Application of Support Vector Machines, Convolutional Neural Networks and Deep Belief Networks to Recognition of Partially Occluded Objects

Joseph Lin Chu and Adam Krzyżak

Department of Computer Science and Software Engineering
Concordia University
1455 de Maisonneuve Blvd. West, Montreal, Quebec, Canada H3G 1M8
jo_chu@encs.concordia.ca, krzyzak@cs.concordia.ca

Abstract. Artificial neural networks have been widely used for machine learning tasks such as object recognition. Recent developments have made use of biologically inspired architectures, such as the Convolutional Neural Network, and the Deep Belief Network. We test the hypothesis that generative models such as the Deep Belief Network should perform better on occluded object recognition tasks than purely discriminative models such as Convolutional Neural Networks. We find that the data does not support this hypothesis when the generative models are run in a partially discriminative manner. We also find that the use of Gaussian visible units in a Deep Belief Network trained on occluded image data allows it to also learn to classify non-occluded images. [1]

1 Introduction

Partially occluded object recognition has historically been a challenging task. Most methods of solving this problem rely on complex preprocessing and feature extraction algorithms, often involving image segmentation and other extra processing [16] [22] [23] [24]. More recently, techniques involving the use of generative model reconstructions have been proposed [18].

Convolutional Neural Networks (CNNs) are feed-forward Artificial Neural Networks (ANNs), while Deep Belief Networks (DBNs) make use of Restricted Boltzmann Machines (RBMs) that use recurrent connections. The fundamental difference between these networks then, is that the DBN is capable of functioning as a generative model, whereas a CNN is merely a discriminative model. A generative model is able to model all variables probabilistically and therefore to generate values for any of these variables. In that sense it can do things like reproduce samples of the original input. A discriminative model on the other hand models only the dependence of an unobserved variable on an observed

[1]This research was supported by the Natural Sciences and Engineering Research Council of Canada

variable, which is sufficient to perform classification or prediction tasks, but which cannot reproduce samples like a generative model can. This suggests that DBNs should perform better on the task of partially occluded object recognition, as they ought to be able to use their generative effects to partially reconstruct the image to aid in classification. This hypothesis is what we wish to test in our work comparing CNNs, and DBNs.

2 Learning Algorithms

In order to contrast the effectiveness of generative models with discriminative models on the occluded object recognition task, we compared several models of ANN, as well as other machine learning algorithms, including: the Support Vector Machine (SVM), the CNN, (two discriminative models) and the DBN, (one generative model). Although the SVM is not a proper ANN strictly speaking, its popularity as a discriminative classifier means that it deserves inclusion as a control.

2.1 Support Vector Machine

The SVM is a powerful discriminant classifier first developed by Cortes & Vapnik [4]. Although technically not considered to be an ANN, Collobert & Bengio [3] showed that they had many similarities to Perceptrons with the obvious exception of learning algorithm.

2.2 Convolutional Neural Networks

The earliest of the hierarchical ANNs based on the visual cortex's architecture was the Neocognitron, first proposed by Fukushima & Miyake [6]. This network was based on the work of neuroscientists Hubel & Wiesel [13], who showed the existence of Simple and Complex Cells in the visual cortex. Fukushima took the notion of Simple and Complex Cells to create the Neocognitron, which implemented layers of such neurons in a hierarchical architecture [5].

Then LeCun et al [14] developed the CNN, which made use of multiple Convolutional and Subsampling layers, while also using stochastic gradient descent and backpropagation to create a feed-forward network that performed exceptionally well on image recognition tasks such as the MNIST. The Convolutional Layer of the CNN is equivalent to the Simple Cell Layer of the Neocognitron, while the Subsampling Layer of the CNN is equivalent to the Complex Cell Layer of the Neocognitron. Essentially they delocalize features from the visual receptive field, allowing such features to be identified with a degree of shift invariance. This unique structure allows the CNN to have two important advantages over a fully-connected ANN. First, is the use of the local receptive field, and second is weight-sharing. Both of these advantages have the effect of decreasing the number of weight parameters in the network, thereby making computation of these networks easier.

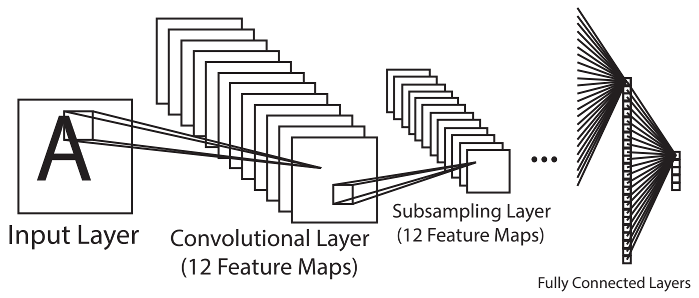

Fig. 1. The basic architecture of the CNN.

2.3 Deep Belief Networks

One of the more recent developments in machine learning research has been the Deep Belief Network (DBN). The DBN is a recurrent ANN with undirected connections. Structurally, it is made up of multiple layers of RBMs, such that it can be seen as a 'deep' architecture. To understand how this is an effective structure, we must first understand the basic nature of a recurrent ANN.

Recurrent ANNs differ from feed-forward ANNs in that their connections can form cycles. The advantage of recurrent ANNs is that they can possess associative memory-like behaviour. Early Recurrent ANNs, such as the Hopfield network [11], showed promise in this regard, but were limited. The Hopfield network was only a single layer architecture that could only learn very limited problems due to limited memory capacity. A multi-layer generalization of the Hopfield Network was developed known as the Boltzmann Machine [1], which while able to store considerably more memory, suffered from being overly slow to train.

A variant of the Boltzmann Machine was first known as a Harmonium [21], but later called a RBM, which initially saw little use. Then Hinton [7] developed a fast learning algorithm for RBMs called Contrastive Divergence, which uses Gibbs sampling within a gradient descent process. The RBM is primarily different from a regular Boltzmann Machine by the simple fact that it lacks the lateral or sideways connections within layers.

By stacking RBMs together, Hinton, Osindero, & Teh, [9] created the DBN. The DBN is trained in a greedy, layer-wise fashion. This generally involves pre-training each RBM separately starting at the bottom layer and working up to the top layer. All layers have their weights initialized using unsupervised learning in the pre-training phase, after which fine-tuning using Backpropagation is performed using the labeled data, training in a supervised manner.

When introduced, the DBN produced then state of the art performance on such tasks as the MNIST. Later DBNs were also applied to 3D object recognition [17]. Ranzato, Susskind, Mnih, & Hinton [18] also showed how effective DBNs could be on occluded facial images.

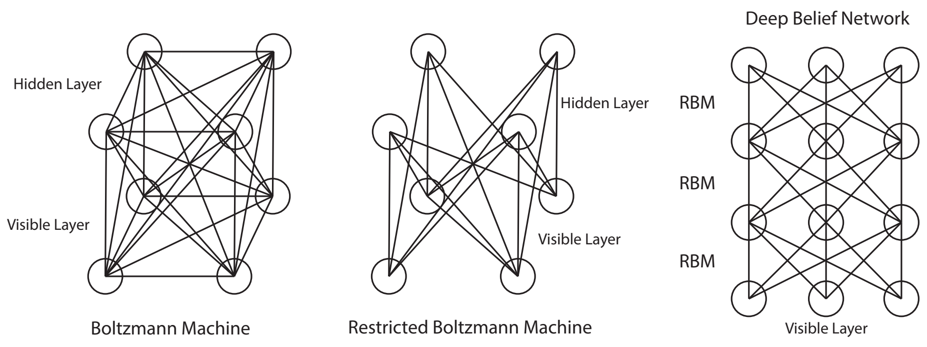

Fig. 2. The structure of the general Boltzmann Machine, the RBM, and the DBN.

3 Methodology

For the object/image dataset, the small NORB [15] was used. The small NORB consists of 5 object categories and several thousand images per category, for a total of 24300 images each in the training and test sets. The small NORB proper includes a pair of stereo images for each training example, but we chose to only use one of the images in the pair. Normal, non-occluded images with the object fully visible in the image are seen in Figure 3. Occluded images were created by occluding a random half of each image in the test set with zeroes (black) as shown in Figure 4.

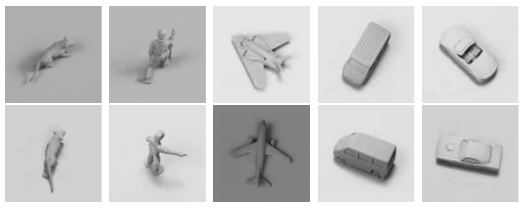

Fig. 3. Images from the small NORB non-occluded data set.

For the SVMs we tested various parameters from the literature, such as Huang & LeCun [12] and Ranzato et al. [19] and eventually settled on a Gamma value of 0.0005, and a C value of 40. Gamma is how far a single training example affects things, with low values being "far" and high values being "close". C is the tradeoff between misclassifying as few training samples as possible (high C) and a smooth decision surface (low C). For code for the SVMs, we used the library "LIBSVM" by Chih-Chung Chang and Chih-Jen Lin from the National Taiwan University [2].

For the CNN, Sirotenko's Matlab library "CNN Convolutional neural network class" (http://www.mathworks.com/matlabcentral/fileexchange/24291-cnn-

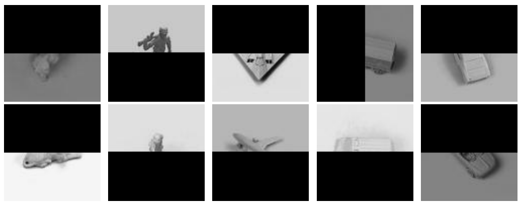

Fig. 4. Images from the small NORB occluded data set.

convolutional-neural-network-class) was used and modified extensively to serve our purposes. Determining the architecture of a CNN requires special considerations. To calculate the reasonable dimensions of a square layer from either its previous layer (or next layer) in the hierarchy requires at least some of the following variables to be assigned. Let x be the width of the previous (or current) square layer. Let y be the width of the current (or next) square layer. Let r be the width of the square receptive field of nodes in the previous (or current) layer to each current (or next) layer node, and f be the offset distance between the receptive fields of adjacent nodes in the current (or next) layer. The relationship between these variables is best described by

$$y = \frac{x - (r - f)}{f} \tag{1}$$

where, $x \geq y$, $x \geq r \geq f$, and $f > 0$.

For convolutional layers $f = 1$ and (1) generalizes to

$$y = x - r + 1 \tag{2}$$

For subsampling layers $r = f$, and thus (1) generalizes to

$$y = \frac{x}{f} \tag{3}$$

From this we can determine the dimensions of each layer. The architecture for the CNN on the NORB dataset is shown in Table 1, where S, C and F represent convolutional, subsampling and fully connected layers, respectively.

Various parameters for the CNN were also experimented with to determine the optimal parameters to use in our experiments. We eventually settled on 100 epochs of training. The CNN learning rate and learning rate decrement parameters were determined by using Huang and LeCun's recommendations [12]. That is to say, the learning rate was initially set to 2.00E-05, and gradually decremented to approximately 2.00E-07.

For the DBN we used Stansbury's Matlab library "Matlab Environment for Deep Architecture Learning (MEDAL)" (https://github.com/dustinstansbury/medal). Experiments were also conducted on the parameters for the DBN. By default, DBNs use binary visible units. A modification has been suggested to use Gaussian visible units for image data [10]. DBNs using both binary and Gaussian visible units were tested.

Table 1. The architecture of the CNN used on the NORB dataset, based on Huang & LeCun [12].

CNN			
Layer	**Nodes**	**k or r**	**Feature Maps**
S1	96x96		
C2	92x92	5	8
S3	23x23	4	8
C4	18x18	6	24
S5	6x6	3	24
C6	1x1	6	24
F1	100	1	
F2	5	1	

Two different amounts of hidden nodes were used, 2000 and 4000 respectively for the binary units. This was because prior experiments used to determine the effectiveness of various parameter configurations found that the binary units in combination with 2000 hidden nodes seemed to actually perform better than the combination of binary units and 4000 hidden nodes, which was different than expected. Gaussian units on the other hand, showed greater effectiveness at 4000 hidden nodes, than at 2000 hidden nodes, which was expected. For this reason, we tested multiple configurations. Eventually, through systematic efforts involving testing various parameters at different values and looking at the change in performance, we settled on the Layer, Learning Rate, and Epoch parameters for the Visible and Hidden Node cases shown in Table 2. Hinton also provided some suggested values that we took into consideration [8].

Table 2. The parameters chosen as an optimal configuration for the DBNs.

Parameters - DBN						
Parameters			Learning Rate		Epochs	
Visible	Layers	Hidden	Pre-Training	Fine-Tune	Pre-Training	Fine-Tune
Binary	2	2000	0.1	0.01	200	50
Binary	2	4000	0.1	0.01	200	50
Gaussian	2	4000	0.001	0.001	200	50

Some more parameters we settled on are shown in Table 3, some of which were based on experimentation, while others were simply default settings that worked well. Details about the various parameters are described by Hinton [8].

Finally, experiments were performed with the optimized parameters for SVMs, CNNs, and DBNs on the small NORB image dataset. Each of the training and testing sets consisted of 24300 images. These experiments consisted of three different methods of training: one which consisted of training exclusively on the non-occluded training set, followed by testing on both a non-occluded test set and an occluded test set; one which consisted of training exclusively on the oc-

Table 3. The parameters chosen as an optimal configuration for the DBNs.

Parameters - DBN	
Momentum	0.5
Weight Penalty	2.00E-04
Batch Size	100
Begin Simulated Annealing At	50
Number of Gibbs Sampling	1
Sparsity	0.01
Start to Vary Learning Rate At	50

cluded training set, followed by testing on both a non-occluded test set and an occluded test set; and finally one which consisted of training on a mixture of non-occluded and occluded images, followed by testing on both a non-occluded test set and an occluded test set. Three replicates were performed for each experimental setup and averaged.

4 Results

4.1 Support Vector Machines

Table 4 provides a direct comparison of the non-occluded, occluded, and mixed trained SVMs.

Table 4. A comparison of the accuracy results of the non-occluded, occluded, and mixed trained SVMs.

SVM - NORB				
Training	Training Test	Mixed Test	Non-Occluded Test	Occluded Test
Non-Occluded	**0.999 ± 0.003**	0.513 ± 0.001	**0.825 ± 0.007**	0.200 ± 0.003
Occluded	0.994 ± 0.0001	0.446 ± 0.0002	0.200 ± 0.0001	0.692 ± 0.0005
Mixed	0.973 ± 0.0003	**0.754 ± 0.001**	0.813 ± 0.001	**0.694 ± 0.0005**

Note: Mean of 3 replicates ± standard error.

4.2 Convolutional Neural Networks

Table 5 provides a direct comparison of the non-occluded, occluded, and mixed trained CNNs.

4.3 Deep Belief Networks

Tables 6-8 provide a direct comparison of the non-occluded, occluded, and mixed trained DBNs, with the differences between each table resulting from the effects of choosing different visible units and number of hidden units in the ANN.

Table 5. A comparison of the accuracy results of the non-occluded, occluded, and mixed trained CNNs.

CNN - NORB				
Training	Training Test	Mixed Test	Non-Occluded Test	Occluded Test
Non-Occluded	**0.955 ± 0.000**	0.515 ± 0.000	**0.831 ± 0.000**	0.199 ± 0.000
Occluded	0.693 ± 0.003	0.444 ± 0.017	0.304 ± 0.031	0.585 ± 0.002
Mixed	0.832 ± 0.002	**0.717 ± 0.003**	0.769 ± 0.009	**0.665 ± 0.010**

Note: Mean of 3 replicates ± standard error.

Table 6. A comparison of the accuracy results of the non-occluded, occluded, and mixed trained DBNs using binary visible units with 2000 hidden nodes.

DBN - Binary Visible Unit w/ 2000 Hidden Nodes				
Training	Training Test	Mixed Test	Non-Occluded Test	Occluded Test
Non-Occluded	**0.993 ± 0.0002**	0.545 ± 0.000	**0.873 ± 0.007**	0.214 ± 0.004
Occluded	0.847 ± 0.007	0.451 ± 0.026	0.193 ± 0.044	**0.708 ± 0.009**
Mixed	0.832 ± 0.013	**0.680 ± 0.013**	0.676 ± 0.037	0.684 ± 0.020

Note: Mean of 3 replicates ± standard error.

Table 6 shows specifically the performance of the DBNs using binary visible units and having 2000 hidden nodes.

Table 7 shows specifically the performance of the DBNs using binary visible units and having 4000 hidden nodes.

Table 7. A comparison of the accuracy results of the non-occluded, occluded, and mixed trained DBNs using binary visible units with 4000 hidden nodes.

DBN - Binary Visible Unit w/ 4000 Hidden Nodes				
Training	Training Test	Mixed Test	Non-Occluded Test	Occluded Test
Non-Occluded	**0.989 ± 0.002**	0.520 ± 0.008	**0.841 ± 0.014**	0.203 ± 0.002
Occluded	0.852 ± 0.007	0.458 ± 0.014	0.208 ± 0.022	**0.708 ± 0.006**
Mixed	0.866 ± 0.008	**0.673 ± 0.001**	0.653 ± 0.004	0.693 ± 0.004

Note: Mean of 3 replicates ± standard error.

Table 8 shows specifically the performance of the DBNs using Gaussian visible units and having 4000 hidden nodes.

5 Discussion

The experiments performed have shown that when training a classifier on only the non-occluded training set, the occluded task is a particularly challenging one for both the discriminative models, such as SVMs and CNNs, and the generative models, namely the DBNs. In general, training on the non-occluded images tends to lead to good performance on the non-occluded test set, but poor performance

Table 8. A comparison of the accuracy results of the non-occluded, occluded, and mixed trained DBNs using Gaussian visible units with 4000 hidden nodes.

DBN - Gaussian Visible Unit w/ 4000 Hidden Nodes				
Training	Training Test	Mixed Test	Non-Occluded Test	Occluded Test
Non-Occluded	**0.981 ± 0.003**	0.550 ± 0.002	**0.832 ± 0.006**	0.258 ± 0.013
Occluded	0.786 ± 0.001	0.673 ± 0.002	0.693 ± 0.006	0.652 ± 0.005
Mixed	0.860 ± 0.016	**0.697 ± 0.023**	0.714 ± 0.044	**0.679 ± 0.006**

Note: Mean of 3 replicates ± standard error.

on the occluded test set, while in most cases, training on the occluded images leads to good performance on the occluded test set, and poorer performance on the non-occluded test set.

However, it appears that training on the occluded training set only, for DBNs using Gaussian visible units at least, produces a highly unusual result of good performance on the non-occluded test set (69% accuracy). This behaviour is not apparent with the DBN using binary visible units (19-21% accuracy). A much less pronounced but similar effect is also visible with the CNN (30% accuracy), which is not seen at all with SVM, which performs at chance (20% accuracy). It may be that this is because the SVM is a purely discriminative model. The CNN while also a discriminative model, is also an ANN, which gives it some similarity to the DBN. Nevertheless, the unexpectedly good performance of the Gaussian visible unit based DBN on the dataset type it wasn't trained on is something perhaps worth looking into for future research. Though this seems to come at a cost to performance on occluded test set, as it is the only classifier that performs better on the dataset type it wasn't trained on (69% accuracy), than on the type it was trained on (65% accuracy).

Training the SVM, the CNN, and the DBN with Gaussian visible units on a mixed training set containing both non-occluded and occluded images leads to slightly worse performance on the non-occluded test set than an exclusively non-occluded trained classifier, and slightly better performance on the occluded test set than an exclusively occluded trained classifier. This result suggests that mixed training actually improves performance on the occluded problem. It is possible that these classifiers are benefiting from the more complete images in the non-occluded part of the training set.

Training a DBN with binary visible units on a mixed training set containing both non-occluded and occluded images performs worse on the non-occluded test set than a pure non-occluded training set, and is worse but is very close in performance on the occluded test set to that trained on a pure occluded training set. This is expected, as a mixed training set should yield mediocre performance on both test sets compared to classifiers trained exclusively on the non-occluded or the occluded training sets.

While training a SVM, CNN, and a DBN with Gaussian visible units on a mixed training set leads to better relative performance on the non-occluded test image set than on the occluded test image set, the reverse appears to be the case with DBNs with binary visible units, which had better relative performance on the occluded test image set than on the non-occluded test image set. This is somewhat curious, and may be indicative of the differences between binary and Gaussian visible units.

In comparison to other work in the literature, the experiments performed on the SVM and CNN did not exceed the performance of the results from Huang and LeCun [12]. Huang and LeCun were able to achieve 88.4% accuracy with their SVM on the small NORB dataset, and 93.8% accuracy with their CNN on the small NORB dataset [12]. The SVM in our experiments, with the same parameters as Huang and LeCun [12], achieved 82.5% $\pm$ 0.7% accuracy, while our CNN achieved 83.1% accuracy. Our best performing algorithm was actually a DBN using binary visible units and 2000 hidden nodes, which achieved 87% accuracy. In comparison, Nair and Hinton [17], achieved 93.5% accuracy with their DBN on the standard small NORB dataset, and 94.8% accuracy with their DBN using extra unlabeled data. Thus, on the non-occluded images, we did not achieve quite as good results as the best in the literature.

A major reason for our relatively inferior performance was that we chose to only take one of the two stereo images in the NORB dataset to be used by our algorithms. The top performing results in the literature on the other hand, generally made use of both of the stereo images. We chose not to use the stereo pair images primarily because of limitations on our part, namely that it would double the size of the dataset in memory, and that in the case of the CNN it would require a considerable modification to the architecture of the network. Thus, we chose to save both memory and time by using only the single image. This was an important choice, because we were limited in the amount of RAM available on our computers, and the amount of time to required to train with even this limited version of the NORB was quite substantial. Also, in reality it often difficult to obtain stereo images without resorting to some special robotic vision setup. Conversely, single images are readily available in many datasets, CCTV cameras, and Internet searches.

As far as occluded images are concerned, there is a lack of results in the literature that are directly comparable to our work. Probably the most similar work done so far would be Ranzato et al. [18]. Their work on classifying facial expressions includes some use of occlusion. Rather than using NORB, they used the Cohn-Kanade (CK) dataset, and the Toronto Face Database (TFD), classifying 7 different facial expressions, rather than 5 objects. Their Type 3 - right half, Type 4 - bottom half, and Type 5 - top half occlusions are most similar to the occlusions we used in our experiments. Unlike our experiments, their deep generative model actually attempts to reconstruct the image first before classifying. This takes full advantage of the unique properties of generative models. As such, they achieve fairly impressive results.

Overall their results in combination with our own results appear to show that the advantage of using a generative model comes from the reconstruction process that Ranzato et al. [18] were able to use, and is not simply a result of classification using a generative model discriminatively as we did. Further research naturally could involve actually implementing some kind of reconstruction process similar to what Ranzato et al. [18] used, except on the small NORB dataset, to see whether or not this conjecture actually holds.

A further possible reason why the performance of the generative DBN did not exceed the discriminative models could be because the DBNs were fine-tuned with Backpropagation. As this process is inherently discriminative rather than generative, the final resulting network perhaps behaves more like a discriminative model than a generative model. If this is the case, we should be able to see some difference in the accuracy of the model when it has only been pre-trained, and not yet fine-tuned with Backpropagation. To truly test this possibility, we may need to find a generative model that is fully generative through and through, such as a Deep Boltzmann Machine (DBM) [20].

6 Conclusions

It thus appears that the original hypothesis that the generative models would perform significantly better on the occluded task than the discriminative models is not well supported by the results of the experiments performed. Rather, when run in a discriminative manner, the generative model, in our case, the DBN appears to perform close to equally well to the discriminative models, the SVM and the CNN. This suggests that, with regards to other findings in the literature which use generative models and are able to show a difference, that this difference is primarily due to the additional use of reconstruction processes, and is not due to merely the architecture and training algorithm itself.

On the other hand, with regards to DBNs using Gaussian visible units, when trained on the occluded training set and tested on the non-occluded dataset, show remarkable performance that perhaps warrants further research. In fact, this may suggest that intentionally occluding data sets may allow for good performance on both the non-occluded and occluded tasks, at least when using this particular variant of DBN. Such could prove useful in tasks in which the original training set is non-occluded, but the real-world test data may well be occluded, such as in the case of real-world face recognition from CCTV cameras.

References

1. Ackley, D.H., Hinton, G.E., Sejnowski, T.J.: A learning algorithm for boltzmann machines. Cognitive Science 9, 147–169 (1985)
2. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2, 27:1–27:27 (2011), software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm
3. Collobert, R., Bengio, S.: Links between perceptrons, mlps and svms. Proceedings of the 21st International Conference on Machine Learning (ICML) p. 23 (2004)
4. Cortes, C., Vapnik, V.N.: Support-vector networks. Machine Learning 20, 273–297 (1995)
5. Fukushima, K.: Neocognitron for handwritten digit recognition. Neurocomputing 51, 161–180 (2003)

6. Fukushima, K., Miyake, S.: Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position. Pattern Recognition 15(6), 455–469 (1982)
7. Hinton, G.E.: Training products of experts by minimizing contrastive divergence. Neural Computation 14(8), 1771–1800 (2002)
8. Hinton, G.E.: A practical guide to training restricted boltzmann machines. Momentum 9(1), 599–619 (2010)
9. Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. Neural Computation 18, 1527–1554 (2006)
10. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science 313, 504–507 (2006)
11. Hopfield, J.J.: Neural networks and physical systems with emergent collective computational abilities. Proceedings of the National Academy of Sciences of the USA 79(8), 2554–2558 (1982)
12. Huang, F.J., LeCun, Y.: Large-scale learning with svm and convolutional nets for generic object categorization. Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) 1, 284–291 (2006)
13. Hubel, D.H., Wiesel, T.N.: Receptive fields, binocular interaction and functional architecture in a cats visual cortex. Journal of Physiology (London) 160, 106–154 (1962)
14. LeCun, Y., Bottou, L., Bengio, Y., P., H.: Gradient-based learning applied to document recognition. Proceedings of the IEEE 86(11), 2278–2324 (1998)
15. LeCun, Y., Huang, F., Bottou, L.: Learning methods for generic object recognition with invariance to pose and lighting. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) 2, 97–104 (2004)
16. Martnez, A.M.: Recognizing imprecisely localized, partially occluded, and expression variant faces from a single sample per class. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(6), 748–763 (2002)
17. Nair, V., Hinton, G.E.: 3d object recognition with deep belief nets. Advances in Neural Information Processing Systems (NIPS) pp. 1339–1347 (2009)
18. Ranzato, M., Susskind, J., Mnih, V., Hinton, G.: On deep generative models with applications to recognition. 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 2857–2864 (2011)
19. Ranzato, M.A., Huang, F.J., Boureau, Y.L., LeCun, Y.: Unsupervised learning of invariant feature hierarchies with applications to object recognition. 2007 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 1–8 (2007)
20. Salakhutdinov, R., Hinton, G.E.: Deep boltzmann machines. International Conference on Artificial Intelligence and Statistics (AISTATS) pp. 448–455 (2009)
21. Smolensky, P.: Information processing in dynamical systems: Foundations of harmony theory. In: Rumelhart, D.E., McLelland, J.L. (eds.) Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol. 1, chap. 6, pp. 194–281. MIT Press (1986)
22. Tsang, P.W.M., Yuen, P.C.: Recognition of partially occluded objects. IEEE Transactions on Systems, Man and Cybernetics 23(1), 228–236 (1993)
23. Winn, J., Shotton, J.: The layout consistent random field for recognizing and segmenting partially occluded objects. 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) 1, 37–44 (2006)
24. Wiskott, L., Malsburg, C.V.D.: A neural system for the recognition of partially occluded objects in cluttered scenes: A pilot study. International Journal of Pattern Recognition and Artificial Intelligence 7(4), 935–948 (1993)