**UDC 004.8**

# LARGE SCALE MULTILAYER PERCEPTRON

B.Sc. Sascha Jonas, B.Sc. Martin P. Herberg, Prof. Dr. Christian Herta,
Prof. Dr. Ingo Claßen

Hochschule für Technik und Wirtschaft Berlin
Germany, Berlin
sascha.jonas@student.htw-berlin.de, m.herberg@student.htw-berlin.de,
christian.herta@htw-berlin.de, ingo.classen@htw-berlin.de

*In this study different paradigms of distributed computing for scaling out the training of neural networks are compared. The ability of MapReduce/In-Memory Computing and bulk-synchronous parallel computing (BSP) was studied. Comparable distributed training algorithms based on mini-batch and big-batch scheme were developed in the different paradigms. For this purpose we used Spark (In Memory Computing/Map Reduce) and Apache Hama (BSP). The scalability of the different approaches has been determined by the training of an autoencoder with 10,000 input and output neurons.*

*Keywords: autoencoder, bulk-synchronous parallel computing (BSP), Deep Learning, Multilayer Perceptron, Spark, Apache Hama.*

It has been shown in the last year that deep learning and unsupervised feature learning based on autoencoder [1, 2] can achieve state-of-the-art performance [3] without the need of feature engineering. The use of big amounts unlabelled data for training of such neural networks requires parallelism of the training procedure. It was further shown that the training of large neural networks is in principle highly scalable and can beat the performance of GPU-based frameworks [4].

In this study, two approaches for the training of multilayer perceptrons in a computer cluster were evaluated. These are based on Apache Hama [5] which is an implementation of the bulk-synchronous parallel computing [6] and Spark a distributed in-memory cluster system [7].

In both approaches the principle of distributed learning is based on a two step procedure. The two steps are successively executed until the stopping criterion is fulfilled.

During the first step the weight changes are computed on each (slave) node by the backpropagation algorithm [8] using a part of the whole training data set. This can be done in parallel without any communication between the (slave) nodes. During this step no weight changes are applied to the current neural network. The second step is to combine all weight changes computed by different slaves. For this purpose the changes have to be transferred over the computing network to a master node. The master adds up all weight changes and updates the neural network. Then the new neural network can be transferred to all slaves and step 1 can be executed again. If all training data (cumulative on all slaves) is used in step 1 it's called batch learning [8]. The use of only a small part of the whole training data in step 1 is called mini batch.

For this procedure the multilayer perceptron model has the restriction that it must fit into the main memory of a cluster computer. But this is not a strong restriction. Typically a huge amount of storage can be available in main memory because machines can be equipped with up to 128 GB today.

Both approaches allow the iterative solution of tasks, where the required data can be kept in main memory. A repeated loading of data from the hard disk like the Hadoop MapReduce algorithm can be avoided. The explicit support of iterative algorithms is an additional advantage, so a better performance in comparison can be expected.

**REFERENCES**

1. Hinton G. E., Salakhutdinov R. R. Reducing the dimensionality of data with neural networks, science: www.sciencemag.org, 2006.
2. Hinton G. E. Learning multiple layers of representation // ScienceDirect.— 2006.— Vol. 10.— N 11.
3. Le Q., Ranzato M., Monga R. et al. Building high-level features using large scale unsupervised learning // Proceed. of International Conference in Machine Learning.— United Kingdom, Edinburgh.— 2012.
4. Dean J., Corrado G., Monga R. et al. Large scale distributed deep networks // Neural Information Processing Systems (NIPS).— United States, Nevada, Lake Tahoe.— 2012.
5. A. S. Foundation, "Apache Hama". http://hama.apache.org/.
6. Valiant L. G. A bridging model for parallel computation // Commun. ACM.— 1990.— Vol. 33, N 8.— P. 103—111.
7. U. B. AMPLab "Spark — Lightning-Fast Cluster Computing".— UC Berkeley AMPLab, [Online]. Available: http://spark-project.org.
8. Bishop C. Neural Networks for pattern recognition.— Oxford University Press, 1996
9. Evert S. The statistics of word cooccurrences: Word Pairs and Collocations.— Institut für maschinelle Sprachverarbeitung, University of Stuttgart, 2005.
10. Lee H., Grosse R., Ranganath R., Ng A. Convolutional deep belief networks for scalable unsupervisedlearning of hierarchical representations // Proceed. of the 26th International Conf. on Machine Learning.— Canada, Montreal: Omnipress.— 2009.— P. 609—616.
11. Vincent P., Larochelle H., Bengio Y., Manzagol P.-A. Extracting and composing robust features with denoising autoencoders // Proceed. of the 25th International Conf. on Machine Learning.— 2008.— P. 1096–1103.
12. Xing, the social career network. [Online]. Available: http://www.xing.com/.
13. Mahout - scalable machine learning and data mining. [Online]. Available: http://mahout.apache.org/.
14. Hadoop - reliable, scalable, distributed computing. [Online]. Available: http://hadoop.apache.org/.
15. Dunning T. Accurate methods for the statistics of surprise and coincidence // Journal Computational Linguistics.— 1993.— Vol. 19, Iss. 1.

———————

С. Джонас, П. М. Герберг, К. Герта, И. Классен

**Крупномасштабный многослойный перцептрон.**

Сравниваются различные парадигмы распределенных вычислений для масштабирования обучения нейронных сетей. Изучена возможность вычислений методом MapReduce/In-Memory и методом синхронных параллельных вычислений. Сопоставимые распределенные алгоритмы обучения на основе mini-batch и big-batch схем были разработаны в различных парадигмах. Для этой цели были использованы Spark (вычисления In Memory/Map reduce) и Apache Hama (синхронные параллельные вычисления). Масштабируемость различных подходов была определена путем подготовки автокодировщика с 10.000 входных и выходных нейронов.

Ключевые слова: *автокодировщик, синхронные параллельные вычисления, Deep Learning, многослойные перцептрон, Spark, Apache Hama.*