10.2478/v10048-010-0008-4

MEASUREMENT SCIENCE REVIEW, Volume 10, No. 2, 2010

On the Optimum Architecture of the Biologically Inspired Hierarchical Temporal Memory Model Applied to the Hand-Written Digit Recognition

Invited paper

Svorad Štolc[1,2*] and Ivan Bajla[1]

[1]Austrian Institute of Technology, Safety & Security Department, Seibersdorf, Austria
[2]Institute of Measurement Science, Slovak Academy of Sciences, Bratislava, Slovakia

In the paper we describe basic functions of the Hierarchical Temporal Memory (HTM) network based on a novel biologically inspired model of the large-scale structure of the mammalian neocortex.The focus of this paper is in a systematic exploration of possibilities how to optimize important controlling parameters of the HTM model applied to the classification of hand-written digits from the USPS database. The statistical properties of this database are analyzed using the permutation test which employs a randomization distribution of the training and testing data. Based on a notion of the homogeneous usage of input image pixels, a methodology of the HTM parameter optimization is proposed. In order to study effects of two substantial parameters of the architecture: the *patch size* and the *overlap* in more details, we have restricted ourselves to the single-level HTM networks. A novel method for construction of the training sequences by ordering series of the static images is developed. A novel method for estimation of the parameter *maxDist* based on the box counting method is proposed. The parameter *sigma* of the inference Gaussian is optimized on the basis of the maximization of the belief distribution entropy. Both optimization algorithms can be equally applied to the multi-level HTM networks as well. The influences of the parameters *transitionMemory* and *requestedGroupCount* on the HTM network performance have been explored. Altogether, we have investigated 2736 different HTM network configurations. The obtained classification accuracy results have been benchmarked with the published results of several conventional classifiers.

Keywords: hierarchical temporal memory (HTM); optimum network architecture, visual pattern recognition; USPS hand-written digits.

1. INTRODUCTION

BIOLOGICAL SYSTEMS transform environmental information into efficient survival behaviors in a remarkably robust and reliable way. Neuroscience has shown that the most sophisticated elements of such an information transformation are performed in the neocortex of the mammalian brain. In a general context, it is accepted that different areas of the neocortex are interconnected to form hierarchical structures [1]. This has been verified by detailed studies of the visual cortex, where in the ventral pathway, information passing from the retina via the lateral geniculate nucleus enters the lowest level (V1) of the visual cortex. Afterward, it passes up in a sequence to the V2, V4 and inferotemporal cortex (IT) areas. As information moves up the hierarchy, each area detects increasingly more general and invariant features of the input visual scene [2].

One of the most promising direction of the neuroscience research is based on the idea that the neocortex represents sensory information probabilistically. Therefore, for modeling these processes, the Bayesian probability theory has been deployed. The powerful aspect of the Bayesian model is that it allows for feedback within the neocortical hierarchy which provides contextual information. Lee and Mumford [3] proposed a model of hierarchical Bayesian inference in the visual cortex that was further elaborated by Dean [4] to produce a pyramidal Bayesian network. This work showed that the task of performing Bayesian inference for *pattern recognition* (PR) in large brain-like structures is becoming tractable.

The *Hierarchical Temporal Memory* (HTM) is another biologically inspired computational theory that provides a large-scale model of the overall structure of the mammalian neocortex. It was first introduced by Hawkins and Blakeslee [5] and later on, by George and Hawkins [6], further improved and developed into a software implementation called *NuPIC* (Numenta Platform for Intelligent Computing, http://www.numenta.com). The HTM extends Lee and Mumford's work explicitly handling temporal sequences of input within a hierarchical Bayesian framework [6, 7, 8]. Hawkins and coauthors see the fundamental task of neocortical processing as *prediction* and so place the temporal aspect of the perception at the center of their model.

The HTM is a memory-prediction network model that takes

*Corresponding author: svorad.stolc@ait.ac.at

advantage of the Bayesian belief propagation and revision techniques. The roles of biological cortical regions and subregions are mimicked by the network nodes which are computationally identical and represent basic building blocks of the HTM architecture. It is the hierarchical structure and temporal relations contained in a training data that serve the HTM nodes for extracting invariant representations of the outer world's causes or, in other words, the categories of input patterns. As a state-of-the-art approach, the HTM has principal potential for resolving difficult PR problems that have not been handled so far.

There are several open issues of the research into the HTM, in particular, only a little attention has been devoted to the construction of the optimum HTM architectures for specific practical applications. For example, in the field of PR problems addressed up to now using the NuPIC tool, for searching optimum values of several model parameters, which control the learning and inference processes in the HTM nodes, the task has been completely delegated to the user's trial-and-error computer experimentation.

No systematic research of the relation of the random nature of the input visual data to the vector quantization process (the parameter *maxDist*) and to the Gaussian inference (the parameter *sigma*) has been carried out. To our knowledge, no analysis of the influence of various node overlap schemes in relation to the utilization of the information originating from individual image pixels has been published.

In this paper, we concentrate ourselves exactly on the research into these aspects of the construction of the optimum single-level HTM network dedicated to the specific PR problem – the recognition of the hand-written digits contained in the internationally accepted USPS database [9]. This choice is based on the fact that the recognition accuracy achieved by various classifiers applied to this database is well documented and benchmarked in the literature [9, 10]. The restriction to the single-level architectures in this study is motivated by the effort to explore in more details effects of substantial controlling parameters of the HTM network on its performance.

Related to the PR problem of USPS, it should be noted that several attempts to apply the HTM network to this task have already been published. In [11, 12], an own implementation of the HTM model was applied to the binary version of the USPS digits. The authors selected a fixed two-level architecture and manually experimented with the learning parameter *maxDist* and inference parameter *sigma.* They did not explore the suitability of the architecture, neither the parameter optimization process. The best recognition accuracy achieved was 96.32%.

In the report [13], the first NuPIC "Pictures Demo" program was adopted to the USPS recognition problem. Again, it was a two-level architecture applied to the binary versions of the USPS digits. The author reported the best recognition accuracy achieved of 96.26%.

The most recent activities related to various applications of the HTM to PR of the USPS digits have been reflected in the Numenta discussion forum [14]. The approach of Gregko involved a novel element of the HTM network, namely, "eye movements". Such an extension of the HTM model corresponds, in machine learning, with the method of the so-called "ensemble learning". The approach already worked with the gray-level images of the USPS digits and the best achieved recognition accuracy was again 96.26%. However, the author did not report on the exact network configuration and he did not cope in his work with the systematic optimization of the controlling parameters of the HTM.

2. Hierarchical Temporal Memory

In [6, 15, 16], the basic HTM theory and terminology, as well as implementation are described. Since our interest has been focused on the research into the optimal design of the HTM network intended for solving image object recognition tasks, in the following we overview briefly the essential concepts of the HTM model that we see as most relevant to the visual data processing.

As already mentioned, the HTM network is organized in the layers of elementary units – *nodes*. All the nodes implement the same learning and inference algorithms, and therefore they operate in computationally identical regimes. They can only differ in the content of their internal memory, i.e., the information gained during the learning phase. The individual layers (or levels) of nodes are usually structured into a tree-like hierarchy. There is always a zero sensory level that serves as an input to the first level of nodes. In our case, the zero level of the network represents a visual field containing raw image pixels. In the NuPIC platform this function is implemented by the *ImageSensor* object. On the other hand, at the very top of the HTM hierarchy, a single node called *Zeta1TopNode* is typically situated. It realizes task of a simple classifier that associates learned top-level belief patterns with known output categories [8]. Of course, one could potentially consider any supervised classifier to be used instead of the Zeta1TopNode.

Since the use of the temporal dependencies of input spatial patterns is essential characteristic of the HTM, it learns either from natural temporal sequences of images (i.e., movies) or artificially generated movies obtained by applying a smooth set of transformations (e.g., translation, rotation, or zooming) to the given training images. In the NuPIC platform, the special ImageSensor plug-ins called *explorers* are responsible for generating the artificial temporal sequences out of the static input images.

To describe individual functions of the single HTM node, a simple example of the visual pattern recognition will be introduced. The goal of the HTM network in this example is to learn invariant representations of a set of binary image categories comprising primitive line drawing objects. Fig. 1 shows a way of interfacing the HTM with the given visual world. Each frame (image) of the training sequences is presented to the sensory field ("retina") of size 32×32 pixels. The nodes at the level 1 are arranged in an 8×8 grid where each node receives input from a 4×4 pixel patch of the retina. Such an arrangement of the level 1 nodes covers the whole

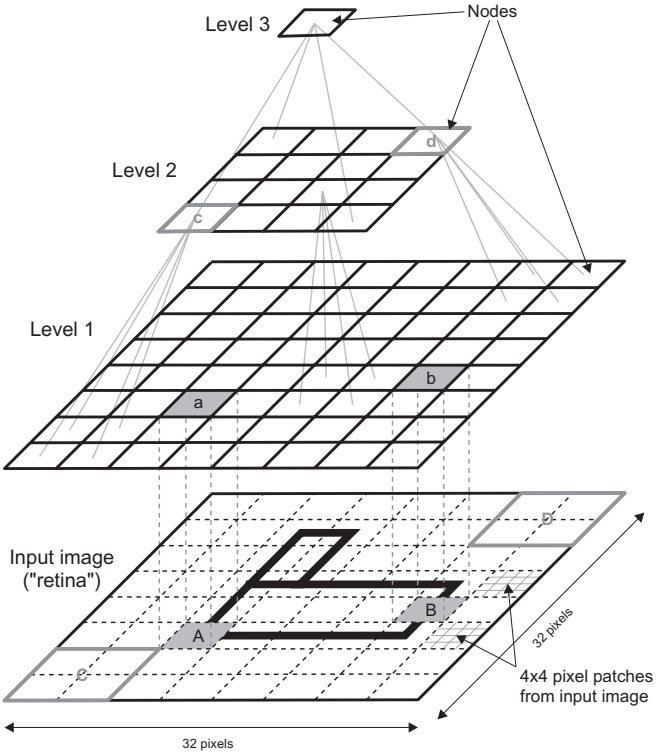


Fig. 1: The structure of the HTM network for learning invariant representations of the binary line drawings. The picture is derived from the one contained in [16].

retina with no overlap between neighboring patches. The effective input area – *patch* – from which any node receives its input is called *field of view* or *receptive field*. At the level 2, each node receives its input from 2×2 patch of the child nodes at the level 1 again with no overlap. The single node at the top-most level 3 is connected to all 4×4 nodes at the level 2 and so it covers the entire network's input by pooling outputs from all its child nodes.

The depicted HTM network is trained by sequences of images that show each object in various smoothly changing positions within the sensory field of view. At any time, only a single frame of the training movie is presented to the network. It is convenient to consider that each HTM node operates in two distinct stages – *learning* and *inference*. During the learning stage, all network layers are being learned in consecutive order from the lower to the higher ones. The network is repeatedly exposed to the training movie until all the nodes at all the levels form their own representations of the input space. Once all the nodes at all the levels of the hierarchy are learned, the whole network can be switched to the inference mode. The task of the HTM network during the inference is to recognize the category of a potentially unseen input pattern. This information can be read at the top of the network hierarchy.

In the following, we describe briefly the algorithms used for learning and inference within the HTM nodes.

2.1. Learning in a node

During the learning stage, each node of the network performs the following basic operations:

1. memorization of input patterns,
2. learning transition probabilities,
3. temporal grouping.

Memorization of input patterns

In the first step of the learning process, the node memorizes the representative patterns that were seen in its receptive field. In general, the memorization of spatial patterns which are represented by fragments of individual movie frames can be considered as a *vector quantization* process of the input data. In the HTM terminology, this process is called the *spatial pooling*. The detected *quantization points* (or *coincidences*) represent the centroids of the pools containing one or more input vector patterns [17]. When the Euclidean distance between the input pattern and the already existing quantization points exceeds the value of the parameter *maxDist,* a new quantization point is generated. After processing all input patterns, the memorization process is finished and all detected quantization points are stored in the node's internal memory.

The mentioned memorization procedure is usually applied only to the lowest level of the HTM network. In the higher levels, a sort of sparsification of the memorized patterns is usually considered, mainly due to significantly reduced memory demands and presumably better scaling to larger problems. The basic idea of the sparsification is that the memorized coincidence vectors are not stored in their dense form, but rather in a sparse format (i.e., majority of the elements are zeros) which preserves only single maximum belief component per each child node. All other beliefs in each memorized vector are zeroed so they do not occupy any memory. There is a significant difference in the calculation of the distance between dense (non-sparse) and sparse vectors. Authors of the NuPIC have considered measuring the vector distance based on non-sparse elements only, whereas all sparse elements are simply excluded from the calculation. From such a point of view, each memorized coincidence behaves exactly as a vector of the length corresponding to the number of the child nodes. For more details on the sparsification used in the NuPIC, see Numenta discussion forum.

Learning transition probabilities

The ultimate goal of the HTM learning is to generate correct invariant representations of the input world's causes based on the temporal relations contained in the training sequence. To achieve this goal, authors of the HTM model proposed to evaluate a frequency of transition events, i.e., cooccurrences of the memorized coincidences in adjacent time instances [16]. The temporal relations are then described in a form of the first-order Markov graph where vertices represent the memorized

coincidences and links stand for the transitions between coincidences in time. Such a graph structure can be expressed as a square matrix called *Time Adjacency Matrix* (TAM), where the rows and columns represent coincidences in adjacent time instances and the particular matrix elements contain the frequencies of transitions between them. In each HTM node, an individual TAM characterizing local temporal transitions is constructed and maintained.

As for the TAM construction during the learning phase, a sequence of the input vector patterns generates a sequence of the closest coincidences within each node. When two coincidences appear nearby in time in this sequence, the relevant TAM element is increased until the whole sequence is presented to the network. In the basic configuration, only transitions between immediately preceding coincidences are being increased by the constant value 1. However, sometimes it may be reasonable not to focus only on immediate temporal neighbors, but also to increase transitions between current and few older coincidences. The number of affected coincidences in the past is defined by the parameter $transitionMemory$. In this case, the increment is not constant but rather ruled by the formula: $I_t = transitionMemory - t + 1$, where t is the time gap between the current and past coincidence. When high values of $transitionMemory$ are being considered, the temporal transitions are smoothed out so that the temporal jitter and repeated states in the input sequence do not produce undesired behavior [17]. Furthermore, it also leads to a higher occupation of the TAM that may result in more stable statistics when the training sequence is short.

Temporal grouping

Once learning of the TAM is finished and numbers of occurrences of each particular transition are recorded, one can construct the so-called normalized Markov graph in which the links represent relative frequencies (i.e., probabilities) of the transitions between individual quantization points.

The last step of the learning process in each HTM node is to analyze the normalized Markov graph with the aim of its partitioning into a set of *temporal groups*. The goal of this partitioning is to group together coincidences (i.e., vertices of the Markov graph) which highly likely follow one another and so they are likely to share a common cause. To form these groups, the HTM nodes use the well-established *Agglomerative Hierarchical Clustering* (AHC) method [18]. The AHC algorithm takes a set of patterns and their pair-wise similarities as an input, and produces a tree-like hierarchy of clusters (dendrogram) such that patterns in the same cluster are as similar as possible. The probability of the transition between any two coincidences is used as the similarity measure for the AHC algorithm. Clustering based on a such measure puts patterns that are likely to follow one another into the same branch of the AHC dendrogram. The final coincidence clusters are obtained by cropping the dendrogram at a certain level according to the requested maximum number of the temporal clusters specified separately for each network level by the parameter $requestedGroupCount$.

In general, the optimal value of this parameter has highly data- and problem-dependent nature, therefore it is considered as one of the most crucial parameters of the HTM model since it may significantly influence the overall performance of the network. On one hand, if the $requestedGroupCount$ takes too small values, the nodes have tendency to overgeneralize the learned information and consequently their performance goes down. On the other hand, if too high values of $requestedGroupCount$ are used, the nodes do not realize a sufficient temporal grouping which leads to an approximation of the simple vector quantization approach.

2.2. Inference in a node

A node that has completed its learning phase can be switched to the inference mode. The characteristics of the input to the node during inference are identical to those during the learning. The moving objects are exposed to the network's retina from where the different image fragments are passing to the particular nodes according to their field of view. A node being in the inference mode produces an output vector for every input pattern being seen. This vector indicates the degree of membership of the input pattern in each of its temporal groups. There are two phases of the inference process – the spatial and the temporal inference.

Given the actual input pattern in a vector form, a closeness to every memorized pattern must be calculated in the first step. Typically, most of the input patterns do not perfectly match any of the memorized coincidences. Let d_i be the Euclidean distance of the i-th coincidence from the input pattern. The larger is the distance, the smaller should be the match between compared vectors. It can be assumed that the match of the patterns can be expressed as a Gaussian function of their Euclidean distances with zero mean:

$$y_i = e^{-\left(d_i/sigma\right)^2}, \tag{1}$$

where *sigma* is a parameter of the node. Calculating this quantity for all coincidences, one produces the overall belief vector $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ which is the result of the first phase of the inference in each node.

In the second phase, the calculated belief vector $\mathbf{y}$ is employed to produce the beliefs that express membership of the actual input pattern to each of the memorized temporal groups. In the NuPIC, this can be performed by two different methods, i.e., *sumProp* and *maxProp*, however, in our experiments we have considered only the latter one. The *maxProp* approach basically associates each temporal group with the maximum belief among all coincidences connected to that particular temporal group. The resulting belief vector, where each element stands for a single temporal group, represents the node's output given the actual input pattern.

The training of the HTM network is performed gradually layer-by-layer from the bottom up to the top. Once the learning process is finished at some level, all HTM nodes at this level are switched from the learning to the inference mode,

thus the learning can continue at the next level. When learning is started at some level, the HTM network must process all training sequences again and each frame must be passed through all the previously trained layers until the information (i.e., generated belief vectors) reaches the nodes that are currently learned. The whole training process is finished only when the top-level of the network is successfully trained.

3. Training sequence generation

3.1. Static image sweeping

The ImageSensor is a NuPIC sensor node which reads data from image files and passes them to the nodes at the first level of the network [19]. In the NuPIC 1.7.1, the ImageSensor can load binary or gray-level images. It can also load color images, but they are treated the same way as the gray-level images. Besides the technical matters, a key capability of the ImageSensor is the ability to generate smoothly-varying patterns forming "virtual" temporal sequences out of the static input images. According to these sequences, the nodes should be able to group together patterns that occur nearby in time, which means, the patterns likely containing similar geometrical structures.

It is the explorer plug-ins that generate the temporal sequences within the ImageSensor object. They are responsible for "exploring" the input space of possible images accomplishing the following two main goals:

- efficiently select patterns that need to be presented to the network, out of a large number of potential input patterns. In a large training set, there can be far more images than needed for successful training of invariant representations;
- generate smooth temporal sequences for learning the temporal transitions. Note that only some explorers generate smooth temporal sequences that are suitable for training the HTM networks.

In the NuPIC, a conventional approach to the construction of the training sequence out of static images, which focuses only on the latter goal, is called *ExhaustiveSweep* explorer. This explorer performs an exhaustive raster scan through all possible translations of the given static image within the network's retina. An original image is gradually shifted line-by-line horizontally and vertically with constant one pixel step. The sequences generated this way are rather long and therefore imply highly memory and time demanding training process and also large trained HTM networks. On the other hand, the networks trained on such sequences are able to learn correct invariances even on very small number of training examples. The ExhaustiveSweep explorer builds mostly positional invariances, however, it does not explicitly strive for any rotational or scale invariance. Nonetheless, if the network is trained using this explorer, it still exhibits some robustness against these transformations too. For more detailed analysis of learning invariances in the HTM model, see [16].

When classification methods are benchmarked on strictly defined training and testing sets of examples, one could raise an objection against the ExhaustiveSweep explorer that it basically blows up the training set while every training pattern is presented to the network several times in various translated positions. In can be expected that if other classification methods got the same expanded training set (i.e., the basic patterns accompanied by all their translations), they would also exhibit improved performance. Therefore, in the context of correct benchmarking methodology, we were obliged to introduce an alternative method for construction of the training temporal sequence consisting of only original training images.

3.2. Training set ordering

The structure of the digit patterns in the USPS database, i.e., quite rich classes of the individual digits, enables to consider an alternative approach to the generation of the virtual temporal sequences. This approach can save memory and computational time significantly without negative influence on the performance of the HTM model. The method consists in ordering the training images, separately in each of the digit classes, that exploits their mutual spatial similarities. The natural attribute of the appropriate ordering is that similar patterns (with respect to some distance measure, e.g., L_2 metric) appear nearby in the sequence while dissimilar ones should be as distant as possible. Given rich enough set of the static images and satisfying this condition, we can produce a reasonably smooth image sequence. Such a generated sequence can be considered as a virtual temporal sequence, even if there are no inherent temporal relations between any of the used images.

In our experiments, we considered the ordering based on the traditional L_2 metric. Note that this metric completely ignore 2-dimensional nature of the visual information and therefore it is not the best choice for this task. Finding the optimal ordering of a finite set of the static images that minimizes the total sum of distances between all adjacent images can be understood as the well-known *Travelling Salesman Problem* (TSP). The problematic point about TSP is that, since it belongs to the NP-complete complexity class, there exists no deterministic algorithm for solving this problem in less than exponential time (i.e., algorithm equivalent to the checking of all possible permutations). However, there is a great deal of heuristic algorithms which are capable of finding satisfactory semi-optimal solutions in a reasonable time.

We have considered the following two possibilities of how to accomplish semi-optimal ordering:

- greedy approach,
- genetic algorithm approach.

Greedy approach

The greedy approach is typical for its strictly local decisions that minimize cost of each particular step, assuming that the sum of all such decisions will lead to the small total cost. Of

course, this concept is not generally true, however, in our experiments, it proved to be quite efficient.

For each pattern class, the algorithm begins with finding the nearest pattern to the class mean and, in every consecutive step, the process continues by finding the closest pattern (in L_2 sense) to the previous one. The result of such an algorithm should be a piecewise smooth sequence of the digits with occasional "jumps", mostly at transitions between the digit classes.

Results of the greedy method applied to the USPS database of the hand-written digits are described in details in Section 5.1.

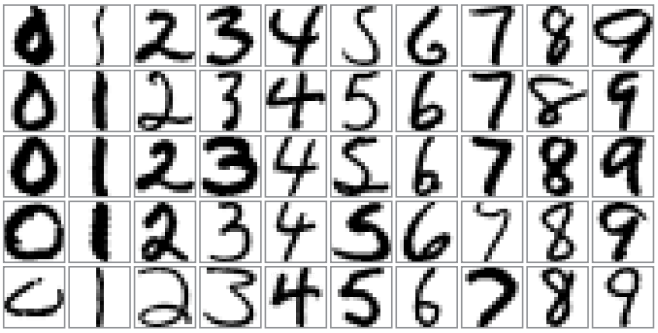


Fig. 2: Examples of the hand-written digits of 10 classes extracted from the USPS database.

Genetic algorithm approach

The *Genetic Algorithms* (GA) are a stochastic search technique that guides a population of solutions using the principles of evolution and natural genetics [20]. Such approaches are often used for solving difficult optimization problems that do not have known analytical solutions, such as TSP. In general, given large enough number and size of the populations, the GA have potential to converge quite close to the global solution, however, in many practical situations, achieving the convergence may be too slow.

Preliminary experiments with the GA used for finding the optimal ordering of the USPS hand-written digits showed that the semi-optimal solutions found by GA are hardly as good as the solution found by the greedy algorithm. When the greedy solution was used for initialization of GA, just a slightly better solution could be achieved only after a vast number of iterations. Due to these two reasons, we did not consider the GA approach in our further analysis.

4. USPS DATABASE OF HAND-WRITTEN DIGITS

The problems of optimum architecture design of the novel HTM network are connected to particular application problems. For the research in the area of applications of the HTM model to the visual pattern recognition we chose a task of the *hand-written digit* recognition that could serve as a suitable starting point for more complex PR problems. The important advantage of selecting such a PR task was that a great number of benchmarks have been performed for different classification methods applied to this problem. For the purpose of testing the performance of HTM applied to this problem, we have decided on the standard USPS (U.S. Post Service) database of hand-written digits collected by CEDAR, Buffalo [21]. Existence of this image database, as well as other databases in different domains, confirms the overall popularity of benchmark testing among the communities of machine learning and optical character recognition. Unfortunately, the experience with this database, as reported in some publications [22, 23], as well as our own experience showed that the generation of this frequently referred database suffers from several drawbacks (e.g., errors in digit class labeling, inclusion of confusing samples into the training and testing sets). Therefore we decided to analyze the data included in this database in a more details.

4.1. Basic description of the USPS database

We start with the basic description of the USPS database, as it has been stated in the literature. The hand-written ZIP-codes containing digits from "0" to "9" were scanned from envelopes at the resolution of 300 pixels per inch [21, 24]. The acquired gray-level images were afterward converted to binary (bi-tonal) format by application of a thresholding algorithm. Later on, LeCun's research group [25] performed conversion from the binary format back to the 256 gray-level format achieved by resampling all digits into the normalized dimensions of 16×16 pixels and applying a linear transformation in order to center the patterns within the given bounding box. Intensity values of the resulting gray-level images were normalized (i.e., scaled and translated) to fall within the range $\langle -1, 1 \rangle$. Few examples of the resulting digit images are shown in Fig. 2. The final database contains 9298 digits (each of 16×16 pixels) which are divided into two non-overlapping groups: 7291 (78.4%) digits for training and 2007 (21.6%) digits for testing. Exact digit quantities in each particular digit class are shown in the left part of Tab. 1. The USPS database can be found, e.g., at the following website: http://www.cenparmi.concordia.ca/∼jdong/HeroSvm/data.zip.

4.2. Labeling errors and severely distorted digits

The first question on the quality of the USPS database was whether all the digits in the training and testing sets are properly labeled and whether these sets contain only symbols uniquely recognizable by humans. To answer this question we have scrupulously analyzed both sets and discovered several discrepancies. In the training set the digits with the following indices have been found as symbols which cannot be recognized as written digits or having undoubtedly incorrect labels: #431 (labeled as the digit "2"), #2466 (as "5"), #5146 (as "8"), #5297 (as "4"), #6761 (as "5"). In the testing set, the following either severely distorted or falsely labeled digits have been found: #528, #995, and #1007 (labeled as the digit "0"), #234 (as "1"), #915 (as "2"), #1358, and #1432

Table 1: Some quantitative characteristics of the USPS training and testing sets. Left: absolute quantities of particular digits in each data set are shown. Right: results of the permutation test for difference between the training and testing sets are summarized. The gray rows stand for the classes where the significant difference (p-value ≤ 0.05) has been obtained. The confidence intervals of the p-values are derived from the fitted binomial distribution given 10000 simulations (trials).

USPS database digit quantities			
Class	**Train set**	**Test set**	**Total**
"0"	1194	359	1553
"1"	1005	264	1269
"2"	731	198	929
"3"	658	166	824
"4"	652	200	852
"5"	556	160	716
"6"	664	170	834
"7"	645	147	792
"8"	542	166	708
"9"	644	177	821
Total	7291	2007	9298

Test set vs train set permutation diff. test		
Class	***P*-value**	**Conf. int.**
"0"	0.026	0.023 - 0.030
"1"	0.000	0.000 - 0.001
"2"	0.076	0.071 - 0.081
"3"	0.047	0.043 - 0.051
"4"	0.047	0.043 - 0.051
"5"	0.000	0.000 - 0.001
"6"	0.157	0.150 - 0.164
"7"	0.012	0.010 - 0.014
"8"	0.008	0.007 - 0.010
"9"	0.510	0.500 - 0.520

(as "3"), #266, and #971 (as "4"), #562, #994, and #1978 (as "5"), #1529 (as "7"), #199 as "8"). The incorrect digits contained in the training set could be removed from this set without violating the equal conditions for benchmark evaluation of the recognition results. However, to ensure a fair comparison of the HTM performance to the performance of other benchmarked classifiers, the removal of any, even falsely labeled, symbols from the testing set is not permissible.

4.3. Distributions of the digit classes, differences between the training and testing sets, noise

Another interesting question concerns a mutual separation of the individual digit classes in the vector space (of dimension $16 \times 16 = 256$) generated by the intensity values of the image pixels. The visualization of such a separation can be demonstrated by means of the PCA method. In the main plot in Fig. 3, one can see the distribution of all the digit classes (including both training and testing data sets) projected onto the first two largest PCA components. The mean values of the digit classes are clearly separated which means that the classifiers applied to the USPS database may benefit from the so-called pixel overlap information (the class can be assessed from its average sample). On the other hand, some of the digit populations are highly overlapped, and so the classification in particular cases may be very difficult. When one deals with such overlapped data, it is very important that the training set describes a manifold associated with a specific digit class in sufficient details. If the training set were less informative and descriptive than the testing set (i.e., the testing set would contain principally new patterns), a low classification accuracy should be expected in that particular digit class.

Our next question has been focused on verification of the hypothesis that the USPS training and testing sets were randomly chosen from a common distribution of digits, despite the fact that this statement was explicitly claimed in the literature [24]. We were particularly interested in discovering the case where not all objects in the testing set have corresponding representatives (i.e., digits of similar spatial characteristics) in the training set. We have decided to verify this assumption by the permutation test method described in [26]. The permutation test of the null hypothesis (H_0) that the distributions that generated two populations A and B are actually the same distributions is based on the so-called randomization distribution constructed by calculating some characteristic function (test statistic) for all possible random divisions of $A \cup B$ into the sets A' and B' of the the same cardinality as the original sets A and B. The p-value and its confidence interval can be then estimated by fitting the binomial distribution into the sequence of successes and failures derived from the simulated randomization distribution compared to the test statistic of the sets A and B. The more samples from the randomization distribution lie below the test statistic of A and B, the higher p-value is obtained, and vice versa. Of course, if the sets A and B contain more than just few samples, it is not computationally feasible to evaluate all possible combinations of the sets A' and B'. Therefore a simulation approach is usually considered for estimating the randomization distribution.

In the case of testing equality of distributions of the USPS training and testing sets, we have proposed a test statistic which is focused on the presence of outliers within the testing set with respect to the training set. The test statistic evaluates the following ratio:

$$T(A,B) = \frac{V_{BB}(A \cap B)}{V_{BB}(A)}, \tag{2}$$

where A and B are the evaluated data sets and $V_{BB}(.)$ estimates the volume of a tightest box covering the data (bounding boxes). The rationale behind this statistic is as follows: if the testing set contains any samples that are not within the range of the training set, the test statistic will likely receive a value close to 0. On the other hand, when the whole testing set lies within the range of the training set, the test statistics value approaches 1. To assure that the bounding boxes follow basic orientation of the data and have nonzero volumes, prior to the test, the data have been projected onto the largest PCA components that altogether cover 99% of the original data variance. By this operation, all insignificant PCA components (i.e., the components with very small or zero eigenvalues) were eliminated. The permutation test was carried out for each digit class separately performing 10000 random simulations. Only after this number of simulations a sufficient convergence of the p-value has been achieved.

The results of the permutation test are shown in the right part of Tab. 1. One can see that the test proved significant differences (p-value ≤ 0.05) between the training and testing sets in 7 from 10 digit classes ("0", "1", "3", "4", "5", "7", "8"). Only 3 digit classes ("2", "6", "9") seem to be generated by the genuine random process. In all other cases the generation process was either not random or the random selection was very improbable. Therefore one should expect

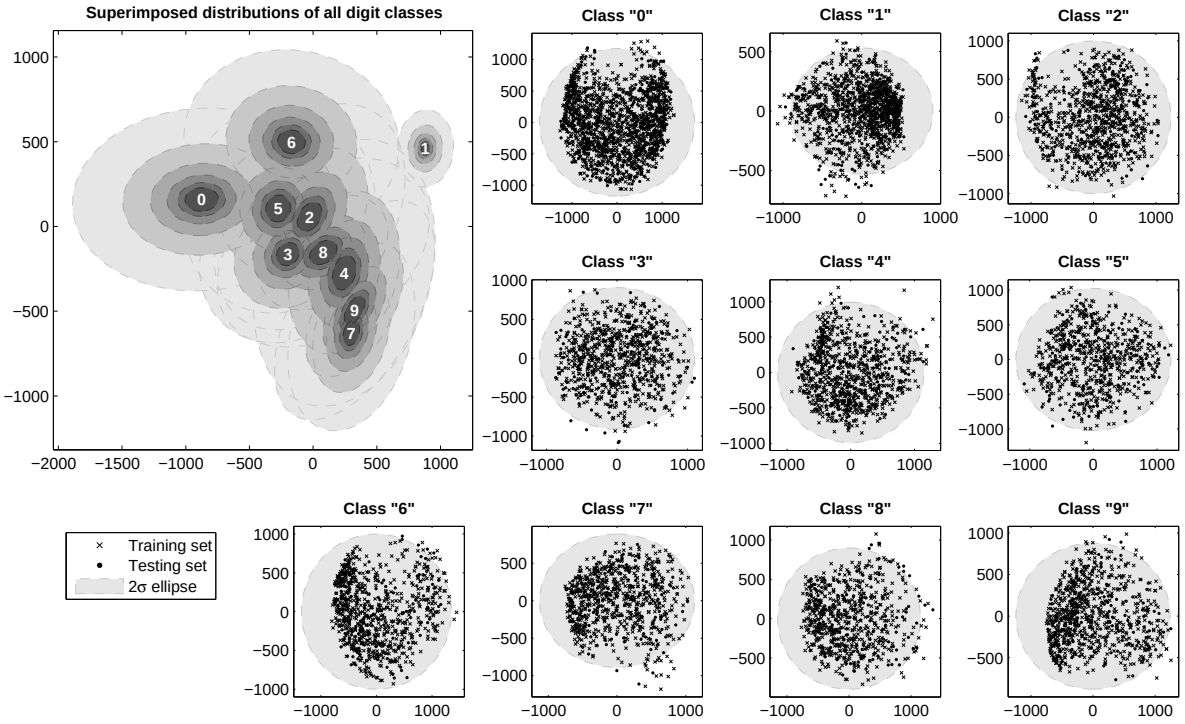


Fig. 3: The visualization of the PCA results for the USPS data using the first two largest components. The main plot (left) shows the distributions of all digit classes plotted as the gray ellipses (2σ, 1σ, $1/2\sigma$, $1/4\sigma$, $1/8\sigma$, and $1/16\sigma$) centered in the class means. In other 10 plots (right and bottom), the PCA results for separate digit classes are displayed.

the most classification errors exactly in these classes due to the expected insufficiency of the training set.

As will be clarified later, the nature of noise in the USPS data, has considerable influence on the design of the optimization procedure of the basic HTM network parameter *maxDist* that controls the generation of quantization centers in each node of the first network level. Given the information on the USPS digit acquisition and post-processing procedures, we can assume that the quantization noise occurring in this process may manifest itself only on the digit boundaries and introduces only insignificant changes into the final recognition operation. We think that the only stochastic component, which may be considered as noise, is generated by the interindividual differences in writing the digits among the people whose envelopes have been processed by the U.S. Post Service. Since in our task this influence is significant, it should be reflected in the search of an optimum value of the *maxDist* parameter.

5. Application of HTM to the USPS classification problem

5.1. USPS training sequence

As described in Section 2, the HTM networks must be trained on the temporal sequences. When a native temporal sequence is not available for certain problem domain, one must generate a virtual temporal sequence out of the available data. Two alternative methods for generating virtual temporal sequences are discussed in Section 3.

In this study we have decided to use the greedy approach for generating the training sequence out of the static images. The obtained sequence consists of 7286 frames (images of digits), since 5 digits were excluded due to the presence of labeling errors or severe distortions. Within the sequence, the digit classes are ordered lexicographically from "0" to "9". The evolution of certain characteristics, such as the L_2 distance between any two consecutive images, the distance from the class or global means, are shown in Fig. 4. It can be seen that within each digit class there is a general upward tendency of the distances between any two neighboring images while the biggest difference is always reached when switching from one class to another. A similar behavior can be observed in the evolution of the distances from the individual class means. However, quite a different picture is generated by the distances from the global mean where no upward trend but the clear oscillations are visible in certain intervals of the sequence. These clues altogether suggest that for each digit class the sequence is likely traversing the data manifold along gradually expanding hyper-spheres with the identical center in the class mean.

5.2. Suitable HTM network architectures

One of the most important attributes of the HTM model addressed in this paper is the network architecture and its suit-

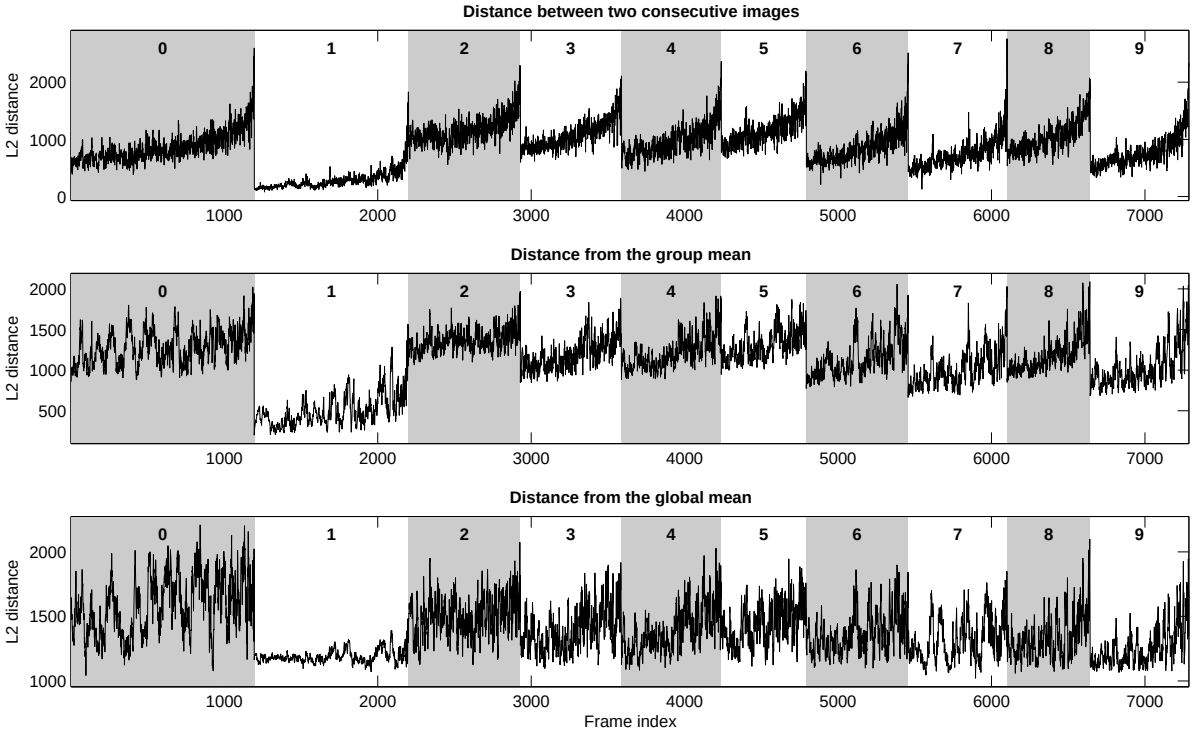


Fig. 4: Some L_2 metric characteristics of the training sequence generated by the greedy ordering method. Top: the plot of the distances between two consecutive digit images as they appear in the training sequence. Middle and bottom: the plots of the distances between individual digit images and the class mean or the global mean, respectively.

ability for the USPS classification problem. As already mentioned, we have restricted ourselves to the single-level HTM networks in order to study in more details effects of two substantial parameters of the architecture: the *patch size* and the *overlap*. The patch size defines the extent of an effective node's input area, while the overlap parameter defines how big is the overlap between patches of two neighboring nodes at a certain level. One can understand the overlap also as a parameter that controls the spatial resolution of each HTM layer. These concepts are graphically demonstrated in Fig. 5. It can be seen that, in case of the single-layer HTM networks, there are only four dependent parameters which define the whole network architecture: *retina size* (retina), *patch size* (patch), *overlap*, and *grid size* (grid). Hereinafter, we introduce the following unified code expressing any single-level HTM architecture: *retina_patch_overlap_grid* (e.g., "28_8_6_11" stands for the network with the retina of 28×28 pixels, the patch of 8×8 pixels, the overlap of $6,6$ pixels, and finally the grid size of 11×11 nodes).

Assuming that the crucial information contributing to a successful classification may appear in any position of an input image (area of 16×16 pixels), the prerequisite of a successful architecture is that it utilizes information from all image pixels homogeneously. In all single-level HTM architectures studied in this paper, Zeta1TopNode exploits homogeneously all the nodes at the first HTM level, i.e., the nodes at the first level have exactly the same influence on the total network's

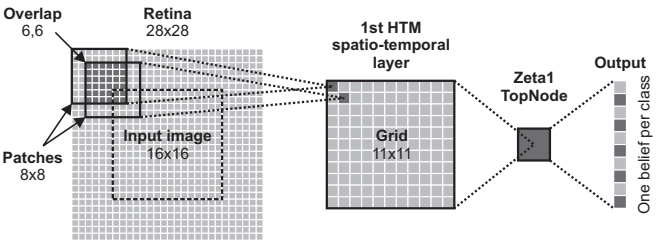


Fig. 5: An example of the single-level HTM network architecture with the retina of 28×28 pixels, the patch of 8×8 pixels, the overlap of $6,6$ pixels, and finally the grid size of 11×11 nodes (i.e., 28_8_6_11). The relations between these four dependent parameters are graphically demonstrated.

output. However, when the nonzero overlaps are being considered, the output of the first network level as a whole may prefer some of the retinal pixels. According to the various overlap schemes one can differentiate between four different modes of the pixel information usage which are illustrated in Fig. 6. From all these cases, only two overlap schemes produce a compact homogeneous area:

- no overlap produces the homogeneous usage of all retinal pixels,
- some large overlaps generate the inhomogeneous border effect, which however preserves a homogeneous zone in the center of the retina.

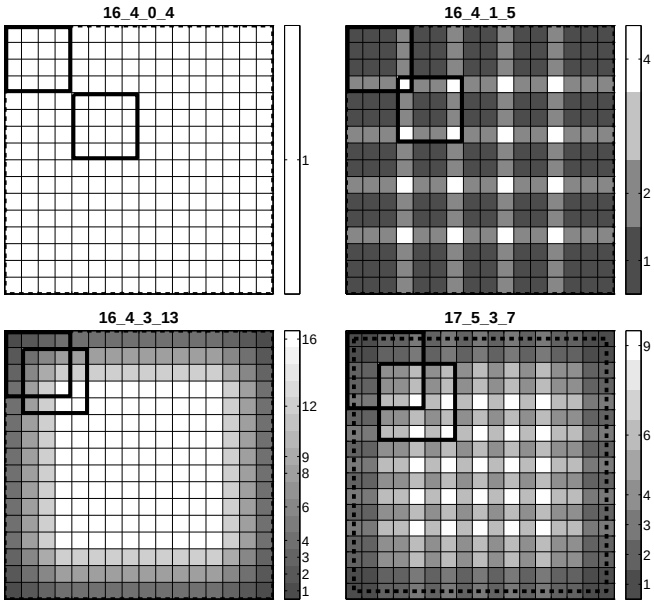


Fig. 6: The examples of a homogeneous and inhomogeneous usage of the retinal 16×16 pixels. Top-left: the 4×4 patch with zero overlap generates fully homogeneous retina usage. Top-right: the 4×4 patch with the one pixel overlap produces an inhomogeneous interlaced usage. Bottom-left: the 4×4 patch with the overlap of $3,3$ pixels results in inhomogeneous usage with a border effect, however, with a homogeneous zone in the center of the retina. Bottom-right: the case with the combined inhomogeneity (patch 5×5 and overlap $3,3$) is displayed.

We assume that only these two modes may become the basis for a successful HTM network architecture. The first non-overlapped case of the network architecture can be applied directly. In the second case with the border effect, for achieving the homogeneous coverage of at least the area of an input image (16×16 pixels), we need to pad the retina by zero-valued dummy pixels. For the given application it is reasonable to assume maximum patch size of 8×8 pixels (a quarter of an input image). To obtain at least some reduction in the network hierarchy, the maximum size of the node's grid is limited to 16×16 nodes. These two assumptions yield only 19 architectures that cover homogeneously the central 16×16 pixel area of the retina (see Fig. 7 and Tab. 2). Only these architectures will further be explored.

5.3. Estimation of the optimal values of the parameter *maxDist*

As described in Section 2.1, the parameter *maxDist* serves for specification of the maximum distance between the closest coincidences created by the vector quantization process during the node's learning. It is known fact that the optimum vector quantization, i.e., classification of M training vectors into N clusters under two optimality conditions ((i) the nearest neighbor condition, and (ii) the centroid condition), is the NP-hard problem. It means that there is no algorithm leading to the globally optimal solution in the polynomial time. However, the HTM vector quantization procedure, as proposed by Numenta, produces a set of coincidences (a codebook) in the polynomial time. Depending on the choice of *maxDist*, various kinds of codebooks can be obtained – from small to large, from underspecified to overspecified ones. The detected coincidences are distributed regularly over the whole extent of the data, no matter how are they distributed in the particular regions. In other words, the HTM spatial pooling procedure ignores the density of the input data, while the only exploited property is the data extent. It is clear that such a method of the codebook generation typically produces highly non-optimal solutions. In the following, we propose a method for estimating the optimum *maxDist* values that produce the most optimal codebook given the HTM vector quantization procedure.

Preserving a sufficient quantization resolution in the crucial dense regions of the data is the essential part of successful training. Too few quantization points lead to the low-specific temporal grouping that may seriously affect the total recognition accuracy. On the other hand, too many quantization points usually result, besides the large memory demands, in too specific temporal grouping that also leads to poor recognition performance. To avoid these undesirable effects, it is very important to assess the optimum *maxDist* value for the given training data. Instead of laborious experimentation, we propose to base the optimization procedure on characterization of the data structure via the method inspired by principles used in fractal theory.

First, the USPS database admits deviations between individual participating writers that is reflected in random differences among digitized hand-written numerals within each class (see Section 4.3). Then, a natural requirement to the quantization process is to find such a value of the parameter *maxDist* that ensures the generation of the coincidences representing the macroscopic structure of the data in maximum details, while maximally ignoring the low-amplitude noisy component.

Second, the number of detected coincidences during the HTM learning corresponds approximately to the number of hyperspheres of the diameter *maxDist* which cover the entire training data. Performing multiple runs of the spatial pooling for various *maxDist* values, it is possible to examine the functional relationship between *maxDist* and the number of detected coincidences.

Third, all hand-written digits contained in the USPS database form a vector cloud embedded in a 256-dimensional space that is bounded by the hypercube of all possible images of the given size (16×16). Based on the PCA results, shown in Fig. 3 (a good separation of digit classes has been achieved already in a projection into two main PCA components), it can be assumed that the digit cloud is rather a *low-dimensional manifold* which does not fill in the whole 256-dimensional hypercube. Using such an interpretation of our input data, it is appropriate to identify the optimum *maxDist* value with the scale at which the useful data and the noisy component are structurally most differentiated.

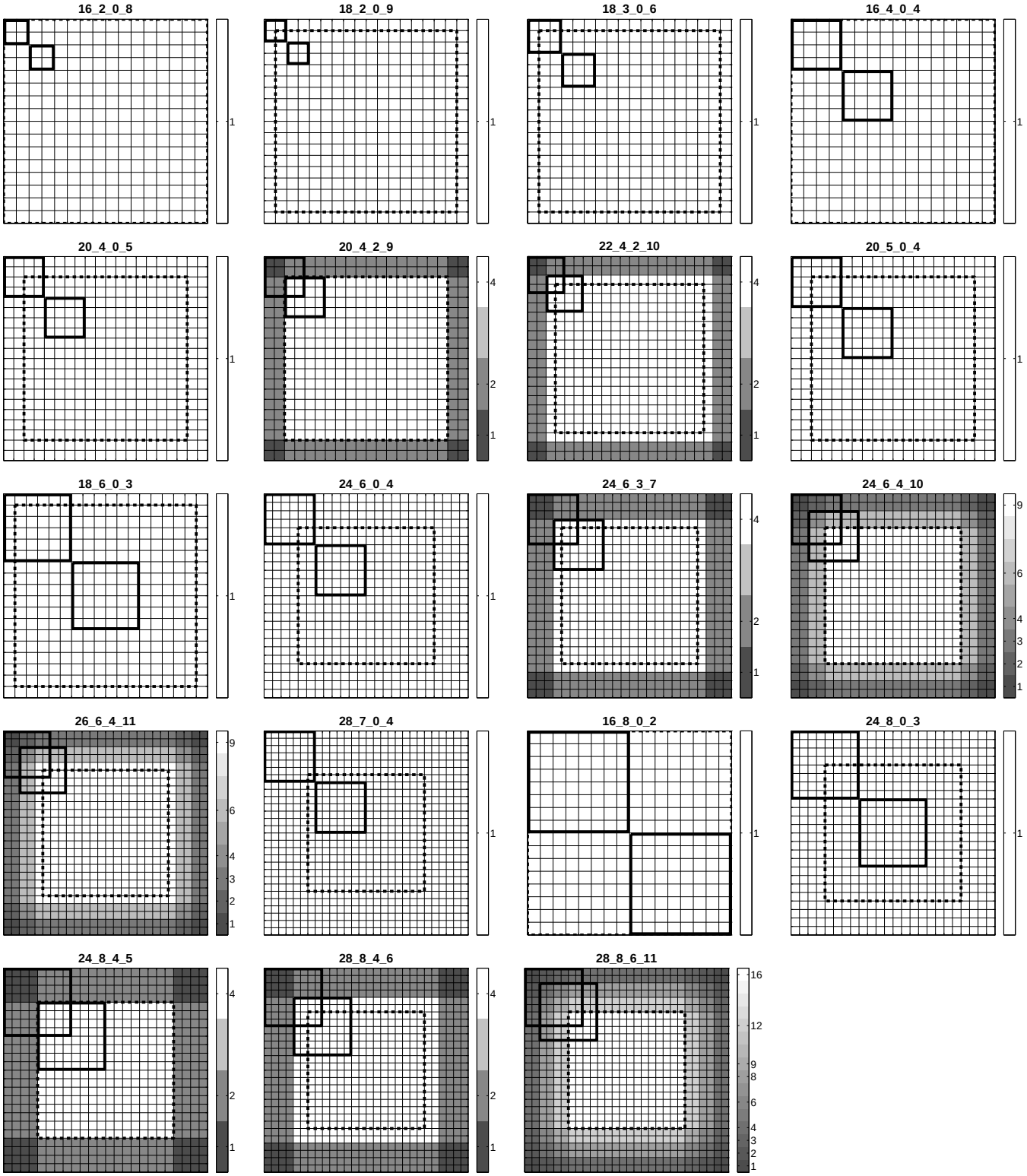


Fig. 7: Usage of the retinal pixels for all 19 explored HTM network architectures. The gray-level scales displayed next to each architecture indicate how many times the particular image pixels are utilized during the learning and inference stages of the network.

Table 2: Left: all considered single level HTM architectures which proved to have the homogeneous coverage of at least the area of 16×16 pixels in the center of the retina. All other subtables: three setups – low, medium, and high – of the parameters *maxDist* and *sigma*. The "spatial coincidences" columns show the mean numbers of coincidences detected by the nodes given the data and the particular *maxDist* value.

Retina	**Overlap**			**Low setup**			**Medium setup**			**High setup**		
	Patch		**Grid**	*maxDist*	**Spat. coinc.**	*sigma*	*maxDist*	**Spat. coinc.**	*sigma*	*maxDist*	**Spat. coinc.**	*sigma*
16	2	0	8	11.90	1451.59	308.32	14.87	1170.44	308.60	17.84	960.16	308.75
18	2	0	9	10.37	1204.67	309.64	12.96	998.79	310.72	15.55	830.25	311.13
18	3	0	6	46.63	1148.44	496.56	58.28	860.92	499.85	69.94	653.44	501.34
16	4	0	4	99.23	1682.75	691.14	124.04	1182.44	693.83	148.84	828.81	695.43
20	4	0	5	94.96	1014.84	676.63	118.70	727.92	683.97	142.44	521.80	688.79
20	4	2	9	96.99	1295.85	682.60	121.24	921.11	687.90	145.49	652.12	691.05
22	4	2	10	94.00	1056.70	679.63	117.49	757.77	685.30	140.99	542.70	688.95
20	5	0	4	145.70	1098.13	831.15	182.12	765.19	848.46	218.55	525.56	859.05
18	6	0	3	193.83	1717.56	948.36	242.28	1159.78	962.82	290.74	758.89	972.57
24	6	0	4	197.34	919.56	1026.48	246.68	661.81	1039.79	296.01	455.31	1047.66
24	6	3	7	198.18	1190.73	981.84	247.72	821.71	998.82	297.27	550.04	1010.26
24	6	4	10	201.22	1304.71	986.02	251.52	898.70	1004.37	301.82	598.69	1018.08
26	6	4	11	197.01	1092.98	981.33	246.27	755.67	999.64	295.52	509.68	1013.08
28	7	0	4	244.59	866.56	1150.60	305.74	611.13	1163.52	366.89	410.56	1176.51
16	8	0	2	309.74	2892.50	1215.58	387.17	1836.00	1235.10	464.61	1095.50	1254.54
24	8	0	3	296.42	981.11	1214.07	370.52	649.44	1275.40	444.62	417.89	1325.68
24	8	4	5	310.67	1530.16	1248.71	388.34	1018.60	1289.06	466.00	645.96	1319.47
28	8	4	6	296.42	1133.25	1237.06	370.52	767.47	1268.66	444.63	493.83	1295.17
28	8	6	11	303.92	1301.15	1243.21	379.90	873.41	1279.49	455.88	557.32	1307.40

In fractal theory, there are methods for estimating various kinds of fractal dimensions of low-dimensional manifolds embedded into high-dimensional spaces. One such method is the *box counting* method for assessing the Minkowski-Bouligand dimension [27]:

$$D_b = \lim_{\varepsilon \to 0} \frac{\log N(\varepsilon)}{\log 1/\varepsilon}, \tag{3}$$

where ε is the box scale and $N(\varepsilon)$ is the count of boxes that cover the data set at the given scale.

It can be shown that the uniformly sampled solid manifold produces the power-law decay of the box count with respect to the scale. The exponent of the power law is related to the manifold dimension. Note that the power-law function plotted in the log-log scale turns into the linear function with the slope identical to the power-law exponent. When the manifold is either sampled randomly or some low-dimensional noise is present in the data (e.g., differences between individual handwriting types), the power-law decay is being eased at the small scales. The turning point, at which the observed decay starts to have a tendency to follow the power law, represents the optimum scale at which the noise already declines in its influence, while the macroscopic structure of the data is not yet significantly affected. Because there is a close analogy between the spatial pooling procedure used in HTM and the box counting method, we propose to apply this methodological approach to the USPS data, interpreted as the low-dimensional manifold, and to set the optimum value of the parameter *maxDist* near to the mentioned turning point.

Since each HTM node at the first level receives its input from a different region of the retina, various nodes may detect various sets of coincidences for the same *maxDist* value. In order to arrive at the common optimum of the parameter *maxDist*, we have considered the mean numbers of coincidences over all nodes instead of analyzing all the nodes individually. For all considered network architectures, the mean numbers of coincidences were evaluated for 50 different *maxDist* values ranging from the minimum to the maximum pairwise distances present in the data. Typically, the number of detected coincidences decrease very slowly at the small scales but it falls much faster at the large scales. For determining position of the turning point we have proposed a method that models the decay by two tangent lines in the log-log scale. The first tangent approximates the function at the very small scales, whereas the other tangent approximates the function in the steepest linear part (i.e., the part which follows the power law). This linear part is usually used as the scaling interval for estimating D_b. The point of intersection of these two tangent lines provide us with an estimate of the turning point that is directly identified with the optimal *maxDist* value. An example of the turning point estimation is shown in Fig. 8.

As the proposed method for estimation of the optimal *maxDist* value uses several approximations, there is a possibility of missing the true *maxDist* optimum. To reduce such a possibility, we have conducted all our experiments with respect to the following three *maxDist* setups:

- **Low setup** – the estimated value reduced by 20%,
- **Medium setup** – the actual estimated *maxDist* value,
- **High setup** – the estimated value increased by 20%.

All the considered *maxDist* values for all the network architectures are summarized in Tab. 2.

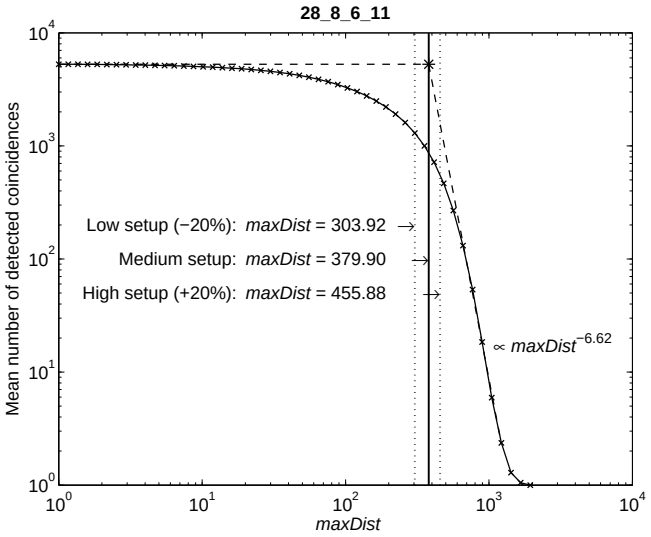


Fig. 8: Estimation of the optimum range of the parameter *maxDist* by exploiting the box counting method. The *mean number of detected coincidences* through all the nodes vs *maxDist* is shown in the log-log scale. The "star" symbol represents the turning point of the function being searched for.

5.4. Estimation of the optimal values of the parameter *sigma*

Assume that we have already performed a successful vector quantization in all the nodes using the optimal *maxDist* value estimated by the method explained in the previous section. During the inference, for calculation of the belief in a particular coincidence given the inferred pattern, the Gaussian inference function (1) is employed. This function is controlled by the only parameter *sigma*. Depending on the choice of *sigma*, one may achieve various inference regimes which may result in a very different classification performance. The small *sigma* values cause that high beliefs are assigned only to the coincidences which are very close to the inferred pattern. On the contrary, when too large *sigma* values are used, all the coincidences receive high belief values regardless of their distance from the inferred pattern.

For the purposes of this study, it is reasonable to derive the *sigma* values from the data and the actual set of coincidences memorized within each node. It can be seen that the Gaussian inference function is a monotonically decreasing function that transforms each distance value from the interval $\langle 0, \infty)$ into the belief from the interval $(0, 1\rangle$. The main task of the Gaussian inference in the HTM model is to appropriately cover gaps between the coincidences to reach a desired generalization behavior.

Let us now focus on a particular coincidence stored within any HTM node. When all unique training patterns are compared with the given coincidence, a set of distances is generated which can be interpreted as a random sample from the distribution of distances associated with this coincidence. By application of the Gaussian inference function, the unbounded distribution of the distances is transformed into the bounded belief distribution. Regardless of character of the distance distribution, we have no particular reason for preferring any specific subrange of the potential belief values. Therefore, our goal in obtaining the optimal *sigma* value is to transform the distance distribution into the interval $(0, 1\rangle$, so that it covers the whole interval as uniformly as possible. This task can be formulated by means of the entropy measure defined for a discrete distribution [28]:

$$H(X) = -\sum_{x \in X} p(x) \log p(x), \tag{4}$$

where X is a discrete random variable and $p(\cdot)$ is its probability mass function. Then, the optimal *sigma* value can be found by maximization of the entropy H of the belief distribution $p(x)$ with respect to the parameter *sigma*.

To be able to calculate the entropy according to the above discrete formula, it is necessary to discretize the belief distribution. Without loss of generality we can use a division of the interval $(0, 1\rangle$ into only two equal bins: $B_1 = (0, 1/2\rangle$ and $B_2 = (1/2, 1\rangle$. In such a scheme, any distribution with an equal $50\% \,/\, 50\%$ occupation of the both bins possesses the maximum entropy. This claim is equivalent with the condition for a distribution to have the median value equal to $1/2$.

An important consequence of the monotonicity of the Gaussian function is that the median of the distance population is always mapped onto the median of the belief population. Based on this property, for maximization of the entropy of the belief distribution within the two-bin discretization scheme, we just need to find such a *sigma* that maps the distance median onto the belief equal to $1/2$. For each coincidence, this operation can be expressed by means of the Gaussian inference function:

$$e^{-\left(\text{median}(D)/sigma_{opt}\right)^2} \quad = \quad 1/2, \tag{5}$$

where D is the distance distribution obtained for this coincidence. Finally, the optimum $sigma_{opt}$ is given as follows:

$$sigma_{opt} = \frac{\text{median}(D)}{\sqrt{-\ln 1/2}}. \tag{6}$$

When all the $sigma_{opt}$ values are calculated for all the coincidences within each individual node, we can proceed towards the estimation of the *sigma* optima characterizing each node as a whole. Finally, based on these node's *sigma* values, it is necessary to calculate a global *sigma* optimum valid for all the nodes at the given network level. Since no assumption can be made about the distributions of the coincidences and the data observed by any particular node, for calculation of the node's optimum, we use the median rather than the mean of the $sigma_{opt}$ values within the node. The global optimum *sigma* value is then calculated as the weighted average of the node's optima. Due to the presence of repeated blank patterns in some nodes (originating mainly from the zero-padded borders of the retina), the weight of each node's $sigma_{opt}$ was derived from the number of unique patterns observed by each

Table 3: Other relevant parameters of the NuPIC platform which were considered in all reported HTM network configurations.

Level 0: ImageSensor	
background	0
blankWithReset	true
Level 1a: SpatialPoolerNode	
clonedNodes	false
sparsify	false
explorer	Flash
Level 1b: TemporalPoolerNode	
clonedNodes	false
equalizeGroupSize	false
temporalPoolerAlgorithm	maxProp
explorer	Flash
Level 2: Zeta1TopNode	
outputElementCount	10
spatialPoolerAlgorithm	dot
mapperAlgorithm	maxProp
explorer	Flash

particular node. As these nodes receive a lower number of relevant training patterns, their outputs is adequately penalized in the ultimate $sigma_{opt}$ estimation.

All the $sigma_{opt}$ values calculated by the proposed method for all the network architectures, as well as for three *maxDist* setups, are shown in Tab. 2.

5.5. Search for the optimal values of the parameters *transitionMemory* and *requestedGroupCount*

Unlike both preceding cases, the parameters *transitionMemory* and *requestedGroupCount* concern the temporal learning that is the crucial part of the HTM model. For estimation of the optimal values of these parameters, we decided to perform a systematic search through the reasonable ranges of parameter values. This approach allowed us to draw interesting conclusions about the nature of the HTM temporal learning.

Since the HTM network learning process is, in general, computationally highly demanding job, we have restricted ourselves to a search through 4 fixed values of *transitionMemory* (i.e., {1, 4, 16, 64}) and 12 fixed values of *requestedGroupCount* (i.e., {2, 5, 10, 22, 46, 100, 215, 464, 1000, 2154, 4642, 10000}). All the parameter combinations have been investigated with respect to each of 19 considered network architectures, and for all three setups of the parameters *maxDist* and *sigma* (see Tab. 2). Altogether, we have analyzed $4 \times 12 \times 19 \times 3 = 2736$ different network configurations.

5.6. Other relevant parameters of the NuPIC platform

For the completeness, we also provide the values of the remaining NuPIC parameters which are relevant to the reported classification problem (see Tab. 3). For detailed description of the stated parameters, see [17].

6. Results

In Section 5 we have described our main theoretical contributions to the estimation of the optimal controlling parameters of the single-level HTM network applied to the recognition of the USPS hand-written digits. The search for actual values of these parameters required an extensive set of computer experiments. In this section we provide the reader with the comprehensive description of the achieved classification accuracy values for the cases explored in our experiments, with the comments on the influences of the individual parameters on the achieved accuracy.

6.1. Performance of individual HTM network architectures

For all 19 considered HTM architectures described in Section 5.2, we have carried out extensive computer experiments for many combinations of the parameter values. We have calculated the *maximum classification accuracy* (MCA) obtained for each architecture separately and collected also the 95-th percentile over all accuracy values obtained for the given architecture. In Fig. 9, the MCA values are plotted in increasing order, whereas the 95-th accuracy percentiles are represented by the lower bounds of the arrow-marked intervals. From this plot it is apparent that the architectures can be divided into two clear groups: (i) the group of architectures which use zero overlap of the node patches, and (ii) the group in which the overlaps with various sizes are used. The former architectures exhibit lower MCA values, while significantly higher MCA values were achieved for the latter ones. The architecture 16_2_0_8 with the minimum patch and having zero overlap is the second worst case, while the architecture 28_8_6_11 with the maximum patch as well as overlap is the best one. Note that the architecture 28_8_6_11 has also a narrow interval of values greater than 95-th percentile, which means that it is quite robust against inaccurately chosen values of the HTM parameters.

6.2. Influence of the *maxDist* and *sigma* parameters on the classification performance

In our experiments we also intended to analyze the influence of the parameters *maxDist* and *sigma* on the classification performance. Based on the technique, we developed in Sections 5.3 and 5.4, the computer experiments have been carried out in the three setup modes: low setup, medium setup, and high setup. As the optimum values of the parameter *sigma* are calculated in relation to the previously found values of the parameter *maxDist*, in all three setups, the optimum values of these parameters are coupled. From Fig.10a it is clear that the most of architectures reach their MCA for the low setup which corresponds to the 20% underestimation of the estimated medium *maxDist* value. Interestingly, this trend is equally present in both groups of the architectures without and with overlaps.

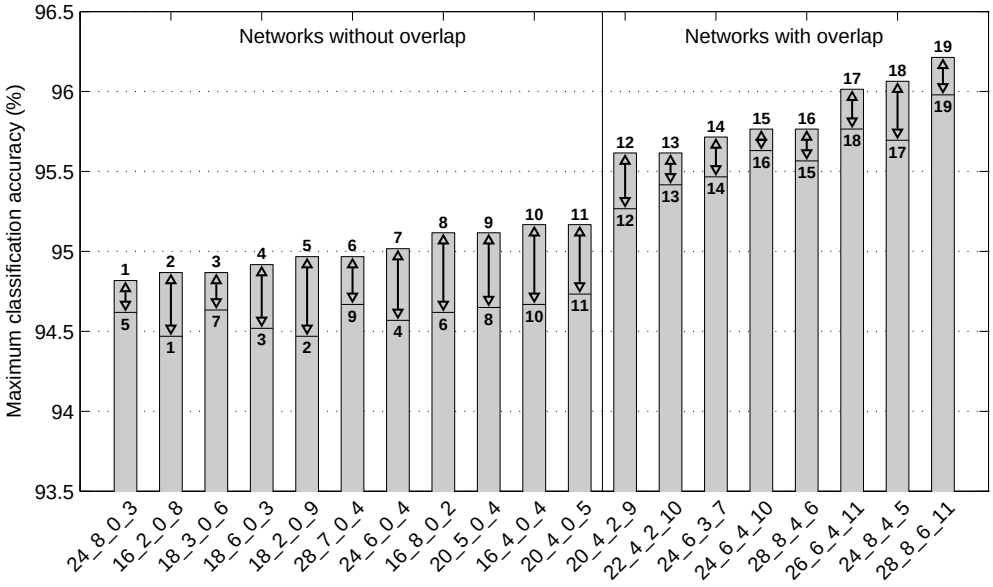


Fig. 9: Maximum classification accuracy obtained for each considered HTM network architecture when the most appropriate parameter values were used. Arrow-marked intervals represent the 5% range of the most successful network configurations (i.e., the upper value is the maximum accuracy while the lower value is the 95-th percentile of all obtained accuracies for the given architecture). The narrower is the range, the more stable is the network architecture or, in other words, the less sensitive is the network to the choice of optimal parameters. Numbers above the arrows stand for the architecture indices when ordered according to the achieved maximum accuracy. Numbers below the arrows stand for the indices of the ordering according to the 95-th accuracy percentile.

6.3. Influence of the *transitionMemory* parameter on the classification performance

The optimization of the further two parameters, namely parameter *transitionMemory* and *requestedGroupCount*, has been performed as a systematic search through reasonable ranges of their values. In case of the parameter *transitionMemory* we have chosen four representative values (see Section 5.5). We have observed the following trends in the results (see Fig.10b). First, in the less successful architectures with zero overlap the higher MCA values have been achieved in majority of cases (7 of 11) for the lower values of the *transitionMemory* (≤ 4). Second, for more successful architectures with various overlaps, the trend is quite opposite. The most of architectures (5 of 8) achieved the maximum MCA for higher values of the *transitionMemory* (≥ 16). This finding indicates that these architectures are capable to better utilize the temporal information contained in data.

6.4. Influence of the *requestedGroupCount* parameter on the classification performance

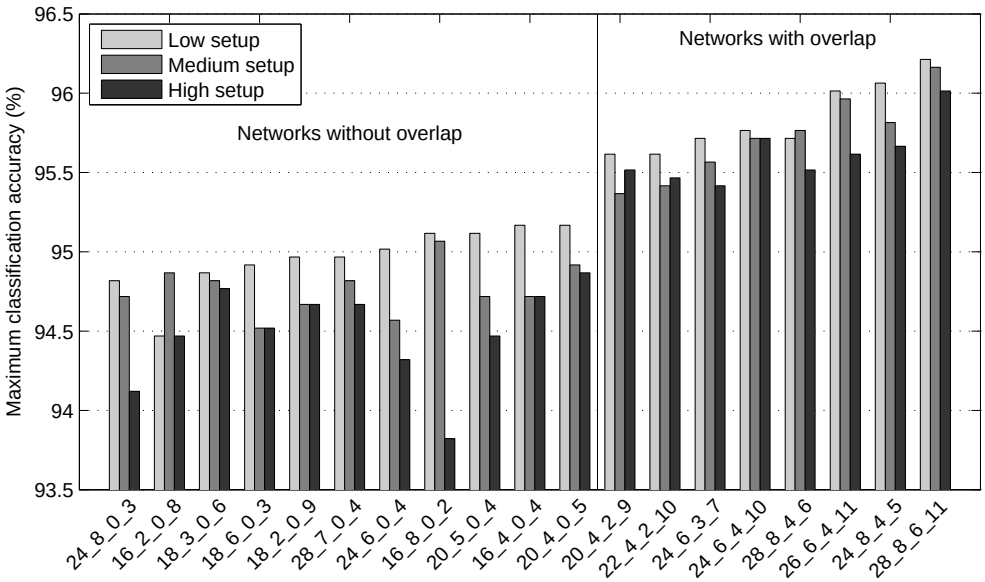
This section is devoted to the most important property of the HTM model – the temporal grouping. In Fig. 11a we show the plots of MCA vs *requestedGroupCount* (the irrelevant parts are cut out) which express the best achieved performance (MCA) given the specific value of the parameter *requestedGroupCount*. In this picture a separation of the network architectures into two sets showing different characteristic behavior is apparent.

The architectures within the first set contain the networks without overlap and exhibit lower MCA values for most values of the *requestedGroupCount* parameter. Furthermore, these architectures are also distinguished by achieving highest MCA values for much higher *requestedGroupCount* values in comparison with the other group of architectures. This means that there is usually only an insignificant difference between the number of temporal groups and the number of spatial coincidences what is also reflected by low temporal reduction factor (see Fig. 11b and Tab. 4). In other words, the performance of such HTM network architectures is comparable with the performance of a simple vector quantization approach.

The architectures contained in the second characteristic set achieve best results for small values of the *requestedGroupCount*. It should be noted that in this case, the difference between the MCA maxima and the values corresponding to the right tails of curves (expressing the performance of a simple vector quantization approach), is much more expressed. Consequently, these architectures can significantly benefit from the temporal information in the training sequence. This indicates that the strong point of HTM can fully be demonstrated exactly by these architectures.

Another finding based on the inspection of Fig. 11b is, that three most successful architectures posses also a significant temporal reduction factor, while it is small and almost constant for all remaining architectures explored. This knowledge suggests a potential diagnostic tool for testing a suitability of the given HTM architecture. If the given architecture would reach the best results without a sufficient temporal re-

a)



b)

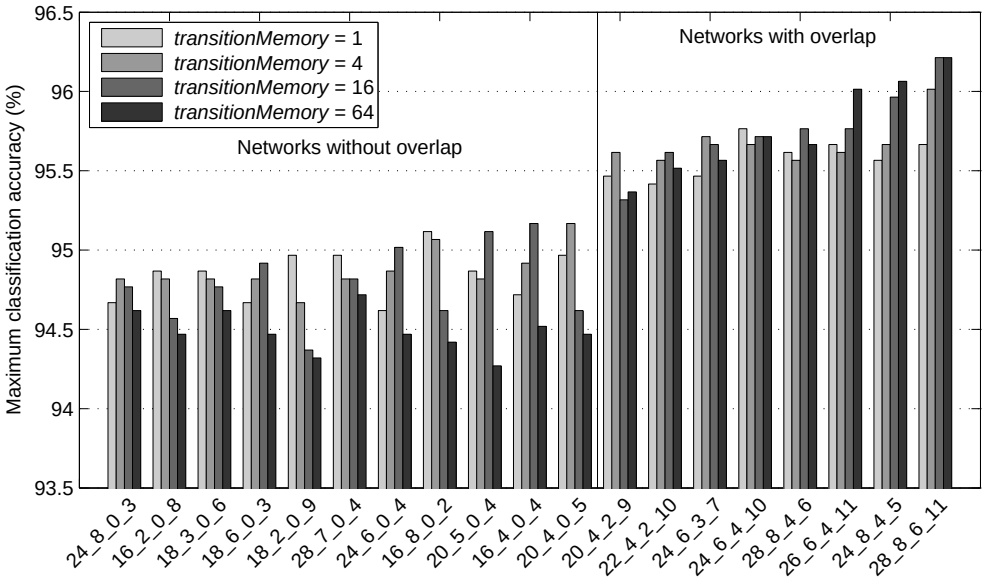
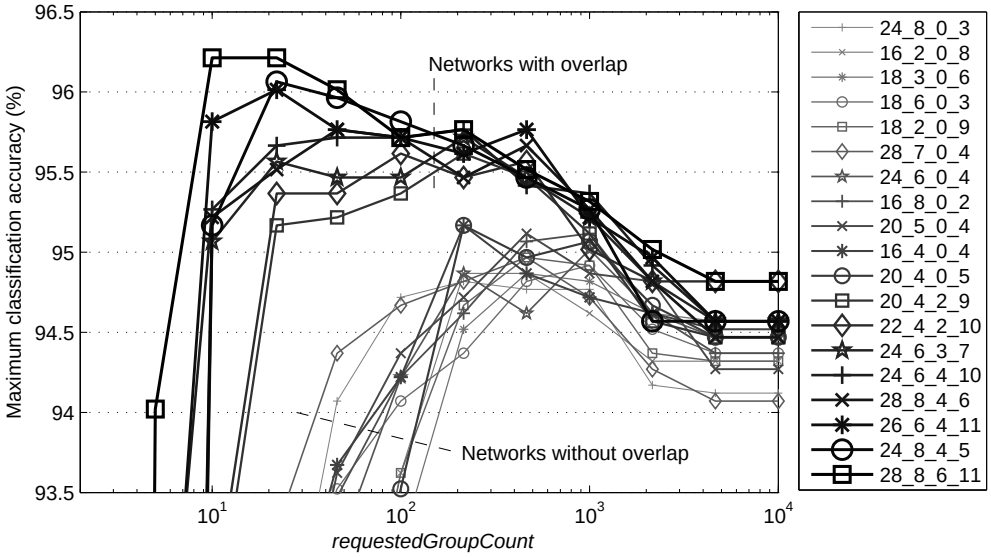


Fig. 10: Maximum classification accuracy values of the individual architectures explored: a) for three setups of the parameters *maxDist* and *sigma*, b) for four representative values of the parameter *transitionMemory*. Both diagrams manifest two characteristic groups of the networks: without and with patch overlap.

a)



b)

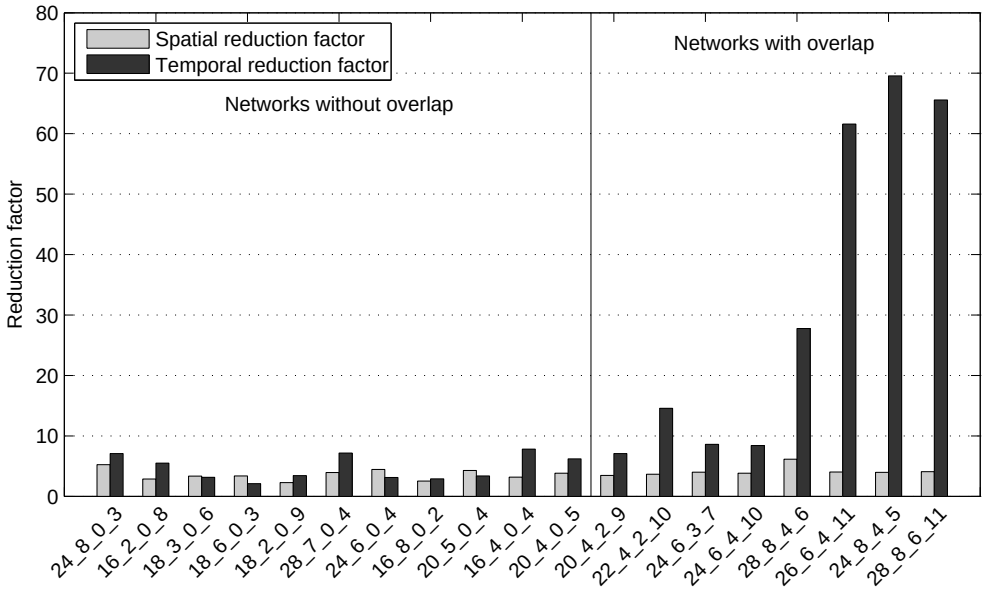


Fig. 11: a) The plot of the maximum classification accuracy vs the parameter *requestedGroupCount* for all 19 networks considered. Two groups of the network architectures (without and with the patch overlap), which show different characteristic behavior, are apparent. b) The spatial and temporal reduction factors calculated for the most successful network configuration with respect to each individual network architecture. The architectures are ordered with respect to the obtained maximum classification accuracy.

Table 4: All considered HTM network architectures ordered with respect to the obtained maximum classification accuracy. The columns "input patterns", "spatial coincidences" and "temporal groups" summarize the mean numbers of the unique patterns, the detected coincidences and the temporal groups within the nodes of the particular HTM architecture, respectively. The gray rows stand for three most successful architectures which are also distinguished by high values of the temporal reduction factor (B/C).

Architecture	Input patterns (A)	Spatial coinc. (B)	Temp. groups (C)	Reduction factor (A/B)	Reduction factor (B/C)	Accuracy (%)
24_8_0_3	5130.33	981.11	138.78	5.23	7.07	94.82
16_2_0_8	3356.03	1170.44	212.45	2.87	5.51	94.87
18_3_0_6	3861.67	1148.44	363.47	3.36	3.16	94.87
18_6_0_3	5810.11	1717.56	813.67	3.38	2.11	94.92
18_2_0_9	2726.19	1204.67	350.64	2.26	3.44	94.97
28_7_0_4	3409.38	866.56	120.81	3.93	7.17	94.97
24_6_0_4	4091.75	919.56	294.13	4.45	3.13	95.02
16_8_0_2	7278.00	2892.50	1000.00	2.52	2.89	95.12
20_5_0_4	4707.69	1098.13	324.00	4.29	3.39	95.12
16_4_0_4	5359.88	1682.75	215.00	3.19	7.83	95.17
20_4_0_5	3871.80	1014.84	163.76	3.82	6.20	95.17
20_4_2_9	4502.93	1295.85	182.95	3.47	7.08	95.62
22_4_2_10	3871.45	1056.70	72.57	3.66	14.56	95.62
24_6_3_7	4750.86	1190.73	138.39	3.99	8.60	95.71
24_6_4_10	5003.13	1304.71	155.33	3.83	8.40	95.76
28_8_4_6	4718.61	767.47	27.64	6.15	27.77	95.76
26_6_4_11	4395.42	1092.98	17.74	4.02	61.60	96.01
24_8_4_5	6072.36	1530.16	22.00	3.97	69.55	96.06
28_8_6_11	5315.35	1301.15	19.84	4.09	65.57	96.21

duction, it would be justified to consider this architecture as non-optimal.

6.5. Overall the best HTM network configuration

Using the best HTM network architecture 28_8_6_11, which resulted from the above described basic parameter optimization procedure, we have performed an additional smooth tuning with slightly modified parameter values. For the following values of the HTM controlling parameters: $maxDist = 312.73$, $sigma = 1247.80$, $TransitionMemory = 64$, $RequestedGroupCount = 11$, we have reached the highest overall classification accuracy of 96.36%. In Tab. 5 we present the ultimate classification confusion matrix for 10 hand-written digit classes from the USPS testing set, classified by the best HTM network architecture, using the mentioned parameter values. The highest classification accuracy (99.44%) has been achieved for the digit "9", and the lowest one (93.37%) for the digit "3".

Eventually, we were interested in a comparison of the performance of the best HTM network configuration in relation to the digits which we declared as "unidentifiable" (erroneous labeling or too distorted shape) in Section 4.2. In Fig. 12 we display all the symbols which have been misclassified by the best HTM network. Note that 13 of 14 "unidentifiable" digits (marked by the crossed squares) are present in the set of HTM misclassified symbols. Despite the presence of substantial distortions, the testing digit #266 ("4") has been correctly classified.

Table 5: The classification confusion matrix of 10 digit classes for the best HTM network configuration 28_8_6_11.

Predicted group	True group "0"	"1"	"2"	"3"	"4"	"5"	"6"	"7"	"8"	"9"
"0"	**355**	0	1	0	0	3	0	0	1	0
"1"	0	**256**	0	0	1	0	0	0	0	0
"2"	1	0	**186**	3	1	1	1	0	2	0
"3"	0	0	5	**155**	0	1	0	1	3	0
"4"	0	4	2	0	**191**	0	4	5	0	1
"5"	0	0	0	7	0	**155**	4	0	1	0
"6"	0	3	0	0	1	0	**161**	0	0	0
"7"	2	1	2	0	1	0	0	**141**	0	0
"8"	0	0	2	0	0	0	0	0	**158**	0
"9"	1	0	0	1	5	0	0	0	1	**176**
Accuracy (%)	98.89	96.97	93.94	93.37	95.50	96.88	94.71	95.92	95.18	99.44

7. CONCLUSIONS

The focus of this paper is in a systematic exploration of possibilities how to optimize several important controlling parameters of the Hierarchical Temporal Memory network when applied to a problem of visual pattern recognition. This model, as outlined in Section 2, represents a biologically inspired memory-prediction network model that takes advantage of the Bayesian belief propagation and revision techniques. As any optimization of the HTM parameters has to be inevitably tailored to the data being processed, the first task was to select suitable data. We decided for the well-known USPS database of hand-written digits which serves as a suitable starting point for applying the HTM to more complex PR problems in the future. The important advantage of selecting the USPS database was that a great number of benchmarks were performed for different classification methods. In spite of this advantage, several recent papers reported on some defects occurred in the generation of this data set. Therefore in Section 4.3 we have analyzed in details the distributions of individual digit classes and differences between the given training and testing subsets of the whole database. The results of the permutation test, based on the so-called randomization distribution, showed significant differences (p-value ≤ 0.05; see the right part of Tab. 1) between the training and testing sets in 7 from 10 digit classes. This means that the generation process was either not random or the current random selection is highly improbable. We think that the PR community which still uses the USPS database should be aware of these findings and may initiate a generation of statistically more reliable data. Furthermore, we have found 5 undoubtedly incorrectly labeled digits in the training set, and 14 severely distorted or falsely labeled digits in the testing set. We have removed defective digits from the training set, however, for preserving equivalent benchmark conditions, we retained the defective digits in the testing set. Only for a demonstration of the influence of these defect digits on the recognition accuracy, we have removed them from the testing set and performed tests with the best found HTM configuration.

For any implementation of the HTM network the design of

Fig. 12: All classification errors produced by the best HTM network. The crossed digits belong to the set that has been suggested for removal from the USPS testing set due to the presence of labeling errors or severe distortions (see Section 4.2). The denotation $x \rightarrow y$ stands for misclassification of the true digit x as the digit y.

a module responsible for forming virtual temporal sequences for the static training data is important. There are several such modules, called explorers, implemented in the NuPIC platform by Numenta. Basically, all these modules generate smooth temporal sequences suitable for learning, thereby blow up the training set, because every training pattern varied and presented to the network in several versions. However, such an expansion of the training set turned out unacceptable regarding the correct benchmarking of the HTM model. Therefore, we have proposed an alternative method for construction of the training sequence consisting of only original training images. The method is based on a greedy algorithm which orders the digits within each class according to their mutual distances (in L_2 metric). Such an ordered sequence of images is then considered as a virtual temporal sequence suitable for training the HTM network. The proposed approach has no negative influence on the HTM performance in this particular application, but it saves memory and computational time significantly.

Each level of any HTM architecture is fully determined by three parameters – *patch size*, *overlap*, and *retina size.* The patch size determines the extent of the node's field of view. The overlap parameter controls how significant is an overlap between patches of any two neighboring nodes. The retina size specifies the size of an input image and defines the possible ranges of the previous two parameters. However, their particular values which would be optimal for the given PR task, are variable and not a priory known. The additional parameter of the HTM architecture is the *grid size* which is dependent on the previous three parameters. It turned out that a deeper study of the influence of the patch size and overlap values on the network performance, needs to restrict our research only to the single-level HTM networks.

In the paper we have introduced a novel view of relation between the first two parameters of the HTM architecture (the patch size and the overlap). Given the retina size, instead of considering all possible overlaps for all possible patch sizes, we set the methodological requirement for a reasonable combination of the patch and overlap values that is given by the homogeneous usage of a certain central part of the retina. This requirement is anchored in observations that the arbitrary value of the overlap for the given patch size may result in preferring some of the retinal pixels in construction of the output of the first network level. To avoid such undesirable behavior, 19 admissible architectures have been selected for the USPS classification task. In Fig. 7 the results of the pixel coverage simulation for all 19 architectures are demonstrated which correspond to such patch sizes and overlaps which guarantee the homogeneous usage of at least the central area containing the original digit images (16×16 pixels). Only these architectures have been included in the further analysis.

At each HTM level, for controlling the node's learning and inference processes, the following four parameters are used: *maxDist, sigma, transitionMemory*, and *requestedGroupCount*. In the existing works on various HTM applications, the search for the optimum values of all these parameters is left solely on the user's computer experimentation. However, this job is feasible only under strong constraints on parameter working spaces, thus enabling to reach only their suboptimal solutions.

The experiments with the HTM using empirically found "optimum" values of the parameter *maxDist* revealed that the spatial pooling procedure tends to ignore the density of the input data, thereby inevitably leads to highly non-optimal recognition rates. In the paper a novel method for estimation of the parameter *maxDist* is proposed that is based on some principles used in fractal theory. We have analyzed properties of the data clouds (representing digit fragments) in high-dimensional spaces, interpreted as low-dimensional manifolds, by the approach similar to the *box counting* method (see Section 5.3). We have developed a procedure for estimating the turning point position of the power-low decay of the scaling function (i.e., *mean number of detected coincidences* vs *maxDist*) in the log-log scale (see Fig. 8). A common *maxDist* optimum for all nodes in the given level is calculated by averaging the coincidence numbers over all nodes. The obtained values of the *maxDist* optimum are summarized in Tab. 2.

For the inference, applied to a pattern presented to the already learned level of the HTM network, the belief values are calculated using the Gaussian function (1) with the only con-

trolling parameter *sigma*. The main task of the Gaussian inference is to appropriately cover gaps between the coincidences memorized in the node. The optimum value of this parameter should guarantee a desired generalization and it is closely related to the distribution of the distances between the fixed set of coincidences and all possible input patterns for the given node. Therefore we were interested in investigation of the way by which the parameter *sigma* can mediate the relation of the distance distribution to the distribution of the beliefs. Since there is no particular reason to prefer a specific subrange of the belief values, we have formulated the task to find such an optimum *sigma* that yields a maximally uniform coverage of the interval $\langle 0,1\rangle$ after the distance transformation. For solving this task we proposed a method that maximizes the entropy of the belief distribution. The assumption of the simplest possible two-bin discretization of this distribution has finally arrived at the formula (6) in which the optimum value of *sigma* for the one fixed coincidence is given by introducing the median of the distance distribution into the Gaussian inference function. The optimum value for the whole node is obtained as the median value over all the coincidences within the node. Finally the global optimum *sigma* value for the whole layer of the nodes is found as a weighted average of the node's optima.

The search for optimum values of the parameters $transitionMemory$ and $requestedGroupCount$ lead to the finding of two characteristically distinguished groups of architectures with respect to the obtained maximum classification accuracy. The more successful architectures with various overlaps achieve greater MCA values for higher values of the parameter $transitionMemory$ (see Fig. 10). A conclusion can be drawn that these architectures are able to better utilize the temporal information contained in the training sequence. If we plot the values of MCA in relation to the parameter $requestedGroupCount$ (see Fig. 11) a similar division into two different architecture groups can be observed. The architectures without patch overlap achieve generally lower MCA values and prefer much higher values of the $requestedGroupCount$. The architectures with various patch overlaps achieve better results for small values of $requestedGroupCount$. They can significantly benefit from the temporal information in the data.

In Tab. 6 the final overview of the classification results is summarized. It contains the classification accuracy values achieved by various conventional classifiers benchmarked in the cited literature and those achieved by the three previously published HTM implementations. The classification accuracy of our HTM network with systematically optimized parameters slightly outperforms all the previous HTM implementations, however, the difference is not statistically significant (see confidence intervals in Tab. 6). Considering the accuracy confidence interval, this result is only significantly better than the *Optimal margin classifier* (95.40%) and classifiers below. On the other hand, it is significantly worse than the *Boosted neural nets* (97.40%) and methods above. Therefore, all four HTM implementations, as well as the both *Virtual SVM*s (97.00% and 96.80%), the *Local learning* method (96.70%) and the *SVM* method (96.00%) form a group of classifiers which are not statistically differentiable given the standard USPS testing set. For the completeness, we report also on the HTM performance achieved for the corrected testing set (i.e., excluding the digits which cannot be correctly classified due to the presence of labeling errors or severe distortions). In this case, the achieved classification accuracy has been as much as 96.99%.

Table 6: The overview of different classification methods applied to the USPS classification problem [9, 10].

Classification method	**Accur. (%)**	**Conf. int. (%) *****
Human error rate (1)	98.49	
Combination of tangent vector and local representation	98.00	
Feature-based virtual SVM	97.66	
Human error rate (2)	97.50	
Tangent distance *	97.40	
Boosted neural nets *	97.40	
Virtual SVM (local kernel)	97.00	
Single level HTM (this study) **	**96.99**	**96.22 – 97.77**
Virtual SVM	96.80	
Local learning *	96.70	
Single level HTM (this study)	**96.36**	**95.52 – 97.20**
Two level HTM (Java implementation) [11]	96.32	
Two level HTM (NuPIC "Pict. Demo") [13]	96.26	
HTM with "eye movements" [14]	96.26	
SVM	96.00	
Optimal margin classifier	95.40	
LeNet1 *	95.00	
Nearest-neighbor *	94.10	

* Classifiers trained on the USPS database extended by some machine-printed patterns.

** In this case, we have considered the testing set without 14 severely distorted or falsely labeled digits.

*** The confidence intervals were calculated at the significance level of 5% ($\alpha = 0.05$) according to Eq. (2) in [29].

The following key original contributions of the presented paper can be summarized:

- the statistical analysis of the USPS digit database,
- a novel method for construction of the training sequences by ordering series of the static images,
- a novel characterization and selection of the acceptable HTM architectures for the given task which uses the analysis of the node's overlaps and the criterion of the homogeneous usage of the retinal pixels in learning and inference procedures,
- a novel method for estimation of the parameter *maxDist* based on the box counting method,
- a novel optimization method for the parameter *sigma* based on the maximization of the entropy of the belief distribution,

- findings based on the analysis of the influences of the parameters $transitionMemory$ and $requestedGroupCount$ on the HTM network performance.

Since the whole study has been focused solely on the single-level HTM architectures which appeared to be sufficient for the given classification problem, no explicit references to the performance of the multi-level architectures could be made. Nevertheless, the proposed methods for estimating operational values of the parameters *maxDist* and *sigma* can equally be applied to the higher levels of the HTM hierarchy. In such a case, however, one may experience memory problems, since the data dimensionality usually rises with the level of the hierarchy.

In the further research into the HTM network structure and methods for optimization of its parameters, we intend to concentrate ourselves on possible improvements of the vector quantization algorithm (spatial pooling) by using more adequate clustering schemes and pattern similarity measures. We will also deal with design and implementational aspects of the multi-level HTM networks which would be suitable for fast detection and classification of visual objects at different scales and locations.

ACKNOWLEDGMENTS

The research reported in this paper has been partially supported by the Slovak Grant Agency for Science (project No. 2/0019/10).

REFERENCES

[1] Felleman, D., van Essen, D. (1991). Distributed hierarchical processing in the primate cerebral cortex. *Cerebral Cortex* (1), 1–47.

[2] Serre, T., Oliva, A., Poggio, T. (2007). A feedforward architecture accounts for rapid categorization. In: *Proc. National Academy of Sciences of the USA*, Vol. 15. pp. 6424–6429.

[3] Lee, T. S., Mumford, D. (2003). Hierarchical Bayesian inference in visual cortex. *Journal of Optical Society of America* A 20(7), 1434–1448.

[4] Dean, T. (2006). Scalable inference in hierarchical generative models. In: *Proc. 9th Int. Symp. on Artificial Intelligence and mathematics*. pp. 1–9.

[5] Hawkins, J., Blakeslee, S. (2004). *On intelligence*. Henry Holt and Company, New York.

[6] George, D., Hawkins, J. (2009). Towards a mathematical theory of cortical micro-circuits. *PLoS Computational Biology* 5(10). DOI 10.1371/journal.pcbi.1000532.

[7] George, D., Hawkins, J. (2005). Hierarchical Bayesian model of invariant pattern recognition in the visual cortex. In: *Proc. Int. Joint Conf. on Neural Networks*. Montreal, Canada.

[8] Numenta (2007). Zeta1 algorithms reference. Document version 1.0.

[9] Dong, J. (2001). Statistical results of human performance on USPS database. Technical report, CENPARMI, Concordia University.

[10] Dong, J. (2005). HeroSvm 2.1. `http://www.cenparmi.concordia.ca/~jdong/HeroSvm.html`.

[11] Thornton, J. R., Gustafsson, T., Blumenstein, M., Hine, T. (2006). Robust character recognition using hierarchical Bayesian network. In: *Proc. 19th Australian Joint Conf. on Artificial Intelligence, Hobart, Australia*. pp. 1259–1264.

[12] Thornton, J. R., Faichney, J., Blumenstein, M., Hine, T. (2008). Character recognition using hierarchical vector quantization and temporal pooling. In: Wobcke, W., Zhang, M. (eds.) *Proc 21st Australasian Joint Conf. Artificial Intelligence*, Vol. Lecture Notes in Computer Science. pp. 562–572.

[13] Bobier, B. (2007). Hand-written digit recognition using Hierarchical Temporal Memory. `http://arts.uwaterloo.ca/~cnrglab/?q=system/files/SoftComputingFinalProject.pdf`.

[14] Numenta (2009). Numenta forum: benchmark with USPS handwritten digit dataset. `http://www.numenta.com/phpBB2/viewtopic.php?t=224`.

[15] Numenta (2008). Hierarchical temporal memory, concepts, theory, and terminology. Document version 1.8.0.

[16] George, D. (2008). How the brain might work: a hierarchical and temporal model for learning and recognition. Ph.D. thesis, Dept. of Electrical Engineering, Stanford University, USA.

[17] Numenta (2009). Numenta node algorithms guide, NuPIC 1.7.

[18] Johnson, S. T. (1967). Hierarchical clustering schemes. *Psychometrika* 32, 241–254.

[19] Numenta (2008). Vision framework guide, NuPIC 1.6.1.

[20] Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[21] Wang, C. H., Srihari, S. N. (1988). A framework for object recognition in a visually complex environment and its application to locating address blocks on mail pieces. *Int. Journal of Computer Vision* 2(2), 125–151.

[22] Dong, J., Krzyzak, A., Suen, C. Y. (2001). Statistical results of human performance on USPS database. Technical report, Centre of Pattern Recognition and Machine Intelligence, Concordia University.

[23] Seewald, A. K. (2005). Digits – a dataset for handwritten digit recognition. Technical Report TR-2005-27, OFAI, Wien.

[24] Hull, J. J. (1994). A database for hand-written text recognition research. *IEEE Transactions on Pattern*

Analysis and Machine Intelligence 16, 550–554.

[25] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computing* 1(4), 541–551.

[26] Ernst, M. D. (2004). Permutation methods: A basis for exact inference. *Statistical Science* 19(4), 676–685. DOI 10.1214/088342304000000396.

[27] Schroeder, M. R. (1991). *Fractals, chaos, power laws : minutes from an infinite paradise*. W. H. Freeman, New York.

[28] Shannon, C. E. (1948). A mathematical theory of communication. *Bell System Technical Journal* 27, 379–423.

[29] Martin, K. J., Hirschberg, D. S. (1996). Small sample statistics for classification error rates II: confidence intervals and significance tests.

Received February 24, 2010.
Accepted April 6, 2010.