VERSITA
Information Technology and Management Science
doi: 10.2478/itms-2013-0009
2013 / 16

Research on the Classification Ability of Deep Belief Networks on Small and Medium Datasets

Andrey Bondarenko[1], Arkady Borisov[2], [1-2] *Riga Technical University*

***Abstract* – Recent theoretical advances in the learning of deep artificial neural networks have made it possible to overcome a vanishing gradient problem. This limitation has been overcome using a pre-training step, where deep belief networks formed by the stacked Restricted Boltzmann Machines perform unsupervised learning. Once a pre-training step is done, network weights are fine-tuned using regular error back propagation while treating network as a feed-forward net. In the current paper we perform the comparison of described approach and commonly used classification approaches on some well-known classification data sets from the UCI repository as well as on one mid-sized proprietary data set.**

***Keywords* – Artificial neural networks, deep belief networks, classification, restricted Boltzmann machines**

I. Introduction

Theoretical foundations for learning deep belief networks (DBNs) were laid down by Geoffrey Hinton, for example, see [1]. Bengio gives a great overview over deep architectures in general – see [2-3]. DBNs are formed by stacked Restricted Boltzmann Machines (RBMs). Recently DBNs, RBMs and other Deep Architectures were successfully applied to a wide range of classification tasks outperforming other approaches [4], [6-8]. In [9] there is evidence that adding more layers helps in recognition/classification tasks. However, [5] showed that DBNs were outperformed on some classification tasks. The current paper aims at comparing RBMs and DBNs classification performance against some well-known classifiers like SVMs and Random Forest Trees on some well-known small classification UCI [10] data sets as well as a single mid-sized proprietary document classification data set. This paper is structured as follows: Section 2 provides theoretical background for RBMs and DBNs as well as describes pre-training procedure for feed-forward error back-propagation artificial neural networks. Sections 3 and 4 describe experimental setup and present experiments results, while Section 5 concludes the paper.

II. Energy-based Models

A. Restricted Boltzmann Machine

RBMs are stochastic generative neural networks that can learn probability distributions over a set of their input vectors. The main consequence of this definition is that such a neural network learns **p(data)** instead of **p(label | data)** – essentially these models are modelling data, not labels. This allows us to deal with unlabelled or partially labelled data. Besides, restricted Boltzmann machines can be represented as a bipartite graph with two sets of neurons – visible and hidden ones **(v, h)**, refer to Fig.1. Neurons in both layers are symmetrically connected. RBMs are Energy-based Models (EBM) [28], that associate scalar energy to each configuration, so an overall network state can be represented as follows:

$$E(v,h) = -\sum_{i \in visible} a_i v_i - \sum_{j \in hidden} b_j h_j - \sum_{i,j} v_i h_j w_{i,j} \quad (1)$$

where v_i, h_j are binary states of visible unit i and hidden units j, a_i, b_j are their biases and w_{ij} is the weight between them. As in general RBM contains stochastic binary units, meaning that its binary unit state is defined by probability of its weights, the shaping of energy function allows obtaining more plausible probability distributions for network neurons.

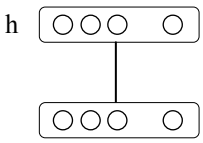

Fig. 1. Restricted Boltzmann machine neural network

This means that RBM network learns distributions of **(v, h)**; in other words, the probability of joint configuration over both hidden and visible units depends on the energy of that joint configuration compared to energy of all other joint configurations – this can be written as follows:

$$p(v,h) = \frac{1}{Z} e^{-E(v,h)} \quad (2)$$

here ***Z*** represents all other possible configurations of visible and hidden units:

$$Z = \sum_{v,h} e^{-E(v,h)} \quad (3)$$

Network assigns probability p(v|h) as follows:

$$p(v,h) = \frac{1}{Z} \sum_{h} e^{-E(v,h)} \quad (4)$$

Thus, to acquire high probability of visible training vector we need to adjust weights and biases of weights to biases and hidden units to lower energy of training vector and raise energy of other training vectors (especially those that have low energy). According to [14], the derivative of a visible training vector with respect to weights is as follows:

$$\frac{\partial \log p(v)}{\partial w_{ij}} = \left\langle v_i h_j \right\rangle_{data} - \left\langle v_i h_j \right\rangle_{model} \quad (5)$$

Here angle brackets denote probability expectations for data

or model distributions. We can notice that (1) can be translated to free energy formula:

$$F(v) = -\sum_{i \in visible} a_i v_i - \sum_i \log \sum_{h_i} e^{h_i (b_i + W_i v)} \quad (6)$$

Please refer to [15] in regard to how this (1) - (6) translation is done. Due to the fact that we deal with stochastic binary neurons, (6) can be even more simplified:

$$F(v) = -\sum_{i \in visible} a_i v_i - \sum_i \log(1 + e^{h_i (b_i + W_i v)}) \quad (7)$$

Free energy is omitted in next formulas, but it will be reused in Conditional Restricted Boltzmann Machines.
Equation (5) can be rewritten as:

$$\frac{\partial \log p(v)}{\partial w_{ij}} = \langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model} \quad (8)$$

Due to the fact that there is no connection between neurons within a layer, it is relatively easy to get expectations for *data* distribution:

$$p(v_i = 1 \mid h) = sigm\left(a_i + \sum_i v_i w_{ij} \right) \quad (9)$$

where *sigm* is the sigmoid function – *sigm(a)=1/(1+exp(a))*. And similarly for visible units:

$$p(h_j = 1 \mid v) = sigm\left(b_j + \sum_i v_i w_{ij} \right) \quad (10)$$

In (9) and (10) *a*, *b* are biases and *v, h, w* are visible and hidden unit states, respectively, w_{ij} is their associated weight. Thus, we assign 1 or 0 to hidden or visible neurons with a defined probability.
It is much more difficult to get model distributions, but in 2002 G.Hinton discovered [29] an elegant solution to this problem. Thus, instead of:

$$\Delta w_{ij} = \varepsilon \left(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model} \right) \quad (11)$$

Hinton proposed to use:

$$\Delta w_{ij} = \varepsilon \left(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{recon} \right) \quad (12)$$

where reconstructed expectation of distribution can be calculated by Contrastive Divergence (CD) algorithm [14], see Fig. 2, which depicts a single step of CD algorithm. As it is highlighted in (6) and (7), RBM uses stochastic binary units (there are real valued extensions). One step of CD contains two phases – positive and negative ones. In the positive phase, one needs to clamp a training visible vector on a visible layer and calculate new states of hidden neurons using (7). In the negative phase, one needs to calculate new states of visible units. This new state of a visible layer can be thought as "fantasy". CD with such a single step is referred to as CD1, the more steps are taken, the better approximation to model distribution will be acquired. It was discovered that even a single step is enough,

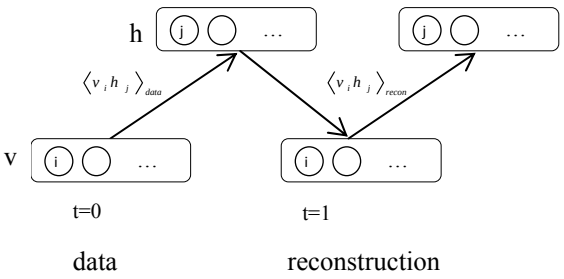

Fig.2. Depicts single step of Contrastive Divergence algorithm

at least at early learning stages. At later stages, one can switch to CD3, CD5 and CD10. Apart from CD, another algorithm, called Persistent Contrastive Divergence, was proposed in [11]. There are some other nuances in regard to CD learning algorithm, which can be found in [14].

B. Deep Belief Networks

When CD was found, it was proposed in [1] to stack trained RBMs in a greedy manner to form the so-called Deep Belief Networks (DBN). The idea was to cleverly train RBM on a training vector, then after finishing the training process to use the first RBM hidden layer neuron activations as input for a visible layer of the second stacked RBM to train it and continue this procedure for all subsequent layers. When overall training is performed, the found network weights can be fine-tuned with a regular Error Back Propagation algorithm. For graphical representation see Fig. 3.

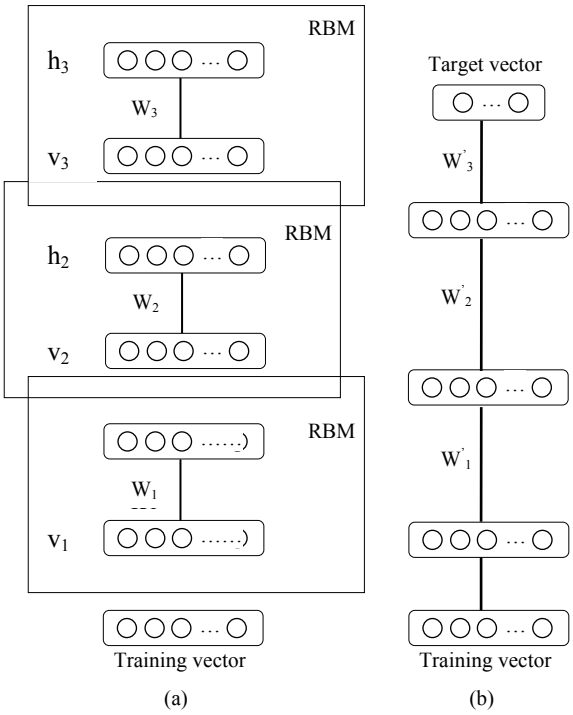

Fig. 3. (a) Denotes DBN formed by stacked RBMs; (b) shows how a regular Feed-Forward Neural Network (FFNN) is formed using weights acquired during DBN training to perform fine-tuning using a standard error back propagation

It is argued that such a deep network is capable of building complex hierarchical feature representations. For example, when one wants to classify digits "3" and "8", it is quite a problematic task because digit "3" is somewhat entangled in the "8"-th digit manifold; thus, the necessity for hierarchical features arise – and in such tasks DBNs and Deep Architectures outperform many other classifiers.

It can be worth noting that the reason why neural networks were abandoned in favour of SVMs is that on the one hand we did not have enough training data and computational power and on the other hand it was quite problematic to train really deep architectures due to a "vanishing gradient" problem, which shows itself at higher levels or during Recurrent Neural Network (RNN) training (each RNN can be represented as regular FFNN with a large number of layers, so this is a common problem for deep layers). For a vanishing gradient problem in regard to RNN, see [12]. In [3] Bengio justifies greedy pre-training, and in [19] the author provides experimental results that show higher accuracy acquired by pre-trained FFNN and demonstrates that solutions found lay in different areas of function space (see page 8 in [19]).

C. Conditional Restricted Boltzmann Machine

Since the introduction of RBMs, different authors have proposed various modifications of RBMs, especially Conditional Restricted Boltzmann Machine (CRBM), proposed in [8], [13], for graphical representation see Fig. 4. The main idea here was to adjust RBMs for more successful application to discriminative problems. Apart from [8], [13], one of the first attempts to use RBMs/DBNs for a classification task was made by Hinton, Osindero and Yee Whye Teh in [16].

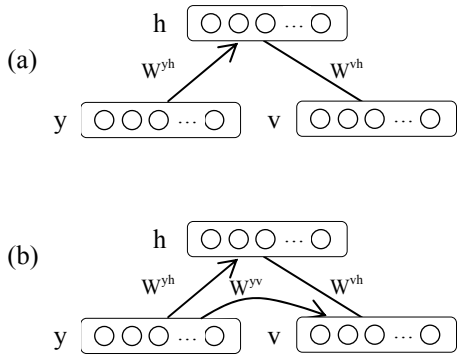

Fig. 4. (a) Classification Restricted Boltzmann Machine, (b) Conditional Restricted Boltzmann Machine

We will consider only (a) type CRBM, which uses target vector u for which it holds two additional weight matrices W^{uh} and W^{uv} (for hidden and visible layers, respectively. According to [16], CRBM models the joint distribution of an input $\boldsymbol{x} = (x_1, \ldots x_n)$ and target class $\boldsymbol{y}$ using a hidden layer of binary stochastic units $\boldsymbol{h} = (h_1,\ldots,h_H)$. This is done by first defining an energy function:

$$E(y,v,h) = -h^T W v - a^T v - b^T h - c^T e_y - h^T Y e_y \quad (13)$$

with parameters $\Theta = (W, a, b, c, Y)$ and where

$$e_y = \left(1_{i=y}\right)_{i=1}^{C} \quad (14)$$

is "one out of C" representation for y. From an energy function, we can assign probabilities to values of y, v and h as follows:

$$p(y,v,h) = \frac{e^{-E(y,v,h)}}{Z} \quad (15)$$

where Z is the normalization constant (partition function) that is already known (from (3)) and ensures that (15) is a valid probability distribution. Similarly to standard RBM computing $p(y,v,h)$ is computationally intractable, but it is possible to do Gibbs sampling, which gives conditional distributions. When conditioning on the visible layer we have:

$$p(h \mid y,v) = sigm\left(b_j + Y_{jy} + \sum_i v_i w_{ij}\right) \quad (16)$$

And when conditioning for the hidden layer we have:

$$p(v \mid h) = sigm\left(a_i + \sum_i v_i w_{ij}\right) \quad (17)$$

$$p(y \mid h) = \frac{e^{\left(c_y + \sum_j Y_{jy} h_j\right)}}{\sum_{y^*} e^{\left(c_{y^*} + \sum_j Y_{jy^*} h_j\right)}} \quad (18)$$

It is also possible to compute $p(y|v)$ exactly and hence perform the classification. Thus, after some transformations (please refer to [16]) it is possible to derive:

$$p(y \mid v) = \frac{e^{(-F(y,v))}}{\sum_{y^* \in \{1,\ldots,C\}} e^{(-F(y,v))}} \quad (19)$$

Here $F(y,v)$ is free energy that is already known.

According to [16], one way of interpreting (19) is that, when assigning probabilities to a particular class y for some input v, the Classification RBM looks at how well the input v fits or aligns with the different filters associated with the rows W_j of W. These filters are shared across the different classes, but different classes will make comparisons with different filters by controlling the class-dependent biases Y_{jy}. Notice also that two similar classes could share some filters in W, that is, both could simultaneously have large positive values of Y_{jy} for some rows W_j. Along with that [16] describes a hybrid RBM learning approach, which uses descriptive learning combined with generative learning adjusted using some parameter alpha. Such a generative approach outperformed RBM+NN approach (RBM used as a pre-training step) on

MNIST (refer to http://yann.lecun.com/exdb/mnist/) digit recognition data set.

III. Experimental Setup

For our experiments we used generative DBN implementation (unmodified source codes taken from https://github.com/rasmusbergpalm/DeepLearnToolbox), which afterwards was used as a pre-training step for fine-tuning FFNN. For all experiments we used 10-fold cross-validation, i.e., we divided the whole data set into ten parts and used nine parts to train model and the last 10^{th} part to run a classification test, in the next run the part used for training was changed to be different. Thus, on all 10 runs the same 10 data parts were used, but the training part was always different. We report classification accuracy testing rates averaged over 10-fold cross-validation runs.

Apart from the mentioned Energy-based models and DBN architecture, for comparison purposes we used Random Forests (RF) implementation (unmodified source code was taken from https://code.google.com/p/randomforest-matlab/), for classifier details see [17]. Along with RF for some data sets we provided SVN accuracy rates taken from other studies [18].

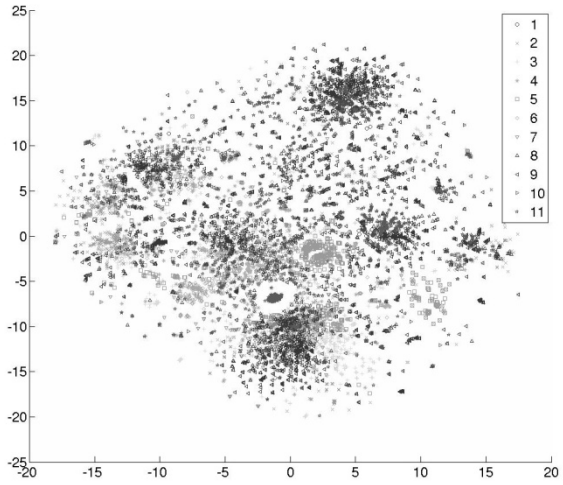

Fig. 5. Visualization of proprietary data set – 12916 binary bags of words representing 11 financial document classes

We performed our tests on 2 standard classification benchmarking data sets: glass identification (http://archive.ics.uci.edu/ml/datasets/Glass+Identification) and ionosphere (http://archive.ics.uci.edu/ml/datasets/ionosphere). Both of them are multivariate real valued datasets related to classification problem. We tested the discussed algorithms on a mid-sized proprietary data set containing 12916 binary data vectors of length 200 (initially there were vectors of length 5000, but we picked only 200 most representative features). These vectors represented bags of words extracted from financial documents. There were 11 classes in this data set and classes were represented by: 608, 1331, 1542, 995, 1009, 500, 731, 1220, 2788, 78 and 2114 data vectors, respectively. As can be seen, class 10 is quite poorly represented. There were numerous overlapping vectors belonging to different classes and in general such a data set could be considered quite hard to classify. Figure 5 represents the visualization of this data set (utilizing data vectors of full length equal to 5000) by means of fast t-Distributed Stochastic Neighbour Embedding, for visualization algorithm details refer to [20-22].

IV. Experimental Results

TABLE I

Classification Accuracy Rates

Accuracy Rates (%)	Propr. data set	Ionosphere	Glass
*Gen*RBM (with fine-tuning)	69.5%	89.17%	35.5%
*Class*RBM (without fine-tuning)	65.17%	89.17%	34.58%
DBN-(with fine-tuning)	18.12%	65.46%	25.45%
FFNN (single hidden layer)	64.76%	89.17%	35.5%
FFNN (2 hidden layers)	56.09%	89.17%	35.5%
Random Forests	77.15%	91.17%	79.91%
Hybrid SVM/ Gaussian SVM/ -	79.7%	94.8%	-

Random Forests were used with default settings for all data sets.

For a proprietary data set generative RBMs were trained with 800 hidden neurons and 3000 training epochs, for Ionosphere and Glass data sets it was trained with 100 hidden neurons and 1000 training epochs.

Classification RBM without fine-tuning for a proprietary data set was trained with 800 hidden neurons and 3000 training epochs, for Ionosphere and Glass data sets it was trained with 100 hidden neurons and 1000 training epochs.

DBN and FFNN were trained using 2 hidden layers with 200 neurons for a proprietary data set and with 10 and 32 neurons in each hidden layer, respectively. On Glass/Ionosphere data sets RBMs were trained on 100 epochs for RBM and for 100 epochs for FFNN fine-tuning on Glass/Ionosphere datacsets. In all cases FFNN (used in general setup and fine-tuning stage) was trained using Cross Entropy as a loss function.

SVM used for classification is fine-tuned implementation based on libSVM library [23].

Table I shows classification accuracy rates on different data sets. It is clearly seen that RBMs and DBN networks clearly lose in terms of accuracy to Random Forests and SVM-based classifiers. It can be seen that FFNN with two hidden layers outperforms DBN. Our results resemble ones in [5]. Moreover, DBN network shows extremely low performance even compared to RBMs. The first observation is that Ionosphere has 32 features and Glass Classification only 10. In contrast, our proprietary data set holds 200 features, but all of them are binary. Thus, it seems that having problems with lower dimensionality (or with several features preselected by some other algorithms) can badly influence RBM classification rates (we should note that we conducted partial experiments on a proprietary data set with larger feature vectors (2000 features), but performance was even worse than with 200 features for DBN). In contrast to real-valued

vectors representing documents in [1], we used binary vectors. Nevertheless, SVM and RF given such initial information were able to outperform RBMs, DBN and FFNN. As to DBN it was trying to build higher-level hierarchical features based on quite poor representation given by RBM in the first layer. However, in our case it seems that all features were uncorrelated and their combination at higher levels provided a low value if at all. The same logic applies to FFNN with two hidden neuron layers. We conducted additional experiments, which showed that adding additional hidden layers did not help DBN to perform better. Looking at DBNs, their main point is to learn a hidden layer of filters or sparse bases (sparse codes) that can be combined in subsequent layer(s) either in FFNN or even SVM (for example, see [24]). In contrast, for our data sets it seems that the learning of such filters that would model the appearance of several bits in a vector instead of the single one is inappropriate for the reviewed data sets. While such sparse coding is a good thing for high-dimensional data, it is obviously not the best choice for dense data sets. In general, our findings somewhat contradict the results in [25], where Hinton argues that DBNs with an exponentially large count of hidden layers and size equal to an input vector can model an arbitrary input vector with arbitrary accuracy, but again we performed only partial experiments with 3 and 4 hidden layers, while Hinton talks about much larger amount. The same discussion about RBM and DBN representational power is held in [26-27]. While such theoretical discussions are important in a way they give theoretical justifications of methods, but as our experiments show for some specific data sets the referenced classification approaches do not work very well using acceptable models (both in terms of size and training time).

All successful DBN and RBM applications reported in the referenced papers are related to high-dimensional data sets, such as documents, images and alike. While these data sets are extremely perspective research area, it is clear that for low-dimensional or pre-cleared data such approaches with default settings are not the best choice. LeCunn generalizes many classification approaches as Energy-based Models and treats them all as Energy-based Learning, so in theory it is possible to leave architecture and inference algorithm, but do adjustments in a loss function and possibly a learning algorithm.

V. Conclusion

We performed comparison of RBM+FNN, CRBM, DBN and FFNN in classification tests using two small benchmarking UCI data sets and single proprietary mid-sized data set. It was shown that RBMs lost in terms of accuracy rates to RF and SVM approaches, while DBN was proven to be useless because showed very poor performance, FFNN showed performance slightly worse than RBM with fine-tuning, which aligned with the reported good influence of pre-training phase. Tests on proprietary 200 feature data set showed that even such number of features could be insufficient to learn good separation hyper-planes for classification. Building hierarchical features through DBN showed to be useless. An increase in the number of neurons in the RBM hidden layer proved to have some positive effect, but it badly influenced training time and proved to give a negligible increase in accuracy. In general, it is obvious that existing approaches allow RBMs and DBNs to deal with high-dimensional data, where we have a large number of sample vectors to be learned from. Moreover, RBMs allow us to perform training on unlabeled data, which is a huge gain in certain scenarios.

Future research directions can include searching for reasons why RBMs are outperformed by RFs and SVMs and looking for possible solutions to increase performance of RBM. Energy-based Model framework [27] is a good candidate that can help in solving the latter problem. Another direction is searching for metrics that would allow us to tell beforehand whether specific data set can be successfully modelled by RBMs and DBNs. Apart from that, experiments with Partially Restricted Boltzmann Machines and Deep Boltzmann Machines can be conducted to see how well they perform.

References

[1] G.E. Hinton, R.R. Salakhutdinov, *Reducing the Dimensionality of Data with Neural Networks*. Science, Vol. 313. No. 5786, 28 July 2006, pp. 504-507.

[2] Y. Bengio, *Learning Deep Architectures for AI* Foundations and Trends in Machine Learning 1(2), 2009, pp. 1-127.

[3] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, *Greedy Layer-Wise Training of Deep Networks* Advances in Neural Information Processing Systems 19 (NIPS'06) 1(2), MIT Press 2007, pp. 153-160.

[4] G.E. Hinton, *Deep Belief Nets*, Tutorial at Deep Learning Workshop NIPS 2007 [Online] Available: http://nips.cc/Conferences/2007/Program/event.php?ID=572 [Accessed September 25, 2013]

[5] L. McAfee, *Document Classification using Deep Belief*, Report for CSC224n, Stanford, 2008.

[6] H. Lee, R. Grosse, Ranganath, A. Y. Ng, *Convolutional Deep Belief Networks for Scalable Unsupervised Learning of Hierarchical Representations*, Proceedings of the 26th Annual International Conference on Machine Learning ICML 2009, pp. 609-616.

[7] S. Zhou, Q. Chen and X.Wang, *Active Deep Networks for Semi-supervised Sentiment Classification* COLING 2010, Proceedings of the 23rd International Conference on Computational Linguistics: Posters, 2010, pp. 1515-1523.

[8] H. Larochelle, M. Mandel, R. Pascanu, Y. Bengio, *Learning Algorithms for the Classification Restricted Boltzmann Machine,* Journal of Machine Learning Research 13, 2012, pp. 643-669.

[9] N. Le Roux, *Neural Networks RBMs and DBNs* presentation at SMILE seminar 10 February 2011 [Online] Available: http://nicolas.le-roux.name/files/SMILE_dbns.pdf [Accessed September 25, 2013]

[10] K. Bache and M. Lichman, UCI Machine Learning Repository, Irvine, CA: University of California, School of Information and Computer Science, 2013 [Online] Available: http://archive.ics.uci.edu/ml/ [Accessed September 25, 2013]

[11] T. Tieleman, *Training Restricted Boltzmann Machines Using Approximations of the Log Likelihood Gradient*, Proceedings of the 25th Annual International Conference on Machine Learning ICML 2008, pp. 1064-1071.

[12] S. Hochreiter, *The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions*, International Journal of Uncertainty, Fuzziness and Knowledge Based Systems, Vol. 6, Issue 2, April, 1998, pp. 107-116.

[13] V. Mnih, H. Larochelle, G. E. Hinton, *Conditional Restricted Boltzmann Machines for Structured Output Prediction*, Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence, UAI-2011, Barcelona, Spain, July 14-17, 2011.

[14] G.E. Hinton. *A Practical Guide to Training Restricted Boltzmann Machines* Version 1, August 2, 2010. [Online] Available:

http://www.cs.toronto.edu/~hinton/absps/guideTR.pdf [Accessed September 25, 2013].

[15] *Deep Learning Documentation – Training RBM* Version 1 [Online] Available: http://deeplearning.net/tutorial/rbm.html. [Accessed September 25, 2013]

[16] G.E. Hinton, S. Osindero, Y. W. The, *A Fast Learning Algorithm for Deep Belief Nets*, Neural Computation, Volume 18, 2006, pp. 2283-2292.

[17] L. Breiman and A. Cutler, Random Forests, [Online] Available: http://www.stat.berkeley.edu/~breiman/RandomForests/cc_papers.htm [Accessed September 26, 2013].

[18] Classification results for various benchmark datasets, [Online] Available: http://www.is.umk.pl/projects/datasets.html [Accessed September 26, 2013].

[19] D. Erhan, P-A. Manzagol, Y. Bengio, S. Bengio and P. Vincent *The Difficulty of Training Deep Architectures and the Effect of Unsupervised Pre-Training*, Proceedings of the 12th International Conference on Artificial Intelligence and Statistics, April 16-18, Clearwater Beach, Florida, USA, 2009.

[20] L.J.P. van der Maaten and G.E. Hinton, *Visualizing High-Dimensional Data using t-SNE*, Journal of Machine Learning Research 9 (Nov), 2009, pp.2579-2605.

[21] L.J.P. van der Maaten, *Learning Parametric Embedding by Preserving Local Structure*, Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics (AI-STATS), JMLR W&CP 5, 2009, pp.384-391.

[22] L.J.P. van der Maaten Barnes-Hut-SNE, Proceedings of the International Conference on Learning Representations, 2013, Available: http://arxiv.org/abs/1301.3342 [Accessed September 26, 2013].

[23] Chang, Chih-Chung and Lin, Chih-Jen, *LIBSVM: A library for support vector machines*, ACM Transactions on Intelligent Systems and Technology, Vol. 2, Issue 3, 2011, pp.1-27, Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm [Accessed September 26, 2013].

[24] H. Tang *A Comparative Evaluation of Deep Belief Networks in Semi-Surpevised Learning*, Report for CSC2515, 2008. Available: http://www.cs.toronto.edu/~hxtang/projects/dbn_eval/dbn_eval.pdf, [Accessed September 26, 2013]

[25] I. Sutskever, G.E. Hinton, *Deep, Narrow Sigmoid Belief Networks are Universal Approximators*, Journal Neural Computation Volume 20, Issue 11, November. 2008.

[26] G. Montufar, N. Ay, *Refinements of Universal Approximation Results for deep Belief Networks and Restricted Boltzmann Machines*, Neural Computation, May, 2011, pp. 1306-1319.

[27] G. Montufar, *Mixture Models and Representational Power of RBM's and DBN's*, Deep Learning and Unsupervised Feature Learning Workshop, NIPS'10, December 19, Vncouver, Canada, 2010.

[28] Y. LeCunn, S. Chopra, R. Hadsell, M.A. Ranzato and F.J. Huang, *A Tutorial on Energy-Based Learning*, v1.0 August 16, 2006, Available: http://yann.lecun.com, [Accessed September 26, 2013]

[29] G.E. Hinton, *Training Products of Experts by Minimizing Contrastive Divergence*, Neural Computation 14(8), August 2002, pp. 1771-1800.

Andrey Bondarenko received the B.Sc. and MSc. degrees in 2004 and 2006 from the University of Latvia, Institute of Transport and Communication (Riga, Latvia). Since 2010 he has been continuing studies at Riga Technical University to obtain a doctoral degree in computer science. At present, major field of study is the extraction of concise rules from trained artificial neural network. Research interests include: computational intelligence, machine learning.
e-mail: andrejs.bondarenko@gmail.com

Arkady Borisov received his Doctoral Degree in Technical Cybernetics from Riga Polytechnic Institute in 1970 and Dr.habil.sc.comp. degree in Technical Cybernetics from Taganrog State Radio Engineering University in 1986.

He is a Professor of Computer Science at the Faculty of Computer Science and Information Technology, Riga Technical University. His research interests include fuzzy sets, fuzzy logic, computational intelligence and bioinformatics. He has 235 publications in the field.

He is a member of IFSA European Fuzzy System Working Group, Russian Fuzzy System and Soft Computing Association, honorary member of the Scientific Board, member of the Scientific Advisory Board of the Fuzzy Initiative Nordrhein-Westfalen (Dortmund, Germany).
Contact information: 1 Kalku Street, Riga, LV-1658, phone: +371 67089530, e-mail: arkadijs.borisovs@cs.rtu.lv.

Andrejs Bondarenko, Arkādijs Borisovs. Dziļas pārliecības tīklu iespēju izpēte mazu un vidēju datu kopu klasifikācijā
Dotajā brīdī mākslīgie neironu tīkli (MNT) tiek pielietoti dažādās nozarēs kur ir nepieciešama daudzdimensionālu datu apstrāde, ka arī neklasificētu datu apstrāde, kur minētie tīkli uzrāda labus rezultātus. Mēs varam droši runāt par MNT jomas otru renesansi. MNT kuri tiek pielietoti darbam ar minētiem datiem parasti tiek būvēti kā dziļi - ar daudziem apslēptiem slāņiem. Šādu dziļu MNT apmācība ir pietiekami grūts uzdevums, jo pastāv gradienta saplūšanas problēma. 2002.gadā, Geoffrey Hinton nodarbojās ar ierobežotiem Boltzmana mašīnām (IBM) un ir atklājis ātru apmācības algoritmu CD-k (kontrastīvā novirze). Šīs algoritms deva iespēju apmācīt IBM pieņemamā laikā. Pats par sevi IBM ir stohastisks radošs modelis, kuru ir iespējams apmācīt bez skolotāja. IBM apmācība tika izmantota kā pirmapmācības solis. Parasti MNT apmācībai iegūtais neironu saistību svars otrajā apmācības posmā tika izmantots kā normālais svars, lai apmācītu parasto MNT ar atgriezeniskās kļūdas izplatīšanu, lai precizētu neironu saišu svaru. Vēlāk tika rādīti dziļas pārliecības tīkli (DPT) (Deep Belief Networks), kuros katri divi blakus slāņi tika apmācīti ar IBM apmācības algoritmu. DPT izmantošana, kā pirmapmācības procedūra dziļiem MNT ar atgriezeniskās kļūdas izplatīšanas apmācību, uzrādīja iespaidīgus rezultātus tādās jomās, kā attēlu atpazīšanas sistēmas, dokumentu klasifikācijas sistēmas, sistēmas runas atpazīšanai un cilvēka kustību klasifikācijai (skriešana/iešana). Šajā rakstā ir aplūkota IBM un DPT teorija. Tika veikti IBM un DPT salīdzinošie testi klasifikācijas uzdevumos ar mazām un vidējām datu kopām ar mazu dimensiju skaitu (10/32/200 pazīmes). Visos gadījumos abi aplūkotie IBM tīkli uzrādīja sliktus rezultātus, bet DPT parādīja sliktāko rezultātu. Tādējādi, runājot par aplūkotiem modeļiem, rodas jautājums - kādas īpašības piemīt datu kopām, kas tik slikti ietekmē IBM un DPT klasifikācijas rezultātus un, no otras puses, kādas izmaiņas ir jāveic IBM un DPT apmācības algoritmā un/vai tīkla arhitektūrā, lai šie modeļi varētu strādāt ar aplūkotajiem datiem.

Андрей Бондаренко, Аркадий Борисов. Исследование возможностей сетей глубокой уверенности для классификации малых и средних наборов данных
В настоящее время применение искусственных нейронных сетей (ИНС) в различных отраслях, требующих работы с данными большой размерности, а также с неклассифицированными данными, показывает хорошие результаты. Можно с уверенностью говорить о втором ренессансе в области ИНС. Нейронные сети, применяемые для работы с упомянутыми данными, как правило, строятся глубокими - с большим количеством скрытых слоёв. Обучение таких сетей является трудоёмкой задачей, так как существует проблема исчезающего градиента. В 2002 году Джеффри Хинтон, занимавшийся ограниченными машинами Больцмана (ОМБ), открыл алгоритм обучения CD-k (Contrastive Divergence). Данный алгоритм позволил обучать ОМБ в приемлемое время. Сами по себе ОМБ - это стохастические генеративные модели, способные к обучению без учителя. Обученные ОМБ стали использовать как шаг предобучения ИНС – веса, полученные при таком обучении, на втором шаге использовались как веса обычной сети с обратным распространением ошибки для более точной финальной настройки весов. Еще позже были созданы сети глубокой уверенности (СГУ) (Deep Belief Networks), которые обучались послойно по алгоритму обучения ОМБ. Использование СГУ для предобучения глубоких сетей с обратным распространением ошибки позволило достичь впечатляющих результатов в системах распознавания образов, документов, речи, распознавания типа движения человека (бег/ходьба). В данной статье приводится теория, лежащая в основе ОМБ и СГУ. В рамках исследования были проведены сравнительные тесты по классификации малых и средних наборов данных малой размерности (10 / 32 / 200 признаков) средствами ОМБ, классификационной ОМБ, а также СГУ. В сравнительных тестах во всех случаях все три рассмотренные архитектуры показали худшие результаты, а СГУ оказалась худшей моделью. Таким образом, применительно к существующим моделям возникают вопросы - какими характеристиками обладают данные, на которых СГУ и ОМБ показывают плохие результаты, а с другой стороны - какие изменения позволят строить конкурентоспособные модели ОМБ и СГУ на рассмотренных наборах данных.