12-2013

Deep maxout networks for low-resource speech recognition

Yajie Miao
Carnegie Mellon University

Florian Metze
Carnegie Mellon University, fmetze@andrew.cmu.edu

Shourabh Rawat
Carnegie Mellon University

Published In
Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU), 398-403.

DEEP MAXOUT NETWORKS FOR LOW-RESOURCE SPEECH RECOGNITION

Yajie Miao, Florian Metze, and Shourabh Rawat

Language Technologies Institute, School of Computer Science, Carnegie Mellon University
{ymiao,fmetze}@cs.cmu.edu, srawat@andrew.cmu.edu

ABSTRACT

As a feed-forward architecture, the recently proposed maxout networks integrate dropout naturally and show state-of-the-art results on various computer vision datasets. This paper investigates the application of deep maxout networks (DMNs) to large vocabulary continuous speech recognition (LVCSR) tasks. Our focus is on the particular advantage of DMNs under low-resource conditions with limited transcribed speech. We extend DMNs to hybrid and bottleneck feature systems, and explore optimal network structures (number of maxout layers, pooling strategy, etc) for both setups. On the newly released Babel corpus, behaviors of DMNs are extensively studied under different levels of data availability. Experiments show that DMNs improve low-resource speech recognition significantly. Moreover, DMNs introduce sparsity to their hidden activations and thus can act as sparse feature extractors.

Index Terms— Deep maxout networks, speech recognition, low-resource conditions, deep learning

1. INTRODUCTION

Deep neural networks (DNNs) have been applied to automated speech recognition (ASR) and shown superior performance over the traditional GMM-HMM models. Applications of DNNs fall into two categories. In hybrid systems, DNNs are trained to classify context-dependent states and estimate their posterior probabilities [1, 2]. In tandem systems, we use DNNs to generate phone posteriors or bottleneck features (BNF), and build normal GMM-HMM models with the discriminative front-end [3, 4, 5, 6]. These acoustic modeling techniques are distinct from the earlier ANN-HMM systems [7] in the sense that there are more hidden layers in the DNN topology. Therefore, DNN based acoustic models tend to have much more parameters than GMM-HMM. For example, in [8], the hybrid system with a 5-hidden-layer fully-connected DNN has 12 times more parameters than its corresponding GMM-HMM system. When DNNs are fine-tuned on small training sets, this large parameter space can cause overfitting easily and degrade the model robustness on unseen decoding data.

Various methods have been proposed to enhance DNNs under low-resource conditions. A potential solution is to build sparse DNNs [8], either through regularizing hidden-layer parameters or through rounding tiny parameters to zero. Although speeding up model training, sparse DNNs fail to improve recognition performance significantly. Meanwhile, dropout is presented as a useful strategy to prevent overfitting in DNN fine-tuning [9]. Random dropout is observed to perform effectively on phone recognition [9] and LVCSR [10, 11], displaying special benefits when language resources become highly limited. Also, a large amount of work has been dedicated to training DNNs over multiple languages, for both hybrid [11, 12] and tandem systems [3, 5]. Multilingual network training enables cross-language knowledge transfer and can happen either in pre-training [13] or in the fine-tuning stage [11, 12].

This paper investigates the utility of maxout networks [14] in low-resource speech recognition. Maxout networks differ from the standard multi-layer perceptron (MLP) in that hidden units at each layer are divided into non-overlapping groups. Each group generates a single activation via the max pooling operation. Due to reduced hidden activations, maxout networks shrink the size of model parameters and thus are particularly suitable for low-resource conditions. Also, training of maxout networks can optimize the activation function for each unit. Used in conjunction with dropout and convolutional layers, maxout networks set the state of the art on computer vision benchmark datasets [14].

In this study, we make the first attempt to apply maxout networks to LVCSR tasks. We extend the maxout model to the *deep maxout networks* (DMNs) architecture and use it for both hybrid and BNF tandem systems. Although DMNs can be viewed as a special case of DNNs, we distinguish them to be different types in this paper. Pre-training based on stacked denoising autoencoders (SDAs) is performed to initialize DMN parameters and facilitate subsequent fine-tuning. We evaluate the effectiveness of DMNs on the Babel Tagalog dataset [6, 15]. Extensive experiments are conducted to determine appropriate DMN settings such as number of maxout layers and size of unit groups. Under the LimitedLP condition with 10 hours of training data, DMNs outperform the standard DNNs significantly, resulting in consistent word error rate (WER) reduction. In addition, DMNs can naturally enforce sparsity on their high-level hidden activations. The sparse feature representations extracted from DMNs further improve hybrid setups.

2. REVIEW OF DNNS

A DNN is an MLP which consists of many hidden layers before the softmax output layer [11]. On each hidden layer, the DNN computes the activations of conditionally

independent hidden units given the input vector. When using sigmoid activation, the emission of the l-th layer, i.e., the input to the $l+1$-th layer, can be computed as follows:

$$\mathbf{u}_l = \sigma(\mathbf{W}_l \mathbf{u}_{l-1} + \mathbf{b}_l), \ 1 \le l < L \quad (1)$$

where $\mathbf{u}_0 = \mathbf{o}_t$, $\mathbf{W}_l$ is the matrix of connection weights between the l-1-th and l-th layers, $\mathbf{b}_l$ is the bias vector at the l-th layer, $\sigma(x) = (1+\exp(-x))^{-1}$ is the sigmoid function.

Training DNNs directly with error back-propagation (BP) may be problematic in that BP easily gets stuck at poor local optima [11]. A common solution is to initialize DNN parameters using unsupervised pre-training such as restricted Boltzmann machines (RBMs) [16] and stacked denoising autoencoders (SDAs) [17]. A denoising autoencoder (DA) has the same structure as the traditional autoencoder, with the only difference of corrupting the input by adding some form of noise. SDAs can be trained in a greedy layer-wise manner. Training of each DA involves reconstructing the clean input from the corrupted version of it. In our experiments, we observe that SDAs based pre-training performs comparably with RBMs in terms of the recognition of DNN acoustic models. However, training of SDAs is more efficient than training of RBMs. Therefore, we use SDAs as the pre-training method through this paper.

2.1. Hybrid Systems

When building hybrid systems, we train a DNN with a softmax output layer to classify the input acoustic features into classes corresponding to context-dependent tied states. The DNN output is an estimate of the posterior probability $P(s \mid \mathbf{o}_t)$ of each state s given the observation $\mathbf{o}_t$:

$$P(s \mid \mathbf{o}_t) = \frac{\exp(\mathbf{W}_L \mathbf{u}_{L-1} + \mathbf{b}_L)}{\sum_{\bar{s}} \exp(\mathbf{W}_L \mathbf{u}_{L-1} + \mathbf{b}_L)} \quad (2)$$

Hybrid systems share the model structure (phone set, HMM topology, tying of context-dependent states) coming from an initial GMM-HMM model that has been maximum likelihood (ML) trained on the same data. That model is also used to generate the true class label of each frame through forced-alignment. During recognition, the emission probability of the HMM state s can be computed by converting state posteriors in Eq. (2) as follows:

$$P(\mathbf{o}_t \mid s) = \frac{P(s \mid \mathbf{o}_t) P(\mathbf{o}_t)}{P(s)} \quad (3)$$

where $P(s)$ is the state prior probability which can be approximately estimated from the training data by simple counting, the observation probability $P(\mathbf{o}_t)$ is independent of the word sequence and can be ignored.

2.2. BNF Systems

The BNF front-end can be extracted from a narrow bottleneck hidden layer in DNNs and used to construct GMM-HMM tandem systems. In this paper, we turn to the previously established deep BNF (DBNF) framework [6] for bottleneck feature generation. The DNN exploited by DBNF inserts multiple hidden layers between the input data and the bottleneck layer, and pre-trains these prior-to-bottleneck layers using SDAs. A hidden layer and the final softmax layer are added on top of the bottleneck layer. DBNF differs from other BNF approaches [4, 5] in that its hidden layers are arranged asymmetrically around the bottleneck layer.

The whole DBNF network is then fine-tuned on the available training data. BNF training has adopted phones or context-independent states as frame-level super-vision. However, we observe that context-dependent states give BNF systems better recognition results. Thus, we use the same frame labels as in hybrid systems during DBNF training. We refer interested readers to [6] for more details.

3. DEEP MAXOUT NETWORKS FOR LVCSR

Deep maxout networks (DMNs) consist of multiple layers which generate hidden activations via the maxout function. Figure 1(a) illustrates the l-th layer in a maxout network where the hidden units are divided into disjunct groups. We denote the number of unit groups as I and the group size (how many units each group contains) as g. The maxout function is imposed on each unit group to generate this layer's activations $\mathbf{u}^{(l)} = [\mathrm{u}_1^{(l)}, \mathrm{u}_2^{(l)}, \cdots, \mathrm{u}_I^{(l)}]$. Each element is computed as

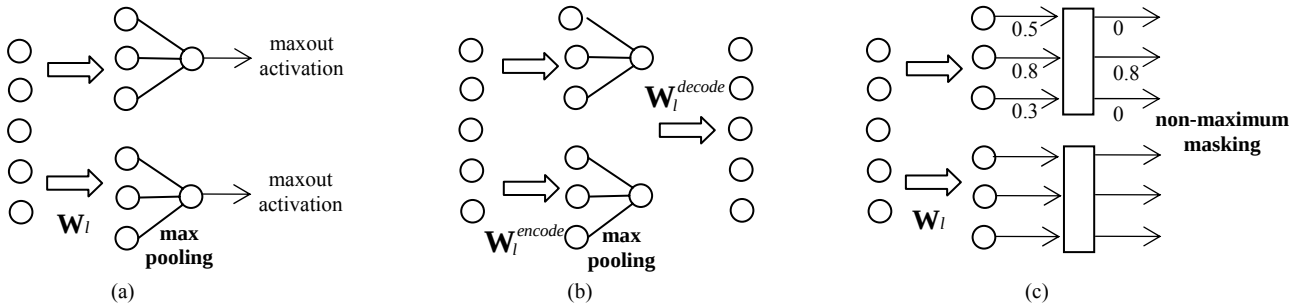
$$\mathrm{u}_i^{(l)} = \max_j (\mathrm{h}_j^{(l)}), \quad (i-1) \times g + 1 \le j \le i \times g \quad (4)$$

where $\mathbf{h}^{(l)} = \mathbf{W}_l \mathbf{u}_{l-1} + \mathbf{b}_l$ represents the linear pre-activation values. We can see that the maxout function applies a max pooling operation on $\mathbf{h}^{(l)}$. The maximal value within each group is taken as the output from the l-th layer. A DMN can be constructed by connecting multiple maxout layers consecutively and finally adding the softmax layer.

No pre-training is carried out for maxout networks in [14]. When applying DMNs to LVCSR, the networks may become really deep to fully capture human speech variability. In this case, pre-training becomes necessary for initializing network parameters properly. We can pre-train a single maxout layer with the autoencoder depicted in Figure 1(b). This structure behaves similarly with the normal autoencoder, except for the maxout activations as the hidden output. Due to the mismatch of their dimensions, the encoding and decoding parameter matrices are not tied. Artificial corruptions can also be added to the input. Training of this autoencoder tries to minimize the difference between the reconstruction output and the clean input. The whole DMN can be pre-trained by stacking autoencoders corresponding to the maxout layers in a layer-wise manner.

3.1. Application to LVCSR

The application of DMNs to speech recognition is easy to accomplish. We replace the DNN modules used in LVCSR systems (see Section 2) with DMNs. In this study we aim at



(a) (b) (c)

Fig. 1. Maxout architectures in this paper: (a) maxout layer with the group size of 3; (b) maxout autoencoder; (c) sparse feature extractor.

improving LVCSR when only limited training data is available. We argue that DMNs are particularly suitable for low-resource tasks because of the following two reasons.

First, compared with standard DNNs, DMNs can reduce the size of network parameters significantly. Suppose that a DNN and DMM contain the same number of units at each hidden layer. Then the size of each connection matrix in the DMN is $1/g$ of the size of the connection matrix in the DNN. This parameter reduction enhances model robustness under limited data. Second, unlike DNNs, DMNs do not fix the shape of the activation function for hidden outputs. By tuning weight vectors of the subsumed hidden units, each maxout activation is capable of approximating any convex functions [14] and thus can be optimized towards specific datasets in hand. This property enables DMNs to capture speech variability from limited data more effectively.

An important part of DMN acoustic modeling is the integration of dropout, a technique performing particularly well for low-resource speech recognition. Maxout networks are found to maximize the model averaging effects caused by dropout. Therefore, we impose dropout on each DMN hidden layer by following the implementation described in [11]. On each presentation of a training example, maxout activations from each hidden layer are randomly omitted via a binomial distribution. This distribution is governed by a pre-specified probability referred to as *drop factor* in [11]. Dropout is applied only during training (fine-tuning). For testing (recognition), network parameters need to be scaled properly according to the value of the drop factor [11].

3.2. DMNs as Sparse Feature Extractors

In addition to acting as acoustic models directly, a trained DMN can also be used as a sparse feature extractor. Sparse outputs are generated from an arbitrary layer by applying a *non-maximum masking* operation, rather than max pooling. Specifically, given each input frame, all the units within each group have their individual outputs, instead of being pooled together into one output. However, only the maximal value in this group is retained, while the other outputs are rounded to 0. An example is presented in Figure 1(c), where the group size is 3 and 2/3 of maxout activations are set to 0. In Section 4.5, we experimentally show that these sparse outputs pose a useful representation for the raw acoustic features and can improve the performance of hybrid systems.

4. EXPERIMENTS

4.1. Experimental Setup

We use the Babel corpus that has been collected and released under the IARPA Babel research program. The goal of the Babel program is to realize rapid deployment of speech recognition and spoken term detection systems for low-resource languages. Up to now, the corpus has covered Cantonese, Tagalog, Turkish, Pashto and Vietnamese. The full language pack (FullLP) of each language consists of around 80 hours of telephony speech for training and 20 hours for system development. Each audio file records spontaneous conversations lasting approximately for 10 minutes. The data collection attempts to cover a variety of acoustic conditions (e.g., street, office, inside vehicles), speaking styles, and various dialects. Also, a notable portion of the audio data are either non-speech events (e.g., breath, laugh, cough, lip smack, ring) or non-lexical speech (e.g., hesitations, fragments and foreign words). Due to all these factors, speech recognition on the Babel corpus is a very difficult task, ending up with much higher WERs than on other benchmark datasets such as Switchboard [6, 15].

In this paper, we conduct our experiments on Tagalog and focus on the limited language pack (LimitedLP, version babel106b-v0.2g-sub-train). This condition only has 10 hours of speech data and the corresponding resources (dictionary, language model) for system building. As a comparison with LimitedLP, 40 hours of training data are selected from Tagalog FullLP (version babel106-v0.2f) to simulate a 40HrLP rich-resource condition. During decoding, we select 2 hours of speech from the entire 20-hour development data as the dev set, and another 2.5 hours as the eval set. The training, dev and eval sets have no overlapping speakers. All decoding runs use a trigram language model built solely from training transcriptions. The Tagalog LimitedLP and 40HrLP datasets have the statistics summarized in Table 1.

4.2. GMM and SGMM Systems

On both LimitedLP and 40HrLP, GMM-HMM systems are built with the same recipe. We first train the initial ML model based on 39-dimensional PLP+delta+acceleration features with per-speaker cepstral mean normalization. Then

Table 1. Statistics of the datasets used in the experiments. The OOV rate is measured on transcriptions of the 2-hour dev set.

Statistics	Conditions	
	LimitedLP	**40HrLP**
# speakers	132	482
training (hours)	10.7	40.3
dictionary size	8k	35k
OOV rate	9.1%	1.8%

9 frames of PLPs are spliced together and projected down to 40 dimensions with linear discriminant analysis (LDA). A maximum likelihood linear transform (MLLT) is applied on the LDA features and generates the LDA+MLLT model. Finally, to deal with speaker variability, speaker adaptive training (SAT) is performed using feature-space maximum likelihood linear regression (fMLLR). On the two datasets, the numbers of context-dependent triphone states are 1920 and 3066 respectively, with an average of 10 and 16 Gaussian components per state.

On top of the SAT systems, we train subspace Gaussian mixture models (SGMM) [18] for better recognition outcomes. Learning of universal background model (UBM) and SGMM parameters is carried out in the fMLLR feature space. We adopt the SGMM configurations (e.g., number of shared Gaussians, subspace dimensions) in [19]. Because of shared subspace parameters, SGMM can model more tied states than GMM. On LimitedLP and 40HrLP, the numbers of tied states are increased to 2851 and 4542, and each state on average has 3 and 5 substates respectively. Discriminative training is further performed based on the maximum mutual information (MMI) criterion. Due to space limit, we don't elaborate on our *MMI-SGMM* setup. More details can be found in [18, 19] and the Kaldi toolkit [20]. Figure 2 shows the WERs of the resulting MMI-SGMM models.

4.3. Effectiveness of DMNs for Hybrid Systems

Hybrid systems inherit the model structure (phone set, HMM topology, tying of context-dependent states) from the SAT models built in the previous section. The class labels for speech frames are generated by SAT GMM-HMM through forced alignment. DNN inputs include 9 fMLLR frames (4 on each side of the current frame) which are further reduced to 250 dimensions by LDA. These speaker adaptive features in our experiments perform better than the uncorrelated PLPs and correlated log filter bank coefficients.

DNN parameters are initialized with SDAs based pre-training. We follow [6] for SDAs learning with masking noise and the denoising factor of 0.2. Pre-training of each layer has the learning rate of 0.01 and runs for 10 epochs. During fine-tuning, an exponentially decaying learning rate schedule is used for gradient descent. Specifically, the learning rate starts from 0.08 and remains unchanged for 15 epochs. Then the learning rate is halved at each epoch until the cross-validation error on a held-out set stops to drop. A momentum of 0.5 is used in both pre-training and fine-tuning for fast converging. The batch size is 128 for pre-training and 256 for fine-tuning. Each DNN hidden layer consists of 1024 units, which is observed to perform better than 512 units and similarly with 2048 units.

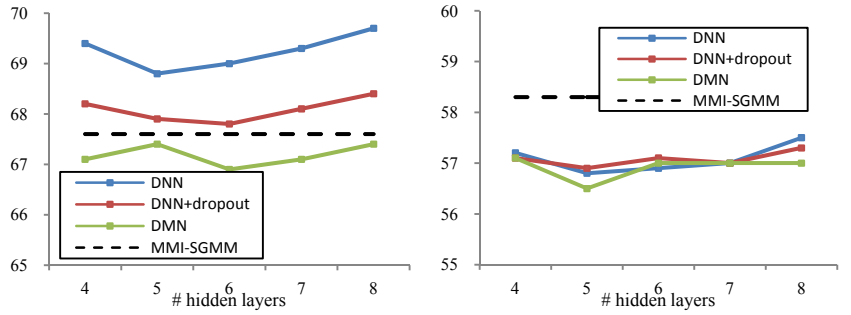
When applying DMNs to hybrid systems, we start with 400 unit groups at each hidden layer and with the group size of 3. This setting gives DMNs approximately the same number (1200 vs. 1024) of hidden units as DNNs. Fine-tuning of DMNs has the identical configuration as that of DNNs. However, with the introduction of dropout, fine-tuning for DMNs must start from a larger learning rate 0.1 [14]. Figure 2 makes a comparison between the DNN, DNN with dropout and DMN models on the 2-hour dev set. Similarly with [11], dropout applied in DNNs and DMNs has the drop factor of 0.2 on each hidden layer. We can see that under LimitedLP, DNN+dropout performs better than DNN with the same pre-training. This confirms the effectiveness of dropout in improving DNNs with limited training data. The DMN model outperforms both DNN and DNN+dropout consistently, resulting in better performance than MMI-SGMM. In contrast, under 40HrLP, the three methods perform comparably as shown in Figure 2(b). These results demonstrate the advantage of DMNs when applied to low-resource LVCSR. Under the LimitedLP condition, Table 2 lists WERs corresponding to the best settings discovered in Figure 2(a). Compared with the DNN baseline, the DMN achieves 1.8% absolute improvement on the dev set and 2.0% on the eval set. In the following experiments, we only work on the LimitedLP condition.

Note that these gains are obtained when the parameters of DMNs are randomly initialized. Now we perform SDAs based pre-training as discussed in Section 3. In this case, the network initial values come closer to the optimum. Thus, we use smaller learning rates for DMN fine-tuning, reducing the starting value from 0.1 to 0.06. The last row of Table 2 shows that SDAs pre-training brings additional improvement to the DMN model, i.e., 0.4% absolute on the dev set and 0.2% on the eval set.

With 400 unit groups and the group size of 3, the DMN has around half as many parameters as its DNN counterpart which has 6 hidden layers and 1024 units at each layer. We also examine how DMNs behave when their topology changes. We fix the number of maxout layers to 6 and the number of hidden units at each layer to 1200. LimitedLP hybrid systems are constructed with different combinations of group number and group size. Their results on the dev set are shown in Table 3. For a fair comparison, no pre-training is performed for the various DMNs architectures. We can see that continuing to decrease the number of unit groups

Table 2. WERs (%) of LimitedLP hybrid systems on the dev and eval sets. The last row shows the DMN with SDAs pre-training.

Model	**Pre-training**	**Dev WER%**	**Eval WER%**
DNN	SDAs	68.8	72.0
DNN+dropout	SDAs	67.8	70.9
DMN	Random	67.0	70.1
DMN	SDAs	**66.6**	**69.9**



(a) The LimitedLP condition (b) The 40HrLP condition

Fig. 2. WERs(%) for MMI-SGMM and hybrid systems on the dev set. DNN and DNN+dropout are pre-trained with SDAs while DMN is randomly initialized.

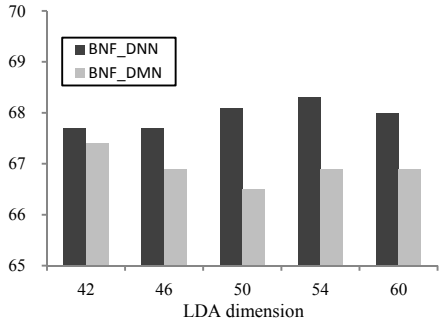


Fig. 3. WERs(%) for LimitedLP BNF systems on the dev set. Comparison is made between DNNs and DMNs.

causes degradation on the WER. This is partly because the aggressive reduction of model parameters hurts DMN's modeling capacity. Further, we verify whether similar gains can be achieved simply by shrinking the size of DNNs. The parameters of the 6-hidden-layer DNN are reduced by half, using 512 units at each hidden layer. On the dev set, this smaller DNN gives the WER of 70.0% with dropout, performing worse than both the original DNN and the DMN.

4.4. Effectiveness of DMNs for BNF Extraction

Now we investigate the performance of DMNs in extracting BNF features. In the DBNF architecture, totally 4 hidden layers are inserted prior to the bottleneck layer. The softmax output layer classifies context-dependent tied states. When using a DNN as the building block of DBNF, the bottleneck layer has 40 hidden units while each of the other hidden layers has 1024 units. When a DMN is used, the group size is set to 3 for every hidden layer and each non-bottleneck layer has 400 unit groups as in hybrid systems. The bottleneck layer consists of 40 unit groups to ensure the same BNF dimensionality as the DNN. For both types of networks, the 4 prior-to-bottleneck hidden layers are pre-trained with SDAs.

When the DBNF network, either a DNN or DMN, has been trained, we build an LDA+MLLT tandem system using the BNF front-end. We observe that a critical variable for BNF system building is the size of the resulting LDA features. We compare BNF systems with different LDA feature dimensions in Figure 3 and show the results corresponding to the best configurations in Table 4. In generally, BNF systems get notable gains over the SAT model built in Section 4.2. The BNF system based on DMN achieves better WERs than the system based on DNN, resulting in 1.2% and 0.9% absolute improvement on the dev and eval sets respectively.

Table 3. DMNs on the dev set with various settings. Model size is measured by the ratio of parameter size between DMN and DNN.

# unit groups	group size	Dev WER%	model size
400	3	**67.0**	0.46
300	4	67.6	0.36
240	5	68.4	0.30

4.5. DMNs as Sparse Feature Extractors

Both DNNs and DMNs can be used to extract high-level representations from the raw acoustic features. One advantage of DMNs is to naturally introduce sparsity in the learned representations. Since our focus is on low-resource tasks, we study feature extraction in the context of cross-lingual speech recognition [12]. Our goal is to improve speech recognition on LimitedLP Tagalog, with the presence of auxiliary languages including LimitedLP Cantonese, Turkish and Pashto also from the Babel corpus. To achieve this, we firstly follow the recipe in [11, 12] to learn a multilingual DNN or DMN. The hidden layers are shared and collaboratively trained on all the auxiliary speech data, while the softmax output layers are specific to individual auxiliary languages. Training data for multilingual networks should have minimum mismatch across languages. As a result, no language-specific transformations such as LDA and fMLLR can be applied to the raw features [12]. In this subsection, our experiments take the 30-dimensional log filter banks generated on each frame as DNN and DMN inputs. Both types of networks in this multilingual setting have 6 hidden layers. Each DNN hidden layer contains 1024 units while each DMN layer has 400 unit groups with the group size of 3. Multilingual fine-tuning of the DNN and DMN is performed in the same manner as in the monolingual scenario. Each epoch needs to traverse data from multiple languages instead of one single language.

The shared layers are then applied to LimitedLP Tagalog as a language-universal feature extractor. With the multilingual DNN, 1024-dimensional features can be generated from *the last hidden layer*. When using the multilingual DMN, we can extract the maxout activations from *the last maxout layer* into 400-dimensional features. Alternatively, sparse features with 1200 dimensions are generated in the way described in Section 3.2.

Table 4. Comparison between DNNs and DMNs for BNF extraction. WERs (%) are reported on the dev and eval sets.

System	Dev WER%	Eval WER%
SAT GMM-HMM	71.1	73.8
DNN BNF tandem	67.7	70.8
DMN BNF tandem	**66.5**	**69.9**

Table 5. Comparison of hybrid systems built on various feature types. WERs (%) are reported on the 2-hour dev set.

Feature type (dimension)	Dev WER%
log filter banks (330, ± 5 frame context)	71.3
Multilingual DNN (1024)	70.2
Multilingual DMN (400)	69.2
Sparse Multilingual DMN (1200)	**67.5**

On LimitedLP Tagalog, we adopt the identical DNN topology to build hybrid systems over various feature types. This DNN has 4 hidden layers each of which has 1024 units and is randomly initialized. Table 5 evaluates these feature types by comparing the WERs of their hybrid systems. Note that the numbers here are not in the same range as the ones in Table 2 simply because we are switching to a different base front-end (fMLLR vs. log filter banks). We can see that deep features, either from the multilingual DNN or DMN, outperform the original log filter banks. Among all the feature types, sparse representations extracted from the multilingual DMN achieve the best WER. This confirms the effectiveness of DMNs acting as sparse feature extractors.

5. CONCLUSIONS AND FUTURE WORK

This paper studied deep maxout networks (DMNs) for low-resource speech recognition. Following experiments on the challenging Babel corpus, we are able to draw the following principal conclusions: 1) Compared with DNNs, DMNs can improve the performance of both hybrid and BNF systems under the LimitedLP condition; 2) SDAs based pre-training performs effectively for DMNs initialization and brings gains when DMNs become really deep; 3) DMNs can be used as sparse feature extractors to generate hierarchical high-level representations. For our future work, we are interested to study RBMs for DMNs initialization in which probabilistic pooling strategies are required to realize a fully generative model. Also, we would like to extend the sparse feature extraction idea to BNF and generate sparse bottleneck features for tandem systems.

6. ACKNOWLEDGMENTS

This work was supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense U.S. Army Research Laboratory (DoD / ARL) contract number W911NF-12-C-0015. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoD/ARL, or the U.S. Government. This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number OCI-1053575.

7. REFERENCES

[1] G. 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(1), pp. 30-42, 2012.

[2] F. Seide, G. Li, X. Chen, and D. Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription," in *Proc. ASRU*, pp. 24–29, 2011.

[3] F. Grezl, M. Karafiat, and M. Janda, "Study of probabilistic and bottle-neck features in multilingual environment," in *Proc. ASRU*, pp. 359-364, 2011.

[4] J. Gehring, W. Lee, K. Kilgour, I. Lane, Y. Miao, and A. Waibel, "Modular Combination of Deep Neural Networks for Acoustic Modeling," in *Proc. Interspeech*, pp. 94-98, 2013.

[5] K. Vesely, M. Karafiat, F. Grezl, M. Janda, and E. Egorova, "The language-independent bottleneck features," in *Proc. SLT*, pp. 336-341, 2012.

[6] J. Gehring, Y. Miao, F. Metze, and A. Waibel, "Extracting deep bottleneck features using stacked auto-encoders," in *Proc. ICASSP*, pp. 3377-3381, 2013.

[7] H. Franco, M. Cohen, N. Morgan, et al., "Context-dependent connectionist probability estimation in a hybrid hidden Markov model-neural net speech recognition system," *Computer Speech and Language*, vol. 8(3), pp. 211-222, 1994.

[8] D. Yu, F. Seide, G. Li, and L. Deng, "Exploiting sparseness in deep neural networks for large vocabulary speech recognition," in *Proc. ICASSP*, pp. 4409-4412, 2012.

[9] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv:1207.0580*, 2012.

[10] G. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for LVCSR using rectified linear units and dropout," in *Proc. ICASSP*, pp. 8609-8613, 2013.

[11] Y. Miao, and F. Metze, "Improving low-resource CD-DNN-HMM using dropout and multilingual DNN training," in *Proc. Interspeech*, pp. 2237-2241, 2013.

[12] J. Huang, J. Li, D. Yu, L. Deng, and Y. Gong, "Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers," in *Proc. ICASSP*, 2013.

[13] P. Swietojanski, A. Ghoshal, and S. Renals, "Unsupervised cross-lingual knowledge transfer in DNN-based LVCSR," in *Proc. SLT*, pp. 246-251, 2012.

[14] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio, "Maxout networks," *arXiv:1302.4389*, 2013.

[15] J. Cui, X. Cui, B. Ramabhadran, et al., "Developing speech recognition systems for corpus indexing under the IARPA Babel program, " in *Proc. ICASSP*, pp. 6753-6757, 2013.

[16] G. E. Hinton, "A practical guide to training restricted Boltzmann machines," UTML TR., 2010.

[17] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P. Manzagol, "Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion," *Journal of Machine Learning Research*, vol. 11, 2010.

[18] D. Povey, L. Burget, M. Agarwal, et al., "The subspace Gaussian mixture model-a structured model for speech recognition," *Computer Speech and Language*, vol. 25(2), pp. 404-439, 2011.

[19] Y. Miao, F. Metze, and A. Waibel, "Subspace mixture model for low-resource speech recognition in cross-lingual settings," in *Proc. ICASSP*, pp. 7339-7342, 2013.

[20] D. Povey, A. Ghoshal, G. Boulianne, et al., "The Kaldi speech recognition toolkit," in *Proc. ASRU*, 2011.