Context-Dependent Deep Neural Networks for Commercial Mandarin Speech Recognition Applications

Jianwei Niu*†, Lei Xie*, Lei Jia† and Na Hu†
* Shaanxi Provincial Key Laboratory of Speech and Image Information Processing
School of Computer Science, Northwestern Polytechnical University, Xi'an, China
† Baidu Inc., Beijing, China
Emails: {niujianwei, jialei, huna}@baidu.com, lxie@nwpu.edu.cn

***Abstract*—Recently, context-dependent deep neural network hidden Markov models (CD-DNN-HMMs) have been successfully used in some commercial large-vocabulary English speech recognition systems. It has been proved that CD-DNN-HMMs significantly outperform the conventional context-dependent Gaussian mixture model (GMM)-HMMs (CD-GMM-HMMs). In this paper, we report our latest progress on CD-DNN-HMMs for commercial Mandarin speech recognition applications in Baidu. Experiments demonstrate that CD-DNN-HMMs can get relative 26% word error reduction and relative 16% sentence error reduction in Baidu's short message (SMS) voice input and voice search applications, respectively, compared with state-of-the-art CD-GMM-HMMs trained using fMPE. To the best of our knowledge, this is the first time the performances of CD-DNN-HMMs are reported for commercial Mandarin speech recognition applications. We also propose a GPU on-chip speed-up training approach which can achieve a speed-up ratio of nearly two for DNN training.**

I. Introduction

The main-stream of traditional automatic speech recognition (ASR) system typically uses hidden Markov models (HMMs) to model the evolvement of speech units (e.g., phonemes) and uses Gaussian mixture models (GMMs) to represent the relationship between acoustic inputs and speech units. Speech co-articulation is modeled by context-dependent (CD) units, such as triphones. This is the well-known generative CD-GMM-HMM architecture in the literature. Expectation-maximization (EM) algorithm is usually used for HMM training, while further recognition accuracy improvement can be achieved using discriminative training algorithms [1-3] such as MMI, MCE and MPE, etc. About two decades ago, artificial neural networks (ANNs), as a kind of discriminative model, were also investigated in speech recognition [4-6] with some limited success. In a typical ANN approach, instead of using GMMs, ANNs with a single layer of nonlinear hidden units are used to predict HMM states from acoustic observations. However, due to the limitations of the computation power and the learning algorithms, such a single-hidden-layer ANN approach was not sufficiently powerful to seriously challenge GMMs. As a result, the main practical contribution of ANNs was to provide useful features, namely tandem or bottleneck features [7], in which the posterior probability of each phone was estimated using ANN.

With the rapid development of machine learning theory and computer hardware in recent years, it is now capable enough to train a much deeper ANN which contains many layers of non-linear hidden units and a very large output layer. In [8], a new context-dependent deep neural network hidden Markov model (CD-DNN-HMM) was proposed for speech recognition. A significant performance improvement was achieved as compared with traditional CD-GMM-HMM. Unlike previous work of ANN in the ASR area, the posterior probability of context dependent triphone state given acoustic input is estimated directly by a DNN with many layers of hidden units. It has been shown that CD-DNN-HMMs can achieve 33% relative word error reduction over discriminatively-trained CD-GMM-HMMs on the switchboard benchmark task [9]. Moreover, CD-DNN-HMMs have been successfully used in several commercial large-vocabulary English speech recognition applications, such as the Bing mobile voice search application [10][11], Google voice input speech recognition task [12][11] and Youtube speech recognition task [12], etc. Especially, Google voice input recognition task used about 5870 hours of training data and achieved 23% relative word error reduction compared to the best GMM-based system for this task [12][13].

In this paper, we report our latest progress on CD-DNN-HMMs for commercial Mandarin speech recognition applications in Baidu. First, we demonstrate that CD-DNN-HMMs can be effectively used in large-scale Mandarin speech recognition tasks with similar accuracy improvement over CD-GMM-HMMs as in English speech recognition tasks. To our best knowledge, this is the first time the performances of CD-DNN-HMMs are reported for commercial Mandarin speech recognition applications. Second, a new efficient DNN training approach is proposed, in which multiple GPU cards in a single server are utilized in parallel. A speed-up ratio of 1.95 is achieved without recognition accuracy loss.

II. Training CD-DNN-HMMs

A. CD-DNN-HMM

A DNN is actually a conventional multi-layer perceptron (MLP) with more than one layers of hidden units between the input layer and the output layer. In each hidden unit of a

certain layer, a nonlinear activation function is used with input as a linear combination of the outputs of the hidden units in the previous layer. The activation function is typically chosen to be a sigmoid function such as a logistic sigmoid function described as

$$\begin{cases} y_j^l = \frac{1}{1+e^{-x_j^l}} \\ x_j^l = b_j^l + \sum_i y_i^{l-1} w_{ij}^{l-1,l} \end{cases} \quad (1)$$

where x_j^l and y_j^l are the input and output of the j^{th} hidden unit in the l^{th} layer, respectively. x_j^l is also a linear combination of the hidden units in $(l-1)^{th}$ layer, b_j^l is the bias parameters of hidden unit j in the l^{th} layer, and $w_{ij}^{l-1,l}$ is the weight on a connection to hidden unit j in the l^{th} from the hidden unit i in the $(l-1)^{th}$ layer.

Since the output of DNN in CD-DNN-HMM consists of all states, DNN is actually an ANN capable of multi-class classification. Therefore, softmax output activation function with the corresponding multi-class cross-entropy error function is naturally used in which output unit j converts its total input x_j into a class probability p_j by using the softmax nonlinear function as follows:

$$p_j = \frac{e^{x_j}}{\sum_k e^{x_k}} \quad (2)$$

where k is an index over all classes.

Online error back-propagation (BP) algorithm is usually used for ANN training with stochastic gradient ascent:

$$(w_{ij}^{\ell-1,\ell}, b_j^\ell) \leftarrow (w_{ij}^{\ell-1,\ell}, b_j^\ell) + \epsilon \frac{\partial D}{\partial (w_{ij}^{\ell,\ell-1}, b_j^\ell)}, \quad 0 < \ell \le L \quad (3)$$

for an objective function D and a learning rate ϵ. With the ground-truth labels $s(t)$, the objective for minimizing cross-entropy is easily to be seen as maximizing the total log posterior probability over the T training samples $o(t)$, i.e.,

$$D = \sum_{t=1}^{T} \log P_{s|o}(s(t)|o(t)). \quad (4)$$

Therefore, the gradients with respect to $w_{ij}^{\ell-1,\ell}, b_j^\ell$ are given as follows:

$$\frac{\partial D}{\partial w_{ij}^{\ell-1,\ell}} = \sum_t y_i^{\ell-1}(t) \cdot e_j^\ell(t) \quad (5)$$

$$\frac{\partial D}{\partial b_j^\ell} = \sum_t e_j^\ell(t) \quad (6)$$

$$e_j^L(t) = \delta_{s(t),j} - \frac{e^{x_j^L(t)}}{\sum_k e^{x_k^L(t)}} \quad (7)$$

$$e_j^{\ell-1}(t) = y_j^{l-1}(t)\left[1 - y_j^{l-1}(t)\right] \sum_k w_{jk}^{\ell-1,\ell} \cdot e_k^\ell(t) \quad (8)$$

where $\delta(a,b) = \begin{cases} 1 & a = b \\ 0 & \text{otherwise} \end{cases}$.

In order to embed a DNN into the HMM structure, the HMM's state emission likelihoods $p_{\mathbf{o}|\mathbf{s}}(o|s)$ are converted from state posterior probability obtained from DNN as follows:

$$p_{\mathbf{o}|\mathbf{s}}(o|s) = \frac{P_{\mathbf{s}|\mathbf{o}}(s|o)}{P_{\mathbf{s}}(s)} \cdot const(s) \quad (9)$$

where classes s correspond to HMM states, and observation vectors o are acoustic feature vectors. $P_s(s)$ is the prior probability of state s.

TABLE I
DNN TRAINING PARAMETER SETUP

Parameter	Value
Sample size per mini-batch	200 in pretrain 200 in the first epoch in fine-tune 500 in the rest epoches in fine-tune
# of hidden layer	7
# of hidden unit in each layer	2048
Learning rate	decreased from 0.005 to 0.0001
Momentum	0.5

B. Training Procedure of CD-DNN-HMMs

1) Basic training process: In order to determine the state structure of a CD-DNN-HMM, a state-of-the-art CD-GMM-HMM is trained in advance. Each clustered triphone state of the CD-GMM-HMM system is mapped to a unique integer *id* which is used as the label. The ground truth of each acoustic sample is derived with state-level forced alignment on the training set using the CD-GMM-HMM system. Moveover, because error back-propagation optimization procedure easily get trapped in poor local optima for deep networks, we use discriminative pre-training [10] to grow the DNN model layer by layer. After pre-training, the DNN model is fine-tuned using a general error back-propagation procedure described as follows.

- Train a state-of-the-art state tied CD-GMM-HMM system, and map each tied state to a unique integer *id*;
- For each training utterance, do state-level forced alignment and get the ground truth state-level label for each training sample;
- Perform discriminative pre-training until the DNN predefined layer number is arrived [10]:
  1) Initialize a three-layer DNN with an input layer, an output layer and a single hidden layer, and use BP algorithm to train the DNN with two epoch;
  2) Discard the weight which connects the output layer and the top hidden layer and add a new top hidden layer which connects the old hidden layer and the output layer, respectively;
  3) Use BP algorithm to retrain the DNN with one epoch;
  4) Repeat step 2) and step 3) until the predefined layer number is arrived.
- Use normal BP algorithm to retrain (fine-tune) DNN until the maximum number of training epoches is achieved.

2) Parameter setup: In order to make the optimization procedure efficient and effective, several parameters should be set up in advance, as shown in Table I. Please note that the mini-batch size in the fine-tune procedure is different from that of pretrain due to the efficiency issue. Given a good initial model which is trained using a mini-batch size of 200, there is no recognition accuracy degradation using 500 as the mini-batch size in the remaining fine-tune epochs. The momentum in our procedure is set to 0.5, which is different from [11]. We tried several momentum values and no significant accuracy difference is observed. In the BP algorithm, the learning rate adjusting strategy is very important. In our procedure, an exponential decrease is used with learning rate range from 0.005 to 0.0001 during the training process. Specifically, we halved the learning rate when the performance of the DNN model increased slightly on the validation set. In addition, the order of samples is globally randomized in each epoch.

3) 2-GPU Core Parallel BP training: The nature of the online BP algorithm is a sequential algorithm that makes the models be hardly trained in parallel. But within each mini-batch, the forward propagation and backward propagation can be done in parallel. However, the model weight update should be done given total temporal parameters calculated in the forward/backward propagations. As a result, if parallel training is used within each mini-batch, the key point is how to efficiently transfer the temporal parameters calculated in the forward/backward propagations among the different computation nodes. In Baidu, Nvidia K10 [14] is used as the GPU card for DNN training. Within each K10 card, there are two GPU cores and the communication bandwidth between them is beyond 10GB/s. It makes sense to utilize these two GPU cores during the DNN training. The detail training strategy is as follows:

- Initialization: divide the samples in the current mini-batch into two parts, one is forward to GPU core one, another is forward to GPU core two;
- GPU core one/two implements the forward/backward propagations in parallel;
- Once forward/backward propagations are both completed in the two GPU cores, GPU core two transfers the temporal parameters to GPU core one;
- GPU core one updates the model weight and transfers the new model to GPU core two;
- Repeat the above steps until the termination criterion (e.g. no recognition accuracy improvement is observed) is satisfied.

III. Experiments and Analysis

A. Experimental Setup

In this paper, two real-world Mandarin ASR tasks, i.e., SMS voice input and voice search, were used for the performance evaluation of CD-DNN-HMMs. Tabel II shows the data setup for the two tasks. Please note that the testing datasets are collected in real usage scenarios from real mobile users and sampled at 8kHz. The two tasks are quite challenging because the speech data contains real-world variations, e.g., noise, accents, sloppy pronunciation and different audio channels.

TABLE II
Experimental datasets

Data	SMS Voice Input	Voice Search
Training	2600hrs	2100hrs
Testing	5.9hrs (8000 sentences)	1.6hrs (2500 sentences)

TABLE III
The CD-GMM-HMM model setup

Parameter	SMS Voice Input	Voice Search
Training data (hrs)	2600	2100
HMM structure	3 states per HMM for voice HMM 5 states per HMM for silence HMM	
State number	8913	11365
Gaussian number per state	64	64

The speech data was analyzed using a 25ms Hamming window with a 10ms frame rate. Both tasks use 13-dimension PLP features with windowed mean normalization and the first and second order delta features, which forms the 39-dimension feature vectors. We used tonal syllable initial/final triphones as the acoustic modeling units. State-of-the-art CD-GMM-HMM systems were used for comparison, where decision-tree based tied-state triphone GMM-HMMs were trained using MLE/MPE/fMPE criteria for each task. Table III shows the details of the CD-GMM-HMM model setup. For all CD-DNN-HMM experiments, we used 11 continuous frames of PLPs as the input features of the DNN. The recognition experiments were performed using a real-time one-pass decoder in which a 4-gram language model is used with 100K words in the recognition vocabulary. Recognition performance was evaluated using word accuracy for the SMS voice input task and sentence accuracy for the voice search task.

TABLE IV
The recognition accuracy (%) on Baidu's SMS voice input and voice search tasks

Model	SMS (Word Acc.)	Voice Search (Sen. Acc.)
CD-GMM-HMM MLE	82.5	60.4
CD-GMM-HMM MPE	83.2	62.5
CD-GMM-HMM fMPE	84.0	63.7
CD-DNN-HMM	88.2	69.5

B. Experimental Results

The best recognition performances on the testing sets are summarized in Table IV. As demonstrated in Table IV, an absolute 4.2% word accuracy improvement and an absolute 5.8% sentence accuracy improvement were observed over the fMPE baselines for the SMS voice input task and the voice search task, respectively. The relative error rate reduction was 26% (word) and 16% (sentence) for the SMS voice input

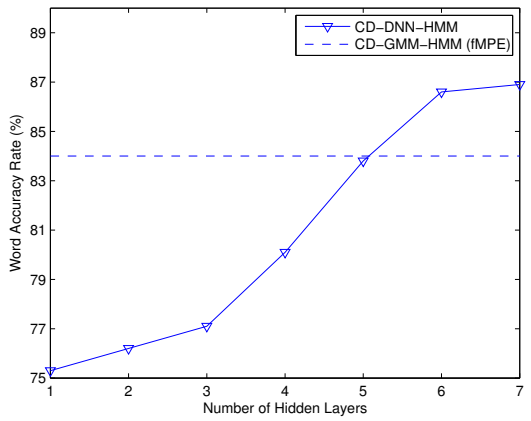

Fig. 1. The relationship between the word accuracy and the number of hidden layers of DNN in the SMS voice input task.

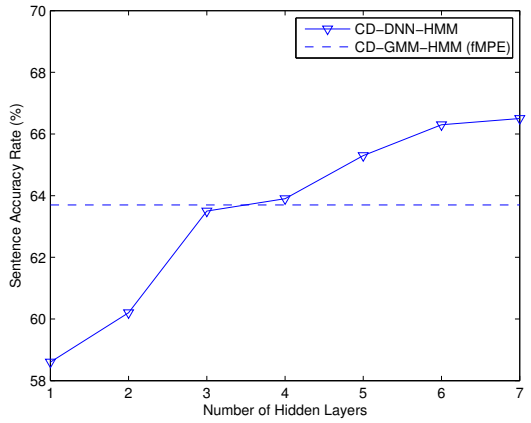

Fig. 2. The relationship between the sentence accuracy and the number of hidden layers of DNN in the voice search task.

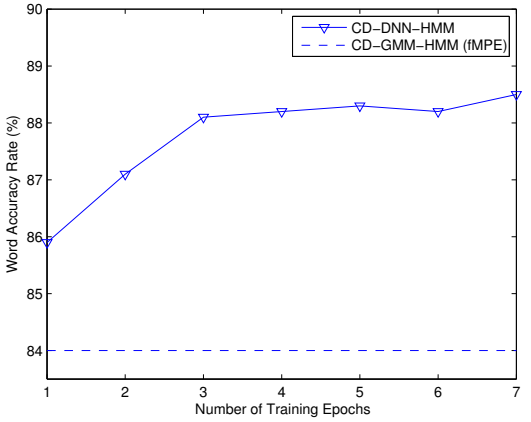

Fig. 3. The relationship between the word accuracy and the training epoch in the fine-tune step of DNN in the SMS voice input task.

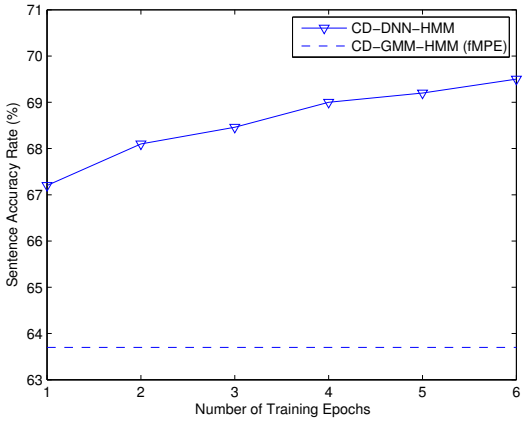

Fig. 4. The relationship between the sentence accuracy and the training epoch in the fine-tune step of DNN in the voice search task.

task and the voice search task, respectively. The results show that the CD-DNN-HMMs are superior to the state-of-the-art CD-GMM-HMMs in real-world Mandarin speech recognition tasks.

C. Effects of Layer Depth

To illustrate the effects of different depths of DNN hidden layers, 7 sets of CD-DNN-HMMs were trained with the hidden layer number ranging from 1 to 7. The relationship between the accuracy and the number of hidden layers for the two tasks are shown in Fig. 1 and Fig. 2, respectively. It can be observed that with the increase of the number of hidden layers, there is a consistent performance improvement accordingly. Moreover, a significant performance improvement comparing with CD-GMM-HMM can be observed when the hidden layer number goes beyond 5 for the SMS voice input task and 3 for the voice search task. Overall, using 7 hidden-layer models provides a relative 13.4% word and a relative 15.4% sentence accuracy improvement over the single hidden-layer system for the two tasks, respectively. Please note that the training epoch in the fine-tune step was fixed to one and the training data set was set to 2600hrs (SMS voice input) and 2100 hrs (voice search).

D. Effects of Training Epoch

In Fig. 3 and Fig. 4, the influence of training epoch on DNN is shown. It is illustrated that the accuracy goes up with the increase of the training epoch. In general, the word accuracy for the SMS voice input task is lifted from 85.9% to 88.5% when the training epoch increases from 1 to 7; the sentence accuracy for the voice search task increases from 67.2% to 69.5% when the training epoch raises from 1 to 6. For the SMS voice input task, a large word accuracy improvement can be obtained with the training epoch increasing from 1 to 3. But with more training epochs, only small improvements are achieved. This is probably mainly due to the ability of the learning rate adjusting strategies. Further studies should be given to this phenomenon. Please note that the depth of the hidden layers was set to 7 and the training data size was set to 2600hrs (SMS voice input) and 2100 hrs (voice search) in the experiments showing the influence of training epoch.

E. Effects of Amount of Training Data

Fig. 5 demonstrates the influence of training data size on DNN for the voice search task. It can be observed that with the increase of the amount of training data, the sentence

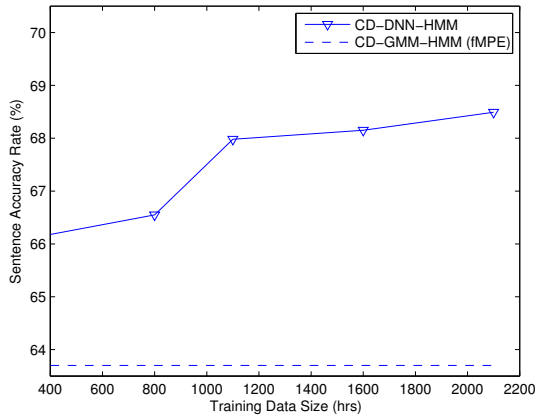

Fig. 5. The relationship between the sentence accuracy and the amount of training data in the voice search task.

TABLE V
THE TRAINING TIME (IN TERMS OF MILLISECOND) PER MINI-BATCH FOR THE 2-GPU PARALLEL TRAINING APPROACH AND THE NORMAL SINGLE GPU TRAINING APPROACH.

Training stage	Samples per mini-batch			
	500		1000	
	2-GPU	1-GPU	2-GPU	1-GPU
Propagation	31.9	70.7	56.7	140.8
Back-propagation	110.2	151.6	161.3	285.3
Total	142.1	222.3	218.0	426.1
Speed-up ratio		1.56		1.95

accuracy can be improved consistently. It is also seen that the accuracy is improved greatly when the amount of training data is increased from 800 hours to 1100 hours. After that, the speed of improvement decreases. The relative accuracy gain is 3.5% when the training data size is increased from 400 hours to 2100 hours. Please note that the depth of the hidden layers was set to 7 and the training epoch was 1 in the fine-tune step in the experiments.

F. Performance of 2-GPU Parallel Training

In Table V, the speedup ability of the proposed 2-GPU parallel training approach is illustrated. We can clearly see the effectiveness of the speed-up strategy. Meanwhile, we observe that the speed-up ratio is highly relevant to the sample size of the mini-batch. Further speed-up occurs with a larger mini-batch size. Specifically, using the 2-GPU parallel training strategy, the speed-up ratio of DNN training is 1.95 when the sample per mini-batch is set to 1000.

IV. CONCLUSIONS

In this paper, we used CD-DNN-HMMs to train acoustic models on very large data sets for two real-world Mandarin speech recognition tasks in Baidu, namely SMS voice input and voice search. Our results illustrate that CD-DNN-HMMs have good capacity to learn from large datasets and can be efficiently applied to different tasks. Experiments demonstrate that CD-DNN-HMMs can get relative 26% word error reduction and relative 16% sentence error reduction in Baidu's SMS voice input and voice search applications, respectively, compared with state-of-the-art CD-GMM-HMMs trained using fMPE. To speed up model training, we proposed a 2-GPU parallel training approach which achieved a speed-up ratio of 1.95 when the mini-bath size was set to 1000. Our future work is to find an efficient training algorithm in order to train DNN models using tens of thousands of hours of data.

V. ACKNOWLEDGEMENTS

This work was supported by the Baidu Inc., the National Natural Science Foundation of China (61175018), the Natural Science Basic Research Plan of Shaanxi Province (2011JM8009) and the Fok Ying Tung Education Foundation (131059).

REFERENCES

[1] S. Kapadia, V. Valtchev, and S. J. Young, "MMI training for continuous phoneme recognition on the TIMIT database," in Proc. ICASSP, vol. 2, 1993, pp. 491-494.
[2] B. H. Juang, W. Chou, and C. H. Lee, "Minimum classification error rate methods for speech recognition," IEEE Transactions on Speech and Audio Processing, vol. 5, no. 3, pp. 257-265, 1997.
[3] D. Povey, "Discriminative training for large vocabulary speech recognition," Ph.D. dissertation, Cambridge University Engineering Department, 2003.
[4] H. Boulard and N. Morgan,"Continuous speech recognition by connectionist statistical methods," IEEE Transactions on Neural Networks, vol. 4, no. 6, pp. 893-909, 1993.
[5] H. Franco, M. Cohen, N. Morgan, D. Rumelhart and V. Abrash,"Context-dependent connectionist probability estimation in a hybrid hidden Markov model-neural net speech recognition system," Computer Speech and Language, vol. 8, pp. 211-222, 1994.
[6] J. Fritsch and M. Finke, "ACID/HNN: Clustering Hierarchies Of Neural Networks For Context-Dependent Connectionist Acoustic Modeling" in Proc. ICASSP, pp. 505–508, May 1998.
[7] H. Hermansky, D. P. W. Ellis, and S. Sharma, "Tandem connectionist feature extraction for conventional HMM systems," in Proc. ICASSP, vol. 3, pp. 1635-1638, 2000.
[8] D. Yu, L. Deng, and G. Dahl,"Roles of Pretraining and Fine-Tuning in Context-Dependent DNN-HMMs for Real-World Speech Recognition," in Proc. NIPS Workshop on Deep Learning and Unsupervised Feature Learning, Dec. 2010.
[9] F. Seide, G. Li, and D. Yu, "Conversational speech transcription using context-dependent deep neural networks," in Proc. Interspeech, 2011, pp. 437-440.
[10] X. Chen, F. Seide, G. Li and D. Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription," in Proc. ASRU, 2011.
[11] G. E. Dahl, D. Yu, L. Deng and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition", IEEE Transactions on Audio, Speech, and Language Processing, vol. 20, no. 1, pp. 30-42, 2012.
[12] N. Jaitly, P. Nguyen, A. Senior, V. Vanhoucke, "Application of pre-trained deep neural networks to large vocabulary conversational speech recognition", Tech. Rep., Department of Computer Science, University of Toronto, 2012.
[13] G. Hinton, L. Deng, D.Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups," IEEE Signal Processing Magazine, pp. 82-97, November 2012.
[14] TESLA K10 GPU Accelerator, http://www.nvidia.com/content/PDF/kepler/Tesla_K10_BD-06280-001_v05.pdf