A Novel Pruning Model of Deep Learning for Large-Scale Distributed Data Processing

Yiqiang Sheng[†a)], Chaopeng Li[††], Jinlin Wang[†], Haojiang Deng[†] and Zhenyu Zhao[*]
[†]National Network New Media Engineering Research Center,
Chinese Academy of Sciences, Beijing 100190, China
[††]University of Chinese Academy of Sciences, Beijing 100049, China
[*]University of Science and Technology of China, Anhui 230026, China
[a)]E-mail: shengyq@dsp.ac.cn

***Abstract*—In this paper, we propose a novel pruning model of deep learning for large-scale distributed data processing to simulate a potential application in the geographical neighbor of Internet of Things. We formulate a general model of pruning learning, and we investigate the procedure of pruning learning to satisfy hard constraint and soft constraint. The hard constraint is a class of non-flexible setting without parameter learning to match the structure of distributed data. The soft constraint is a process of adaptive parameter learning to satisfy an inequality without any degradation of accuracy if the size of training data is large enough. Based on the simulation using distributed MNIST image database with large-scale samples, the performance of the proposed pruning model is better than that of a state-of-the-art model of deep learning in case of big data processing.**

***Keywords*—deep learning, big data, distributed data, cloud computing, internet of things.**

I. INTRODUCTION

Deep learning [1][2][3] which aims at discovering higher layers of representations with abstract concepts has achieved impressive performance in many applications such as image recognition, speech recognition, multimedia data processing, information retrieval, human motion modeling and so on. With the rapid development of communication technologies, however, smart devices with huge amount of geo-distributed data have been paving the way for the new era of distributed big data and Internet of Things (IoT) [4][5]. Efficient distributed big data processing based on things connected to Internet has become a crucial requirement for many industrial applications and services.

By collecting a number of computational resources together, cloud computing [6] was one of the state-of-the-art platforms to partly meet the challenge of big data by centralized data storage and data processing [7][8][9]. However, ever-growing requirements and techniques such as mobility-aware spatio-temporal event processing and mobility-driven distributed event processing [10][11] have been making the current cloud computing inefficient due to expensive communication of distributed data. For examples, cloud-based real-time services of image, video, and multimedia are expensive because the distances of communication between cloud and Internet of Things are geographically far.

To improve the performance of cloud computing, geo-distributed computing such as on-site service [12] and fog [13] extends cloud to the verge of the Internet to geographically improve the communication cost and the quality of real-time services in many scenarios, such as smart traffic lights in vehicular networks. Since the computational resources of distributed computing is in the geographical neighbor of IoT, the communication cost is reduced. It makes easy to provide real-time services for end-users, but it leads to another challenge to improve the efficiency of geo-distributed data processing and to reduce the communication cost between cloud computing and geo-distributed computing.

The main contributions of this research are as follows. (1) We propose a pruning model of deep learning for large-scale data processing to simulate a potential application in the geographical neighbor of Internet of Things. (2) We formulate the pruning learning to satisfy hard constraint and soft constraint, and we define a cost function to evaluate the generation error including the training error, the reconstruction error as well as the regularization terms. (3) We present the procedures of pruning learning, and we modify the existing MNIST handwritten image database as distributed benchmark to simulate geo-distributed data processing. (4) Simulation shows that the communication cost of the proposal is much better than that of a state-of-the-art model of deep learning with shorter learning time and without degradation of accuracy for big data processing. Simulation also shows that the cost of above improvement is the speed of convergence with respect to data size, but the final learning time is actually improved.

The rest of this paper is organized as follows. Section 2 provides a review of foundations and related works. Section 3 formulates the pruning learning. Section 4 is the procedure of pruning learning with constraints. Section 5 shows the simulation of the proposal comparing with a state-of-the-art model. Finally, Section 6 concludes this research.

II. FOUNDATIONS AND RELATED WORKS

The abstract of learning a mapping from training data from the viewpoint of function optimization was discussed by A. Hirabayashi and H. Ogawa [14] with respect to projection learning, partial projection learning, and averaged projection learning to obtain good generalization capability, and devised the concept of a family of projection learning which includes

three kind of projection learning. This provided a framework to discuss a general learning model.

As one of the pilot researches, a feature extraction scheme as a pre-processor for neural network classification was introduced by C. H. Chen and G. G. Lee [15]. It showed the feature extraction scheme implemented by a non-stationary Gaussian Markov random field could provide effective features for neural network classification. By using Bayesian learning, the further enhancement of the segmented result was achieved. The formulation of the maximum a posterior estimator was based on Gibbs prior assumption. The maximum a posterior estimator could be found from neural networks such as the Boltzmann machines. That is one of the foundations of the proposed pruning learning model.

Based on the semi-supervised deep learning algorithms, D. Luo, R. Yang and J. Huang [16] proposed a method to detect the double compressed adaptive multi-rate audio, which was served as a tool for authenticating the originality of audio recordings and detecting the forgery positions. On the other hand, R. Rui and C. Bao [17] proposed a supervised learning algorithm for automatic classification of individual musical instrument sounds deriving from the idea of supervised non-negative matrix factorization algorithm. These are two of typical applications of semi-supervised and supervised learning model.

For distributed data processing, R. Tudoran et al [18] proposed a system of data management for scientific applications running across geographically distributed sites. The environment-aware solution monitors and models the global cloud infrastructure, and offers predictable data handling performance for transfer cost and time. It provides the applications with the possibility to set a trade-off between money and time and optimizes the transfer strategy accordingly. The system was validated on Microsoft's Azure Cloud across the 6 EU and US data centers. The experiments show that the system is able to model and predict well the cloud performance and to leverage this into efficient data dissemination.

For distributed deep learning system using MapReduce, K. Zhang and X. Chen [19] investigated a deep learning model for restricted Boltzmann machines and back-propagation algorithm. As one of the state-of-the-art models of deep learning, deep belief net was trained in a distributed way by stacking a series of distributed restricted Boltzmann machines for pre-training and a distributed back-propagation for fine-tuning. Through validation on the benchmark data sets of various practical problems, the experimental results demonstrated that the distributed deep belief net are amenable to large-scale data with a good performance in terms of accuracy and efficiency.

However, it is difficult for the above researches to meet the increasingly higher requirement of distributed data processing. Especially, the communication cost of distributed big data has been one for the most serious problems. The state-of-the-art model such as distributed deep belief net [19] was limited to cloud computing. It is a big issue for distributed big data processing to improve the overall performance such as communication cost, computational efficiency, scalability, security, etc.

III. Formulation of Pruning Learning with Constraints

Deep learning is a class of neural networks with deep architecture. It attracts wide attention due to the excellent performance of applications [20][21] with open source tools and libraries [22][23].

Let the data sets be marked as D which includes D_{train} as train sets, D_{valid} as validation sets to select hyper-parameter and D_{test} as test sets to evaluate the generalization error with a fair comparison between different models. Each data set D could be a set of labeled data $Y = \{x_i, y_i\}$ or a set of unlabeled data $X = \{x_i\}$. Let N_L be the number of labels. Let N_D be the number of data. Let y be output. Let x be input. Let F be the mapping of the model. Let θ be the set of all parameters.

The mapping of a pruning learning model is defined as the following.

$$y = F(x = x_i \mid \theta^p \subset \theta) \tag{1}$$

where θ^p is the set of necessary parameters after the pre-training of pruning learning.

The training error with respect to each label y_i is defined as the following.

$$E_i^t = \frac{1}{2}\left\| y - y_i \right\|^2 \tag{2}$$

If the mapping between input and output is bidirectional or symmetric, the reconstruction error with respect to each input x_i is defined as the following.

$$E_i^r = \frac{1}{2}\left\| \hat{x} - x_i \right\|^2 \tag{3}$$

The final objective is the generation error which measure the error of a model with respect to a generalized data set. A bidirectional cost function of a general learning model is defined as the following.

$$E = \sum_{i=1}^{l} E_i^t + \sum_{i=1}^{n} E_i^r + \sum_{i=1}^{q} \lambda_i L_i \tag{4}$$

where l is the number of labeled data, n is the number of unlabeled and labeled data, q is the number of regularization terms, E_i^t is the training error with respect to each labeled datum $\{x_i, y_i\}$, E_i^r is the reconstruction error with respect to each unlabeled and labeled datum, λ_i is the weights of regularization terms, L_1 is the Lasso regularization term, L_2 is the Ridge regularization term, and so on.

To evaluate the training error more efficiently, the pruning version of training procedure in experiments could be defined

as minimizing the negative log-likelihood (*NLL*) with or without regularization terms as the following.

$$NLL(\theta^p, D) = -\sum_i \log P(y = y_i \mid x_i, \theta^p) \quad (5)$$

To improve the scalability of deep learning model and match the distributed structure of big data, the hard constraint is designed as a class of non-flexible parameter setting without learning process. The sets of big data are divided into M sub sets according to the geographical sites, where M is the number of end-user's groups. As a learning system, deep neural network with H +2 layers, where $0 \le h \le H+1$, is divided two levels by the layer of $h = H^*$ in the vertical direction of the system. The layer of $h = 0$ is the input, the layer of $h = H^*$ is the partition, and the layer of $h = H+1$ is the output. Accordingly, the first level of the system is from the layer of $h = 0$ to the layer of $h = H^*$, and the second level is from the layer of $h = H^*$ to the layer of $h = H+1$. Then, the first level of the system is divided M sections in the horizontal direction of the system, where M is the number of end-user's groups. The weights among different sections are zero as the hard constraint.

Let the width of the deep learning system with H +2 layers be N_w, then the width of each section in the first level of the system is d_m, where $1 \le m \le M$, to satisfy $N_w = \sum d_m$. Let the set of all neurons in the m^{th} section of the first level be S_m. Let any neuron on the $(h\text{-}1)^{\text{th}}$ layer in the m^{th} section be $S_{i(m),(h-1)} \in S_m$, where $1 \le h \le H^*$. Let the set of all neurons in the o^{th} section of the first level be O_m. Let any neuron on the l^{th} layer in the o^{th} section be $S_{j(o),\,h} \in S_o$, where $m \ne o$. Then, the weights between $S_{i(m),(h-1)}$ and $S_{j(o),\,h}$ satisfy a hard constraint of $w_{i(m),\,j(o),\,h} = 0$.

To improve the communication cost without degradation of accuracy, the soft constraint, named min-k-degree inequality, is designed as an adaptive process of parameter learning. For a non-hierarchical network, the min-k-degree inequality is defined that the out-degrees of each neuron in positive direction are at least k, where k is a hyper parameter. The positive direction is from input to output.

Without the loss of generation, we assume all neurons are completely connected with adjustable weights. Let the number of all neurons be N. Let the network of neurons be connected with non-zero weights, and let the non-connected weight be zero. For the output of any neuron x_j, where $j = 1, 2, ..., N$, with the set of its inputs $X = \{x_i\}$, where $i = 1, 2, ..., k_j$, the following equation is satisfied.

$$x_j = f\left(\sum_{i=1}^{k_j} w_{ij} x_i + b_j\right) \quad (6)$$

where f is the activation function, b_j is the bias, and w_{ij} is the weight from neuron i to neuron j. For a general non-hierarchical network, the min-k-degree inequality is defined as the following.

$$k_j \ge k \quad (7)$$

where k is a user-defined parameter.

For the h^{th} hidden layer, the min-k-degree inequality is defined as the following.

$$k_j^{(h)} \ge k^{(h)} \ge k \quad (8)$$

where $k_j^{(h)}$ is the number of the set of inputs $X = \{x_i\}$ for the output of neuron $x_j^{(h)}$ on the h^{th} hidden layer, $k^{(h)}$ is a user-defined parameter for the h^{th} hidden layer.

To evaluate the communication cost, a computer cluster was described as a graph (V, E, ED, EF), where V is the vertex set $\{i\}$, E is the edge set $\{e_{ij}\}$, ED is the set of edge distance $\{ed_{ij}\}$ which is the geographical distance of communication between cloud and internet of things, and EF is the set of edge flow $\{ef_{ij}\}$ which is the data flow of parameter communication between cloud and internet of things. The communication cost was defined as $CC = \sum_{ij} ed_{ij}\, ef_{ij}$. For two-stage tree structure of computer cluster with a core machine and M verge machines, the communication cost was simplified as $CC_{tree} = \sum_i ed_i\, ef_i$. where ed_i is the geographical distance between the core machine and the verge machines, ef_i is the data flow between the core machine and the verge machines using parameter communication, and M is the number of the verge machines. In case of the parameter communication, the data flow ef_{ij} of communication between cloud and Internet of Things equals to the number of parameters $N_{ij}(\theta)$.

The pre-training procedure is designed to minimize the total number of parameters $N(\theta) = \sum_{ij} N_{ij}(\theta)$ with some given constraints with respect to the reconstruction error or the generation error.

$$\theta^p = \arg\min_{\theta} N(\theta) \quad (9)$$

where θ is the set of all parameters and θ^p is the set of parameters after the pre-training of pruning learning.

Then, the training procedure is designed to minimize the cost function with respect to the generation error as the following.

$$\theta^{p*} = \arg\min_{\theta^p} E(\theta^p, D) \quad (10)$$

where θ^{p*} is the set of optimized parameters after the whole training procedure of pruning learning.

IV. Procedure of Pruning Learning with Constraints

The pruning learning is designed to construct the deep learning system by gradually pruning the connection of neural network until all constraints are satisfied during the pre-training stage of minimizing the cost function. It is an

unsupervised learning method using the set of unlabeled data. The detailed pre-training procedure is as the following.

Procedure (I): The pre-training procedure of the pruning learning.

Input: A given model of deep learning with the set of parameter θ and the set of unlabeled data $\{x_i\}$.

Output: A deep learning model with the set of pruning parameters θ^p to satisfy min-*k*-degree inequality.

Step 1: Coding from Input layer as h=0 to Output layer as h=H+1, where H is the number of hidden layers. Let h be -1.

Step 2: Initiating the parameters between the h^{th} layer and the $(h+1)^{th}$ layer. Let h be h+1.

Step 3: Adjusting the weights between the h^{th} layer and the $(h+1)^{th}$ layer and the bias of the $(h+1)^{th}$ layer to minimize the cost function of the current two layers by using the set of unlabeled data.

Step 4: If the weight is smaller than a threshold value, calculating ΔE_i^r which is the change of reconstruction error with and without the current connection. Deleting the current connection with the probability of $\min[1, \exp(-\Delta E_i^r/E_i^r)]$.

Step 5: Judging that the out-degrees of the current neuron in the positive direction is equal to k or not. If the answer is NO, returning to step 3. If the answer is YES, shifting to the next step.

Step 6: Judging that the cost function is smaller than the second threshold value or not. If the answer is NO, returning to step 3. If the answer is YES, shifting to the next step.

Step 7: Judging that the current h is larger than H or not. If the answer is NO, returning to step 2. If the answer is YES, ending the procedure.

V. SIMULATION

The evaluation of all models with optimization algorithms was implemented by the simulation of a computer cluster which consisted of core machines and verge machines connected by 10Gbs switch using Python 2.7.8 [22] and Theano 0.6 [23]. The core machine had 24 processors of 2.10GHz Intel Xeon E5-2620 CPU 32GB RAM and NVIDIA GPU Grid K2 8GB GDDR5. The verge machine had two processors of 2.50GHz Intel Core i7-4710MQ CPU 16GB RAM and NVIDIA GEFORCE GTX 760 GPU 2GB GDDR5.

To simulate the structure of distributed data from Internet of things, each MNIST handwritten image was splitted into M parts as distributed MNIST database to allocate each part to one of verge machines, and each splitted data set is with 50, 000 samples for training to optimize the given parameters, 10, 000 samples for validation to select hyper-parameter and 10,000 samples for testing to evaluate the generalization error with a fair comparison between different models. Since the MNIST images were size-normalized and centered in a fixed size of 784 pixels, all splitted images were with a fixed size of 784/M pixels. The intensity was normalized to have a value in [0, 1]. The labels are integers in [0, 9] indicating which digit the image presents. Besides, the training set of MNIST with 50, 000 samples was copied with X times to evaluate the scalability performance, where X = 2, 3, 4, ..., 10, ..., 50, ..., 100, ..., 1000, 10000.

We set M = 5 and k = 10 to test the performance of the algorithms. The input layer has 784 neurons. The output layer has 10 neurons. The default setting of hidden layers for the tested model in each machine is as follows. The first hidden layer has 580 neurons. The second hidden layer has 450 neurons. The third hidden layer has 310 neurons. The fourth hidden layer has 160 neurons. The fifth hidden layer has 75 neurons. The regularization terms (L_j) include L_1 and L_2 with $\lambda_1 = 0.00001$ and $\lambda_2 = 0.00009$. The remaining parameters of each tested model are gradually optimized during the same parameter learning process of deep learning systems with different constraints.

The training procedure of the pruning learning is as the following.

Procedure (II): The parameter optimization process of the pruning learning.

Input: The set of pruning parameters of a given model of deep learning θ^p and the set of unlabeled data $\{x_i\}$ and labeled data $\{x_i, y_i\}$.

Output: A minimized cost function $E(\theta, D)$ with an optimized set of parameters θ^{p*}.

Step 1: Let h be 1. Training the first layer (h) as a restricted Boltzmann machine with given constrains that models the raw input $x = h_0$ as its visible layer (h-1).

Step 2: Using the above first layer (h) to obtain a representation of the input that will be used as data for the second layer (h+1). Two common solutions exist. This representation can be chosen as being the mean activation values $P(h_1=1 \mid h_0)$ or samples of $P(h_1 \mid h_0)$.

Step 3: Training the second layer (h+1) as a restricted Boltzmann machine with constrains and taking the transformed data as training examples for the visible layer of that restricted Boltzmann machine with constrains.

Step 4: Let h be h+1. Iterating Step 2 and Step 3 for the desired number of layers (h=H) with constrains, each time propagating upward either samples or mean values.

Step 5: Fine-tuning all the parameters of this deep architecture with respect to the cost function $E(\theta, D)$ or its proxy as a training criterion, i.e. the negative log-likelihood (NLL) with regularization terms.

We evaluated a non-pruning model of deep belief net (DBN) [19], i.e. one of the state-of-the-art deep learning models, which was trained by stacking a series of restricted Boltzmann machines for pre-training and a back-propagation for fine-tuning in a distributed way to get the best performance in terms of error rate, running time and communication cost. The communication cost of the parameter optimization processes of the model is shown in Fig. 1. We can see that the communication is expensive with respect to the scaling of data size, so it is a necessary and urgent problem that we have to solve. The learning time of the parameter optimization processes of the model is shown in

Fig. 2. The learning time is gradually increased with respect to the scaling of data size.

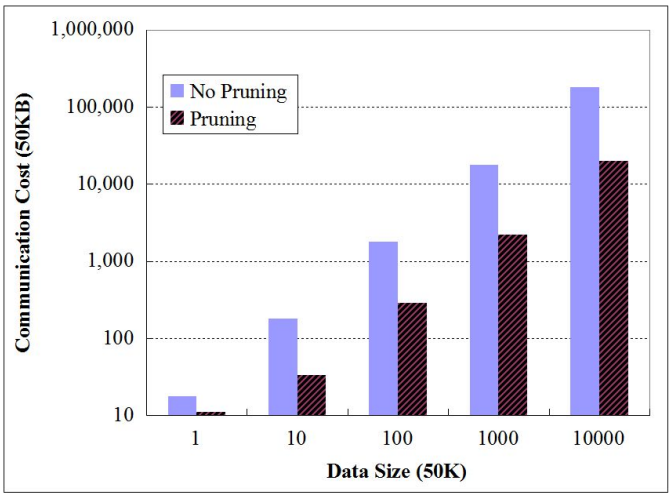

Fig. 1 Comparison of the communication cost of distributed learning systems.

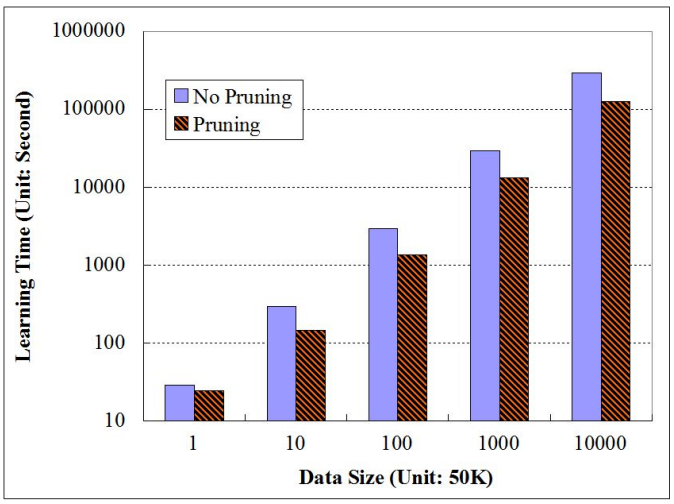

Fig. 2 Comparison of the learning time of the parameter optimization processes of distributed learning systems.

We evaluated the proposed pruning learning with hard constraint and min-k-degree inequality in order to improve the communication cost and the computational performance of distributed data processing. As shown in procedure (I), the pruning learning is used to construct an efficient deep learning system by gradually pruning the connection of a given neural network until all constraints are satisfied. The communication cost of the parameter optimization processes of the proposed pruning learning is shown in Fig. 1. For distributed data processing, the communication cost of the proposal is much smaller than that of the state-of-the-art model of deep learning. The learning time of the parameter optimization processes of the proposed pruning learning is shown in Fig. 2. When the data size is 50, 000, the learning time is much faster than that of the non-pruning model. When the data size increases 1000 times, the learning time is more than 2 times faster than that of the state-of-the-art deep learning model.

In fact, the cost of above improvements is the slower speed of convergence with respect to data size, but the total learning time is actually improved because the pruning leads to the reduction of the learning time for similar data size by minimizing the number of parameters. Let the convergence speed denote the improvement of error rates divided by the data size as the following.

$$CS_i = \frac{N_0(d)(E_0 - E_i)}{N_i(d)} \tag{11}$$

where CS_i is the current convergence speed, E_i is the current error rate, E_0 is the basic error rate, $N_i(d)$ is the current data size and $N_0(d)$ is the basic data size. Let the relative cost denote the relative degradation of the convergence speed. Based on experiments, the details of error rate, convergence speed and relative cost with respect to data size are shown in Table 1.

Table 1 The cost of above improvements is the speed of convergence with respect to data size (Basic error rate: 10%; Basic data size: 50K)

Data Size (50K)	Error Rate (%)		Convergence Speed (%)		Relative Cost (%)
	No Pruning	Pruning	No Pruning	Pruning	
1	4.92	5.98	5.08	4.02	20.92
2	4.46	5.49	2.77	2.26	18.54
3	4.09	5.20	1.97	1.60	18.75
4	3.77	4.92	1.56	1.27	18.45
5	3.53	4.68	1.29	1.06	17.85
6	3.38	4.46	1.10	0.92	16.40
7	3.30	4.31	0.96	0.81	15.07
8	3.15	4.11	0.86	0.74	14.03
9	3.07	3.97	0.77	0.67	12.95
10	2.95	3.77	0.70	0.62	11.52

The basic error rate to calculate the convergence speed and the relative cost is 10%. The error rate of the non-pruning model and the proposal is gradually improved with respect to the data size of learning. When the data size is small, the error rate of the proposal is not as good as that of the state-of-the-art model of deep learning. And the relative cost of improvement with 10 times of data size is high (near 10%). But it does not matter, because we care more about the final result after learning and the error rate is gradually improved with the data size increases. And the relative cost of improvement with 30 times of data size is much lower (near 3%). When the data size is large enough, the error rate or accuracy of the proposal is almost as good as that of the state-of-the-art model of deep learning [19]. That is to say, the big improvement of communication cost and learning time is with

no degradation of accuracy when the number of training samples are large enough according to the simulation.

VI. CONCLUSION

This paper has proposed a pruning model of deep learning to process distributed big data more efficiently. We formulate a general model of pruning learning. We investigate the pruning learning with hard constraint and soft constraint to satisfy a so-called min-*k*-degree inequality. Simulation shows the considerable improvement of performance including learning time and communication cost with scalability by comparing with the state-of-the-art pruning model of deep learning. For large-scale distributed data processing, the communication cost of the proposal is much smaller than that of the state-of-the-art model of deep learning with shorter learning time and without any degradation of accuracy when the number of training samples are large enough. The cost of the above improvement is the slower convergent speed with respect to data size, but the total learning time is actually improved for distributed big data processing.

ACKNOWLEDGMENT

This work is supported by CAS Pioneer Hundred Talents Program and Special Fund for Strategic Pilot Technology of CAS under Grant No. XDA06040501. The authors would like to thank Prof. X. Zeng, Dr. L. Wang and Dr. W. Qi at Chinese Academy of Sciences, thank Prof. A. Takahashi and Prof. S. Ueno at Tokyo Institute of Technology, and thank the anonymous reviewers for their valuable comments.

REFERENCES

[1] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," Science, vol. 313, no. 5786, pp. 504-507, 2006.

[2] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," Advances in Neural Information Processing Systems, vol. 19. p. 153, 2007.

[3] Y. LeCun, K. Kavukcuoglu and C. Farabet, "Convolutional networks and applications in vision," Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS), pp. 253-256, 2010.

[4] S. Priyankara, K. Kinoshita, H. Tode, and K. Murakami,"A generalized spatial boundary analysis method for clustering/multi-hop hybrid routing in Wireless Sensor Networks," Proceedings of IEEE GLOBECOM Workshops (GC Wkshps), pp.281 - 286, 2011.

[5] M. Amarlingam, I. Adithyan, P. Rajalakshmi, Y. Nishimura, M. Yoshida, and K. Yoshihara,"Deployment adviser tool for wireless sensor networks," Proceedings of IEEE World Forum on Internet of Things (WF-IoT), pp.452 - 457, 2014.

[6] Z. Xiao, W. Song, and Q. Chen, "Dynamic Resource Allocation Using Virtual Machines for Cloud Computing Environment", IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 6, June 2013.

[7] Y. Yang, Y. Zhou, L. Liang, D. He, Z. Sun,"A Sevice-Oriented Broker for Bulk Data Transfer in Cloud Computing,"Proceedings of 9th International Conference on Grid and Cooperative Computing (GCC), pp.264-269, 2010.

[8] K. Yang, X. Jia, "An Efficient and Secure Dynamic Auditing Protocol for Data Storage in Cloud Computing", IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 9, September 2013.

[9] Y. Low, J. Gonzalez, A. Kyrola, D. Bickson, C. Guestrin, and J. M. Hellerstein, "Distributed graphlab: A framework for machine learning in the cloud," Proceedings of the VLDB Endowment (PVLDB), pp. 716-727, 2012.

[10] K. Hong, D. Lillethun, U. Ramachandran, B. Ottenwälder, and B. Koldehofe, "Opportunistic spatio-temporal event processing for mobile situation awareness," in Proceedings of the 7th ACM International Conference on Distributed Event-based Systems, pp. 195–206, 2013.

[11] B. Ottenwalder, B. Koldehofe, K. Rothermel, and U. Ramachandran, "Migcep: Operator migration for mobility driven distributed complex event processing," in Proceedings of the 7th ACM International Conference on Distributed Event-based Systems, pp. 183–194, 2013.

[12] J. Wang, J. You, H. Deng, Z. Liu, G. Chen, "A System and Method of On-Site Service Provision", Patent Cooperation Treaty (PCT), CN2014/081300, 201410083167.9, 2014.

[13] S. J. Stolfo, M. B. Salem, A. D. Keromytis, "Fog Computing: Mitigating Insider Data Theft Attacks in the Cloud", Proceedings of IEEE Symposium on Security and Privacy Workshops (SPW), pp. 125-128, 2012.

[14] A. Hirabayashi and H. Ogawa, "A class of learning for optimal generalization," Proceedings of International Joint Conference on Neural Networks, Vol.3, pp.1815 - 1819, 1999.

[15] C. H. Chen and G. G. Lee, "Multiresolution wavelet analysis based feature extraction for neural network classification," Proceedings of IEEE International Conference on Neural Networks, Vol.3, pp.1416 - 1421, 1996.

[16] D. Luo, R. Yang and J. Huang,"Detecting double compressed AMR audio using deep learning," Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp.2669 - 2673, 2014.

[17] R. Rui and C. Bao, "A novel supervised learning algorithm for musical instrument classification," Proceedings of the 35th International Conference on Telecommunications and Signal Processing, pp.446 - 449, 2012.

[18] R. Tudoran, A. Costan, R. Wang, L.Bouge, G. Antoniu, "Bridging Data in the Clouds: An Environment-Aware System for Geographically Distributed Data Transfers," Proceedings of 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), pp.92-101, 2014.

[19] K. Zhang and X. Chen ,"Large-Scale Deep Belief Nets With MapReduce,"IEEE JOURNALS and MAGAZINES,Vol.2, pp.395-403, 2014.

[20] X. Yin, C. Yang, W. Pei and H. Hao,"Shallow Classification or Deep Learning: An Experimental Study,"Proceedings of IEEE 22nd International Conference on Pattern Recognition (ICPR), pp.1904-1909, 2014.

[21] B. Li, E. Zhou, B. Huang, J. Duan, Y. Wang, N. Xu, J. Zhang and H. Yang, "Large scale recurrent neural network on GPU," Proceedings of International Joint Conference on Neural Networks (IJCNN), pp. 4062 - 4069, 2014.

[22] Python: https://www.python.org/

[23] Theano: http://deeplearning.net/software/theano/