ART.

The approaches mentioned above are not applicable to supervised learning tasks such as classification. However several extensions exist that enable their application to such problems, e.g., ARTMAP [4] for ART networks, Bayes' decision rule [28] for mixture models, and Life-long Learning Cell Structures [16] for prototype-based incremental topology-learning neural networks. The resulting supervised learning methods usually inherit the characteristics of their unsupervised components and ancestors, respectively. In particular, they learn locally; i.e., adaptations are restricted to a limited set of parameters.

The Perceptron [22], a very early approach to supervised on-line learning, possesses a distributed memory. As a consequence, all trainable parameters are altered during learning rendering Perceptrons prone to catastrophic forgetting. Furthermore, they have a fixed structure limiting the complexity of the knowledge that can be stored. These problems were inherited by multi-layer Perceptrons (MLPs) [23]. Cascade-Correlation neural networks [11] partially solve them by means of an incremental structure. But they are restricted to off-line (batch) learning.

[1] Applied Informatics, Bielefeld University, Germany, email: marko@techfak.uni-bielefeld.de, marko@tscherepanow.de



a
b
$\underline{y}^{F2_a}(t)$, $\underline{c}^{F2_a}(t)$
node candidate
$\underline{y}^{F2_b}(t)$, $\underline{c}^{F2_b}(t)$
$F2_a$
$F2_b$
$\underline{W}^{F2_a}(t)$
$\underline{W}^{F2_b}(t)$
ρ_a
ρ_b
$F1_a$
$F1_b$
$\underline{x}^{F1}(t)$
$\underline{x}^{F1}(t)$
$\underline{x}^{F1}(t)$
F0
ϕ
$\underline{x}^{F0}(t)=\underline{x}(t)$

Figure 1. TopoART neural networks consist of two modules sharing the input layer $F0$. Both modules function in an identical way. However, input to module b is controlled by module a.

Support vector machines (SVMs) [8], an alternative extension of Perceptrons, learn by solving a quadratic programming problem based on a fixed training set; i.e., off-line. A subset of the training samples, called support vectors, is chosen to construct separating hyperplanes. Although there are approaches to on-line SVMs (e.g., [2] and [6]), the underlying model imposes several problems: an adequate kernel has to be selected in advance, the number of occurring classes needs to be known or is limited to two, and a possibly large set of input samples has to be collected in addition to the support vectors to stabilise the learning process.

Recently, several incremental classification frameworks based on ensemble learning, e.g, ADAIN [17] and Learn++.NSE [10], have been proposed. These approaches assume that data be provided in data chunks containing multiple samples. As for each of these chunks an individual base classifier is trained, a voting mechanism is required so as to obtain a common prediction. Furthermore, additional learning methods may be necessary; ADAIN, for instance, uses an MLP to construct a mapping function connecting past experience with present data.

TopoART-C, the classifier presented in this paper, is based on the unsupervised TopoART network. TopoART was chosen as a basis in order to take advantage of its beneficial properties, namely its capability of stable incremental on-line learning of noisy data. TopoART-C constitutes an extension of TopoART for classification tasks like the Simplified Fuzzy ARTMAP approach [27] used for Fuzzy ART. The usage of an additional mask layer further allows predictions to be made based on incomplete data.

3 TopoART

TopoART is strongly related to Fuzzy ART [5]: it shares its basic representations, its choice and match functions, and its principal search and learning mechanisms. However, TopoART extends Fuzzy ART in such a way that it becomes insensitive to noise and capable of topology learning. One important part of the noise filtering mechanism is the combination of multiple Fuzzy ART-like modules, where preceding modules filter the input for successive ones. Therefore, the standard TopoART architecture as proposed in [26] (see Fig. 1) consists of two modules (a & b). Besides noise filtering, these modules cluster input data at two different levels of detail.

The clusters are composed of hyperrectangular categories, which are encoded in the weights of neurons in the respective $F2$ layer. By learning edges between different categories, clusters of arbitrary shapes are formed.

If an input vector

$$\underline{x}(t) = \left[x_1(t), \dots, x_d(t)\right]^T \tag{1}$$

is fed into such a network, complement coding is performed resulting in the vector

$$\underline{x}^{F1}(t) = \left[x_1(t), \dots, x_d(t), 1-x_1(t), \dots, 1-x_d(t)\right]^T. \tag{2}$$

As a consequence of this encoding that was inherited from Fuzzy ART, all elements $x_i(t)$ of the input vector $\underline{x}(t)$ must be normalised to the interval $[0,1]$.[2]

$\underline{x}^{F1}(t)$ is first propagated to the $F1$ layer of module a. From here, it is used to activate the $F2$ nodes of module a based on their weights given by the matrix $\underline{W}^{F2_a}(t)$.

As TopoART networks learn incrementally and on-line, training and prediction steps can be mixed arbitrarily. During training, the activation (choice function)

$$z_j^{F2}(t) = \frac{\left\|\underline{x}^{F1}(t) \wedge \underline{w}_j^{F2}(t)\right\|_1}{\alpha + \left\|\underline{w}_j^{F2}(t)\right\|_1} \tag{3}$$

of each $F2$ node j is computed first. $\|\cdot\|_1$ and $\wedge$ denote the city block norm and a component-wise minimum operation, respectively (cf. [5]). The node with the highest activation becomes the best-matching node bm. Its weights are adapted if the match function

$$\frac{\left\|\underline{x}^{F1}(t) \wedge \underline{w}_j^{F2}(t)\right\|_1}{\left\|\underline{x}^{F1}(t)\right\|_1} \geq \rho \tag{4}$$

is fulfilled for $j{=}bm$. Otherwise, the current node bm is reset and a new best-matching node is determined. If a suitable best-matching node bm has been found, a second-best-matching node sbm fulfilling Eq. 4 is sought.

The categories of the best-matching node and the second-best-matching node are allowed to grow in order to enclose $\underline{x}^{F1}(t)$ or partially learn $\underline{x}^{F1}(t)$, respectively:

$$\begin{aligned} \underline{w}_{bm}^{F2}(t+1) &= \underline{x}^{F1}(t) \wedge \underline{w}_{bm}^{F2}(t) && (5)\\ \underline{w}_{sbm}^{F2}(t+1) &= \beta_{sbm}\left(\underline{x}^{F1}(t) \wedge \underline{w}_{sbm}^{F2}(t)\right) \\ &\quad +(1-\beta_{sbm})\underline{w}_{sbm}^{F2}(t) && (6) \end{aligned}$$

In order to learn the topology of the data, bm and sbm are connected by an edge. Already existing edges are not modified. If the $F2$ layer is empty or no node is allowed to learn, a new node with $\underline{w}_{new}^{F2_a}(t+1){=}\underline{x}^{F1}(t)$ is incorporated.

According to Eq. 4, the maximum size of the categories is limited by the vigilance parameter ρ. Here, the vigilance parameter ρ_b of module b is determined depending on the vigilance parameter ρ_a of module a:

$$\rho_b = \frac{1}{2}(\rho_a + 1) \tag{7}$$

A value of $\rho_a{=}0$ means that a single category of module a can cover the entire input space, while a value of $\rho_a{=}1$ results in categories containing single samples.

[2] This normalisation usually requires an estimation of the minimum and maximum values for each x_i.

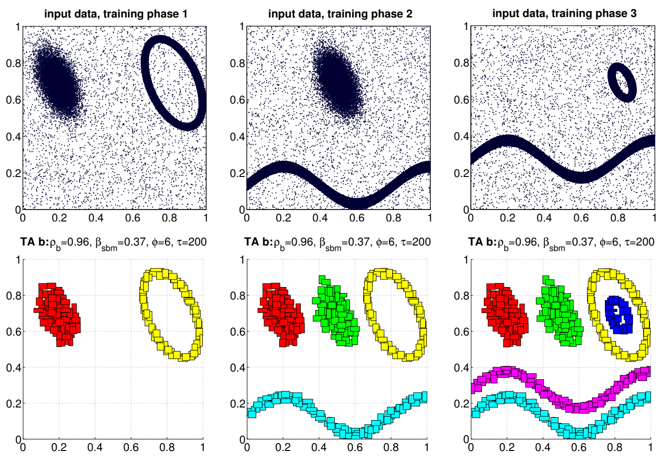


Figure 3. Results for non-stationary data. The training was performed in three successive phases (top row). In the bottom row, the corresponding clusters formed by a TopoART network after finishing the respective phase are shown. The network parameters were adopted from Fig. 2.

The $F2$ neurons of both modules possess a counter denoted by n_j. Each time a node is adapted, the corresponding counter is incremented. Furthermore, all $F2$ nodes j with $n_j{<}\phi$ are removed every τ learning cycles of the respective module. Therefore, such neurons are called node candidates. In contrast, nodes with $n_j{\geq}\phi$ are permanent; i.e., their categories are completely stable. The node candidates and the permanent nodes can be considered as the short-term memory and the long-term memory of the network, respectively.

$\underline{x}^{F1}(t)$ is only propagated to module b for training if the best-matching node of module a is permanent. In module b, the processes of category search and weight adaptation are repeated for the corresponding $F2$ nodes using ρ_b instead of ρ_a. In conjunction with the input filtering and the node-removal process, this constitutes a powerful noise reduction mechanism. Figure 2 illustrates this mechanism in comparison to two other popular unsupervised learning methods. The applied dataset consists of six clusters with 15,000 samples each as well as ten percent of uniformly distributed random noise (100,000 samples in total). In order to create a stationary data distribution, the samples were presented once in random order.

TopoART and SOINN were able to determine a detailed clustering reflecting the six clusters of the input distribution in a high level of detail. Here, the representation was refined from module a to module b and from SOINN layer 1 (SOINN 1) to SOINN layer 2 (SOINN 2). The representation of oKDE also reflects the underlying components, but does not include the topological structures. Furthermore, several Gaussians exclusively represent noise regions.

The capability of TopoART to incrementally learn stable representations from noisy non-stationary data is demonstrated in Fig. 3. Here, the input data used before were reordered and presented in three consecutive phases. After each training phase, TopoART has learnt the respective new clusters and the clusters formed during earlier training phases remained stable.

For prediction, $\underline{x}^{F1}(t)$ is directly propagated to both modules where the respective best-matching nodes are determined. Here, usually the alternative activation function

$$z_j^{F2}(t) = 1 - \frac{\left\| \left(\underline{x}^{F1}(t) \wedge \underline{w}_j^{F2}(t) \right) - \underline{w}_j^{F2}(t) \right\|_1}{d} \qquad (8)$$

that is independent from the category size is applied and the match function is not computed. The output of a module consists of a vector $\underline{y}^{F2}(t)$ with

$$y_j^{F2}(t) = \begin{cases} 0 & \text{if } j \neq bm \\ 1 & \text{if } j = bm \end{cases} \qquad (9)$$

and a vector $\underline{c}^{F2}(t)$ reflecting the clustering structure. For reasons of stability, node candidates are ignored during prediction.

Details on the adjustment and the effects of the parameters ρ_a, β_{sbm}, ϕ, and τ can be found in [26].

4 TopoART-C

In contrast to TopoART, which clusters presented data, a classifier requires additional information. In particular, a class label λ_i is associated with each input vector $\underline{x}(t)$ comprising one or more features $x_i(t)$. This label is either presented for training or predicted based on $\underline{x}(t)$. $\Lambda(t)$ denotes the set of all known class labels at time step t. In incremental learning scenarios $\Lambda(t)$ may grow if new data become available. The current number of known classes is given by $|\Lambda(t)|$. In order to construct a classifier inheriting the advantageous properties of TopoART, the principal structure of TopoART was preserved and extended by three additional layers (see Fig. 4).

Both modules obtained a classification layer $F3$, the nodes of which represent possible classes λ_i. These layers receive the output $\underline{y}^{F2}(t)$ of the respective F2 layer as input. Furthermore, a mask layer $F0_m$ was incorporated so as to enable predictions based on incomplete input data.

4.1 Training

TopoART-C is trained in a similar way to TopoART. But in order to account for the class labels, the match function (cf. Eq. 4) was modified:

$$\frac{\left\| \underline{x}^{F1}(t) \wedge \underline{w}_j^{F2}(t) \right\|_1}{\left\| \underline{x}^{F1}(t) \right\|_1} \geq \rho \quad \text{and} \quad \text{class}(j) = \lambda(t) \qquad (10)$$

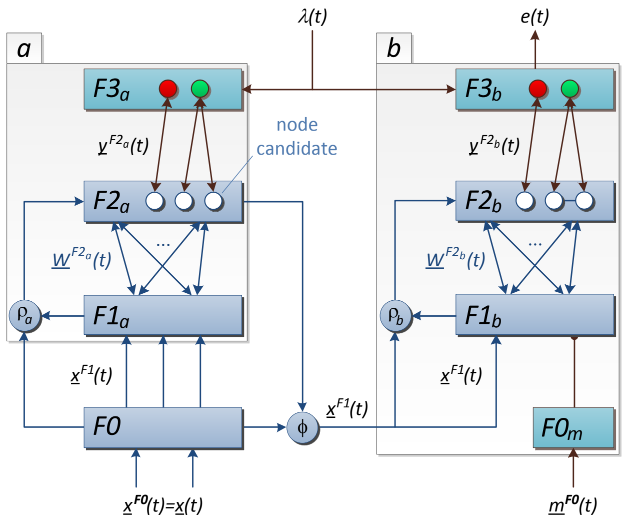


Figure 4. Structure of TopoART-C. TopoART-C extends the structure of TopoART by adding a classification layer $F3$ to each module and a mask layer $F0_m$ to module b.

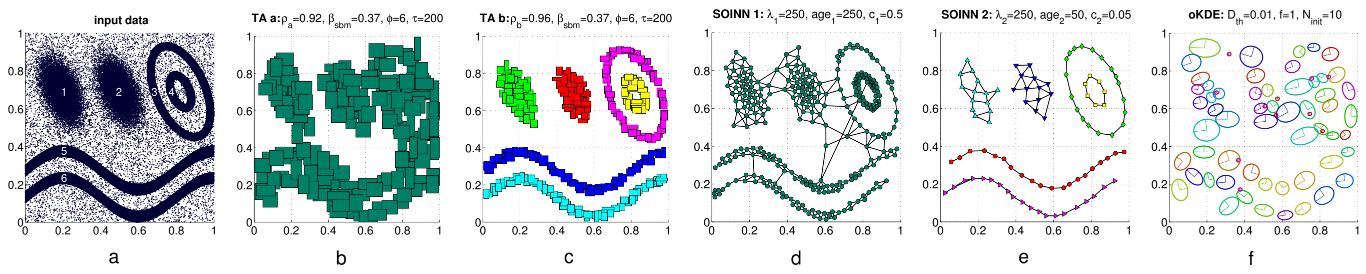


Figure 2. Clustering results for stationary data. A two-dimensional synthetic data distribution was learnt by TopoART (TA), SOINN, and oKDE. The relevant parameters were manually chosen in such a way as to fit the data. Different clusters of TopoART (b & c) and SOINN (d & e) are coloured differently. For SOINN, the edges connecting individual prototype vectors are shown, as well. In contrast to TopoART and SOINN, the distribution estimated by oKDE (f) consists of a mixture of Gaussians drawn as ellipses marking the standard deviations. It does not reflect the topological structure of the data.

Here, $\lambda(t)$ denotes the class label of $\underline{x}(t)$ and class(j) the class encoded by the $F3$ node that is connected with node j.[3]

Assuming the match function cannot be fulfilled for any existing $F2$ node, a new one with $\underline{w}^{F2}_{new}(t+1)=\underline{x}^{F1}(t)$ is incorporated like in the original TopoART network. Additionally, it is linked to the $F3$ node representing $\lambda(t)$. If the network does not know $\lambda(t)$, a new $F3$ node representing this label is inserted.

4.2 Prediction

During prediction, the class label λ_i which best fits the input vector $\underline{x}(t)$ is computed by module b using the decision rule:

$$e(t) = \arg\max_{\lambda_i \in \Lambda(t)} d_{\lambda_i}(t) \tag{11}$$

The discrimination function $d_{\lambda_i}(t)$ measures the similarity of $\underline{x}(t)$ with the internal representation of class λ_i:

$$d_{\lambda_i}(t) = \sum_{\substack{j \in \Upsilon_b(t) \\ \text{class}(j)=\lambda_i}} y_j^{F2_b}(t) \tag{12}$$

$d_{\lambda_i}(t)$ depends on the output $\underline{y}^{F2_b}$ of the $F2$ layer of module b. The set of all nodes of this layer is denoted by $\Upsilon_b(t)$. Module a is completely neglected, as it is only required for training in order to filter irrelevant data.

If the original output function (Eq. 9) is applied, $e(t)$ yields the class label of the permanent node whose category has the closest distance to $\underline{x}(t)$. But depending on the class boundaries, the resulting prediction may be suboptimal. Therefore, we propose a more general output function, which can consider more than one node and allows for problem specific adaptations. Here, two principal cases have to be distinguished: either $\underline{x}(t)$ lies inside one or more categories or it is enclosed by no category. In the first case, the class label can be derived from the enclosing categories summarised in the set

$$\mathcal{E}(t) = \left\{ j \in \Upsilon_b(t) : z_j^{F2_b}(t) = 1 \right\}. \tag{13}$$

These nodes are characterised by a maximum activation $z_j^{F2_b}(t)$ according to Eq. 8. Using the modified output function

$$y_j^{F2_b}(t) = \begin{cases} 1 & \text{, if } j = \arg\min_{k \in \mathcal{E}(t)} S_k(t) \\ 0 & \text{, otherwise,} \end{cases} \tag{14}$$

the prediction corresponds to the class label associated with the smallest category containing $\underline{x}(t)$. The category size $S_k(t)$ is defined as

$$S_k(t) = \sum_{i=1}^{d} \left| \left(1 - w^{F2}_{k,d+i}(t)\right) - w^{F2}_{k,i}(t) \right|. \tag{15}$$

In the second case, i.e., if no category encloses $\underline{x}(t)$, predictions are computed based on the set $\mathcal{N}$ of closest neighbours:

$$\mathcal{N}(t) = \left\{ j \in \Upsilon_b(t) : z_j^{F2_b}(t) \geq \mu + 1.28\sigma \right\} \tag{16}$$

μ and σ denote the arithmetic mean and the standard deviation of $z_j^{F2_b}(t)$ over all $F2_b$ neurons, respectively. If the activations were normally distributed, $\mathcal{N}(t)$ would only contain those 10% of the neurons that have the highest activations.

The contribution of each neighbour to the output function is inversely proportional to the distance between its category and $\underline{x}(t)$:

$$y_j^{F2_b}(t) = \begin{cases} \dfrac{\frac{1}{1-z_j^{F2_b}(t)}}{\sum\limits_{n \in \mathcal{N}(t)} \frac{1}{1-z_n^{F2_b}(t)}} & \text{, if } j \in \mathcal{N}(t) \\ 0 & \text{, otherwise} \end{cases} \tag{17}$$

Depending on the data distribution and for computational reasons it may be advantageous to further limit the number of considered nodes. Therefore, we incorporated an additional parameter ν which denotes the maximum cardinality of $\mathcal{E}(t)$ and $\mathcal{N}(t)$. It does not affect the underlying representations and may be changed during the application of the network. To obtain repeatable results, elements need to be added to both sets in a predefined way (cf. Eqs. 13 and 16). Therefore, we decided to add nodes in increasing order of their indices to $\mathcal{E}(t)$ and in decreasing order of their activations to $\mathcal{N}(t)$. Provided that the cardinality has reached the value of ν, the insertion of new elements is stopped. As a result, established and certain knowledge is preferred over recently acquired and uncertain knowledge. Due to this difference to the original output function (cf. Eq. 9), which treats all nodes equally, we decided to consider not only permanent nodes but also node candidates for prediction. As a result, the predictions become slightly less stable. However, the network is better adapted to recent input, in particular if no established knowledge is available.

In order to make predictions based on incomplete input vectors, the mask layer $F0_m$ is used. Its neurons, the output of which is given by the mask vector

$$\underline{m}^{F0}(t) = \left[m_1^{F0}(t), \ldots, m_d^{F0}(t) \right]^T, \tag{18}$$

[3] Each $F2$ node can only be connected with a single $F3$ node.

inhibit the network connections that encode elements of the input vector that are not available; i.e., presented elements are characterised by a mask value $m_i^{F0}(t)$ of 0 and unknown elements by a value of 1. Hence, the indices of the relevant elements of $\underline{x}^{F1}(t)$ (cf. Eq. 2) are given by the index set

$$\mathcal{M}^0 = \left\{i, i{+}d : m_i^{F0}(t) = 0\right\}. \tag{19}$$

Using $\mathcal{M}^0$, the activation of the $F2$ nodes of module b can be determined solely based on the non-inhibited F1 neurons:

$$z_j^{F2_b}(t) = 1 - \frac{\sum\limits_{i \in \mathcal{M}^0} \left| \min\left(x_i^{F1}(t), w_{ji}^{F2_b}(t)\right) - w_{ji}^{F2_b}(t) \right|}{\frac{1}{2}|\mathcal{M}^0|} \tag{20}$$

If required, the maximum activation over all $F2$ nodes of module b can be applied as a measure of the degree of knowledge the network has about a certain input vector. Then, input vectors can be rejected as unknown if the maximum activation is below a threshold.

5 Results

We evaluated TopoART-C using several real-world datasets from the UCI machine learning repository [12]. In order to show the beneficial properties of TopoART-C, we selected datasets with varying numbers of classes and features, with and without missing values, and with balanced and imbalanced classes (see Table 1). Here, the class ratio denotes the ratio between the number of samples contained in the smallest class and in the largest class, respectively. Thus, a class ratio of 1 shows that a dataset is completely balanced, while class ratios close to 0 indicate imbalanced datasets.

dataset	$\|\Lambda\|$	d	missing values	class ratio
iris	3	4	no	1.000
ISOLET*	26	617	no	0.992
optical digits*	10	64	no	0.967
ozone level (1 hour)	2	73	yes	0.030
ozone level (8 hours)	2	73	yes	0.067
page blocks	5	10	no	0.006
pen digits*	10	16	no	0.921
wine	3	13	no	0.676
wine quality (red) [9]	6	11	no	0.015
wine quality (white) [9]	7	11	no	0.002
yeast	10	8	no	0.011

Table 1. Number of classes $|\Lambda|$, number of features d, existence of missing values and class ratio for the considered datasets. Those datasets marked with $*$ contain an independent test set.

For comparison, we used several well-known on-line and off-line classifiers: the k-nearest neighbour classifier[3] (kNN), the naïve Bayes classifier[3] (NB), random trees[3] (RTs), the Simplified Fuzzy ARTMAP (SFAM) [27], and support vector machines[4] (SVMs) using different kernels. These classifiers were compared based on the harmonic mean accuracy

$$\overline{\mathrm{ACC}}_{\mathrm{hm}} = \frac{|\Lambda|}{\sum\limits_{\lambda_i \in \Lambda} \frac{1}{\mathrm{ACC}(\lambda_i)}} \tag{21}$$

[3] implemented in OpenCV v2.2 [3]
[4] implemented in LIBSVM v3.11 [7]

as proposed in [25] for imbalanced datasets. Here, $\mathrm{ACC}(\lambda_i)$ denotes the fraction of correctly classified samples of class λ_i. In contrast to the total accuracy[5] and the arithmetic mean of the class-specific accuracies $\mathrm{ACC}(\lambda_i)$, $\overline{\mathrm{ACC}}_{\mathrm{hm}}$ prevents large correctly classified classes from dominating the classification results; e.g., if one class cannot be recognised at all, $\overline{\mathrm{ACC}}_{\mathrm{hm}}$ drops to zero, independent of the number of test samples available for this class. Provided that the classification problem is entirely balanced and the class-specific accuracies are equal, all three accuracy measures are equal.

Table 2 shows the classification results. These results were either obtained using five-fold cross-validation or refer to the independent test set that has neither been used for training nor the optimisation of model parameters before, if available (cf. Table 1). The relevant parameters of the classifiers were determined by means of grid search.[6] For training, all features were normalised to the interval $[0.05, 0.95]$. Input vectors containing missing values were ignored (training and prediction) and counted as errors (prediction) if the respective classifier was not able to process incomplete data.

TopoART-C achieved excellent results for the majority (6 of 11) of the datasets. In particular, it outperformed the other classifiers on 4 of 6 imbalanced[7] datasets including those with missing values and reached comparatively high accuracies for the remaining two datasets 'wine quality (red)' and 'wine quality (white)'. Regarding balanced data, SVMs performed better, especially on the 'ISOLET' dataset, which is most likely caused by its large number of features. Nevertheless, TopoART-C achieved the maximum accuracy on 2 of 5 balanced datasets. In addition, TopoART-C often reached very high accuracies after a single presentation of all training samples, which is a good benchmark for incremental on-line learning; without the capability of stable incremental learning, an on-line learning approach such as TopoART (cf. Eqs. 5 and 6) would be prone to catastrophic forgetting resulting in worse results.

6 Conclusion and Outlook

We presented the novel incremental classifier TopoART-C that is capable of fast on-line learning (cf. Table 2). Accurate predictions can even be made if the input vectors contain missing values. As TopoART-C contains the unsupervised TopoART network as major learning component, it is insensitive to noise (cf. Fig. 2) and can be applied to non-stationary data (cf. Fig. 3). These properties of TopoART-C make it an excellent choice for the application to real-world on-line learning tasks, as they occur, for instance, in cognitive robotics. In addition, the clusters learnt by the TopoART subnet could provide additional information on the underlying data.

Furthermore, TopoART-C is not restricted to the usage of TopoART; alternative neural networks with a TopoART-like structure such as Hypersphere TopoART [24] can be applied as well. Since Hypersphere TopoART does not perform complement coding, the resulting classifier could process arbitrarily scaled values even if their range is not completely known in advance.

[5] overall ratio of correctly classified samples
[6] **kNN:** $k \in \{1, 2, \ldots, 25\}$; **RTs:** `use_surrogates` $\in$ {`false`, `true`}, `variableImportance` $\in$ {`false`, `true`}, `nactive_vars` $= d^x$ with $x \in [0.1, 0.9]$ and step size 0.1; **SFAM:** $\rho \in [0.75, 1]$ with step size 0.01, $\beta \in [0.2, 1]$ with step size 0.2; **SVMs:** $C = 10^x$ with $x \in [-4, 4]$ and step size 0.5, $\gamma = 10^x$ with $x \in [-4, 4]$ and step size 0.5 (RBF kernel), `coef0` $= 10^x$ with $x \in [-4, 4]$ and step size 0.5 (polynomial kernel), degree $\in \{1, 2, 3, 4, 5\}$ (polynomial kernel); **TopoART-C**: $\rho_a \in [0.75, 1]$ with step size 0.01, $\beta_{sbm} \in [0, 1]$ with step size 0.2, $\phi \in \{1, 2, 3, 4, 5\}$, $\nu \in \{1, 2, \ldots, 25\}$
[7] class ratio < 0.1

dataset	**kNN**^{o}	**NB**	**RTs**^{tp}	**SFAM**^{o}		**SVMs**		**TopoART-C**^{op}	
	1 it.			1 it.	≤25 it.	polynomial	RBF	1 it.	≤25 it.
iris	97.4±2.7	97.0±2.8	96.7±2.4	97.0±2.0	96.9±3.0	98.4±2.0	96.8±3.1	**98.9±1.4**	98.2±2.7
ISOLET	90.9	0.0	94.5	92.6	93.9	96.3	**96.7**	91.2	91.6
optical digits	97.9	94.6	96.9	96.9	97.7	97.5	**98.4**	97.4	97.4
ozone level (1 hour)	29.5±10.7	0.0±0.0	2.2±4.3	42.8±11.6	42.7±11.6	34.1±6.5	26.3±14.4	50.6±27.5	**54.4±22.1**
ozone level (8 hours)	45.0±6.8	5.5±4.8	10.7±6.9	50.1±9.1	53.7±6.7	50.2±4.1	45.6±4.2	**63.9±3.8**	60.6±10.3
page blocks	69.6±6.3	79.6±6.0	**82.5±4.1**	75.4±3.4	75.5±3.3	77.9±3.0	76.4±6.5	**82.5±4.0**	**82.5±6.5**
pen digits	97.7	95.8	96.4	97.4	97.9	97.5	**98.2**	97.7	97.7
wine	96.6±3.1	98.8±1.5	98.8±1.5	98.7±1.7	98.7±1.7	98.3±1.6	98.8±1.0	98.7±1.7	**99.1±1.7**
wine quality (red)	0.0±0.0	0.0±0.0	0.0±0.0	15.8±20.1	**15.9±20.2**	0.0±0.0	0.0±0.0	9.1±18.2	9.3±18.6
wine quality (white)	0.0±0.0	5.6±11.2	0.0±0.0	6.2±12.3	0.0±0.0	2.2±4.4	**8.5±16.9**	7.3±14.5	7.7±15.5
yeast	11.4±14.0	0.0±0.0	0.0±0.0	24.9±20.9	24.4±20.8	7.3±14.5	11.7±23.4	**37.7±19.5**	36.7±19.7

Table 2. Harmonic mean accuracies and their standard deviations (over the cross-validation runs) in percent. The best results for each dataset are highlighted. In order to alleviate the comparison, some relevant capabilities of the classifiers are indicated by superscripts: o = on-line learning, t = accept missing values for training, and p = accept missing values for prediction. In order to compensate for the negative effects of a possibly too small number of training steps, the respective training sets were presented to the on-line learning approaches except for the kNN classifier up to 25 times. The results are given for the first iteration (1 it.) and when they converged or the maximum number of iterations was reached (≤25 it.).

ACKNOWLEDGEMENTS

This work was partially funded by the German Research Foundation (DFG), Excellence Cluster 277 "Cognitive Interaction Technology".

REFERENCES

[1] Elmar Berghöfer, Denis Schulze, Marko Tscherepanow, and Sven Wachsmuth, 'ART-based fusion of multi-modal information for mobile robots', in *Proceedings of the International Conference on Engineering Applications of Neural Networks*, volume 363 of *IFIP AICT*, pp. 1–10, Corfu, Greece, (2011). Springer.

[2] Antoine Bordes, Seyda Ertekin, Jason Weston, and Léon Bottou, 'Fast kernel classifiers with online and active learning', *Journal of Machine Learning Research*, **6**, 1579–1619, (2005).

[3] Gary Bradski and Adrian Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*, O'Reilly, 2008.

[4] Gail A. Carpenter, Stephen Grossberg, and John H. Reynolds, 'ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network', *Neural Networks*, **4**, 565–588, (1991).

[5] Gail A. Carpenter, Stephen Grossberg, and David B. Rosen, 'Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system', *Neural Networks*, **4**, 759–771, (1991).

[6] Gert Cauwenberghs and Tomaso Poggio, 'Incremental and decremental support vector machine learning', in *Neural Information Processing Systems*, pp. 409–415, (2000).

[7] Chih-Chung Chang and Chih-Jen Lin, 'LIBSVM: A library for support vector machines', *ACM Transactions on Intelligent Systems and Technology*, **2**(3), 27:1–27:27, (2011). Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[8] Corinna Cortes and Vladimir Vapnik, 'Support-vector networks', *Machine Learning*, **20**, 273–297, (1995).

[9] Paulo Cortez, António Cerdeira, Fernando Almeida, Telmo Matos, and José Reis, 'Modeling wine preferences by data mining from physicochemical properties', *Decision Support Systems*, **47**(4), 547–553, (2009).

[10] Ryan Elwell and Robi Polikar, 'Incremental learning of concept drift in nonstationary environments', *IEEE Transactions on Neural Networks*, **22**(10), 1517–1531, (2011).

[11] Scott E. Fahlman and Christian Lebiere, 'The cascade-correlation learning architecture', in *Neural Information Processing Systems*, pp. 524–532, (1989).

[12] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[13] Bernd Fritzke, 'A growing neural gas network learns topologies', in *Neural Information Processing Systems*, pp. 625–632, (1994).

[14] Shen Furao and Osamu Hasegawa, 'An incremental network for on-line unsupervised classification and topology learning', *Neural Networks*, **19**, 90–106, (2006).

[15] Stephen Grossberg, 'Competitive learning: From interactive activation to adaptive resonance', *Cognitive Science*, **11**, 23–63, (1987).

[16] Fred H. Hamker, 'Life-long learning cell structures—continuously learning without catastrophic interference', *Neural Networks*, **14**, 551–573, (2001).

[17] Haibo He, Sheng Chen, Kang Li, and Xin Xu, 'Incremental learning from stream data', *IEEE Transactions on Neural Networks*, **22**(12), 1901–1914, (2011).

[18] Marc Kammer, Marko Tscherepanow, Thomas Schack, and Yukie Nagai, 'A perceptual memory system for affordance learning in humanoid robots', in *Proceedings of the International Conference on Artificial Neural Networks*, volume 6792 of *LNCS*, pp. 349–356. Springer, (2011).

[19] Stephan Kirstein and Heiko Wersing, 'A biologically inspired approach for interactive learning of categories', in *Proceedings of the International Conference on Development and Learning*, pp. 1–6. IEEE, (2011).

[20] Matej Kristan, Aleš Leonardis, and Danijel Skočaj, 'Multivariate online kernel density estimation with Gaussian kernels', *Pattern Recognition*, **44**(10–11), 2630–2642, (2011).

[21] Yann Prudent and Abdellatif Ennaji, 'An incremental growing neural gas learns topologies', in *Proceedings of the International Joint Conference on Neural Networks*, volume 2, pp. 1211–1216. IEEE, (2005).

[22] Frank Rosenblatt, 'The perceptron: A probabilistic model for information storage and organization in the brain', *Psychological Review*, **65**(6), 386–408, (1958).

[23] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams, 'Learning internal representations by error propagation', in *Parallel Distributed Processing – Explorations in the Microstructure of Cognition*, volume 1, 318–362, MIT Press, seventh edn., (1988).

[24] Marko Tscherepanow, 'Incremental on-line clustering with a topology-learning hierarchical ART neural network using hyperspherical categories', in *Poster and Industry Proceedings of the Industrial Conference on Data Mining*, pp. 22–34. ibai-publishing, (2012).

[25] Marko Tscherepanow, Nickels Jensen, and Franz Kummert, 'An incremental approach to automated protein localisation', *BMC Bioinformatics*, **9**(445), (2008).

[26] Marko Tscherepanow, Marko Kortkamp, and Marc Kammer, 'A hierarchical ART network for the stable incremental learning of topological structures and associations from noisy data', *Neural Networks*, **24**(8), 906–916, (2011).

[27] Mohammad-Taghi Vakil-Baghmisheh and Nikola Pavešić, 'A fast simplified fuzzy ARTMAP network', *Neural Processing Letters*, **17**(3), 273–316, (2003).

[28] Andrew R. Webb and Keith D. Copsey, *Statistical Pattern Recognition*, Wiley, third edn., 2011.

[29] James R. Williamson, 'Gaussian ARTMAP: a neural network for fast incremental learning of noisy multidimensional maps', *Neural Networks*, **9**(5), 881–897, (1996).