An Extended TopoART Network for the Stable On-Line Learning of Regression Functions

Marko Tscherepanow

Applied Informatics, Bielefeld University
Universitätsstraße 25, 33615 Bielefeld, Germany
marko@techfak.uni-bielefeld.de

Abstract. In this paper, a novel on-line regression method is presented. Due to its origins in Adaptive Resonance Theory neural networks, this method is particularly well-suited to problems requiring stable incremental learning. Its performance on five publicly available datasets is shown to be at least comparable to two established off-line methods. Furthermore, it exhibits considerable improvements in comparison to its closest supervised relative Fuzzy ARTMAP.

Keywords: Regression, On-line learning, TopoART, Adaptive Resonance Theory.

1 Introduction

For many machine learning problems, the common distinction between a training and an application phase is not reasonable (e.g., [13,18]). They rather require the gradual extension of available knowledge when the respective learning technique is already in application. This task can be fulfilled by on-line learning approaches. But in order to use on-line learning, additional problems have to be tackled. Probably the most important question is how new information can be learnt without forgetting previously gained knowledge in an uncontrolled way. This question is usually referred to as the *stability-plasticity dilemma* [11]. In order to solve it, Adaptive Resonance Theory (ART) neural networks were developed, e.g., Fuzzy ART [3] and TopoART [17].

In this paper, a regression method based on the recently published TopoART model [17] is presented. As well as being able to incrementally learn stable representations like other ART networks, TopoART is less sensitive to noise as it possesses an effective filtering mechanism. But since ART networks constitute an unsupervised learning technique, TopoART had to be extended in order to adapt it to the application field of regression.

In Section 2, an overview of regression methods, in general, and particularly related approaches is provided. Then, TopoART is briefly introduced in Section 3. Afterwards, the required extensions of TopoART are explained in Section 4. The resulting regression method is referred to as TopoART-R. It is evaluated using several datasets originating from the UCI machine learning repository [9] (see Section 5). Finally, the most important outcomes are summarised in Section 6.

B.-L. Lu, L. Zhang, and J. Kwok (Eds.): ICONIP 2011, Part II, LNCS 7063, pp. 562–571, 2011. The original publication is available at www.springerlink.com.

2 Related Work

Regression analysis estimates a regression function f relating a set of p independent variables i_k to q dependent variables d_k:

$$\underline{d} = f(\underline{i}) \quad \text{, with } \underline{i} = [i_1, \ldots, i_p]^T \text{ and } \underline{d} = [d_1, \ldots, d_q]^T . \tag{1}$$

The models and techniques used to approximate f vary considerably; for example, a linear model can be used [6]. Although this model is only capable of reflecting linear dependencies, its parameters (slope, y-intercept) can directly be derived from observed data without the need for an explicit optimisation. In contrast, more advanced models such as support vector regression (SVR) [16] or multi-layer perceptrons (MLPs) [8] can be applied so as to model complex dependencies. But the underlying models have to be optimised by solving a quadratic optimisation problem and by gradient descent, respectively. Recently, extreme learning machines (ELMs) [12] have been proposed as a special type of MLPs possessing a single hidden layer. Here, the weights and biases of the hidden nodes are randomly assigned and the weights of the output nodes are analytically determined based on a given training set.

In recent years, several approaches to on-line SVR have been proposed [14,15]. Since new input may change the role of previously learnt data in the model, they require the complete training set to be stored. In contrast to SVR, MLPs are inherently capable of on-line learning. But the training with new input alters already-learnt representations and the network topology has to be chosen in advance. The latter problem was solved by the Cascade-Correlation (CasCor) architecture [8,7]. CasCor incrementally creates a multi-layer structure, but demands batch-learning.

As mentioned above, ART networks [3,17] constitute a solution to the stability-plasticity dilemma. They learn a set of templates (categories) which efficiently represents the underlying data distribution; new categories are incorporated, if required. Therefore, they are particularly well-suited to incremental on-line learning. ART networks can be applied to supervised learning tasks using the ARTMAP approach [2]. ARTMAP combines two ART modules, called ART_a and ART_b, by means of an associative memory (map field). While ART_a clusters $\underline{i}$, ART_b clusters $\underline{d}$. Furthermore, associations from categories of ART_a to categories of ART_b are learnt in the map field. Although, in principle, the dependent variables can be reconstructed based on the associated categories, ARTMAP cannot directly be applied as a regression method. But there exist ARTMAP variants dedicated to classification such as Default ARTMAP [1]. Default ARTMAP has a simplified structure omitting the map field and ART_b. Moreover, it enables a distributed activation during prediction, which increases the classification accuracy.

In this paper, a regression method based on TopoART is proposed. In order to increase its accuracy, a distributed activation during prediction similar to Default ARTMAP was incorporated.

3 TopoART

Like Fuzzy ART [3], TopoART [17] represents input samples by means of hyper-rectangular categories. These categories as well as the associated learning mechanisms avoid catastrophic forgetting and enable the formation of stable representations. Similar to the Self-Organising Incremental Neural Network (SOINN) [10], TopoART is capable of learning the topological structure of the input data at two different levels of detail. Here, interconnected categories form arbitrarily shaped clusters. Moreover, it has been shown to be insensitive to noise as well. But TopoART requires significantly fewer parameters to be set and can learn both representational levels in parallel. Figure 1 shows the clusters resulting from training TopoART[1] with a 2-dimensional dataset comprising 20,000 samples, 10 percent of which are uniformly distributed random noise.

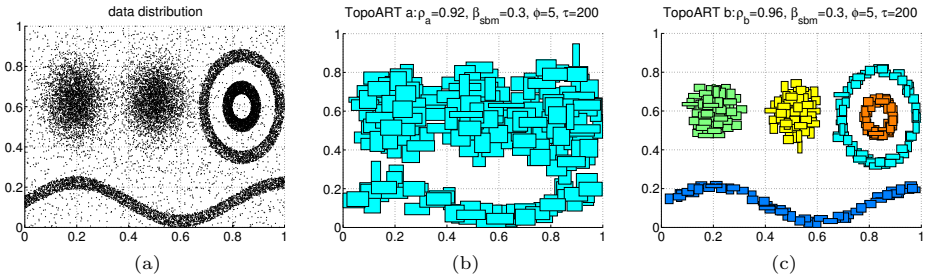

Fig. 1. Input distribution and clustering results of TopoART. After presenting each training sample of the dataset (a) to a TopoART network, it created a noise-free representation at two levels of detail. While only one cluster was formed by TopoART *a* (b), TopoART *b* distinguishes five clusters reflecting the data distribution in more detail (c). The categories associated with the same cluster share a common colour.

The two representational levels are created by two identical modules called TopoART *a* and TopoART *b*. As TopoART *a* controls which input samples are propagated to TopoART *b*, it functions as a filtering mechanism; in particular, only samples, which are enclosed by a category of TopoART *a* are propagated to TopoART *b*. In this way, noise regions are filtered effectively. Furthermore, the maximum category size is reduced from TopoART *a* to TopoART *b*. As a result, the structures represented by TopoART *b* exhibit a higher level of detail.

4 Using TopoART for Regression Analysis

Even though regression analysis constitutes a completely new application field for TopoART, its principal structure and mechanisms were directly adopted (see Fig. 2): TopoART-R consists of two modules (TopoART-R *a* and TopoART-R *b*)

[1] LibTopoART (version 0.20), available at `www.LibTopoART.eu`

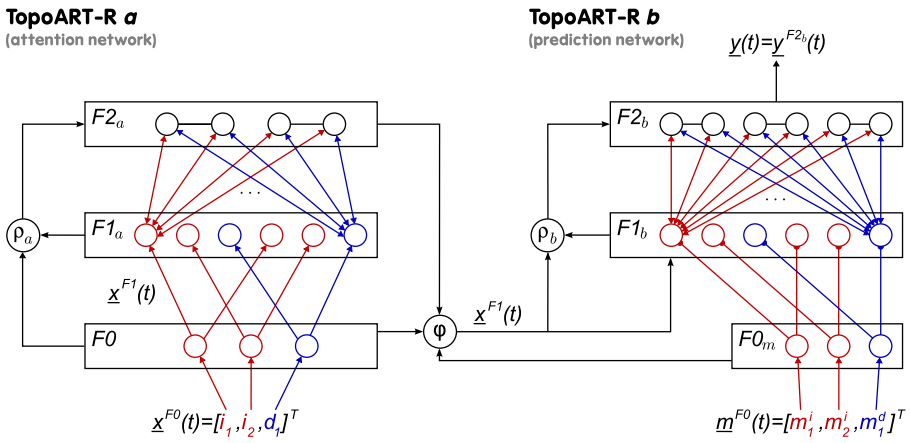

Fig. 2. Structure of TopoART-R. Like TopoART, TopoART-R encompasses two modules (TopoART-R a and TopoART-R b) sharing the input layer $F0$. But the connections of the F2 neurons can either be traced back to $\underline{i}$ or to $\underline{d}$. Furthermore, TopoART-R b has an additional input control layer ($F0_m$) that is required for prediction.

performing a clustering of the input at different levels of detail. As a consequence, the properties mentioned in Section 3 hold for the new application field as well. Nevertheless, several extensions had to be incorporated.

During training, the propagation of input to TopoART-R b depends on the activation of TopoART-R a: only input samples lying in a subspace defined by TopoART-R a reach TopoART-R b. Therefore, it is also called the 'attention network'. Predictions are provided by TopoART-R b. In order to fulfil this task, it requires the additional control layer $F0_m$.

4.1 Training TopoART-R

During training, the independent variables i_k and the dependent variables d_k are treated in the same way. For each time step t, the corresponding vectors $\underline{i}(t)$ and $\underline{d}(t)$ are concatenated and fed as input $\underline{x}^{F0}(t)$ into the TopoART-R network:

$$\underline{x}^{F0}(t) = \begin{bmatrix} \underline{i}(t) \\ \underline{d}(t) \end{bmatrix} = \big[i_1(t), \ldots, i_p(t), d_1(t), \ldots, d_q(t)\big]^T. \tag{2}$$

At the $F0$ layer, the input vectors $\underline{x}^{F0}(t)$ are encoded using complement coding:

$$\underline{x}^{F1}(t) = \big[i_1(t), \ldots, d_q(t), 1 - i_1(t), \ldots, 1 - d_q(t)\big]^T. \tag{3}$$

Due to the usage of complement coding, each element of an input vector $\underline{x}^{F0}(t)$ has to lie in the interval $[0, 1]$.

The set $\mathcal{I}$ summarises the indices of the elements of $\underline{x}^{F1}(t)$ related to $\underline{i}(t)$ and its complement, while the set $\mathcal{D}$ gives the indices for $\underline{d}(t)$ and its complement:

$$\mathcal{I} = \{1, \ldots, p, p{+}q{+}1, \ldots, 2p{+}q\}, \tag{4}$$
$$\mathcal{D} = \{p{+}1, \ldots, p{+}q, 2p{+}q{+}1, \ldots, 2p{+}2q\}. \tag{5}$$

The complement-coded input vectors $\underline{x}^{F1}(t)$ are propagated to the $F1$ layer of TopoART-R a. Then, the $F2$ nodes j of TopoART-R a are activated:

$$z_j^{F2_a}(t) = \frac{\left|\underline{x}^{F1}(t) \wedge \underline{w}_j^{F2_a}(t)\right|_1}{\alpha + \left|\underline{w}_j^{F2_a}(t)\right|_1} \quad \text{, with} \quad \alpha > 0. \tag{6}$$

Here, $|\cdot|_1$ and $\wedge$ denote the city block norm and an element-wise minimum operation, respectively. The activation $z_j^{F2}(t)$ (choice function) measures the similarity between $\underline{x}^{F1}(t)$ and the category represented by neuron j. Like with the original TopoART, the weights $\underline{w}_j^{F2_a}(t)$ span hyperrectangular categories.

The $F2$ node that has the highest activation is selected as the best-matching node bm. But it is only allowed to learn $\underline{x}^{F1}(t)$ if it fulfils the match function

$$\frac{\left|\underline{x}^{F1}(t) \wedge \underline{w}_{bm}^{F2_a}(t)\right|_1}{\left|\underline{x}^{F1}(t)\right|_1} \geq \rho_a; \tag{7}$$

i.e., if the category represented by its weights $\underline{w}_{bm}^{F2_a}(t)$ is able to enclose the presented input vector without surpassing a maximum size defined by the vigilance parameter ρ_a.

Using the original match function (7), a high variance of the dependent variables d_k could be compensated for by a low variance of the independent variables i_k. The result would be a high regression error. Therefore, the match function is independently computed for both components of the input vector $\underline{x}^{F0}(t)$:

$$\frac{\sum_k \min\left(x_k^{F1}(t), w_{bm,k}^{F2_a}(t)\right)}{\sum_k x_k^{F1}(t)} \geq \rho_a \quad \text{, for} \quad k \in \mathcal{I} \quad \text{and for} \quad k \in \mathcal{D}. \tag{8}$$

If (8) can be fulfilled, resonance of TopoART-R a occurs. Otherwise, the activation of neuron bm is reset and a new best-matching node is searched for. If no existing neuron is able to represent $\underline{x}^{F1}(t)$, a new node with $\underline{w}_{new}^{F2_a}(t{+}1){=}\underline{x}^{F1}(t)$ is incorporated.

Provided that TopoART-R a reached resonance, the weights $\underline{w}_{bm}^{F2_a}(t)$ are adapted as follows:

$$\underline{w}_{bm}^{F2_a}(t+1) = \underline{x}^{F1}(t) \wedge \underline{w}_{bm}^{F2_a}(t). \tag{9}$$

If a second-best-matching neuron sbm fulfilling (8) can be found, its weights are adapted as well:

$$\underline{w}_{sbm}^{F2_a}(t+1) = \beta_{sbm}\left(\underline{x}^{F1}(t) \wedge \underline{w}_{sbm}^{F2_a}(t)\right) + (1 - \beta_{sbm})\underline{w}_{sbm}^{F2_a}(t). \tag{10}$$

This is intended to reduce the sensitivity to noise, since the growth of categories in relevant areas of the input space is intensified.

As the weights are adapted after the presentation of single input samples and TopoART-R does not rely on the processing of whole datasets in order to compute weight changes (batch learning), it is always trained on-line.

In contrast to TopoART, no edge needs to be established between node *bm* and node *sbm*, as the topological structure of the input data is not used by TopoART-R. However, TopoART-R could learn topological structures, as well, if required by future applications.

Besides its weights, each $F2$ neuron j has a counter denoted by n_j^a, which counts the number of input samples it has learnt. Every τ learning cycles, all neurons with a counter smaller than ϕ are removed. Therefore, they are called *node candidates*. After n_j^a has reached the value of ϕ, the corresponding neuron can no longer be removed; i.e., it has become a permanent node.

$\underline{x}^{F1}(t)$ is only propagated to TopoART-R b if one of the two following conditions is fulfilled:

(i) TopoART-R a is in resonance and $n_{bm}^a \geq \phi$.
(ii) The input control layer $F0_m$ is activated; i.e., $\left|\underline{m}^{F0}(t)\right|_1 > 0$.

As during training all elements of $\underline{m}^{F0}(t)$ are set to 0, only input samples which lie in one of the permanent categories of TopoART-R a are learnt by TopoART-R b. By means of this procedure, the network becomes more insensitive to noise but is still able to learn stable representations.

After input has been presented to TopoART-R b, it is activated and adapted in the same way like TopoART-R a. Just the vigilance parameter is modified:

$$\rho_b = \frac{1}{2}(\rho_a + 1). \tag{11}$$

As a result of the increased value of the vigilance parameter, TopoART-R b represents the input distribution in more detail.

4.2 Predicting with TopoART-R

In order to predict missing variables with TopoART-R, the mask vector $\underline{m}^{F0}(t)$ must be set accordingly. Consequently, TopoART-R a can be neglected, as $\left|\underline{m}^{F0}(t)\right|_1 > 0$ (see Section 4.1). The mask vector comprises the values m_k^i and m_k^d which correspond to the elements of the input vector $\underline{x}^{F0}(t)$:

$$\underline{m}^{F0}(t) = \begin{bmatrix} \underline{m}^i(t) \\ \underline{m}^d(t) \end{bmatrix} = \left[m_1^i(t), \ldots, m_p^i(t), m_1^d(t), \ldots, m_q^d(t)\right]^T. \tag{12}$$

If these mask values are set to 1, the corresponding variables are to be predicted. Hence, they cannot be given in $\underline{x}^{F0}(t)$ and the respective elements of $\underline{x}^{F0}(t)$ are ignored. Presented variables are characterised by a mask value of 0. Hence, $m_k^i = 0$ and $m_k^d = 1$ for usual regression tasks. TopoART-R can even predict based on incomplete information; if the value of an independent variable i_l is unknown, m_l^i has to be set to 1. Then, i_l is not required as input and will be predicted like the dependent variables.

Each connection of all $F2_b$ neurons can be traced back to a specific element of the input vector $\underline{x}^{F0}(t)$ and to two elements of the complement-coded input vector $\underline{x}^{F1}(t)$ (see Fig. 2). Depending on the corresponding mask values, two disjunct sets $\mathcal{M}^0$ and $\mathcal{M}^1$ of $F1_b$ nodes are generated:

$$\mathcal{M}^0 = \left\{x, x{+}p{+}q : m_x^{F0}(t) = 0\right\}, \tag{13}$$

$$\mathcal{M}^1 = \left\{x : m_x^{F0}(t) = 1\right\}. \tag{14}$$

As the neurons of the mask layer $F0_m$ inhibit the corresponding $F1_b$ nodes (see Fig. 2), the activation of the $F2_b$ neurons is computed solely based on the non-inhibited F1 neurons summarised in $\mathcal{M}^0$. The activation function suggested for prediction with TopoART (cf., [17]) had to be adapted accordingly:

$$z_j^{F2_b}(t) = 1 - \frac{\sum_{k\in\mathcal{M}^0} \left|\min\left(x_k^{F1}(t), w_{jk}^{F2_b}(t)\right) - w_{jk}^{F2_b}(t)\right|}{\frac{1}{2}\sum_{k\in\mathcal{M}^0} 1}. \tag{15}$$

The activation $z_j^{F2_b}(t)$ computed according to (15) therefore denotes the similarity of $\underline{x}^{F1}(t)$ with $\underline{w}_j^{F2_b}(t)$ along those dimensions for which $m_x^{F0}(t){=}0$. The corresponding hyperrectangle is called a partial category.

In order to reconstruct the missing variables using a distributed activation, two cases are distinguished. Firstly, $\underline{x}^{F1}(t)$ lies inside the partial categories of one or more $F2_b$ neurons j. Then, the activation $z_j^{F2_b}(t)$ equals 1 for these neurons. Secondly, $\underline{x}^{F1}(t)$ is not enclosed by any partial category; i.e., the activation of all $F2_b$ neurons is lesser than 1.

In the first case, the missing variables are determined based on the information encoded in the partial categories: a temporary category $\underline{\tau}(t)$ is computed as the intersection of all categories that enclose $\underline{x}^{F1}(t)$. This intersection decreases in size if more neurons are involved. Thus, the more partial categories contain $\underline{x}^{F1}(t)$, the better is it represented by the network.

Since the weight vectors encode lower and upper bounds along all coordinate axes, the intersection is computed as the hyperrectangle with the respective largest lower bound and the smallest upper bound over all considered categories. Due to the usage of complement coding, this operation can be performed using the element-wise maximum operator $\vee$:

$$\underline{\tau}(t) = \bigvee_j \underline{w}_j^{F2_b}(t) \quad , \forall j : z_j^{F2_b}(t) = 1. \tag{16}$$

As $\underline{\tau}(t)$ covers all dimensions including those corresponding to the missing variables, it can be applied for computing predictions. These predictions are summarised in the output vector $\underline{y}(t)$. Its elements $y_k(t)$ are set to -1 if the corresponding variable was contained in the input vector $\underline{x}^{F0}(t)$. Otherwise, it gives a prediction which is computed as the mean of the temporary category's upper and lower bound along the k-th axis of the input space:

$$y_k(t) = \begin{cases} -1 & \text{, for } k \notin \mathcal{M}^1 \\ \frac{1}{2}\tau_k(t) + \frac{1}{2}\big(1 - \tau_{k+p+q}(t)\big) & \text{, for } k \in \mathcal{M}^1 \end{cases}. \tag{17}$$

In the second case, i.e, if no partial category contains $\underline{x}^{F1}(t)$, an intersection similar to (16) does not lead to a valid temporary category. Therefore, the temporary category is constructed as a weighted combination of the categories with the smallest distances to $\underline{x}^{F1}(t)$:

$$\underline{\tau}(t) = \frac{\sum_{j \in \mathcal{N}} \left(\frac{1}{1 - z_j^{F2_b}(t)} \cdot \underline{w}_j^{F2_b}(t) \right)}{\sum_{j \in \mathcal{N}} \frac{1}{1 - z_j^{F2_b}(t)}}. \tag{18}$$

The contribution of each node j is inversely proportional to $1 - z_j^{F2_b}(t)$; i.e., more similar categories have a higher impact. The set $\mathcal{N}$ of very similar categories is determined as follows:

$$\mathcal{N} = \{x : z_x^{F2_b}(t) \geq \mu + 1.28\sigma\}. \tag{19}$$

Here, μ and σ denote the mean and the standard deviation of $z_j^{F2_b}(t)$ over all $F2_b$ neurons. Assuming a Gaussian distribution, $\mathcal{N}$ would only contain those 10% of the neurons that have the highest activations. For computational reasons, $\mathcal{N}$ is further restricted to a maximum of 10 nodes.

5 Results

For the evaluation of TopoART-R, we chose five different datasets from the UCI machine learning repository [9]: Concrete Compressive Strength [19], Concrete Slump Test [20], Forest Fires[2] [5], and Wine Quality [4]. These datasets were selected, since they can be used with regression methods and contain real-valued attributes without missing values. For computational purposes and comparison reasons, all variables were normalised to the interval $[0, 1]$.

The performance of TopoART-R was compared to three different state-of-the-art methods: ν-SVR (with a radial basis function kernel) implemented in LIBSVM (version 3.1), CasCor, and Fuzzy ARTMAP. SVR and CasCor learn the regression function in batch mode; i.e., the training requires a complete dataset to be available. In contrast, Fuzzy ARTMAP and TopoART-R learn a sample directly after its presentation independently of other samples (on-line learning). Since Fuzzy ARTMAP learns a mapping to categories representing the dependent variables rather than a mapping to the dependent variables themselves (cf. Section 2), the centre of the ART_b category connected to the best-matching node of the map field was used as prediction.

For all regression methods, the mean squared error (MSE) was computed for each dataset using five-fold cross-validation. The most relevant parameters

[2] The integer attributes X and Y as well as the nominal attributes month and day were ignored.

were determined by means of grid search.[3] The minimum MSEs reached by each approach using the optimal parameter setting are given in Table 1. For SVR and CasCor, the respective batch learning scheme was applied. Since the number of samples contained in the datasets is rather small (e.g., 103 samples in the Concrete Slump Test dataset), the training sets were repeatedly presented to Fuzzy ART and TopoART until their weights converged. Although these methods learn on-line, they require a sufficiently high number of training steps which depends on the chosen learning rates (β, β_{ab}, and β_{sbm}).

Table 1. Minimum MSEs. The bold numbers indicate the best result for each dataset.

dataset	SVR	CasCor	Fuzzy ARTMAP	TopoART-R
Concrete Compressive Strength	**0.0054**	0.0069	0.0302	0.0119
Concrete Slump Test	0.0656	**0.0370**	0.0597	0.0475
Forest Fires	0.0034	0.0035	0.0037	**0.0032**
Wine Quality (red)	0.0161	0.0164	0.0188	**0.0143**
Wine Quality (white)	0.0122	0.0147	0.0173	**0.0105**

Table 1 shows that TopoART-R achieved the lowest MSEs for three of five datasets. Furthermore, it performed always better than Fuzzy ARTMAP, which is its closest supervised relative. Thus, TopoART-R constitutes a promising alternative to established regression methods.

6 Conclusion

In this paper, a regression method based on the unsupervised TopoART network was introduced. Due to its origins in ART networks, it is particularly suited to tasks requiring stable on-line learning. The performance of TopoART-R on standard datasets has been shown to be excellent. This is most likely a result of its noise reduction capabilities inherited from TopoART as well as the distributed activation during prediction. Finally, TopoART-R offers some properties which might be of interest for future applications: it can learn the topological structure of the presented data similar to TopoART and predict based on incomplete information if the mask vector is set appropriately. The latter property could be crucial if predictions are to be made using data from sensors with different response times.

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

[3] SVR: ν, C, and γ; CasCor: learning rate and activation function of the output nodes (logistic, arctan, tanh); Fuzzy ART: ρ, β, and β_{ab}; TopoART: ρ_a, ϕ, and β_{sbm}

References

1. Carpenter, G.A.: Default ARTMAP. In: Proceedings of the International Joint Conference on Neural Networks. vol. 2, pp. 1396–1401. IEEE (2003)
2. Carpenter, G.A., Grossberg, S., Reynolds, J.H.: ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network. Neural Networks 4, 565–588 (1991)
3. Carpenter, G.A., Grossberg, S., Rosen, D.B.: Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. Neural Networks 4, 759–771 (1991)
4. Cortez, P., Cerdeira, A., Almeida, F., Matos, T., Reis, J.: Modeling wine preferences by data mining from physicochemical properties. Decision Support Systems 47(4), 547–553 (2009)
5. Cortez, P., Morais, A.: A data mining approach to predict forest fires using meteorological data. In: Proceedings of the Portuguese Conference on Artificial Intelligence. LNAI, vol. 4874, pp. 512–523. Springer, Berlin (2007)
6. Edwards, A.L.: An Introduction to Linear Regression and Correlation. W. H. Freeman and Company, San Francisco (1976)
7. Fahlman, S.E., Lebiere, C.: The cascade-correlation learning architecture. In: Neural Information Processing Systems 2. pp. 524–532. Morgan Kaufmann, San Mateo (1989)
8. Fausett, L.: Fundamentals of Neural Networks – Architectures, Algorithms, and Applications. Prentice Hall, New Jersey (1994)
9. Frank, A., Asuncion, A.: UCI machine learning repository (2010), `http://archive.ics.uci.edu/ml`
10. Furao, S., Hasegawa, O.: An incremental network for on-line unsupervised classification and topology learning. Neural Networks 19, 90–106 (2006)
11. Grossberg, S.: Competitive learning: From interactive activation to adaptive resonance. Cognitive Science 11, 23–63 (1987)
12. Huang, G.B., Zhu, Q.Y., Siew, C.K.: Extreme learning machine: Theory and applications. Neurocomputing 70, 489–501 (2006)
13. Lee, D.H., Kim, J.J., Lee, J.J.: Online support vector regression based actor-critic method. In: Proceedings of the Annual Conference of the IEEE Industrial Electronics Society. pp. 193–198. IEEE (2010)
14. Ma, J., Theiler, J., Perkins, S.: Accurate on-line support vector regression. Neural Computation 15, 2683–2703 (2003)
15. Martin, M.: On-line support vector machine regression. In: Proceedings of the European Conference on Machine Learning. LNCS, vol. 2430, pp. 173–198. Springer, Berlin (2002)
16. Schölkopf, B., Smola, A.J., Williamson, R.C., Bartlett, P.L.: New support vector algorithms. Neural Computation 12, 1207–1245 (2000)
17. Tscherepanow, M.: TopoART: A topology learning hierarchical ART network. In: Proceedings of the International Conference on Artificial Neural Networks. LNCS, vol. 6354, pp. 157–167. Springer, Berlin (2010)
18. Tscherepanow, M., Jensen, N., Kummert, F.: An incremental approach to automated protein localisation. BMC Bioinformatics 9(445) (2008)
19. Yeh, I.C.: Modeling of strength of high performance concrete using artificial neural networks. Cement and Concrete Research 28(12), 1797–1808 (1998)
20. Yeh, I.C.: Modeling slump flow of concrete using second-order regressions and artificial neural networks. Cement and Concrete Composites 29(6), 474–480 (2007)