Handwritten digit recognition using biologically inspired features

Ângelo Cardoso, Andreas Wichert

INESC-ID Lisboa and Instituto Superior Técnico, Technical University of Lisbon
Av. Prof. Dr. Aníbal Cavaco Silva, 2744-016 Porto Salvo, Portugal

Abstract

Image recognition problems are usually difficult to solve using raw pixel data. To improve the recognition it is often needed some form of feature extraction to represent the data in a feature space. We use the output of a biologically inspired model for visual recognition as a feature space. The output of the model is a binary code which is used to train a linear classifier for recognizing handwritten digits using the MNIST and USPS datasets. We evaluate the robustness of the approach to a variable number of training samples and compare its performance on these popular datasets to other published results. We achieve competitive error rates on both datasets while greatly improving relatively to related networks using a linear classifier.

Keywords: image recognition, simple and complex cells, handwritten digits, feature extraction

1. Introduction

Handwritten digit recognition despite being a well studied problem is still an active topic of research. This problem is relevant for tasks like postal mail sorting or form data processing. Several works have been devoted to the problem from a feature extraction or classification perspective. In this text we analyze the application of the map transformation cascade (MTC) [1] to this task, which works as feature extractor combined with a classifier. MTC is a model for visual recognition where simple and complex cells are arranged

Email addresses: angelo.cardoso@ist.utl.pt (Ângelo Cardoso), andreas.wichert@ist.utl.pt (Andreas Wichert)

in a hierarchy like proposed by Hubel and Wiesel for the visual cortex [2] and incorporated in several models like Neocognitron [3] and HMAX [4]. In [1] the MTC relation and comparison with Neocognitron was established using a nearest neighbor classifier. In this text we discuss how it relates to HMAX [4] and compares with other pattern recognition methods on two popular datasets of handwritten digits using a linear classifier. A combination of HMAX's features and a classifier has been shown to achieve good results on object recognition [5].

In the next section we make a short overview of biological vision and computational models for visual recognition. Afterwards we describe MTC and finally evaluate its performance of MTC on handwritten digit recognition using the USPS and MNIST datasets. We analyze how the performance of the approach is affected by the number of training samples and finally measure the error rate on the entire dataset.

2. Related Work

The classical hypothesis of Hubel and Wiesel [6] has been transposed into several computational models for visual recognition. The key idea is that two kinds of cells are arranged in layers, being the simple cells selective for a particular stimulus and a position of that stimulus in the visual field and complex cells also selective for a particular stimulus but less selective for its position in the visual field. These two types of cells are then arranged in a hierarchy where the cells' primary stimulus becomes increasingly more sophisticated. The cell's receptive field size increases gradually along the ventral stream [7, 8]. The complexity of the preferred stimuli also increases as we move away from the input [6, 8]. In the temporal visual cortex (IT) some neurons are tuned to specific views of objects [9] while others are invariant to these views and respond identically to an object independently of the view [10, 11].

Cells in the primary visual cortex are organized in columns according to their preferences, being a hypercolumn a block of cortical cells in which all orientation preferences are represented for a small portion of the visual field [7]. A plausible reason for their existence is sharpening selectivity [12, 13], selectivity may be sharpened by lateral inhibitory connections from neighbor cells with slightly different orientations.

From a computational perspective Neocognitron [14, 3] established key principles for a neural network for visual recognition based on Hubel and

Wiesel's classical hypothesis. Simple cells preferred stimuli in Neocognitron is tuned by competitive learning in which the stronger the response of neighboring units, representing lateral connections, the smaller the chance of the unit becoming winner. The simple cells are arranged in cell-planes and each of these planes reacts to a specific stimulus in different positions, after learning each cell-plane becomes independent. HMAX [4, 15, 5] also builds on the classical hypothesis of Hubel and Wiesel. A key difference between HMAX and Neocognitron is the complex cells responses which on the first is the maximum of the afferent responses and on the second the sum[16] or squared sum [17] of the afferent responses. Another important difference between HMAX and other works is that the model parameterization aims at replicating biological measurements [18] and biological performance [5].

The key difference between MTC [1] and both Neocognitron and HMAX is in simple cell responses. Among all cells which share the same receptive field, only the cell whose preferred stimulus is most similar to the current one is active, while others are silent. This sparse simple cell response can be biologically implemented through lateral inhibition as in LISSOM [19]. This response is analogous to LISSOM activity bubble stabilizing with only one active cell. MTC's complex cells are active if any of its afferent simple cells are active, which is therefore equivalent to HMAX's complex cells.

The MTC architecture is related to convolutional networks [20] which are trained by error back-propagation. It can be described as a filter bank learned over regularly-spaced patches by quantization, a winner-take-all operation over the filter bank and a maximum pooling operation of filters over patches. The model is multi-stage and the filter bank for the first stage is learned over the input patterns (i.e. image patches), while on the following layers is learned over the filter responses of previous layers. A MTC with only a simple and a complex layer is related to a spatial bag of features model [21] where the filter bank is learned by quantization of dense SIFT [22] features. The filter bank is chosen so as to minimize the mean squared error. In Deep Belief Networks [23, 24] each layer is a Restrict Boltzman machine which is trained to minimize the energy of the input units over a set of hidden units, resulting in the hidden units learning to represent features that capture higher-order correlations in the input data.

MTC is trained in an unsupervised and greedy manner, one layer at a time, analogously to Neocognitron's intermediate layers [25, 26]. After all layers are trained, the output of the last layer to a set of input patterns is used to train a classifier. Unsupervised greedy layer-wise training has been used

in related network architectures [23, 27, 24]. In MTC the filter weights are chosen so to minimizing the mean squared error produced by the filter bank at the simple layer. In encoder–decoder methods [27] the filter weights are optimized according to the reconstruction error after the maximum pooling. A semi-supervised method for training deep nets was proposed in [28] which learns filter banks which produce similar responses for samples with the same label.

MTC produces a binary sparse code as the result of the lateral competition in the responses after learning. In [27] a rather opposite approach is taken after learning, the stochastic binary units are replaced by continuous sigmoid units to avoid quasi-binary codes. In [29] the responses are made sparser by suppressing the ones below an adaptive threshold. In [27] the sparsity constraint is introduced during learning by a sparsifying logistic between the encoder and decoder and in [30] the sparsity is also induced during learning using the PSD algorithm [31].

The first layer filter bank tends to recognize orientations as is explicitly done by Gabor filters [4] or direction gradients [32]. The expansion of the MNIST dataset through distortions has been shown to improve recognition when using a convolutional network [33].

3. Map Transformation Cascade

In this section we describe MTC which was previously proposed in [1]. The model was proposed to retain the functional principles of Neocognitron in a computationally simpler way. MTC is composed by two types of cells arranged hierarchically. Simple cells are responsible for selectivity by reacting to a particular stimulus. Complex cells are responsible for invariance to position of the stimulus. The two types of cells are arranged in layers of the same cell type. Layers are arranged in ordered pairs where the first has simple cells and the second complex cells. The number of pairs of layers can vary. The operation of a network with two pairs of layers as later used in the experiments is illustrated in Fig. 1 and the connectivity between the cells is illustrated in Fig. 2. In a given layer all cells have the same number of afferent connections, except the cells whose receptive field is partly outside the input pattern as defined by the frame parameter. The receptive field part which lays in the frame region provides no afferent connections. All simple and complex layers are unsupervised and their output works as a dictionary to describe the input patterns in a feature space.

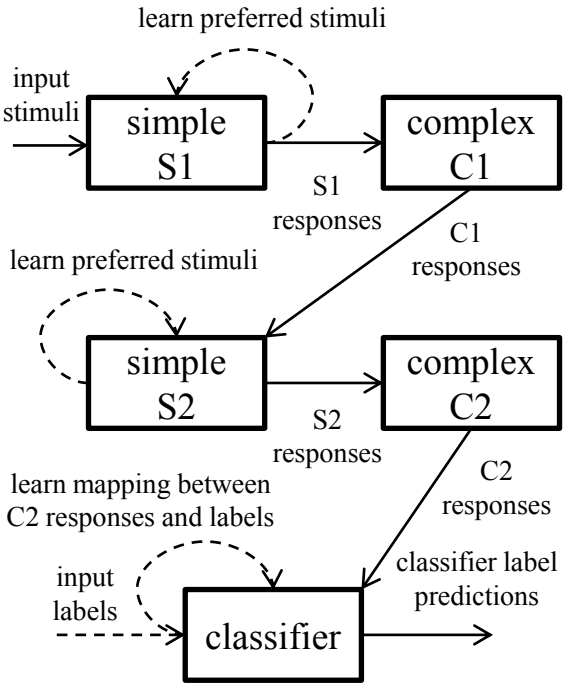

Figure 1: MTC operation — The model operation is sequential. During learning the first simple layer (S1) receives the input stimuli and learns a set of preferred stimuli. Afterwards it responds according to the learnt preferred stimuli. This response is then fed into the first complex layer (C1) which performs a fixed operation and passes its responses to the second simple layer (S2). In S2 a set of preferred stimuli is learnt and then the responses are passed into the second complex layer (C2) which performs a fixed operation producing the C2 responses. The responses of the last layer (C2) are then used to train a classifier. Operations related only to learning are represented by a dashed line and therefore do not apply after the training.

3.1. Simple cell layer

Simple cells react to a particular stimulus in a particular location. Among all cells which have the same receptive field, only the cell whose preferred stimulus is most similar to the current one is active, while other cells are silent. This sparse simple cell response can be biologically implemented through lateral inhibition. In MTC the cells preferred stimulus is learned by K-means clustering. Other unsupervised learning methods can be used. A related model [34] uses a self-organizing map. A biologically plausible model for how simple cells preferred stimulus develops along with orientation columns has been proposed in [35].

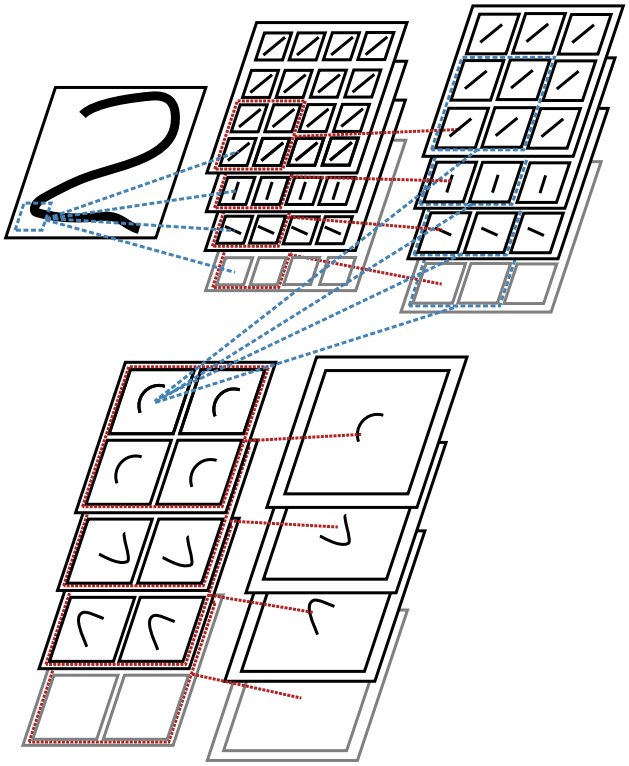

Figure 2: MTC layer connectivity — For a given receptive field in the input pattern there are several simples cells with different preferred stimulus. Several simple cells in contiguous locations reacting to the same preferred stimulus are afferent connections to a complex a cell. The second simple layer receives afferent connections from cells in the first complex layer in contiguous locations across all preferred stimulus. The second complex layer, like the first complex layer, pools over cells in contiguous locations reacting to the same preferred stimulus.

The input pattern is tiled with a squared mask M of size $j \times j$. Each position of the mask, representing a receptive field in the input pattern, results in a sub-pattern and the n sub-patterns $\vec{x}_h$, with $h \in \{1, 2, ..., n\}$, are the input to a clustering algorithm (K-means). The output of the clustering is a set of k preferred stimulus represented by the cluster centers c_v given by $\vec{c}_1, \vec{c}_2, \vec{c}_3, ..., \vec{c}_k$ of the clusters $C_1, C_2, C_3, ..., C_k$, with

$$C_v = \{\vec{x}_h | d(\vec{x}_h, \vec{c}_v) = \min_i d(\vec{x}_h, \vec{c}_i)\}, \tag{1}$$

$$c_v = \frac{1}{|C_v|} \sum_{\vec{x}_h \in C_v} \vec{x}_h. \tag{2}$$

After learning the set of k classes (preferred stimuli) is used to describe the input pattern over the different positions of the mask M. In each position, a mask is applied to the input, and the sub-pattern is compared with the previously learned classes.

During mapping, for each sub-pattern $\vec{x}$, representing a receptive field in the input pattern, the most similar class i is determined according to the Euclidean distance:

$$i = \min_l d(\vec{x}, \vec{c}_l), l \in \{1, ..., k\}. \tag{3}$$

The cell i whose preferred stimulus is more similar to the sub-pattern is active (response is 1), while all other cells with same receptive field are silent (response is 0).

3.2. Complex cell layer

The output of a complex cell layer mask is represented by a vector. As in the simple cell layer, a 'one' stands for a class being active in the corresponding position of the vector; its inactivity is denoted by a 'zero'. The class representation of a pattern is tiled m times with a squared mask M of size $l \times l$. In each position, a vector $\vec{c}_h$, with $h \in \{1, 2, ...m\}$, of dimensions $l \times l$, is determined. The vector $\vec{c}_h$ describes the presence of some classes inside the mask. Complex cells are active if any of its afferent simple cells are active. This is similar to HMAX's maximum response in complex cells.

Each position of the vector $\vec{c}_h$ $(l \times l)$ is represented by a class activity vector $\{1, 2, ..., p\}$ of dimension p, e.g. the presence of the classes $\{1, 4, 3\}$ with k=5 is represented by the vector $\vec{u}$ = [1 0 1 1 0], with ones in the corresponding positions 1, 3, and 4. The result of a transformation of m squared masks M covering a class pattern is a $(m \times p)$-dimensional class activity vector $\vec{U}$. The index 1 to m concerns the position of the mask M. This binary class vector is composed of m activity vectors:

$$\vec{U} = [\vec{u}_1, \vec{u}_2, ..., \vec{u}_m].$$

This is equivalent to a maximum operation. The complex layer operation is therefore predetermined and is not the result of learning.

3.3. Additional Layers

A MTC network can have a variable number of layers (see Fig. 2). The training of the network is performed sequentially, i.e. each layer training is finished before the next layer is trained, starting from the layer closer to the input. If the network has more than one simple layer the output of the first complex layer is then used to train the second simple layer analogously. Each cell in the second and following simple layers has afferent connections from complex cells with different preferred stimulus (e.g. different orientations). After learning, the second simple layer classifies the output of the first stage complex layer and passes it's output to the second complex layer. Only simple layers are modified during learning since complex layers have a predetermined operation. The process is repeated till the last layer is reached.

The response of the last layer can then be used with a classifier for tasks like image recognition.

4. Experiments

In the experiments we evaluate the performance of MTC combined with a linear SVM.

A SVM, as originally proposed, solves a binary classification problem [36]. For the multi-class problem we used the "one-against-one" approach [37, 38]. Therefore we solve a binary classification problem for all the two class combinations, training $k(k-1)/2$ binary classifiers. The output of the binary classifiers is then combined by voting [39]. Another possible approach is the "one-against-all", for a comparison between the two refer to [40]. We use a linear kernel and cost parameter $c = 10$ in all experiments. We use the SVM implementation from [39]. To reduce the cost of the SVM training and due to MTC producing sparse binary vectors we use a linear SVM as it is common for large datasets in text classification. We also empirically observed it achieves similar results to a Gaussian SVM.

To limit the computational cost when training the simple layers, we use a common practice in quantization which is to use a subsample of the entire samples available. When training the simple layer classes using K-means, we take at most $100\times$ more patches than the number of classes K by subsampling or all of the available if less.

We evaluate the performance on the USPS and MNIST datasets (see Section 4.1) and start by finding a parameterization for MTC independently for each dataset (see Section 4.2). We finally evaluate the performance for a

variable number of training samples and compare the results with previous works (see Section 4.3).

4.1. Datasets

The USPS dataset contains gray scale images of size 16×16. The dataset is divided in a training set with 7291 samples and a test set with 2007 samples. The MNIST dataset contains gray scale images of size 28×28. We resize both datasets to 64×64 using bilinear interpolation and scale them to [0:1].

The number of samples of each digit differs (see Table 1) for both datasets. The human error rate for the USPS dataset is estimated at 2.5% [41] while for the MNIST dataset at 0.2% [42]. Several published results for USPS dataset are shown in Table 2 and for the MNIST dataset in Table 3.

	USPS		MNIST	
Digit	Train	Test	Train	Test
0	1194	359	5923	980
1	1005	264	6742	1135
2	731	198	5958	1032
3	658	166	6131	1010
4	652	200	5842	982
5	556	160	5421	892
6	664	170	5918	958
7	645	147	6265	1028
8	542	166	5851	974
9	644	177	5949	1009
Total	7291	2007	60000	10000

Table 1: USPS and MNIST digit samples

4.2. Parameterization

First we performed a random search using cross-validation on the train set to tune the parameters of the network like described in [1]. We fix as in [26] the shift in simple layers (S1,S2) to 1 and in complex layers (C1,C2) to 2, representing a 2:1 thinning out from simple cells to complex cells. The remaining parameters for each layer (see Fig. 3) are the size of the mask and the size of the frame around the input to the layer. Additionally for the simple layers the number of classes k is also a parameter. We find these

Method	Error Rate
nearest neighbor classifier (Euclidean)	5.6%
Relevance Vector Machine [43]	5.1%
Convolutional network (LeNet-1) [44]	4.2%
Support Vector Machine [45]	4.0%
Invariant Support Vectors [46]	3.0%
tangent distance [47]	2.2%
Human [41]	2.5%

Table 2: Other results on USPS.

Method	Error Rate
linear SVM [48]	12.0%
nearest neighbor classifier (Euclidean)	3.09%
Deep Belief Network + linear SVM [48]	1.90%
Convolutional Net LeNet-1 [20]	1.7%
stacked RBM network [24]	1.2%
polynomial SVM [20]	1.1%
Convolutional Net LeNet-4 [20]	1.1%
tangent distance [20]	1.1%
Convolutional Net LeNet-5 [20]	0.95%
large Conv. Net, unsup. pretraining [27]	0.60%
large Conv. Net, unsup. pretraining [30]	0.53%
direction gradient [32]	0.42%
Human [42]	0.2%

Table 3: Other results on MNIST.

remaining parameters by random search. We initialize the random search with $size = 3$, $frame = 1$ for all layers and $classes = 20$ for all simple layers.

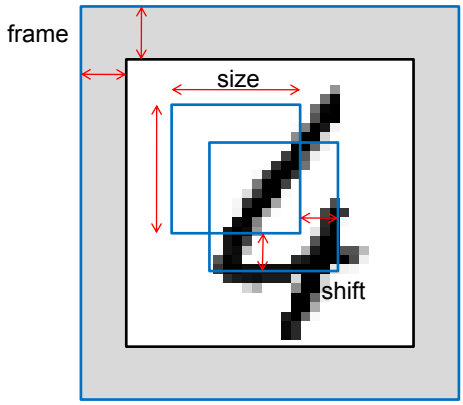

Figure 3: MTC parameter illustration — the smaller squares represent two different positions of mask M. The size refers to the size of the mask, and the shift to distance between different positions of mask M. The frame represented by the gray area determines the extra area without activity which is added to the input patterns, e.g. for the first simple layer this is represent by white background and for the complex layer by no activity in the previous simple layer cells. Additionally for simple layers the number of classes k must also be chosen.

The train and validation sets for each iteration of the optimization were randomly sampled from the entire train set, taking 1000 samples for each. In each iteration the best three parameterizations regarding validation set error rate were used to generate six new parameterizations (two copies from each of three) where each parameter p is updated according to the following equation

$$p_{new} = \begin{cases} \lceil (p_{old} + \epsilon)^{1+\theta r} \rceil & \text{if } r \geq 0 \\ \lfloor (p_{old} + \epsilon)^{1-\theta r} \rfloor & \text{otherwise.} \end{cases} \tag{4}$$

where r is sampled from a uniform distribution $\mathcal{U}(-1, 1)$, θ controls the strength of the change and is set to 0.2 and ϵ is small positive constant which prevents p from settling in 0.

The resulting nine parameterizations are then evaluated and the best three kept. The search ends if the best parameterization is the same for three consecutive iterations. The resulting parameters are shown in Table 4, the number of units results from the parameters. The number of units

Layer	Cell Property	USPS	MNIST
S1	size	6	3
	shift	1	1
	frame	4	2
	classes	20	16
	#units	$67 \times 67 \times 20$	$66 \times 66 \times 16$
C1	size	7	4
	shift	2	2
	frame	2	1
	#units	$33 \times 33 \times 20$	$33 \times 33 \times 16$
S2	size	3	6
	shift	1	1
	frame	2	2
	classes	129	171
	#units	$35 \times 35 \times 129$	$32 \times 32 \times 171$
C2	size	10	10
	shift	2	2
	frame	0	0
	#units	$13 \times 13 \times 129$	$12 \times 12 \times 171$

Table 4: MTC parameters and resulting number of units used in the experiments for each of the datasets.

of layer C2 is therefore the dimensionality of the vectors used for the linear classifier, i.e. 21801 for USPS and 24624 for MNIST.

4.3. Results

We then evaluated the performance of the proposed approach with a linear SVM for a varying number of training samples using the parameterizations from Table 4.

We generated pairs of train and test sets with a different number of samples $10, 50, 200, 1000$ samples (10 of each size). Samples were randomly selected for the generated train and test sets from the respective original sets. The results of the approach for a varying number of training samples are shown in Table 5 for USPS and in Table 6 for MNIST.

It is worth noticing that MTC performs well with few training samples. The error rate for 1000 training examples is 2.22% which compares favorably

to 3.21% in [27].

We finally evaluated the performance of MTC on the entire datasets. For the unsupervised learning of simple cell classes we use on the USPS dataset all train samples and on the MNIST dataset we use 10000 out of the 60000 available to reduce the computational cost. To choose the best preferred stimuli for simple cell layers, we repeat the unsupervised learning 10 times and keep the best by cross-validation on the train set. We randomly pick to train 6291 samples for the USPS and 10000 samples for the MNIST. To validate we randomly pick 1000 samples out of the respective remaining training sets. Finally we take the best unsupervised learning according to validation error to produce the second complex layer (C2) responses for all patterns. We obtain a test error rate of 2.64 % in USPS and 0.71 % in MNIST, which is close to the best results on both datasets and significantly better than several other published results (see Table 2 and 3).

It is also noticeable that this is major improvement (0.71 % vs. 1.90%) over using a deep belief network with a linear SVM [48].

#train	#test	test error rate
100	100	11.90±4.48%
200	200	6.85±1.97%
500	500	5.36±0.67%
1000	1000	4.72±0.70%
7291	2007	2.64%

Table 5: Results on USPS

#train	#test	test error rate
100	100	8.70±3.09%
200	200	5.35±2.01%
500	500	2.88±0.69%
1000	1000	2.22±0.35%
60000	10000	0.71%

Table 6: Results on MNIST

5. Conclusion

We evaluated the combination of MTC with a linear classifier. MTC showed good generalization for a small number of training examples. The combination of MTC and a linear SVM achieved competitive results on both USPS (2.64%) and MNIST (0.71%) datasets. MTC greatly improves the results relatively to using a deep belief network with a linear SVM [48]. It is also interesting that in [27] quasi-binary codes are unsuitable for classification, while the MTC binary codes can be used for classification with competitive results using a linear classifier.

Acknowledgments. The authors would like to thank João Sacramento for much helpful comments. This work was supported by Fundação para a Ciência e Tecnologia (INESC-ID multiannual funding) through the PIDDAC Program funds and through an individual doctoral grant awarded to the first author (contract SFRH/BD/61513/2009).

[1] Â. Cardoso, A. Wichert, Neocognitron and the Map Transformation Cascade, Neural Networks 23 (2010) 74 – 88.

[2] D. Hubel, T. Wiesel, Receptive fields and functional architecture in two nonstriate visual areas (18 and 19) of the cat, Journal of Neurophysiology 28 (1965) 229.

[3] K. Fukushima, Neocognitron: a self organizing neural network model for a mechanism of pattern recognition unaffected by shift in position., Biological Cybernetics 36 (1980) 193–202.

[4] M. Riesenhuber, T. Poggio, Hierarchical models of object recognition in cortex, Nature Neuroscience 2 (1999) 1019 – 1025.

[5] T. Serre, A. Oliva, T. Poggio, A feedforward architecture accounts for rapid categorization, Proceedings of the National Academy of Sciences USA 104 (2007) 6424.

[6] D. Hubel, J. Wensveen, B. Wick, Eye, brain, and vision, Scientific American Library New York, 1988.

[7] D. H. Hubel, T. N. Wiesel, Uniformity of monkey striate cortex: A parallel relationship between field size, scatter, and magnification factor., The Journal of Comparative Neurology 158 (1974) 295.

[8] E. Kobatake, K. Tanaka, Neuronal selectivities to complex object features in the ventral visual pathway of the macaque cerebral cortex, Journal of Neurophysiology 71 (1994) 856.

[9] H. Bülthoff, S. Edelman, Psychophysical support for a two-dimensional view interpolation theory of object recognition, Proceedings of the National Academy of Sciences USA 89 (1992) 60–64.

[10] M. C. Booth, E. T. Rolls, View-invariant representations of familiar objects by neurons in the inferior temporal visual cortex., Cerebral Cortex 8 (1998) 510.

[11] N. Logothetis, J. Pauls, T. Poggio, Shape representation in the inferior temporal cortex of monkeys, Current Biology 5 (1995) 552–563.

[12] D. H. Hubel, T. N. Wiesel, Sequence regularity and geometry of orientation columns in the monkey striate cortex, The Journal of Comparative Neurology 158 (1974) 267 – 294.

[13] C. Blakemore, E. A. Tobin, Lateral inhibition between orientation detectors in the cat's visual cortex, Experimental Brain Research 15 (1972) 439 – 440.

[14] K. Fukushima, Cognitron: A self-organizing multilayered neural network, Biological Cybernetics 20 (1975) 121–136.

[15] M. Riesenhuber, How a Part of the Brain Might or Might Not Work: A New Hierarchical Model of Object Recognition, Ph.D. thesis, Citeseer, 2000.

[16] K. Fukushima, Neocognitron: A hierarchical neural network capable of visual pattern recognition, Neural networks 1 (1988) 119 – 130.

[17] K. Fukushima, Increasing robustness against background noise: Visual pattern recognition by a neocognitron, Neural Networks 24 (2011) 767 – 778.

[18] T. Serre, M. Riesenhuber, Realistic Modeling of Simple and Complex Cell Tuning in the HMAXModel, and Implications for Invariant Object Recognition in Cortex, Massachusetts Institute of Technology Computer Science and Artificial Intelligence Laboratory (2004).

[19] R. Miikkulainen, J. Bednar, Y. Choe, J. Sirosh, Computational Maps in the Visual Cortex, Springer, 2005.

[20] Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (1998) 2278 – 2324.

[21] S. Lazebnik, C. Schmid, J. Ponce, Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, in: Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, volume 2, Ieee, pp. 2169 – 2178.

[22] D. Lowe, Towards a computational model for object recognition in IT cortex, in: Biologically Motivated Computer Vision, Springer, 2000, pp. 141 – 155.

[23] G. E. Hinton, S. Osindero, Y. W. Teh, A fast learning algorithm for deep belief nets, Neural computation 18 (2006) 1527 – 1554.

[24] H. Larochelle, Y. Bengio, J. Louradour, P. Lamblin, Exploring Strategies for Training Deep Neural Networks, Journal of Machine Learning Research 1 (2009) 1 – 40.

[25] K. Fukushima, N. Wake, Handwritten alphanumeric character recognition by the neocognitron, IEEE Transactions on Neural Networks 2 (1991) 355–365.

[26] K. Fukushima, Neocognitron for handwritten digit recognition., Neurocomputing 51 (2003) 161–180.

[27] M. A. Ranzato, F. J. Huang, Y. L. Boureau, Y. Lecun, Unsupervised learning of invariant feature hierarchies with applications to object recognition, in: Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on, Ieee, pp. 1 – 8.

[28] J. Weston, F. Ratle, R. Collobert, Deep learning via semi-supervised embedding, in: Proceedings of the 25th international conference on Machine learning, ACM, 2008,pp. 1168 – 1175.

[29] J. Mutch, D. G. Lowe, Object class recognition and localization using sparse features with limited receptive fields, International Journal of Computer Vision 80 (2008) 45 – 57.

[30] K. Jarrett, K. Kavukcuoglu, M. A. Ranzato, Y. LeCun, What is the best multi-stage architecture for object recognition?, in: Computer Vision, 2009 IEEE 12th International Conference on, IEEE, pp. 2146 – 2153.

[31] K. Kavukcuoglu, Y. Lecun, Fast inference in sparse coding algorithms with applications to object recognition, in: Technical report, Computational and Biological Learning Lab, Courant Institute, NYU

[32] C. L. Liu, K. Nakashima, H. Sako, H. Fujisawa, Handwritten digit recognition: Benchmarking of state-of-the-art techniques, Pattern Recognition 36 (2003) 2271 – 2285.

[33] P. Y. Simard, D. Steinkraus, J. C. Platt, Best practices for convolutional neural networks applied to visual document analysis, in: Document Analysis and Recognition, 2003. Proceedings. Seventh International Conference on, IEEE, pp. 958 – 963.

[34] A. Wichert, MTCn-nets, in: E. Lawrence (Ed.), Proceedings World Congress on Neural Networks, volume IV, 1993, pp. 59–62.

[35] J. Sirosh, R. Miikkulainen, Cooperative self-organization of afferent and lateral connections in cortical maps, Biological Cybernetics 71 (1994) 65 – 78.

[36] V. Vapnik, The Nature of Statistical Learning Theory, Springer, New York, 1995.

[37] J. Friedman, Another approach to polychotomous classification, 1996.

[38] U. H.-G. Kreßel, Pairwise classification and support vector machines, in: B. Schölkopf, C. Burges, A. Smola (Eds.), Advances in Kernel Methods: Support Vector Learning, MIT Press, Cambridge, MA, 1999, pp. 255–268.

[39] C.-C. Chang, C.-J. Lin, LIBSVM: A library for support vector machines, ACM Transactions on Intelligent Systems and Technology 2 (2011) 27:1–27:27.

[40] C. Hsu, C. Lin, A comparison of methods for multiclass support vector machines, IEEE transactions on Neural Networks 13 (2002) 415–425.

[41] P. Simard, Y. LeCun, J. S. Denker, Efficient pattern recognition using a new transformation distance, in: Advances in Neural Information Processing Systems 5, Morgan Kaufmann, 1993, pp. 50–58.

[42] Y. LeCun, L. Jackel, L. Bottou, A. Brunot, C. Cortes, J. Denker, H. Drucker, I. Guyon, U. Muller, E. Sackinger, et al., Comparison Of Learning Algorithms For Handwritten Digit Recognition, in: International Conference on Artificial Neural Networks, pp. 53 – 60.

[43] M. E. Tipping, The relevance vector machine., in: S. A. Solla, T. K. Leen, K.-R. MAller (Eds.), Advances in Neural Information Processing Systems 12, The MIT Press, 2000, pp. 652–658.

[44] P. Simard, Y. LeCun, J. Denker, B. Victorri, Transformation invariance in pattern recognition, tangent distance and tangent propagation, in: G. Orr, M. K. (Eds.), Neural Networks: Tricks of the trade, Springer, 1998.

[45] B. Schölkopf, C. Burges, V. Vapnik, Extracting support data for a given task, in: Proceedings, First International Conference on Knowledge Discovery & Data Mining. AAAI Press, Menlo Park, CA, 1995.

[46] B. Schölkopf, P. Simard, A. Smola, V. Vapnik, Prior knowledge in support vector kernels, in: Advances in neural information processing systems 10, MIT Press, 1998, pp. 640–646.

[47] D. Keysers, J. Dahmen, T. Theiner, H. Ney, Experiments with an extended tangent distance, in: Pattern Recognition, 2000. Proceedings. 15th International Conference on, volume 2, pp. 38 –42 vol.2.

[48] K. Yu, T. Zhang, Y. Gong, Nonlinear learning using local coordinate coding, in: Advances in Neural Information Processing Systems 22, 2009, pp. 2223 – 2231.