Big Data and Deep Learning

B. M. Wilamowski[**], Bo Wu[*], Janusz Korniak [**]

University of Information Technology and Management, Sucharskiego 2, Rzeszow 35-225, Poland
Dept. of Electrical and Computer Engineering
Auburn University, Auburn, AL 36849, USA
wilam@ieee.org, bowu1004@gmail.com, jkorniak@wsiz.rzeszow.pl

***Abstract*—Traditional data processing algorithms are usually not capable to process big data. As matter of fact, usually big data is being defined as such which cannot be processed with traditional techniques. At the same time a progress of technology makes that humans are now overwhelm by big data. One way of processing big data is to use deep neural networks, which are difficult to train so often a combination of several learning algorithms are used. This approach however requires a lot of human involvement in design of such systems for very specific cases. In this presentations and universal architectures and training methods are described which can handle very complex systems with minimal human interactions. Another important problem is a visualization of multidimensional data sets in 2 or 3 dimensions so humans can see them. A new very effective visualization algorithm is also presented. The third issue is the pattern clustering. Round clusters can be separated relatively easy. In this presentation new and very fast clustering of complex shape is also presented.**

Keywords—deep learning, big data; visualization; clustering; classification;

I. INTRODUCTION

The traditional approach for solving complex problems and processes usually follows the following steps: At first we are trying to understand them, and then we are trying to describe them in the form of mathematical formulas. This classical Da Vinci approach was used for the last several centuries, and unfortunately it cannot be applied for many current complex problems. These problems are very difficult to understand and process by humans. Notice that many environmental, economic, and often engineering problems cannot be described by equations, and it seems that adaptive learning architectures are the only solution to tackling these complex problems. Many smaller scale problems were already solved using shallow architectures such as ANN [1,2], SVM [3,4], or ELM [5-7]. However, for more complex problems, more deep learning systems with enhanced capabilities are needed.

It has already been demonstrated that much higher capabilities of super compact architectures have 10 to 100 times more processing power than commonly used learning architectures like MLP [8]. It is possible to train them very efficiently if the network is shallow like SVM or ELM. It turns out that the power of learning systems grows linearly with their widths and exponentially with their depth. For example, such a shallow MLP architecture with 10 neurons can solve only a Parity-9 problem, but a special deep FCC architecture with the same 10 neurons can solve as large a problem as a Parity-1023. Therefore, a natural approach would be to use these deep architectures. Unfortunately, because of the vanishing gradient problem, these deep architectures are very difficult to train, so a mixture of different approaches is used with a partial success. Until now, it is assumed that it is not possible to train neural networks with more than 6 hidden layers. We have demonstrated that it is possible to efficiently train much deeper networks. This became possible by the introduction of additional connections across layers and to use our new very powerful NBN algorithm. Among other things the tutorial may address following topics:

New architectures with 10 to 100 times larger capabilities than MLP processing

- Training algorithms for these new neural network architectures
- How to solve the "diminishing gradient problem" when training deep architecture
- Training neural networks without the backpropagation computations
- Adaptive second order algorithm for on-line training
- Up to 100 times faster algorithms than traditional EBP
- How to break the problem size limit for second order algorithms,
- RBF networks training resulting with 30 to 100 times smaller network size than obtained with SVM or ELM algorithms

II HOW DEEP LEARNING IS CONNECTED WITH BIG DATA

Big data term is being applied to large data sets, which cannot be processed by traditional data processing techniques. The big data area is growing very rapidly because with fact growth of technology with mobile devices, intelligent sensors, etc. it is much easier now to collect huge amount of data, which need to be processed. Big data usually has multiple dimensions and this make it much more difficult to process

This work was supported by the National Science Centre, Kraków, Poland, under Grant No. 2015/17/B/ST6/01880,

because data the processing complexity grows rapidly with dimensionality increase.

Neural networks are common tools for processing large number of data, because with neural networks the design process can be replaced with learning. However it was already demonstrated that capabilities of neural networks are growing linearly with their width and exponentially with their depth [8,9,10]. For example to solve a Parity-N problem using most commonly used SHN single hidden neural network then with n neurons it is possible to solve the Parity N problem where [8,9]

$$N = n - 1 \tag{1}$$

However if the FCC Fully Connected Cascade Architecture is used then with the same n neurons it is possible to solve the Parity-N problem as large as:

$$N = 2^n - 1 \tag{2}$$

Fig. 1 shows comparison of capabilities of SHN and FCC networks. Notice that with 15 neurons with SHN architecture only Parity-14 can be solves, while if the FCC is used with the same 15 neurons the parity-N problem as big as Parity-32767 can be solved.

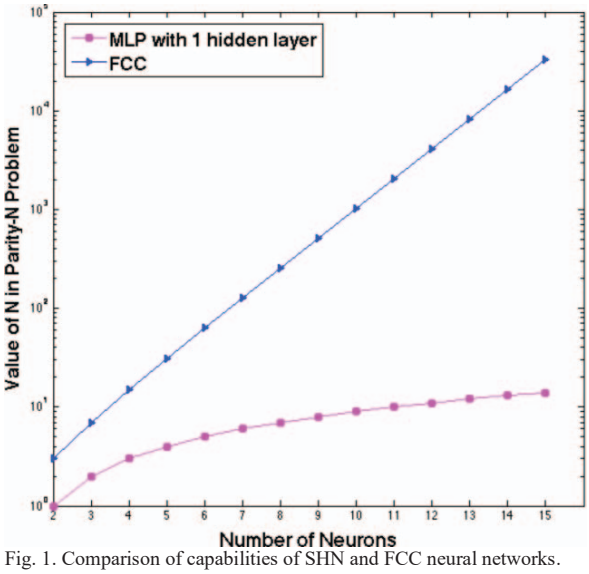

Fig. 1. Comparison of capabilities of SHN and FCC neural networks.

One can withdraw an obvious conclusion that for complex problems it is better to use a deep FCC architecture.
The problem is that deep networks are very difficult to train if there are more than 3 hidden layers. Fig. 2 shows experimental results with training of the 2-spiral problem [12] with number of layers and neurons in each layer in commonly used MLP multi-layer perceptron architectures. One may notice that if the network is deeper than 6 hidden layers it is almost impossible to train.

The prime reason for difficulties of training deep neural architectures is the vanishing gradient problem [13], where with an increase of number of hidden the error gradient is significantly reduced so the commonly used gradient based methods cannot be used. As the consequence the deep learning community [14,15,16] is trying to use a combination of several of unsupervised and supervised methods. Often special data preprocessing and transformation are used for specific problem. With this approach highly-skilled researchers of artificial intelligence has to be engaged to design a problem specific approach.

Recently it has been shown that it is possible to have an universal learning systems where human involvement can be minimal. This was possible by introduction additional concertinas in neural networks across layers in MLP architectures. Such architectures were named as BMLP bridged multi-layer perceptron architectures. Fig, 3 shows experimental results [9] of training BMLP networks, using dedicated NBN algorithm [17.18,19] for arbitrarily connected networks.

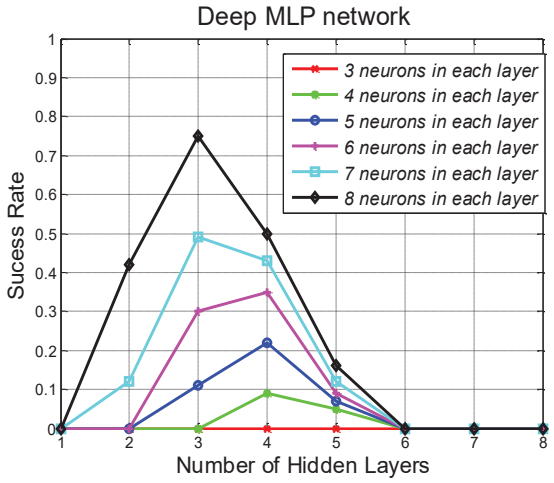

Fig. 2. Experimental results of training the 2 spiral problem with various MLP architectures.

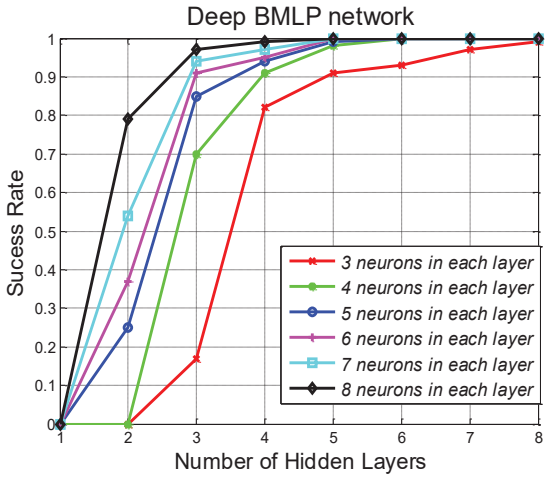

Fig. 3. Experimental results of training the 2 spiral problem with various BMLP architectures.

One may notice that BMLP networks are much easier to train than popular MLP networks and even very deep networks can be easy to train. Please notice that earlier described FCC architecture is just a special case of BMPL architecture where all neurons in the network are directly connected.

III VISUALIZATION OF BIG DATA

Big data has often many dimensions and such problems cannot be visualized by humans. Therefore, it is important to develop algorithms to visualize them just in two or three dimensions. There are many attempt were made for such visualizations. Some are more and other less successful. The issue is to do it relatively fast without losing important properties of the analyzed data set. The issue is to do it relatively fast without losing important properties of the analyzed data sets. The t-SNE, an embedding technique that is commonly used for the visualization of high-dimensional data in scatter plots, is developed t-SNE method by Laurens van der Maaten [20], claiming to be able to solve the crowding problem in SNE [21]. There are many publications about using t-SNE to visualize high-dimensional datasets, i.e. [22]. Even though Laurens van der Maaten developed variants of the Barnes-Hut algorithm and of the dual-tree algorithm [23] to accelerate t-SNE and reduce the computation time to get a visualized image of high-dimensional data sets, it is still time-consuming.

Relatively recently a new very fast and efficient algorithm for fast visualization of big data in many dimensions was published [24]. Several examples bellow will demonstrate the advantages of the new algorithm in comparison the popular t-SNE algorithm

A. Case of Iris plant [25]

This benchmark consist:

- 4 dimensions: sepal length, sepal width, petal length, and petal width
- 150 patterns
- 3 categories: Iris Setosa, Iris Versicolour, and Iris Virginic

Iris plant is a very old benchmark, which is known that patterns there are no linearly separable. Fig. 4 shows result of 2-dim visualization obtained with popular t-SNE algorithm [20] and Fig. 5 shows result obtained with a new visualization algorithm [24].

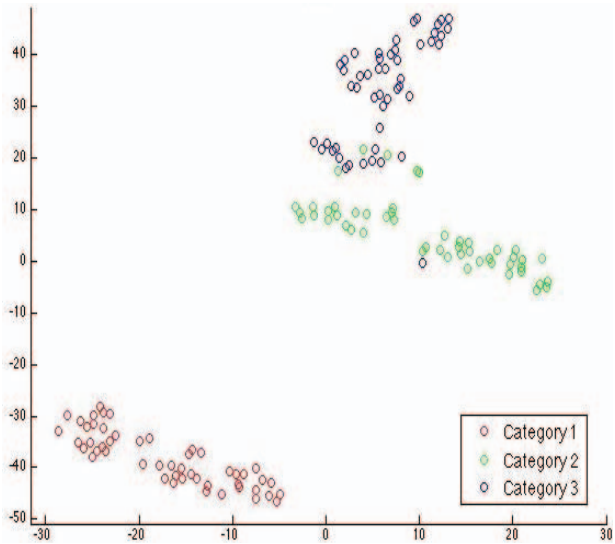

Fig. 4 Result of 2-dim visualization of Iris Plant obtained with popular t-SNE algorithm. Computation time is 1.55sec

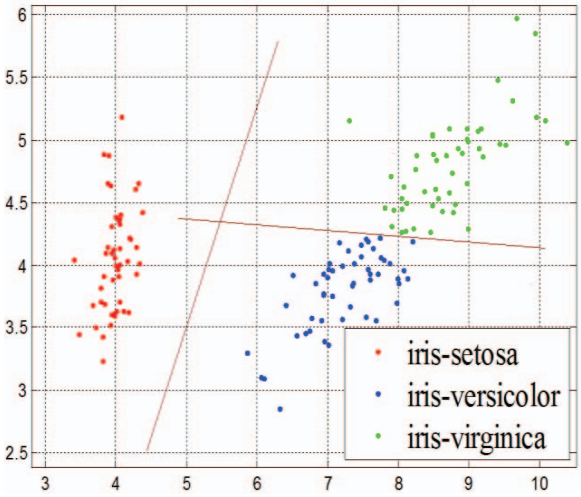

Fig. 5 Result of 2-dim visualization of Iris Plant obtained with new algorithm. Computation time is 0.084sec

B. Case of Human Activities Recognition [26]

This Human Activities Recognition benchmark consist:

- 561 dimensions, with time and frequency domain variables.
- 10,299 patterns
- 6 categories: walking, walking upstairs, walking downstairs, sitting, standing, and laying.

Fig. 6 shows result of 2-dim visualization obtained with popular t-SNE algorithm [20] and Fig. 7 shows result obtained with a new visualization algorithm [24].

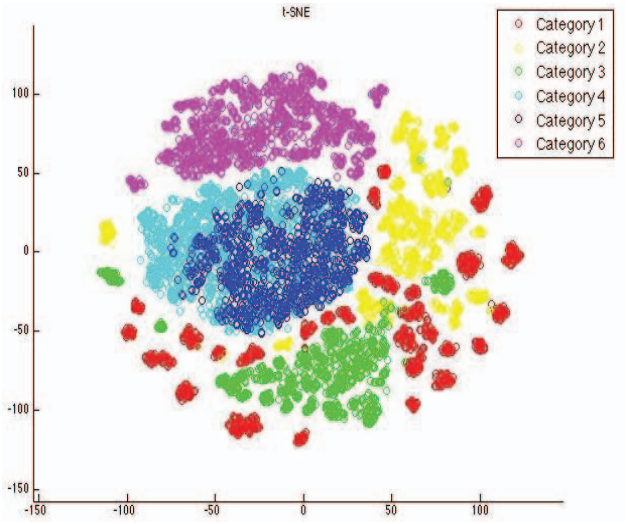

Fig. 6. Human Activity Recognition with Smartphones Data Set.Here, Category-1~Category-6 cooresponding to walking, walking upstairs, walking downstairs, sitting, standing, and laying using t-SNE algoritm. (time 1990sec)

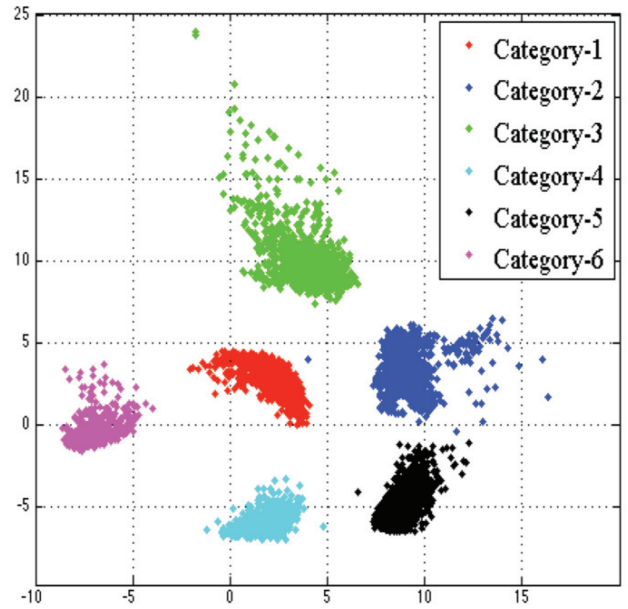

Fig. 7. Human Activity Recognition with Smartphones Data Set. Here, Category-1~Category-6 cooresponding to walking, walking upstairs, walking downstairs, sitting, standing, and laying using new algoritm. (time 5.22sec)

C. *MINST [27]*

This MINST benchmark consist:

- 28*28=784 dimensions (image)
- 60,000 patterns
- 10 categories (Digit 0~9)

Fig. 8 shows result of 2-dim visualization obtained with popular t-SNE algorithm [20] and Fig. 9 shows result obtained with a new visualization algorithm [24].

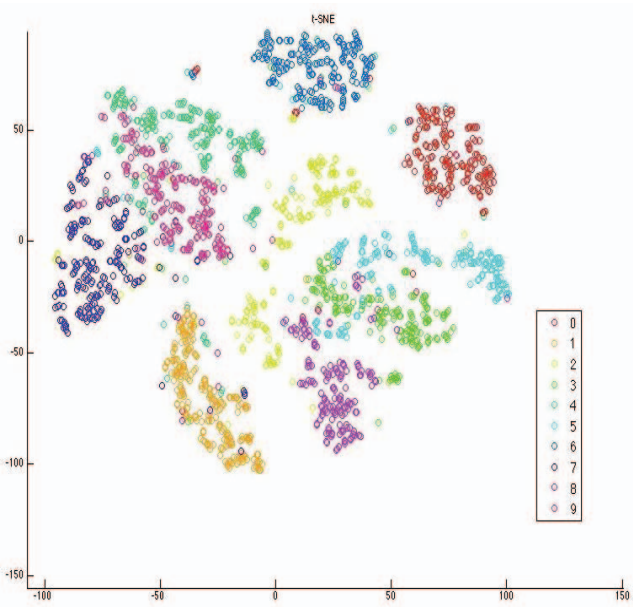

Fig. 8. MINST Benchmark visualized using popular t-SNE algoritm. Number of patterns were limited to 2,000 and processing time was 114.38 sec.

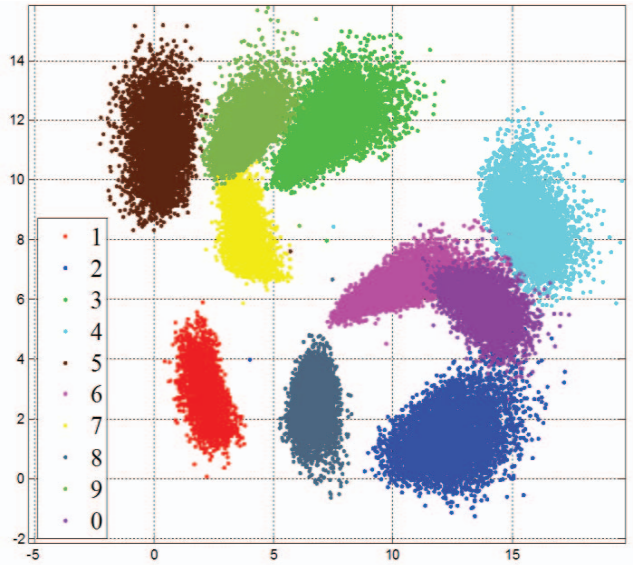

Fig. 9. MINST Benchmark visualized using the algoritm in [24]. All 60,000 patterns were ptocessed and processing time was 49.54 sec.

IV Organizing Clusters with Arbitrarily Shapes

Transitional clustering techniques, such as K-means [28] etc., can handle only clusters with round shapes. In real life clusters do not have round shapes and clustering methods are much more complex. Recently very interesting method of clustering of arbitrarily shapes was published in the *Science* [29], called FSFDP. This is very efficient method bit it is very time consuming and for 1000 patterns the processing time is about 1000 sec. Because the computation complexity of this method is $0(n^2)$ therefore for 10,000 patterns computation time

is 100,000 sec which corresponds to almost 28 hours. So the method FSFDP described in *Science* is not suitable to handle big data.

It turn out that using grid based clustering it is possible to obtain the exactly the same results but over 10,000 times faster. So the example with 1000 patterns can be processed with less than 0.1 sec and 10,000 can be clustered with less than second. The computation complexity of grid based clustering method is *0(n)*.

Figures 10 to 13 show obtained clustering results using the new grid based approach, which are identical to results obtained with the FSFDP algorithm, but the processing time is significantly smaller.

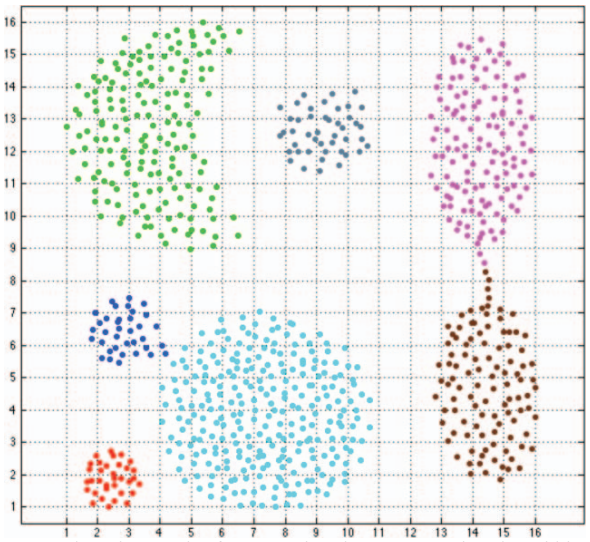

Fig. 10. Clustering result of Aggregation dataset [30] using the grid-based clustering method. Identified results can be obtained as well using the FSFDP [29]. Notice that the FSFDP requires about 414.44 seconds to process this *2spiral* dataset while the grid-based clustering method obtains the same result with about 0.0425 seconds.

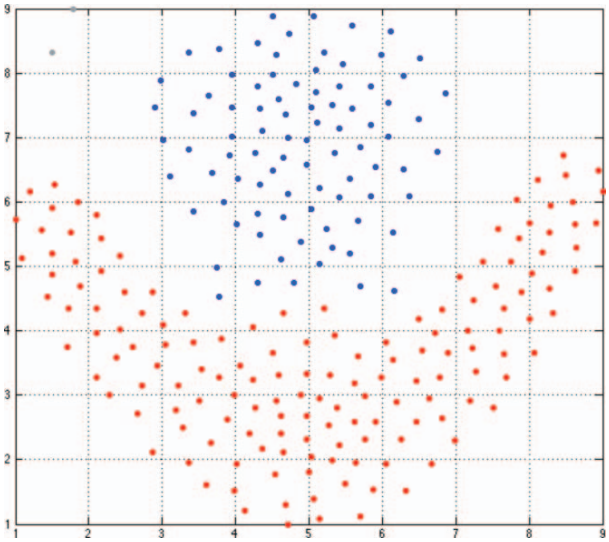

Fig. 11. Clustering result of FLAME dataset [31] with unregular shape using grid-based clustering method. Gray dots are the noise.

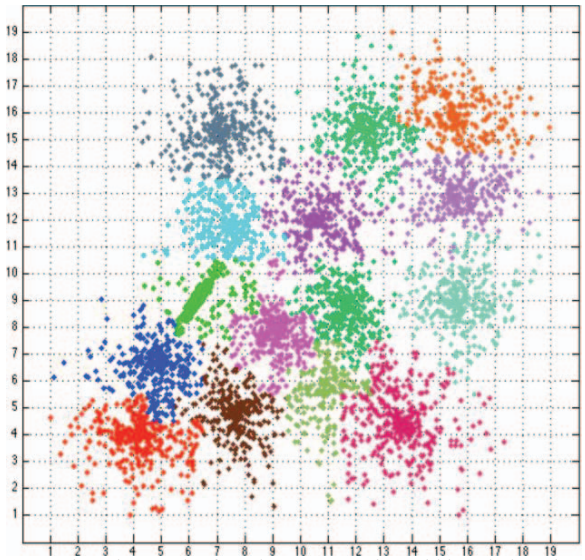

Fig. 12. Clustering result of S3 dataset [32] using grid-based clustering. Its computation time is 0.1586 seconds, as a contrast, it will cost more than 22 hours to get a final result when using the FSFDP method.

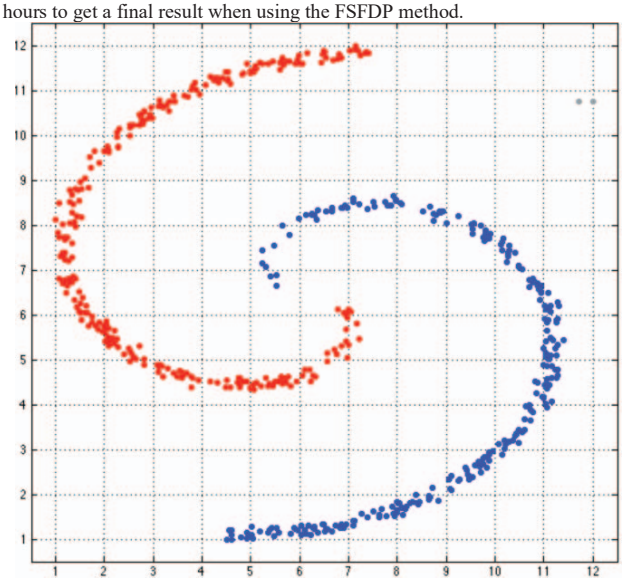

Fig. 13. Classified *2spiral* patterns (derived from [33]) using grid-based clustering method. Gray dots are the noise.

Table I shows computation time comparison of the FSFDP and grid based algorithms (GBC).

Table 1. Processing time comparisons of six benchmarks.

Benchmarks	FSFDP	GBC
Aggregation [30] (np=788;ndim=2)	414.4445 sec	0.0425 sec
FLAME [31] (np=240;ndim=2)	1.5482 sec	0.0264 sec
S3 [32] (np=5,000;ndim=2)	8.0236e+04 sec	0.1586 sec
2spiral [33] (np=537;ndim=2)	71.1021 sec	0.0109 sec

(np: number of patterns; ndim: number of dimensions.)

V. Conclusions

Three different techniques for processing big data were presented:

(1) Special neural network architecture and very efficient second order learning algorithm to train them was presented. These very powerful architecture can be easy trained no mater of their depth.

(2) A new fast algorithm for visualization of multidimensional data

(3) A new very fast algorithm for finding clusters with complex shapes was described.

All these algorithms were verified with multiple benchmarks to shown their usefulness.

References

[1] P. J. Werbos, "Back-propagation: Past and Future". *Proceeding of International Conference on Neural Networks*, San Diego, CA, 1, 343-354, 1988.

[2] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533-536, 1986.

[3] V. N. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.

[4] Smola and B. Schölkopf, A tutorial on support vector regression. NeuroCOLT2 Tech. Rep. NC2-TR-1998-030, 1998.

[5] Guang-Bin Huang; Lei Chen; Chee-Kheong Siew; , "Universal approximation using incremental constructive feedforward networks with random hidden nodes," Neural Networks, IEEE Transactions on , vol.17, no.4, pp. 879- 892, July 2006.

[6] G.-B. Huang and L. Chen, "Convex incremental extreme learning machine," Neurocomputing, vol. 70, no. 16–18, pp. 3056–3062, Oct. 2007.

[7] G.-B. Huang and L. Chen, "Enhanced random search based incremental extreme learning machine," Neurocomputing, vol. 71, no. 16-18, pp. 3460-3468.

[8] D. Hunter, Hao Yu, M. S. Pukish, J. Kolbusz, and B.M. Wilamowski, "Selection of Proper Neural Network Sizes and Architectures—A Comparative Study", *IEEE Trans. on Industrial Informatics,* vol. 8, May 2012, pp. 228-240.

[9] B. M. Wilamowski, D. Hunter, and A. Malinowski, "Solving parity-N problems with feedforward neural networks," *Proc. 2003 IEEE IJCNN,* 2546-2551, IEEE Press, 2003.

[10] P. Różycki, J. Kolbusz, T. Bartczak, B. Wilamowski, Using Parity-N Problems as a Way to Compare Abilities of Shallow, Very Shallow and Very Deep Architectures, *Lecture Notes in Computer Science, vol. 9119, ICAISC 2015*, pp. 112-122

[11] P. Różycki, J. Kolbusz, and B.M. Wilamowski, "Dedicated Deep Neural Network Architectures and Methods for Their Training", *INES'15-Intelligent Engineering Systems,* Bratislava, Sept 3-5, 2015.

[12] K. L. Lang, M.J. Witbrock, "Learning to Tell Two Spirals Apart" *Proceedings of the 1988 Connectionists Models Summer School,* Morgan Kaufman.

[13] Sepp Hochreiter. The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. *Int. J. Unc. Fuzz. Knowl. Based Syst*., 06, 107 (1998)

[14] Bengio, Y. (2009). Learning deep architectures for AI. Foundations and Trends in Machine Learning, 2(1), 1–127. Also published as a book. Now Publishers, 2009.

[15] Dahl, G. E., Ranzato, M., Mohamed, A., and Hinton, G. E.(2010). Phone recognition with the mean-covariance restricted Boltzmann machine. In NIPS'2010

[16] Socher, R., Huang, E. H., Pennington, J., Ng, A. Y., and Manning, C. D.(2011). Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In NIPS'2011

[17] B. M. Wilamowski, "Neural Network Architectures and Learning algorithms- How Not to Be Frustrated with Neural Networks", *IEEE Industrial Electronics Magazine,* vol 3, no 4, pp.56-63, (2009)

[18] B. M. Wilamowski, N. J. Cotton, O. Kaynak, and G. Dundar, "Computing Gradient Vector and Jacobian Matrix in Arbitrarily Connected Neural Networks," *IEEE Trans. on Industrial Electronics*, vol. 55, no. 10, pp. 3784-3790, Oct 2008.

[19] B. M. Wilamowski, H. Yu, "Improved Computation for Levenberg Marquardt Training," *IEEE Trans. on Neural Networks*, vol. 21, no. 6, pp. 930-937, June 2010.

[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), pp. 2579-2605, 2008.

[21] G. E. Hinton and S. T. Roweis, "Stochastic Neighbor Embedding," *Advances in Neural Information Processing Systems*, vol. 15, pp. 833-840, 2003.

[22] A. Joulin, L.J.P. van der Maaten, A. Jabri, and N. Vasilache. "Learning Visual Features from Large Weakly Supervised Data," *arXiv* 1511.0225, 2015.

[23] L. J. P. van der Maaten, "Accelerating t-SNE using tree-based algorithms," *The Journal of Machine Learning Research*, vol. 15, issue 1, pp. 3221-3245, 2014.

[24] B. Wu and B.M. Wilamowski, "An Algorithm for Visualization of Big Data in a Two-Dimensional Space", *41st Annual Conference of the IEEE IECON 2015*, pp. 53-58, 9-12 Nov. 2015.

[25] R. A. Fisher, "The use of multiple measurements in taxonomic problems", *Annual Eugenics*, 7, Part II, pp. 179-188, 1936.

[26] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra and Jorge L. Reyes-Ortiz. A Public Domain Dataset for Human Activity Recognition Using Smartphones. *21th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 24-26, April 2013.

[27] Yann LeCun, Corinna Cortes, and Christopher J.C. Burges, online at http://yann.lecun.com/exdb/mnist/, *THE MNIST DATABASE of handwritten digits*, last seen on 23 March 2016.

[28] J.B. MacQueen, "Some Methods for classification and Analysis of Multivariate Observations," *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, University of California Press., pp. 281-297, 1967.

[29] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492-1496, 2014.

[30] A. Gionis, H. Mannila, and P. Tsaparas, "Clustering aggregation," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1): pp.1-27, 2007.

[31] Limin Fu and Enzo Medico, "FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data," *BMC Bioinform* 8(1): 3, 2007.

[32] P. Fränti and O. Virmajoki, "Iterative shrinking method for clustering problems," *Pattern Recognition*, 39(5), pp. 761-775, 2006.

[33] Kevin J. Lang and Michael J, Witbrock, "Learning to Tell Two Spirals Apart," *in Proceedings of the 1988 Connectionist Models Summer School*, 1988.