Self Learning Machines using Deep Networks

Ahmad A. Al Sallab
Department of Electronics and Communications
Cairo University
Cairo, Egypt
ahmad.elsallab@gmail.com

Mohsen A. Rashwan
Department of Electronics and Communications
Cairo University
Cairo, Egypt
mrashwan@rdi-eg.com

Self learning machines as defined in this paper are those learning by observation under limited supervision, and continuously adapt by observing the surrounding environment. The aim is to mimic the behavior of human brain learning from surroundings with limited supervision, and adapting its learning according to input sensory observations. Recently, Deep Belief Nets (DBNs) [1] have made good use of unsupervised learning as pre-training stage, which is equivalent to the observation stage in humans. However, they still need supervised training set to adjust the network parameters, as well as being non-adaptive to real world examples. In this paper, Self Learning Machine (SLM) is proposed based on deep belief networks and deep auto encoders

I. INTRODUCTION

Recognition in human brains develops first by unsupervised observation of surroundings to learn differences between separate entities. Once the basic structure of the environment is captured inside the brain, supervision role starts so as to give labels to different categories. This role could be achieved by transfer of experience, or by asking pertinent questions to clarify ambiguity and learn the names of different entities. As the task proceeds, learning is adapted as more examples flow in and more experience is gained. New examples, never seen before, of the different categories are learned automatically to belong to the correct class, and hence the system is adapted.

Self learning machines, as defined in this paper, are those following the process described above to learn to classify patterns. A clear example of the above process is human child learning the names of figures. First, he observes different figures with no supervision to learn the differences between them. Then he is told their names, or he asks explicitly for the names of certain examples of each category and generalizes the name to the whole class. As he gets older, he faces more and more different examples, with new shapes never seen before, and yet he can adapt and develop himself to learn those new examples to belong to the right category according to his own belief.

Three components are identified to compose this self learning system. The first one is the unsupervised pre-training, which is responsible of feeding unsupervised examples to the machine to adapt itself to different clusters. The second is the supervised training to learn to classify examples based on true labels, so as to fine tune the network parameters based on the supervised data. And finally adaptive learning continuously adapting the machine to the new examples coming from real data, which could have never been seen before in unsupervised pre-training or supervised training phases. The unsupervised pre-training has proved to be effective in Deep Belief Networks [1] [3]. In this paper, the supervised training and adaptive learning are focused on.

In many practical learning domains, there is a large supply of high-dimensional unlabeled data and very limited labeled data. Applications such as information retrieval and machine vision are examples where large amounts of unlabeled data is readily available [3]. The need to supervised data with semi or minimal supervision in such applications is high, to compensate the lack of labeled data.

Minimum supervision means using few labeled examples, and using them to generalize to broader dataset. If efficient clustering of the unsupervised dataset is possible, then it is sufficient to know the labels of the means of clusters, or at least one supervised example of each cluster to apply the same labeling to the whole cluster automatically.

In [3] non-linear dimensionality reduction is performed using deep auto-encoders, learned using greedy layer by layer learning algorithm in [1]. Applying linear mapping, like Neighborhood Component Analysis (NCA), on the non-linear generated codes of the deep auto-encoder creates non-linear mapping and gets the data to a new space where the examples can be efficiently clustered.

The continuous flow of data during operation is essential for adaptation of the system to real world

978-1-4577-1194-7/11/$26.00

examples. A system needs to be developed to handle learning adaptation based on incoming data. To be able to perform this task, the learning algorithm itself should exploit the unsupervised data in the first place. A typical candidate algorithm for that is the one presented in [1] and [3] of learning deep networks, or Deep Belief Networks (DBN) based on greedy Deep Boltzmann Machine (DBM) learning algorithm. This greedy layer by layer learning algorithm can make efficient use of very large sets of unlabeled data, and so the model can be pre-trained in completely unsupervised fashion. The limited labeled data can then be used to only slightly fine-tune the model for a specific task at hand using standard gradient based optimization. Adaptation as presented in our paper proposes that, the same learning process can be repeated whenever more supply of unsupervised data is available from normal operation.

A novel approach is developed in this paper based on deep networks concepts to achieve two goals; the first one is utilizing few examples during supervised learning phase by automatic labeling the unsupervised training data set using non-linear unsupervised clustering on top of the deep auto-encoder. The second one is network adaptation with the real world unsupervised examples, based on the greedy layer by layer unsupervised learning of DBN, then fine tune the network again with the already available supervised examples. With every batch of new input data, the learning is repeated and the network is updated periodically or on need basis.

Merging the two components of automatic labeling and adaptive learning constitutes the Self Learning Machine (SLM) as defined earlier. Those machines are able to learn with minimal supervision and adapt to real world data. SLMs are generic learning systems in terms that they can be adapted to any pattern recognition task. All what is needed are few representative examples of each class to be recognized, and then the system will build the network from the unsupervised input data flowing in. Recursively the network adapts its parameters as more and more new data is fed to the system, and the performance is enhanced automatically.

The next sections are organized as follows; first related work is demonstrated. The next section presents the automatic labeling component. Then adaptive learning component is described. And finally the paper is concluded with the conclusion and future work

II. Related Work

An approach to imitate the behavior of V1 cortex in humans is found in recent work of Hinton et al. [1] on DBN, where the structure of the world is first captured by unsupervised pre-training, so the system is capable of generating similar examples of those learned without supervision [2]. The role of unsupervised pre-training for classification task is to get the network to a point in space near to the global minimum so that back propagation can start without getting stuck in local minima [1].

In [1],[3] and [4] it is proved that high-level representations can be built from a large supply of unlabeled inputs and the very limited labeled data can then be used to only slightly adjust the model for a problem-specific task.

In [3] work has been done on learning nonlinear mappings that preserve class neighborhood structure. This is the main idea on which automatic dataset labeling is built on. In this work, it was demonstrated how K-nearest neighbor classification can perform well on non-linear transformation of the input.

Future work in [3] suggests semi-supervised learning of deep Boltzmann's machines to target the applications of large supply of high-dimensional unlabeled data and very limited labeled data, like information retrieval and machine vision.

III. Automatic labeling system

Artificial intelligence tasks requiring classification and regression are mostly based on statistical learning of supervised training set, on the hope that this training set is sufficient to generalize well to the test set and real world examples.

This classical model requires large supply of labeled training set. One of the major obstacles facing development of such systems is the availability of such huge labeled training set. In many practical learning domains, there is a large supply of high-dimensional unlabeled data and very limited labeled data. Applications such as information retrieval and machine vision are examples; where large amounts of unlabeled data are readily available.

A self learning system shall follow the same model, but it should not require such large amount of supervised examples, instead it should be able to learn with few supervised examples and generalize the learned concept to other examples in the training set.

The root cause of the problem is that if the input raw data or features are well structured, then the system could discover similarities in the examples and generalize the learned labels to all similar examples. However, the input data is not necessarily structured by nature.

Suppose for the moment that somehow structured input data is obtained, such that an efficient clustering algorithm could be run successfully to discover related examples, such that the number of clusters is the number of classes at hand. In this case only few labeled representative examples are needed of every cluster, and the system can then apply the same labeling on all the

members of the cluster, and hence the whole dataset can be fully labeled. However, the assumption of structured input data or features is not always valid. It depends on the type of features extracted for the data.

A. Deep versus shallow architectures

To discover similarity between cluster members, it is traditional to make some transformation on the original data to improve the clustering algorithm performance. Among the most common algorithms used for such purpose are Pricipal Component Analysis (PCA), Linear Discriminant Analysis (LDA) and Neighborhood Component Analysis (NCA), which performs linear transformation on the data to map it to another space where within class similarities are improved. A linear transformation has a limited number of parameters and it cannot model higher-order correlations between the original data dimensions.

Using a nonlinear transformation function low-dimensional representations that work much better than existing linear methods can be discovered, provided that the dataset is large enough to allow the parameters to be estimated [3].

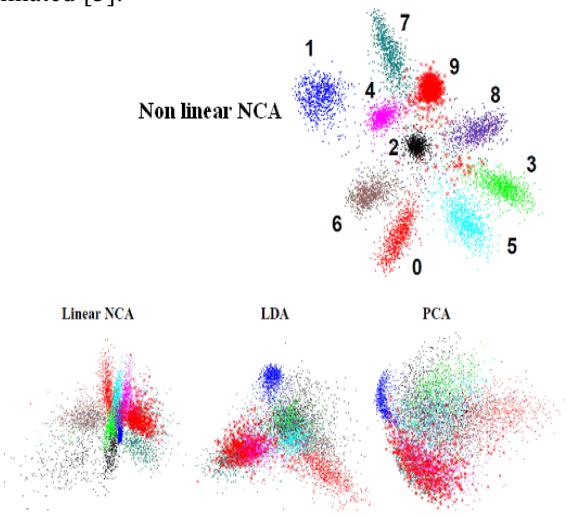

Figure 1 Non-linear NCA performance vs. other linear methods[3]

Figure 1 [3] describes the different mapping results of the MNIST dataset using different methods, linear methods: LDA, PCA and Linear NCA give poor structure for different classes of digits. The upper right graph shows the mapping with NCA on top of deep auto-encoder; non-linear NCA. It is clear how the different classes are well structured such that the operation of a clustering algorithm like K-means is enhanced.

B. Deep auto encoder clustering

To obtain the non-linearity discussed in III.A using greedy unsupervised learning algorithm, a multilayer, nonlinear encoder network that transforms the input data vector x into a low-dimensional feature representation F(x;W) can be trained. After the initial pre-training, the parameters can be fine-tuned by performing gradient descent in the Neighborhood Component Analysis (NCA) objective function [3]

C. Minimal supervision labeling

After having obtained the desired structured data, an efficient clustering algorithm can be run to discover the structure of the data. Number of clusters will be the number of classes, and the initial means of clusters will be the most representative examples of each class. For example, for the digit recognition task, the number of classes shall be 10, and the initial clusters means shall be representative image of each digit class. If the right labels for such representative means are available, then the whole cluster shall hold the same label, and hence a labeled dataset is obtained.

If generative model like deep auto-encoder is used, the system could be designed to generate the original raw input corresponding to the mean of each cluster, and ask the user explicitly for the label of this data, and then generalize the label to the whole cluster. This is very close to the behavior of human learning achieving minimal supervision.

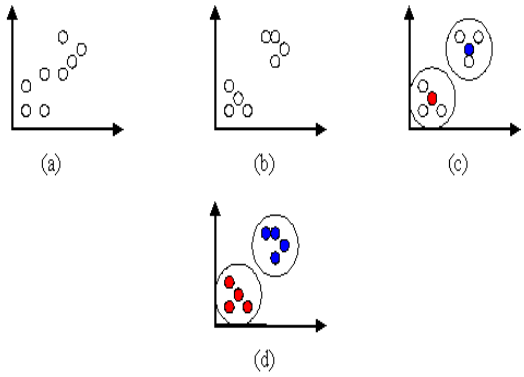

Figure 2 (a) Raw data (b) Structured data after non-linear transformation in the deep auto-encoder (c) Clustering algorithm discovers clusters means with their labels obtained (d) Having the labels of the means the whole cluster is labeled

Figure 2 describes the different mappings and processing on the input unlabeled data until a labeled dataset is obtained. First non-linear mapping is done using deep auto-encoder, such that structured dataset is obtained. Then a clustering technique (like K-means) is applied to the obtained codes. The labels of the clusters means or some representative class examples are provided by the user as supervised examples, form which the whole cluster can be labeled, and hence a supervised dataset is obtained.

IV. ADAPTIVE LEARNING SYSTEM

Traditional pattern recognition model is composed of training and test phases. After training the model, the system goes into test phase or normal operation. While the system is in normal operation, new patterns keep flowing in to be classified. According to the generality of the training set, the empirical classification error is determined. So, if the new patterns were encountered in the training phase or similar ones, then they should be correctly classified. On the other hand, if the training set was not covering some part of the feature space, and a new pattern is encountered in that part during operation, it will be misclassified.

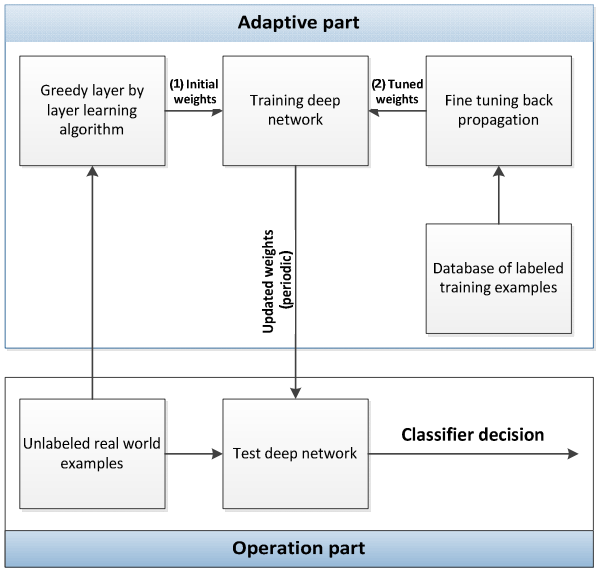

Figure 3 Adaptive learning system

A. *Unsupervised examples improve learning*

In conventional pattern recognition system the training and test phases are sequential. In adaptive system, they are done in parallel. Figure 3 describes the adaptive learning system.

To use real world patterns to update the model a learning algorithm that makes use of unsupervised examples is needed. Deep Belief Nets (DBN) trained using greedy layer wise learning algorithm [1] are typical candidates for such purpose.

The learning algorithm can make efficient use of very large sets of unlabeled data, and so the model can be pre-trained in completely unsupervised fashion. The very limited labeled data can then be used to only slightly fine-tune the model for a specific task at hand using standard gradient based optimization [3]. The unsupervised examples help in getting the initialization of weights to a point near to the minimum of the objective function to be optimized, protecting against getting stuck in poor local minima.

During operation of the system, the more unsupervised examples flowing in can be used to adjust the network to a better initial location. This process will be performed offline, in parallel with normal classification going on in the network, and the whole learning process is repeated to calculate the new weights. New updated weights calculated offline are then applied to the network periodically and only if test error is improved with the new weights, otherwise the old weights are kept.

The DBN architecture in [1] performs back propagation for weight adjustment after the unsupervised training phase. In adaptive learning, there are two options; the first is to apply the new unsupervised examples as separate unsupervised pre-training phase, then run back propagation, and the other one is to append the already existing database of old unsupervised examples with the new ones, then re-run the whole learning process from scratch. The choice depends on timing and storage issues.

B. *Over-fitting or user adaptation*

The above proposed system may be considered to suffer from over fitting to certain examples that flow during operation. For example if it is applied in handwriting recognition, then the system shall over fit to the user style that is using the system, because the network will be continuously adjusted to the examples of his own style. On the other hand, this could be viewed as user style adaptation.

Both points of view could be desirable or not according to the application and the method of implementation.

C. *Implementation*

When it comes to practical implementation, there are two options to implement adaptive learning model; the first one is the on-board implementation, the other one is the distributed implementation.

1) *On board implementation*

In this scheme, both the adaptive learning module and the classifier modules are on the same system, operating on the same user inputs. In this case user style adaptation is desired, and hence over fitting is not an issue. This type is probable in hand-held devices and embedded systems. The choice could be constrained by the cost of the system and resources available. A separate parallel coprocessor shall be used to handle the adaptation separate from the ongoing classification task. Updates of the network weights should be scheduled periodically.

2) *Distributed client-server model*

In cases of multi-users system, over fitting to certain user style is not desirable. In such cases; it is more

convenient to use distributed client-server model; where the client is the light weight device performing classification with the already calculated weights, under real time constraints, while the server part is performing adaptation using the classification examples. Multi-clients communicate their examples to the server, which performs adaptation, and then communicate back the updated weights to the clients. Updates could be scheduled periodically or on need basis.

To avoid over fitting, the server should perform adaptation by balancing the examples coming from all clients to be used in the learning process, and avoid being biased to certain user examples.

D. Experimental results

Adaptive learning is tested on MNIST dataset [16] using modified version of the MATLAB code in [15]. The DBN architecture used was 3 layer network; with 500 neurons in the first layer, 500 neurons in the second layer and 1000 neurons in the third layer. A ''softmax'' layer of 10 units is added on the top layer and tuned to give the labels of characters. The number of iterations for greedy RBM training or back propagation fine tuning was 50 epochs. The input character images are 28x28 pixels. 60,000 examples were used for training and 10,000 for testing. To simulate adaptation, the training set was subdivided into balanced mini-batches each containing 100 examples with total of 600 batches. The experiment goes on by feeding more batches and performing back propagation each time. With every update of the network the classification error performance is tested.

Note that; the target was not to achieve best error rate in comparison to existing systems, but to prove that feeding more unsupervised examples do improve network performance in terms of classification error rate.

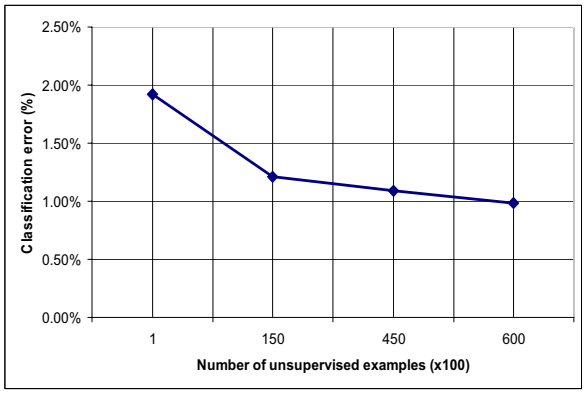

Figure 4 Adaptive learning performance curve

Figure 4 shows that the error performance is improved as more unsupervised examples are fed to the network. As more examples are used to pre-train the DBM network, a better initialization point of weights is obtained in weight space, so that supervised training phase using back propagation can start from a point nearer to the global minimum, instead of falling in a poor local minimum. In addition, as the greedy algorithm is trained on more unsupervised real world examples, its probabilistic model is improved to make those examples more probable, and hence improving the regeneration performance of those examples. This is similar to teaching a human person some figures, so that he can generate similar figures from his imagination, which indicates a better learning of those figures.

V. CONCLUSION AND FUTURE WORK

In this paper a self learning machine is proposed in terms of its ability to learn with minimal supervision and adapt to real world examples during operation. The first contribution of this paper is the automatic labeling component based on non-linear transformation using deep auto-encoders, followed by clustering step. The second contribution is the adaptive learning component based on unsupervised pre-training of Deep Belief Nets. Merging the two components constitutes a Self Learning Machine (SLM). Those machines are able to learn with minimal supervision and adapt to real world data. SLMs are generic learning systems in terms that they can be adapted to any pattern recognition task. All what is needed are few representative examples of each class to be recognized, and then the system will build the network from the unsupervised input data flowing in. Recursively the network adapts its parameters as more and more new data is fed to the system, and the performance is enhanced automatically. The proposed system target is to mimic human learning behavior that it needs few supervision and the ability to build own beliefs based on experience.

Future work includes testing the proposed auto labeling algorithm proposed in this paper using deep auto-encoder architecture to perform non-linear dimensionality reduction, to test the performance of the system. Also, different implementation issues described in adaptive learning are to be addressed. The on board implementation choice could be studied to evaluate practically the parallel co-processor hardware needed to perform adaptation. Also, the communication method needed in case of distributed implementation needs to be studied. Over fitting avoidance strategy in case of adaptive distributed implementation need to be addressed too.

Practical results on MNIST dataset prove the adaptive learning concept, showing improved classification error performance as more unsupervised examples are used for pre-training. This is due to better initialization weights for back fitting supervised stage, thus, improving the generation performance of the network of real world

examples. This is similar to the ability of human being of drawing familiar figures that are well learned.

REFERENCES

[1] G. E. Hinton, S. Osindero, and Y. Teh, ‘‘A fast learning algorithm for deep belief nets’’ *Neural Computation*, vol. 18, pp. 1527–1554, 2006.

[2] G. E. Hinton, ‘‘To Recognize Shapes, First Learn to Generate Images’’ *Computational Neuroscience: Theoretical Insights into Brain Function, Elsevier*, UTML TR 2006 – 004, October 26, 2006

[3] Ruslan Salakhutdinov, ‘‘Learning Deep Generative Models’’ PhD thesis, *Graduate Department of Computer Science, University of Toronto*, 2009

[4] G. Montavon, M. Braun, K. R. Müller, ‘‘Layer-wise Analysis of Deep Networks with Gaussian Kernels’’ *Advances in Neural Information Processing Systems (NIPS)*, 2010

[5] Yoshua Bengio, and Yann Lecun, ‘‘Scaling Learning Algorithms towards AI’’, *Large-Scale Kernel Machines, MIT Press* (2007)

[6] Yoshua Bengio, Pascal Lamblin, Dan Popovici, Hugo Larochelle. Greedy Layer-Wise Training of Deep Networks. *In Proceedings of NIPS'2006*. pp.153~160

[7] Arel, I., Rose, D.C., Karnowski, T.P., ‘‘Deep Machine Learning - A New Frontier in Artificial Intelligence Research [Research Frontier]’’, *Computational Intelligence Magazine, IEEE*, volume: 5, issue:4, pp: 13 – 18, Nov 2010

[8] Olshausen BA, Field DJ. ‘‘How close are we to understanding v1?’’ Neural Comput. 2005 Aug;17(8):1665-99.

[9] Matthew Blaschko, Andrea Vedaldi, Andrew Zisserman, ‘‘Simultaneous Object Detection and Ranking with Weak Supervision’’, *Advances in Neural Information Processing Systems (NIPS)*, 2010

[10] George E. Dahl, Marc'Aurelio Ranzato, Abdel-rahman Mohamed, and Geoffrey E. Hinton, ‘‘Phone Recognition with the Mean-Covariance Restricted Boltzmann Machine’’, Advances in Neural Information Processing Systems (NIPS), 2010

[11] Li Deng, Mike Seltzer, Dong Yu, Alex Acero, Abdel-rahman Mohamed, and Geoff Hinton , ‘‘Binary Coding of Speech Spectrograms Using a Deep Auto-encoder’’, Interspeech 2010

[12] Yuanqing Lin, Tong Zhang, Shenghuo Zhu, Kai Yu, ‘‘Deep Coding Network’’, Advances in Neural Information Processing Systems (NIPS), 2010

[13] Ruslan Salakhutdinov, Geoffrey E. Hinton ‘‘An Efficient Learning Procedure for Deep Boltzmann Machine’’, Computational Cognitive Science, MIT Press (2010)

[14] Marc' A. Ranzato, Christopher Poultney, Sumit Chopra, Yann Lecun , ‘‘Efficient Learning of Sparse Representations with an Energy-Based Model’’, Advances in Neural Information Processing Systems (NIPS), 2006

[15] http://www.cs.toronto.edu/~hinton/MatlabForSciencePaper.html, 5/24/2011 2:31 PM

[16] http://yann.lecun.com/exdb/mnist/, 5/24/2011 2:31 PM