Pattern Recognition Letters xxx (2011) xxx–xxx

Contents lists available at SciVerse ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

On the optimization of Hierarchical Temporal Memory

Ioannis Kostavelis, Antonios Gasteratos [*]

Laboratory of Robotics and Automation, Department of Production and Management Engineering, Democritus University of Thrace, Vasilissis Sophias 12, GR-671 00 Xanthi, Greece

ARTICLE INFO

Article history:
Received 21 June 2011
Available online xxxx
Communicated by A. Petrosino

Keywords:
Hierarchical memory
Supervised learning
Correlation matrix
Logpolar

ABSTRACT

In this paper an optimized classification method for object recognition is presented. The proposed method is based on the Hierarchical Temporal Memory (HTM), which stems from the memory prediction theory of the human brain. As in HTM, this method comprises a tree structure of connected computational nodes, whilst utilizing different rules to memorize objects appearing in various orientations. These rules involve both the spatial and the temporal module. As HTM is inspired from brain activity, its input should also comply with the human vision system. Thus, for the representation of the input images the logpolar was given preference to the Cartesian one. As compared to the original HTM method, experimental results exhibit performance enhancements with this approach, in recognition and categorization applications. Results obtained prove that the proposed method is more accurate and faster in training, whilst retaining the network robustness in multiple orientation variations.

1. Introduction

Object recognition and classification is an open challenge in computer vision systems (Kjellström et al., 2010). Humans are able to recognize a great variety of objects with little effort, even when they are distorted in terms of orientation, color, shape etc., or even when they are partially occluded. Although computer vision techniques provide adequate solutions to such problems, there is still a long way to go to achieve human-like capabilities. The theory of Hawkins and Blakeslee (2004) constitutes such an attempt. This theory supports the view that the machine learning techniques should follow a hierarchical structure, similar to that of the brain. This idea was further expanded with the introduction of the HTM model (Hawkins and George, 2006), which is a computing method replicating the structural and algorithmic properties of the human neocortex. HTM models are hierarchical networks capable to provide efficient solutions in different domains, such us pattern recognition, control theory and behavior generation among others (Numenta, 2006). HTM has been implemented both for unsupervised and supervised learning applications. Bundzel and Hashimoto (2010) introduced a system which enables a mobile robot to explore the environment and recognize different types of objects with the use of an external labeling mechanism. Their method involves a stereo camera as input to the lowest level of a HTM and learns to distinguish objects by using the frequency of occurrence of spatiotemporal patterns. Rozado et al. (2010) attempted to extend the traditional HTM function providing a solution to the problem of sign language recognition. In that work they proposed an additional top node in the HTM's hierarchy to manipulate more efficiently the input patterns that should be learned from the system. Highly motivated from the work of Stenger et al. (2004), where hand pose estimation was achieved using hierarchical detection, Kapuscinski (2010) utilized HTM for hand shape recognition. A two-level HTM was implemented taking multiview orientations of different hand shapes as input. Recently, Melis and Kameyama (2009) implemented a HTM for a traffic sign recognition task, proving that the color information is of particular importance in object recognition. A rather more complicated algorithm was introduced by Csapó et al. (2007), relying on the combination of Visual Feature Array model (VFA) with a HTM network; able to distinguish objects presented in different scales. Therefore, it is evident that the utilization of a HTM in an object recognition problem can significantly improve the performance, even when classic machine learning algorithms fail to handle efficiently such issues.

In this paper, a hierarchical structured machine learning technique has been developed, inspired from the design of a HTM network as described by George and Jaros (2007). The proposed work constitutes a supervised learning method used to recognize objects in different orientations. Although the idea of the HTM network has already become mature enough, specific rules for a robust design of such a network have not been established yet. Contrary to the aforementioned techniques, the proposed method introduces specific alternative rules for the design of each building block of a HTM. These rules expand both the spatial and the temporal module of the network. Concerning the spatial module, the storage of quantization centers is governed by a criterion that minimizes the scatter of the centers placed within the same node. In object recognition and classification applications, where the data used for the training are limited and do not exhibit any strict time

[*] Corresponding author. Tel.: +30 25410 79359; fax: +30 25410 79331.
E-mail addresses: gkostave@pme.duth.gr (I. Kostavelis), agaster@pme.duth.gr (A. Gasteratos).

0167-8655/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2011.11.017

succession, the temporal module does not possess a natural meaning. Thus, the temporal module has been altered using the Pearson Correlation Matrix at each node. Considering the biological inspiration from the human brain the input in the bottom level was chosen to comply with the human vision system. Therefore, the logpolar transformation (Sandini and Tagliasco, 1980), i.e. a mapping from the Cartesian coordinate system to the retinal one, has been applied onto the dataset images. Logpolar mapping has been utilized in several bio-inspired applications (Manzotti et al. (2001), Metta et al. (2004)). The performance of the proposed method has been evaluated using the ETH-80 datasets (CogVis, 2003), which include eight different objects in various orientations. For a comparative experimental validation, the HTM described by George and Jaros (2007) has also been implemented.

2. Hierarchical Temporal Memory structure

Hawkins and George of Numenta Inc. created the HTM network with roots to the Bayesian networks (Mittal and Pagalthivarthi, 2007). The most important feature of such a model is its resemblance to the algorithmic structure of the neocortex. The HTM model is considered as a memory system organized in a tree-shaped hierarchy. This hierarchical structure is divided into several levels, each of which consists of adjoint computational nodes. Ascending the hierarchy, the number of nodes decreases and the top level consists only of a single node. The number of nodes comprising level v is $2^{2\lambda-v}$, where λ is the number of the levels in the network. *Level* 0 corresponds to the images presented to the network. The input images are also divided into patches of n by n pixels. Each patch of the input image corresponds to a single computational node at the first level. Nodes receive inputs from spatially-specific areas, namely the receptive fields. Inference flow from one node to the other follows a bottom-up path. Lower level nodes find causes belonging only to a limited time and input scale while higher-level nodes are able to observe causes on a larger time scale and wider image input. Each node follows the same algorithmic procedure independent of the level it belongs to (George and Jaros, 2007); it comprises two different modules, viz. the *spatial* and the *temporal* and it undergoes two distinct operation mechanisms: the first one is the training mode, which includes the spatial and the temporal procedures, whilst the second one is the inference mechanism, where the node produces outputs to be fed into the higher nodes.

2.1. Spatial module

A node at *level* 1 receives a patch of size n by n pixels from the corresponding receptive field of *level* 0. This patch is reshaped into an input vector $\boldsymbol{q} = (q_1, q_2, \ldots, q_{nxn})$, where q_i correspond to pixel values. The node is exposed to the training images sequentially and learns the quantization space of such input vectors. The subspace of such vectors, eventually added to the node, are the quantization centers. This learning procedure is implemented by a simple algorithm, governed by a threshold parameter D, which corresponds to the minimum Euclidean distance, above which an input vector is considered different from the existing quantization centers. The first input vector presented in the network is considered a new quantization center and is maintained in the node. Each new input vector appearing, is checked against a quantization center that stands closer than distance D from the input vector. In the absence of such a quantization center, the new input vector is pooled into the node and becomes a new quantization center, otherwise, it is ignored. The value of the threshold D should be carefully chosen since a significantly low value will lead to a node with excessively many quantization centers. On the other hand, a large threshold may lead to a node with centers corresponding to very different input vectors. Such a learning procedure converges when the rate of adding new centers becomes lower than a predefined value. Once the learning procedure is concluded, the spatial module is trained and can produce outputs to the temporal module.

2.2. Temporal module

While the spatial module receives inputs from *level* 0 and infers, the temporal module obtains this output and starts learning. For each new input vector presented to a node, all the Euclidean distances between the quantization centers and the query input vector are then calculated. The probability distribution that the input vector belongs to each center is then calculated and is fed to the temporal module, to form the so called time adjacency matrix (TAM). Considering that the spatial module consist of N quantization centers, an NxN zero matrix T is first created. The rows and the columns of the matrix correspond to the centers triggered at time t and $t+1$, respectively. From this procedure, the ith, jth quantization centers are selected, corresponding to the query input vectors at times t and $t+1$, respectively. Following this, the $T(i,j)$ element of the TAM is then increased by one and the same operation is repeated until the T matrix is stabilized sufficiently. The physical meaning of this procedure is that quantization centers with time proximity are grouped together in T. This matrix is used by the system to learn a mixture of first order Markov chains, defined over the set of the quantization centers.

2.3. Temporal grouping module

The next step includes the partitioning of the TAM, which is a large transition matrix. The main objective of this operation is to cluster the quantization centers into temporal coherent subgroups. Each subgroup corresponds to a Markov chain and includes those quantization centers that are likely to occur sequentially in time (George and Hawkins, 2009). An example of the learned 1st order Markov chains for one node of the first level of the implemented HTM could be found at (Kostavelis and Gasteratos, 2011). Considering the TAM as a weighted directed network this problem is treated as a graph partitioning (Hayes, 2000). The algorithm that has been adopted as a solution to this problem is (George and Jaros, 2007):

1. Locate the quantization center with the greatest connectivity.
2. Select the M quantization centers with the greatest connectivity to the quantization center in step 1.
3. Repeat step 2 for each newly-added quantization center.
4. Create a new group comprising the quantization centers that arise from steps 2, 3.
5. Repeat, from step 1, until all the quantization centers have been arranged into a specific group.

Assuming the number of the temporal groups created for a node are G, the output of the temporal module is also a vector of size G. The completion of the graph partitioning indicates the end of the training, i.e. the node is able to draw inferences. During this mode two cases are distinguished to treat the temporal groups resulting from the grouping procedure. The first one being the utilization of noiseless data and, therefore, the output of the node would be a binary vector and the second one the use of noisy data, where the output of the node is considered as a probability distribution over the temporal groups. The proposed network treats the general, noisy case, and incorporates this method for the transmission of information through all levels of the network.

Please cite this article in press as: Kostavelis, I., Gasteratos, A. On the optimization of Hierarchical Temporal Memory. Pattern Recognition Lett. (2011), doi:10.1016/j.patrec.2011.11.017

3. Optimized HTM network

In our approach, both the spatial and the temporal modules are replaced with customized routines, whereas the logpolar transformation of the Cartesian input images has been adopted. In the spatial module, the proposed algorithm stores new centers in a uniform fashion, whereas the temporal module has been replaced by a correlation matrix derived from the correlations between the quantization centers (Tyler, 2008).

3.1. Logpolar transformation

In the human vision system, it has been established that the excitation of the cortex can be approximated by a logpolar mapping of the eye's retinal image. By altering the Cartesian input to the bottom level of the network with a cortical one, as depicted in Fig. 1, we move a step ahead towards a biologically based learning system. The logpolar mapping can be expressed as a transformation between the polar (retinal) plane (ρ,θ), the logpolar (cortical) one (ξ,η) and finally the Cartesian (image) one (x,y). The respective equations follow:

$$\eta = q \cdot \theta \tag{1}$$

$$\xi = \log_a \frac{\rho}{\rho_0} \tag{2}$$

where ρ_0 is the minimum spatial resolution corresponding to the radius of the innermost cycle, $1/q$ is the minimum angular resolution of the logpolar layout and (ρ,θ) are the respective polar coordinates, which in turn are related to the conventional Cartesian reference system by:

$$x = \rho \cos\theta \tag{3}$$

$$y = \rho \sin\theta \tag{4}$$

3.2. Modified spatial module

The spatial module is sequentially exposed to new input vectors. As all the nodes in the hierarchy store new quantization centers in the same way, it is sufficient to describe the operations in the first level of the network. The inputs to the nodes of the first level are cortical images, corresponding to the input vectors, for which the spatial module has to learn a representative subset. The stored input vectors are the quantization centers that encode the knowledge of the network. It is imperative that these centers are carefully selected to ensure that the spatial module will be able to learn a finite space of quantization centers from an infinite number of input vectors. At this point a balance between the number of the quantization centers stored and the rest of the space, that is not represented by any quantization center, should be reached. The unlimited storage of quantization centers for each node, given their large amount in such a network, will soon run the system out of memory; whilst the conservative storage of quantization centers will lead to a subspace unable to distinguish noisy input vectors. Given these constraints, the selected quantization centers should delimit efficiently the theoretically infinite input space.

In the beginning of the learning, the first input vector is considered as a quantization center at the respective node. Assuming that the learned quantization space in the spatial module of a node is $\boldsymbol{Q} = [\boldsymbol{q_1}, \boldsymbol{q_2}, \ldots, \boldsymbol{q_N}]^T$, where $\boldsymbol{q_i}$ corresponds to quantization centers and N is the number of the existing centers, all the Euclidean distances d between these centers are calculated and their sum S is then computed:

$$S = \frac{1}{2} \sum_{i}^{N} \sum_{j}^{N} d(\boldsymbol{q_i}, \boldsymbol{q_j}) \tag{5}$$

As a new input vector $\boldsymbol{q_c}$ appears in the receptive field of the node, all the distances d within the existing centers $\boldsymbol{Q}$ and the new input vector $\boldsymbol{q_c}$ are computed. The sum S_c is then calculated:

$$S_c = S + \sum_{i}^{N} d(\boldsymbol{q_i}, \boldsymbol{q_c}) \tag{6}$$

The value of S represents the scatter of the existing quantization centers in the node. Therefore, any new input vector should be subsumed in the node only when the resulting scatter S_c is much greater than the previous one. This approximation ensures that input vectors without any substantial information are not considered as new centers. Therefore, for each new input vector the alteration $(alt = (S - S_c)/S)$ between S and S_c should be examined against a threshold T. If $alt > T$ the query input vector becomes a new quantization center; otherwise, the next input vector is examined.

The learning of the spatial module is completed when the quantization centers added describe sufficiently the space. At the beginning of the learning function, the node adds new centers rapidly, yet this center pooling procedure decreases with time, as described in (George and Jaros, 2007). The learning is completed when the rate of adding new quantization centers falls bellow a predefined threshold. The proposed method manages to store only those quantization centers that contain substantial information of the

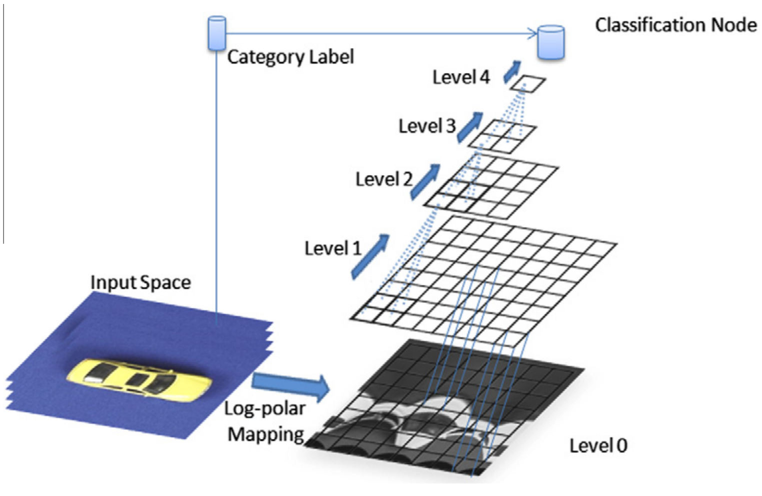

Fig. 1. The proposed HTM network structure for supervised learning.

Please cite this article in press as: Kostavelis, I., Gasteratos, A. On the optimization of Hierarchical Temporal Memory. Pattern Recognition Lett. (2011), doi:10.1016/j.patrec.2011.11.017

great variety of images that may appear in an object recognition problem. The main advantage of this approach is the decrease of the computational cost and the minimization of memory resources. In the example presented in Fig. 2, the two different methods are examined according to their ability to learn a given two-dimensional circular space. Every point belonging to the image is considered as a 2D vector. These vectors are presented randomly in time and the different methods should add to their nodes those quantization centers that characterize the input space in a more representative way. The rate threshold was set to the value of 0.1 in both methods; i.e. for every 100 new input vectors, when more than 10 new centers were added, the learning procedure should go on. For the pooling of new quantization centers in the spatial module, very strict thresholds were utilized. In our example, during the learning procedure the proposed method converges in a few iterations as exhibited in Fig. 2(b), where the center adding rate for the proposed method is indicated by a dashed line and that for the original HTM in a continuous one. Additionally, the learned space is more uniform and the stored quantization centers in Fig. 2(d) represent sufficiently all the area of the input space. Obviously the proposed method exhibits better attributes during the learning procedure, leading to more efficient structures.

3.3. Correlation module

The approximation, as introduced in Section 2.2, is very suitable in applications where the input corresponds to a time-based sequence of images, e.g. video, where the successive images share a great amount of common information. Thus, the input vectors presented in the receptive field of a node over time mostly represent the same content shifted by a small portion of pixels. Therefore, it could be said that in such cases the input vectors with time proximity would also have spatial one. Albeit, in object recognition and categorization problems, where the system has to be trained from a queue of images with no time proximity, such an approximation does not have a physical meaning. The input images presented successively at the lowest level rarely share any amount of common information. As a result, the receptive field of a node receives input vectors can differ remarkably. However, matrix T is highly affected by the order the images are fed into the lowest level of network. Evaluating such a technique in a multi-class object recognition problem, where the images are not time sequential, the groups formed, as described in Section 2.3, leak coherence.

In the present work we handle the temporal module in a different manner. In the first instance, the temporal module it is replaced by a correlation coefficient matrix. The correlation matrix R for the N quantization centers is then calculated. Besides the great variety, the most common measure of correlation is Pearson's coefficient (Miller, 2006), which obtain values in $[-1, 1]$. The correlation matrix is an N by N matrix containing the Pearson correlation coefficients between all the possible pairs of quantization centers. The larger the absolute value of correlation, the stronger the association between the two variables and the more accurate the prediction of any variable. A correlation value greater than 0.7 denotes strong positive correlation between the two variables. The correlation coefficient r within all the quantization centers is calculated by:

$$\boldsymbol{R}(\boldsymbol{q_i}, \boldsymbol{q_j}) = \frac{E[(\boldsymbol{q_i} - \mu_{\boldsymbol{q_i}})(\boldsymbol{q_j} - \mu_{\boldsymbol{q_j}})]}{\sigma_{\boldsymbol{q_i}} \sigma_{\boldsymbol{q_j}}} \tag{7}$$

where E is the expected value operator and $\sigma_{\boldsymbol{q}}$ is the standard deviation of the respective quantization center. The $\boldsymbol{R}(\boldsymbol{q_i}, \boldsymbol{q_j})$ is a diagonal

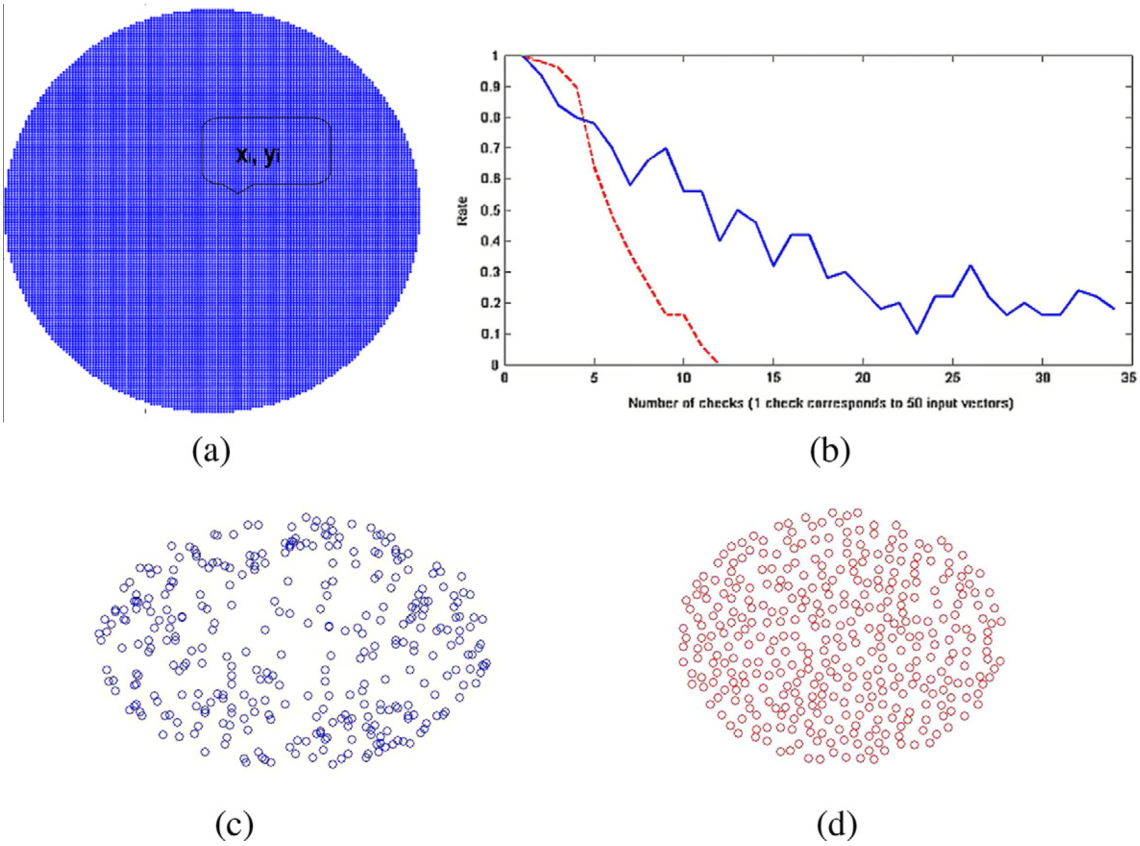

Fig. 2. (a) The circular input space presented in the receptive field of the node, (b) the rate of adding new quantization centers, (c) a mapping of the learned quantization centers using Numenta's Inc. method and (d) a mapping of the learned quantization centers according to the proposed method.

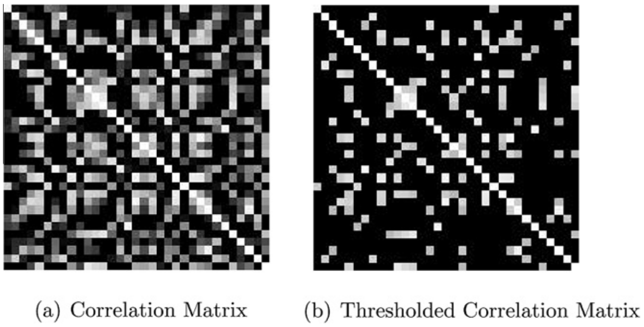

Fig. 3. (a) The correlation matrix between the quantization centers in a node and (b) the thresholded correlation matrix with a threshold of 0.8 corresponding to statistical significance greater than 95%.

matrix as any quantization center is fully correlated with itself. The resulted correlation matrix (Fig. 3(a)) is thresholded and only correlation values greater than 0.8 are kept (Fig. 3(b)). In order to validate the existence of correlation between the quantization centers, the statistical significance of the hypothesis that there is no correlation is also examined. If the correlation $r(\boldsymbol{q_i}, \boldsymbol{q_j})$ is greater than 0.8 with statistical significance greater than 95% then the correlation value is kept, otherwise it is discarded. The resulting correlation matrix is then grouped following the same technique as described in Section 2.3. During this procedure, the quantization centers are separated into highly correlated coherent subgroups, which do not depend on the order of appearance of the query input vectors. The main advantage of this technique is that the learning of the temporal module is by-passed. In the original approximation of this module as described in 2.2, new input vectors are required in order to form a sufficiently stabilized TAM. Consequently, in the proposed solution, the spatial module should not infer, in order to form the resulted matrix. The formation of the correlation coefficient matrix does not demand an iterative learning procedure and the computational cost is reduced, as a result.

4. Experimental validation

In this section the proposed method is evaluated. The network described by George and Jaros (2007) has been implemented for benchmarking against the optimized HTM. The two different networks complied to the same tree-shaped structure, ensuring thus the comparability of the experimental results. In both versions of the HTM, respective 4-level networks were implemented ($v = 4$). The node placed on the top of the hierarchy is the classification one with input images of size 32 by 32 pixels. Thus, each node in *level* 0 receives patches of 4 by 4 pixels, which are then reshaped to the input vectors. For the experimental validation the ETH-80 dataset was utilized that contains 8 different classes, each one represented by 10 similar objects in 41 orientations. The size of the images is 256 by 256 pixels and, therefore, the input images should be modified in order to fit into the receptive field of *level* 1 of the network. In the original version of the HTM the initial images of the dataset were linearly scaled down to 32 by 32 pixels; whereas in the proposed method the input images were mapped according to the logpolar transformation and the resulting images were also of size 32 by 32 pixels. The proposed modification outperforms the simple Cartesian down-sampling procedure, as it retains greater detail at the regions of interest, without significant local loss of information. During the learning the dataset was divided into a train and a test set corresponding to 70% and to 30% of the total size of the dataset, respectively.

In a HTM network, to maximize the performance, the selection of the parameters for the spatial module is of great importance. The parameters need to be determined is the distance threshold *D*, which is the criterion for adding new quantization centers in the spatial module and the rate that terminates the learning procedure. The learning rate was set to 0.1 and assumed fixed for the rest of the experimental procedure. The distance thresholds both for the original and the proposed spatial module were regulated through an exhaustive search, which maximizes the classification rate. In the top node of the hierarchy, a *k*-nearest neighbor classifier was utilized in both methods. The most appropriate threshold for the original and the proposed implementation is 0.45 and 0.08, respectively, achieving corresponding classification rates of 76.21% and 84.44%. Fig. 4 depicts the average efficiency for all the classes based on the distance threshold of the spatial module. The parameter *k* for the nearest neighbor classifier was examined among others and the most suitable one was found to be $k = 3$.

The next step is the comparison between the two different networks. The contribution of the proposed modules in the HTM has thus been evaluated. Firstly, the efficiency of the original version of the HTM was examined. Secondly, every building block of the network was replaced individually by the corresponding, modified one and the resulting efficiency was also examined. Thus, the contributions of each separate module, viz. the input, the spatial and the correlation one, were evaluated. Lastly, the total efficiency of the combined HTM network was also examined. The overall results are summarized in Table 1. Every single improvement in the basic building blocks results in a better classification rate than the original version. The correlation module is seen to boost the accuracy (79.06%), whereas the combination effect of all the proposed changes into a single network results in an accuracy of 84.44%.

The efficiency of the examined methods could be further increased with the utilization of a stronger classifier in the top node of the hierarchy. Therefore, the *k*-nearest neighbor classifier was replaced by a Support Vector Machine (SVM). This is a binary

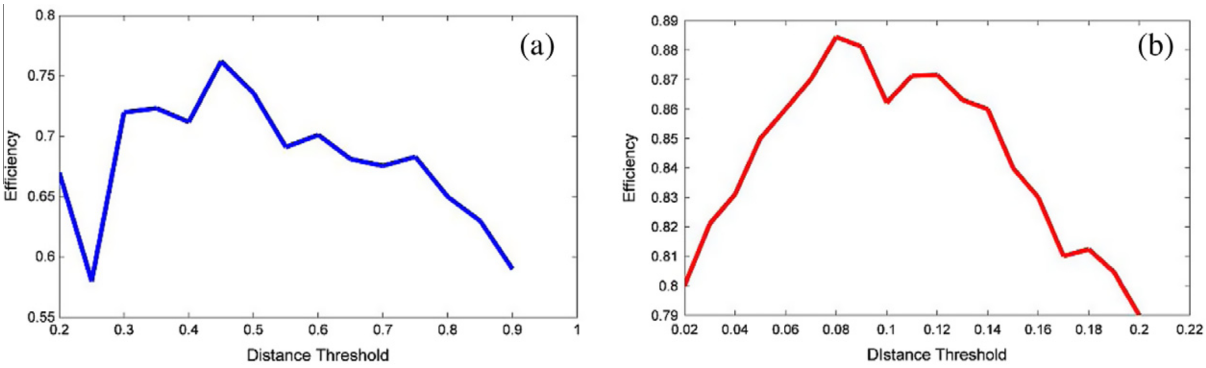

Fig. 4. The average efficiency for all the classes based on the distance threshold of the spatial module (a) for the original implementation and (b) for the proposed method.

Please cite this article in press as: Kostavelis, I., Gasteratos, A. On the optimization of Hierarchical Temporal Memory. Pattern Recognition Lett. (2011), doi:10.1016/j.patrec.2011.11.017

Table 1
Classification rate for the original and the proposed HTM using *k*-nn.

Class type	Original version (%)	Original with logpolar input (%)	Original with new spatial module (%)	Original with correlation module (%)	Proposed HTM (%)
Apple	75.23	77.10	76.11	77.32	83.32
Car	78.46	79.34	83.21	80.45	89.51
Cow	76.17	76.34	76.61	79.41	82.50
Cup	79.56	80.12	82.52	84.35	89.96
Dog	77.32	77.89	78.80	79.34	84.12
Horse	72.45	74.10	73.17	77.40	81.23
Pear	79.11	80.11	81.00	82.00	84.31
Tomato	71.34	73.45	72.16	72.21	80.56
Average efficiency	**76.21**	77.31	77.95	79.06	**84.44**

Table 2
Classification rate for the original and the proposed HTM using SVM.

Class type	Original HTM version		Optimized HTM Version	
	Train set (%)	Test set (%)	Train set (%)	Test set (%)
Apple	97.30	85.12	98.45	93.35
Car	99.00	90.12	99.21	98.65
Cow	98.10	87.35	99.10	94.22
Cup	97.34	91.45	97.00	98.65
Dog	98.10	90.46	98.34	93.31
Horse	99.20	88.30	98.26	95.28
Pear	97.44	91.80	97.14	94.67
Tomato	98.35	86.12	98.25	93.46
Average efficiency	98.10	88.84	98.22	95.20

classifier, yet the one versus all methodology was utilized for the testing procedure. The selected library for the implementation was the LIBSVM, proposed by Chang and Lin (2011). Linear, polynomial and Gaussian kernels have been tested, although the linear kernel proved capable enough to distinguish efficiently the different classes. The model regularization parameter C, which penalizes large errors, is chosen equal to 1000, in order to optimize data separation. The utilization of SVM in the top node of the hierarchy yielded greater classification accuracy for both methods, as shown in Table 2.

These are learning techniques for object recognition and categorization and, therefore, they should be further compared with classic methodologies capable to solve similar problems. Consequently, a single SVM classifier was used as a reference against the HTM learning framework applied to the same configuration of the dataset. The only modification was a dimensionality reduction of the dataset, utilizing principal component analysis (PCA). The resulting dimensions retained more than 95% of the initial information, simplifying the computational burden considerably. A linear kernel was selected for the training procedure to maximize the resulting classification rate, with the model regularization parameter C set to 100. Fig. 5 shows the average classification accuracy for the SVM-only, the original HTM plus the SVM and the optimized HTM plus the SVM method. A common issue in pattern recognition problems is that the classification accuracy is highly sensitive to the size of the training set. The experiments were performed utilizing different dataset sizes throughout the training procedure. Both the original HTM and the proposed method achieve high classification accuracy even with only the 55% of the dataset, proving thus the generalization ability of the specific topology.

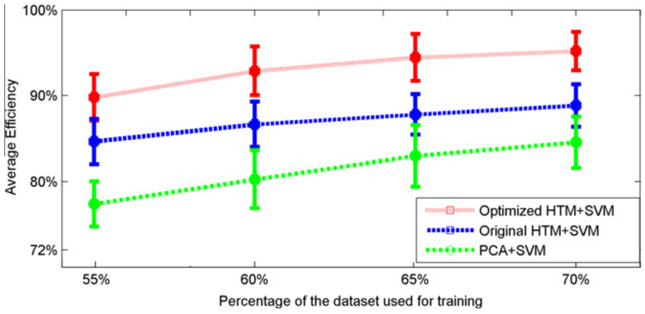

Fig. 5. Average efficiency for the three different classification methods with 1 σ tolerance versus the size of the training set.

5. Conclusions

In the present work an optimized HTM method for object recognition and classification has been presented. The optimization process involves changes in all the three different modules of the network. Firstly, the input images are mapped into the logpolar coordinate system, in similitude to the biological nature of the HTM network. Furthermore, in the spatial module of the network the quantization centers are stored more uniformly. The last and most crucial modification concerns the temporal module, where the time adjacency matrix has been replaced by the correlation matrix for the existing quantization centers in a node. The learning procedure in the temporal module was omitted leading to reduced computational burden and, consequently, to decreased time for the training of the entire network. The original approach has also been implemented in benchmarking the proposed method, while for the performance evaluation the ETH-80 dataset was utilized. Moreover, the parameters of the implemented networks were so selected in order to maximize the average efficiency among all the different classes of the dataset. In this procedure the *k*-nearest neighbor classifier ($k = 3$) was utilized and the proposed method succeeded with an 84.44% classification accuracy against the 76.21% accuracy of the original one. As a proof of HTMs efficiency, we compared them with a classic machine learning technique for object classification, viz. the PCA plus SVM. The original HTM architecture achieved a greater classification rate than the simple SVM, while the optimized HTM network resulted in the higher classification rate among all the evaluated classifiers. Consequently, the proposed method is considered an optimized HTM network capable of solving effectively the object recognition and categorization problems. Additionally, both the original and the optimized HTM networks succeed great generalization ability against the simple SVM classifier, proving the robustness of such networks, regardless of the number of the instances utilized for the training procedure. The optimized HTM is computationally effective as it stores only the most crucial quantization centers in the spatial learning and excludes the training routine in the temporal module.

References

Bundzel, M., Hashimoto, S., 2010. Object identification in dynamic images based on the memory-prediction theory of brain function. J. Intell. Learn. Syst. Appl. 2, 212–220.

Chang, C.C., Lin, C.J., 2011. LIBSVM: A library for support vector machines. ACM Trans. Intell. Syst. Technol. 2, 27:1–27:27, Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

CogVis, 2003. Website. http://www.mis.tu-darmstadt.de/Research/Projects/categorization/eth80-db.html.

Csapó, A., Baranyi, P., Tikk, D., 2007. Object categorization using vfa-generated nodemaps and hierarchical temporal memories. IEEE International Conference on Computational Cybernetics, ICCC 2007. IEEE, pp. 257–262.

George, D., Hawkins, J., 2009. Towards a mathematical theory of cortical micro-circuits. PLoS Comput. Biol. 5, e1000532.

George, D., Jaros, B. 2007. The htm learning algorithms. Whitepaper, Numenta Inc.

Please cite this article in press as: Kostavelis, I., Gasteratos, A. On the optimization of Hierarchical Temporal Memory. Pattern Recognition Lett. (2011), doi:10.1016/j.patrec.2011.11.017

Hawkins, J., Blakeslee, S., 2004. On Intelligence: How a New Understanding of the Brain will Lead to the Creation of Truly Intelligent Machines. Henry Holt & Company, New York,NY.

Hawkins, J., George, D., 2006. Hierarchical temporal memory: Concepts, theory and terminology. Whitepaper, Numenta Inc.

Hayes, B., 2000. Graph theory in practice: Part II. Amer. Sci. 88, 104–109.

Kapuscinski, T., 2010. Using hierarchical temporal memory for vision-based hand shape recognition under large variations in hands rotation. Artif. Intell. Soft Comput., 272–279.

Kjellström, H., Romero, J., Kragic, D. 2010. Visual object-action recognition: Inferring object affordances from human demonstration. Computer Vision and Image Understanding.

Kostavelis, I., Gasteratos, A., 2011. Laboratory of Robotics and Automation. http://utopia.duth.gr/~gkostave/downloads/operation_of_nodes_in_the_first_level.rar.

Manzotti, R., Gasteratos, A., Metta, G., Sandini, G., 2001. Disparity estimation on log-polar images and vergence control. Comput. Vision Image Underst. 83, 97–117.

Melis, W., Kameyama, M., 2009. A study of the different uses of colour channels for traffic sign recognition on hierarchical temporal memory, in: Fourth International Conference on Innovative Computing, Information and Control (ICICIC), 2009, pp. 111–114.

Metta, G., Gasteratos, A., Sandini, G., 2004. Learning to track colored objects with log-polar vision. Mechatronics 14, 989–1006.

Miller, J., 2006. Earliest known uses of some of the words of mathematics. Tomado de la página Web http://members.aol.com/jeff570/mathword.htmlel 15.

Mittal, A., Pagalthivarthi, K., 2007. Temporal bayesian network based contextual framework for structured information mining. Pattern Recognition Lett. 28, 1873–1884.

Numenta, 2006. Problems that fit HTM. Technical Report. Numenta.

Rozado, D., Rodriguez, F., Varona, P., 2010. Optimizing hierarchical temporal memory for multivariable time series. In: Artificial Neural Networks – ICANN 2010. Springer, pp. 506–518.

Sandini, G., Tagliasco, V., 1980. An anthropomorphic retina-like structure for scene analysis. Comput. Graph. Image Process. 14, 365–372.

Stenger, B., Thayananthan, A., Torr, P., Cipolla, R., 2004. Hand pose estimation using hierarchical detection. In: Computer Vision in Human–Computer Interaction. Springer, pp. 105–116.

Tyler, D., 2008. Robust statistics: Theory and methods. J. Amer. Stat. Assoc. 103, 888–889.

Please cite this article in press as: Kostavelis, I., Gasteratos, A. On the optimization of Hierarchical Temporal Memory. Pattern Recognition Lett. (2011), doi:10.1016/j.patrec.2011.11.017