MULTILINGUAL DEEP BOTTLE NECK FEATURES A STUDY ON LANGUAGE SELECTION AND TRAINING TECHNIQUES

Markus Müller[⋆]*, Sebastian Stüker*[⋆]*, Zaid Sheikh*[†]*, Florian Metze*[†] *and Alex Waibel*[⋆†]

International Center for Advanced Communication Technologies
[⋆]Institute for Anthropomatics and Robotics, Karlsruhe Institute of Technology, Germany
[†]Language Technologies Institute, Carnegie Mellon University, Pittsburgh PA, U.S.A.

ABSTRACT

Previous work has shown that training the neural networks for bottle neck feature extraction in a multilingual way can lead to improvements in word error rate and average term weighted value in a telephone key word search task. In this work we conduct a systematic study on a) which multilingual training strategy to employ, b) the effect of language selection and amount of multilingual training data used and c) how to find a suitable combination for languages. We conducted our experiment on the key word search task and the languages of the IARPA BABEL program. In a first step, we assessed the performance of a single language out of all available languages in combination with the target language. Based on these results, we then combined a multitude of languages. We also examined the influence of the amount of training data per language, as well as different techniques for combining the languages during network training. Our experiments show that data from arbitrary additional languages does not necessarily increase the performance of a system. But when combining a suitable set of languages, a significant gain in performance can be achieved.

Index Terms— bottle neck features, multilingual acoustic modeling, low-resource ASR, time-delay neural networks, data selection

1. INTRODUCTION

The goal of IARPA's program *BABEL*[1] is to build systems for keyword search (KWS) in telephone speech in a rapid manner and on limited amounts of data. Within the program progress is measured through annual evaluations. For the primary condition of the evaluation at the end of second year performers were only allowed to use 10h of transcribed data in the target language.

Since state-of-the-art key word search systems make use of *Large Vocabulary Continuous Speech Recognition* (LVCSR) systems, the task of rapidly building KWS systems includes the task of rapidly building LVCSR systems.

Building LVCSR systems for a new language requires large amounts of data in the target language in order to estimate the system's model parameters in a robust way. While the Babel evaluation's primary condition only allows for using data from the target language, another condition exists in which participants are allowed to use any data available within the BABEL program from any language in addition to the limited data of the target language.

In previous work we have shown for the Babel task that using multilingual data for training the neural network of the bottle neck feature (BNF) component of the pre-processing of the LVCSR system can either reduce training time [1] or the system's word error rate (WER) [2]. To the best of our knowledge, there has not been a concise analysis about which languages and data to choose. We therefore conducted a detailed study of how to combine the different languages within the BABEL program to improve a system given a specific target language.

In this paper we conduct a systematic series of experiments on training multilingual BNFs for the Babel task studying three aspects: a) which is the best technique for training the multilingual BNFs, b) is it more important to increase the total amount of training data or to vary the number of languages in BNF training, c) which is the best selection of languages for multilingual training.

The rest of the paper is structured as follows. In Section 2 we review related work. Then in Section 3 we describe how we trained our DBNFs. Section 4 describes our experimental set-up while Section 5 presents our experimental results.

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 effort uses the IARPA Babel Program language collection releases IARPA-babel{102b-v0.4,103b-v0.3,101b-v0.4c,201b-v0.2b,203b-v3.1a,104b-v0.4bY,106-v0.2f,204b-v1.1b,105b-v0.4,107b-v0.7,206b-v0.1d}.

[1]http://www.iarpa.gov/index.php/research-programs/babel

2. RELATED WORK

2.1. Bottle Neck Features Extracted via Deep Belief Neural Networks

State-of-the-art LVCSR systems often use *deep belief neural networks* (DNNs) [3] for extracting features with the help of *deep belief bottle neck features* (DBNFs) [4, 5, 6]. For DBNFs a deep-belief network with several hidden layers and one bottleneck layers is trained, that classifies extracted feature vectors as, e.g., phonemes, context-dependent phonemes, or even model states. The layers of the DBNF are usually pre-trained, either by using *Restricted Boltzmann Machines* (RBMs) [4] or denoising auto-encoders [7]. After that, back-propagation training, in one of several possible variants, e.g., stochastic gradient descent combined with mini-batch training, is applied which we will call *fine-tuning* in this paper.

Past research has also shown that the use of *Time Delay Neural Networks* (TDNNs) for DBNF front-ends, which we sometimes also call shifting DBNFs, leads to performance improvements over the standard feed-forward networks used for DNNs [2].

2.2. Multilingual DBNFs

Recently the concept of multilingual acoustic modeling has also been extended to feature extraction via DBNFs. This is motivated by the fact that neural networks have been shown to be good at learning shared hidden representations across different tasks. With respect to multilingual modeling for DBNFs, the different languages that might partly overlap and partly differ in their phoneme inventory correspond to the different tasks, while the aspects common to the sounds across languages are the hidden aspects learned by the network. E.g., [8] has shown that the pre-training stage of the training of DBNFs is language independent. Training multilingual DBNFs can either be done by using one shared phoneme set [9], as it is done for ML-Mix [10], or by using different language dependent output layers, one for every language [11, 12, 13, 14]. The latter is possible, as the output layer is later discarded anyway and only the bottle neck layer of the network is retained for constructing the final feature vector. And just like the regular DBNFs, DBNFs using TDNNs can also be trained in multilingual fashion and lead to performance improvements [2].

Several strategies combining data from different languages have been explored. Thomas et al. analyzed the influence of varying the amount of data used from the target language in [15]. They built a multilingual system using a fixed amount of data from two training languages and studied the influnce of adding different amounts of data from the target language. Knill et al. merged the data from different languages during training thus creating a common phoneme set in [16]. They used a fixed set of multilingual data for training the acoustic model as well as the neural network and obtained an increase in performance. Grezl et al. used a similar approach in [17]. They trained a neural network using a fixed set of multilingual training data in combination with a limited set of data from the target language for adaptation.

While the work cited here has shown that multilingual training of DBNFs can lead to performance improvements, to the best of our knowledge no systematic study has been conducted that answers the three questions we aim to answer in this paper: a) is it more important to have more training data or to vary the number of languages in training, b) what is the best combination of multilingual training and the two stages of pre-training and fine tuning in DBNF training, c) and how important is the selection of languages when performing the multilingual training.

3. DEEP BOTTLE NECK FEATURES

3.1. Input Features for the DBNF Neural Network

There are several approaches towards the preprocessing the audio data before feeding it into the DBNF network. Among them are features such as mel-scaled cepstral coefficients (MFCC) and logarithmic mel-scaled spectral coefficients (lMel). Preliminary experiments have shown, that the use of MFCCs and lMels lead to similar results. We thus decided to use only lMels for our experiments. In addition to lMel features, we also include features derived from the fundamental frequency variation [18] and a pitch tracker[19]. These features are then combined and used as input to the DBNF neural network.

3.2. Deep Bottle Neck Features

The use of Deep Bottle Neck features as part of a speech recognition system has first been described by Grézl at al. in [20]. The common approach is to use a feed-forward neural network which is trained as a discriminative feature extractor. Our network contains a narrow, so called *bottle neck*, hidden layer. That layer consists of only a fraction of neurons in comparison to the other layers.

This network performs a non-linear discriminative dimensionality reduction. It has been shown that the activation of the bottle neck units are well suited as input features for HMM/GMM systems leading to improved recognition accuracy. In our set-up, we pre-train the network in an unsupervised fashion using denoising auto-encoders, and fine-tune the network via mini-batch training using stochastic gradient descent, adapting the learning rate via the new-bob algorithm [7].

3.3. Time-delay Neural Network Features

There are various methods and training strategies for neural networks. In a study conducted earlier[2] it as been shown that the use of TDNNs, which we sometimes call Shifting

Deep Bottle Neck features (SDBNFs) leads to improvements in performance. SDBNFs are based on the idea of a time delay neural network[21]. The concept of this approach is to bring the stacking of input features to the neural network level. Here, in the forward pass of the fine tuning step, the gradients of adjacent frames are being averaged over a window of several frames, thereby capturing the information over a longer period in time than just a single frame [2].

3.4. Multilingual Deep Bottle Neck Features

Data from multiple languages can be included at various stages of the neural network training. The first step in which data can be added is the pre-training. [8] showed that the use of multilingual data in an unsupervised way can be beneficial. In our experiment, the data we use from different languages has similar properties. It was recorded under the same recordings conditions. Therefore, the network can learn to extract features from human speech recorded under similar conditions in a language independent way. The role of the pre-training is to initially guide the network parameters into the right direction prior to the fine-tuning. By using more data, the network has the ability to generalize more due to the fact that the network parameters can be estimated in a more robust way.

The fine-tuning takes place as a second step. It offers the possibility to add data from multiple languages as well. Like when pre-training a network, here we also have the opportunity to include data from other languages as well. The same holds true when applying the shifting step. But since these steps fine-tune the parameters of the network, they are somewhat more language-dependent as they need to extract features resembling the individual sounds of a language. Thus, the question remains at which stages to work with multilingual data and at which only with the target data.

4. EXPERIMENTAL SETUP

We conducted our experiments with the help of the Janus Recognition Toolkit (JRTk) [22] which features the IBIS decoder [23]. As target language in our experiments we used Tamil, for which we trained speech recognition systems using different kinds of multilingual DBFNs. In our experiments, the DBNFs are the only part trained multilingually. The HMM/GMM system itself is only trained on the LLP dataset (10h) from Tamil. As we wanted to focus our study to the DBNF component of the system, we kept everything else fixed.

We assessed the performance of the systems on the development data set provided for Tamil. It consists of 10 hours of audio data. The systems were evaluated using two different metrics: Word error rate (WER) and average term weighted value[24] (ATWV). The latter requires a set of keywords; for this we used the given development keyword list. ATWV gives scores in the range between 0 and 1. For better readability, we multiplied the ATWV score by 100. We used a class based language-model and an automatic segmentation. Throughout our experiments, we keep the decoding parameters identical.

4.1. Corpora

We used data from the IARPA BABEL project. The IARPA provided data for several languages. These are: Assamese, Bengali, Cantonese, Haitian Creole, Lao, Pashto, Tagalog, Tamil, Turkish, Vietnamese and Zulu. Table 1 provides an overview of all the languages used, including details about the language family and the phoneme inventory. The languages selected for the BABEL program cover a wide variety of different language families. The number of phonemes per language ranges from 32 (Haitian Creole) to 68 (Vietnamese). As Tamil is the target language in our experiments, we also looked into the amount of phonemes that Tamil shares with each language. This information is presented in the last column of table 1

Language	**Language Family**	**# Ph.**	**# Ph. w. T.**
Tamil	Dravidian	34	-
Assamese	Indo-European	50	20
Bengali	Indo-European	51	21
Hait. Creole	(French) Creole	32	17
Lao	Tai-Kadai	41	20
Pashto	Indo-European	43	24
Tagalog	Austronesian	46	20
Turkish	Turkic	41	25
Vietnamese	Austroasiatic	68	18
Cantonese	Sino-Tibetan	37	14
Zulu	Niger-Congo	47	16

Table 1: Language overview, including the **language family**, size of **ph**oneme set and amount of **ph**onemes that each language shares **w**ith **T**amil

For each language, two data sets were provided: a limited language pack (LLP) and a full language pack (FLP). The LLP of a language consists of 10h of transcribed conversational speech. The FLP of a language consists of approximately 100h of transcribed data and includes the data from the LLP. The data itself is mainly narrowband telephone speech sampled with 8kHz. Some languages from the second year of the project (Assamese, Bengali, Haitian Creole, Tamil and Zulu) include some recordings with higher, CD-quality resolution. For our experiments, we resampled those down to 8kHz. The recordings contain different types of noises, as they were performed on the street, while driving a car or in an office with some machinery running in the background.

4.2. Baseline

The target language for our experiments is Tamil. For our baseline, we trained a system on the LLP dataset from Tamil only. First, we built a context-independent system from scratch using a flatstart approach. On top of that, we built a context-dependent system with 2,000 models.

Using that context-dependent system, we created the data required to train a DBNF. Our DBNFs consists of five hidden layers. With the exception of the bottle neck layer, each layer consists of 1,000 neurons. The bottle neck layer has only 42 neurons. For pre-training, we are using denoising auto-encoders with Gaussian noise and a corruption rate of 20%. To extract training data from the other languages for the neural network training, we used the FLP per language to train systems in a similar manner and to create that data.

In order to create forced alignments for languages other than Tamil, we trained a context-dependent system on the FLP dataset of that particular language. For selecting different amounts of training data, we randomly choose sets of speakers resembling the defined amount of audio data.

5. RESULTS

In our results we examined three different aspects: a) whether it is more important to use more data in multilingual DBNF training, or whether it is more important to have data from more, different languages; b) at which stages in the training of DBNFs is multilingual training data helpful; c) how to select the languages from which to train the DBNF for a specific language.

Therefore, we initially conducted an analysis to determine the performance of data from a single language in combination with the target language in Section 5.1. Parallel to that, we varied in Section 5.2 the amount of data for a selection of languages. We also investigated the use of additional language data at different points of neural network training in Section 5.3. Finally, we combined the best fitting languages together and as a contrastive experiment the worst fitting languages in Section 5.4.

5.1. Combination of a Single Language with Tamil

In order to establish a baseline for multilingual DBNFs we trained multilingual DBNFs on only two languages, by combining the data from the Tamil LLP with 40h of one other language. This will give a first impression of the usefulness of adding training data from other languages and will show the variance in performance depending on the exact language that was chosen to be added. We compare the resulting WERs against a baseline in which the DBNF was trained on Tamil LLP only. For this experiment, we used the multilingual data during pre-training, fine-tuning and the shifting step.

The results are shown in Table 2. One can see that the choice of language is important for the performance of the resulting DBNF. Some combinations lead to better performance, while others decrease the performance. The best results can be archived using Turkish, Pashto or Haitian Creole where we see gains of up to 1.6% relative in terms of WER over the monolingual baseline. Similar gains can be observed for ATWV. Here the best system (Turkish) improves from 2.67 to 3.96. However in the worst case, when choosing Vietnamese as additional language, the WER increases by 4.7% relative, while ATWV drops to -1.34.

The gains and losses correspond to some degree with the amount of shared phonemes between Tamil and each language. As shown in Table 2 the best fitting languages (Turkish and Pashto) share the largest amount of phonemes with Tamil, whereas the worst fitting languages (Vietnamese, Cantonese and Zulu) share the least phonemes with Tamil. But the amount of shared phonemes should only be considered as an approximation of the expected performance gain since for example Haitian Creole fits equally well to Tamil like Bengali and Pashto, although it only shares 17 phonemes with the target language.

Language	WER	ATWV	# Ph. w. T.
Baseline	82.6	2.67	-
+ Assamese	82.7	3.00	20
+ Bengali	81.5	3.26	21
+ Hait. Creole	81.5	3.82	17
+ Lao	82.3	2.97	20
+ Pashto	81.5	3.48	24
+ Tagalog	82.0	3.40	20
+ Turkish	**81.3**	**3.96**	**25**
+ Vietnamese	*86.5*	*-1.34*	*18*
+ Cantonese	83.3	1.53	14
+ Zulu	84.6	-0.04	16

Table 2: Tamil LLP plus additional 40h of another language. The last column shows the amount of **ph**onemes that each language shares **w**ith **T**amil

5.2. Varying the Amount of Additional Data

In the next experiment, we looked into varying the amount of foreign language data to Tamil LLP. Just as in the experiment before we added only one language to the Tamil training data, but this time either added the FLP (ca. 100h), 40h or 10h of training data of that language. We performed these experiments with the languages from the second year of the BABEL program. We added the language data to the whole training process of the neural network, including the pre-training, fine-tuning and shifting step.

Language	FLP	40h	10h
Assamese	82.4 / 2.54	82.7 / 2.37	82.0 / 3.28
Bengali	82.0 / 2.61	81.5 / 3.26	81.7 / 3.03
Hait. Creole	82.2 / 2.30	81.5 / 3.82	81.6 / 3.14
Lao	82.5 / 2.20	82.3 / 2.97	81.6 / 3.31

Table 3: Use of different amounts of data in combination with Tamil LLP. The number on the left denotes WER, the one on the right ATWV.

Table 3 shows the performance of the resulting systems. The results show that selecting the right amount of training data in addition to the 10h of Tamil training data is also important. Using all available data per language leads to performance degradation over the baseline. Matching the 10h of Tamil data with 10h of data from another language always leads to improvements over the baseline. For two out of the four languages taking 40h instead of 10h improves system performance even further, while for the other two languages this seems to be already too much training data, as performance starts to degrade again.

5.3. Methods of Using Data from Additional Languages

There are several steps in the training process of the neural network at which training data is used and therefore data from multiple languages can be added. Our training setup for neural networks consists of up to four steps: Pre-training, fine-tuning, shifting and again a fine-tuning step. For this experiment, we trained the networks used in three different ways, whereas multilingual data is being used in more and more steps: a) Using multilingual data only during pre-training, then performing the fine-tuning and shifting using data from Tamil LLP only. b) Using multilingual data for pre-training, fine-tuning and shifting. c) Using multilingual data for pre-training, fine-tuning, shifting and adding a fine-tuning step using data from Tamil LLP only.

This time we also looked at not only adding one language, but multiple languages to the DBNF training. We used 40h of data per language and the LLP for Tamil. Again, as in the previous experiment, we used data from the Babel second year languages Haitian Creole, Lao, Assamese and Bengali. This results in up to 160h of training data in addition to the 10h of data from the target language Tamil.

	H	H+L	H+L+A	H+L+A+B
a)	82.5 / 2.18	83.3 / 1.47	82.3 / 2.93	82.2 / 2.42
b)	81.5 / 3.82	81.2 / 3.63	80.8 / 4.06	80.6 / 4.05
c)	81.2 / 3.85	80.7 / 4.13	80.8 / 4.34	79.9 / 5.05

Table 4: Tamil LLP plus additional languages (**H**aitian Creole, **L**ao, **A**ssamese and **B**engali) and training methods: a) ML pre-training, b) ML pre-training and shifting, c) additional fine-tuning on Tamil LLP after shifting. The number on the left denotes WER, the one on the right ATWV.

As shown in Table 4, using the multilingual data in set-up a) (only for pre-training) yields only small gains, if at all. Set-up b) (using the multilingual data not only during pre-training, but also for fine-tuning and shifting) results in a gains of performance in all cases. The WER decreases up to 2.5% relative and the ATWV increases by 2.16. After applying another round of fine-tuning on Tamil only, and thus resulting in set-up c), performance is increased even further, by 0.5% relative in Terms of WER, and 1.0 in terms of ATWV.

As a general result, using additional data at all steps of the neural network training increases the performance the best. Likewise does an extra fine-tuning step on the target language after the multilingual training improve system performance further.

5.4. Combining Multiple Languages

Following our initial experiments using multiple languages and determining the performance of individual languages in combination with Tamil, we created two sets of four languages to do a first investigation into the best combination of multiple languages. One set consists of the best four languages according to Section 5.1. The second set contains the worst four languages. The languages are listed in Table 5.

Best fitting	Worst fitting
Turkish	Vietnamese
Hait. Creole	Zulu
Pashto	Cantonese
Bengali	Assamese

Table 5: Overview of languages fitting best and worst to Tamil. The best fitting languages are sorted starting with the best fitting language, the worst fitting languages are starting with the worst fitting language.

For this experiment, we use two schemes to determine the performance of the combination of the different languages. First, we used 40h per language, resulting in an additional amount of data of 40h, 80h, 120h and 160h of speech data. In a second set of experiments, we kept the amount additional data fixed to 40h, resulting in an amount of data per speech

depending on the number of additional languages. This results in an amount of 40h, 20h, 13h and 10h of additional data per language .

Language	40h p. l.	40h total	# Ph. w. T.
Baseline	82.6 / 2.67	82.6 / 2.67	-
T	81.3 / 3.96	81.3 / 3.96	25
T+H	81.0 / 4.50	80.9 / 4.21	25
T+H+P	80.3 / 5.41	80.5 / 4.66	28
T+H+P+B	79.7 / 5.65	79.9 / 5.52	28

Table 6: Use of additional languages (**T**urkish, **H**aitian Creole, **P**ashto and **B**engali) with either 40h of data **p**er **l**anguage or **40h** in **total** for all additional languages. The number on the left denotes WER, the one on the right ATWV. The last column shows the amount of **ph**onemes shared **w**ith **T**amil.

The results for combining the best systems are shown in Table 6. Integrating the best four languages into system training in addition to Tamil LLP decreases the WER by 3.6% relative and improves ATWV by 2.98. The results show that the more languages one adds, the better the performance gets. The difference between using 40h per additional languages and 40h of additional data in total is marginal. We see this as an indicator that the total amount of data used is not as important as the variety in the languages used for the multilingual training. Thus, when faced with the question of whether it is better to collect more data in few languages or more languages with fewer data, it is better to go for language diversity. The amount of shared phonemes corresponds here to some extent to the gain in recognition performance. Combining multiple languages increases the phoneme coverage of the target language.

Language	40h p. l.	40h total	# Ph. w. T.
Baseline	82.6 / 2.67	82.6 / 2.67	-
V	86.5 / -1.34	86.5 / -1.34	18
V+Z	82.4 / 1.98	82.5 / 1.91	20
V+Z+C	82.0 / 3.05	81.9 / 3.19	22
V+Z+C+A	81.6 / 3.90	81.7 / 3.85	24

Table 7: Use of additional languages (**V**ietnamese, **Z**ulu, **C**antonese and **A**ssamese) with either 40h of data **p**er **l**anguage or 40h in total for all additional languages. The number on the left denotes WER, the one on the right ATWV. The last column shows the amount of **ph**onemes shared **w**ith **T**amil.

When comparing the results of adding the best languages in Table 6 against the results in Table 7 where we added the worst languages, one sees that in the end, when adding enough languages, also adding bad performing languages gives gains over the baseline. Adding more languages increases the coverage of the phonemes here as well. The observed gain is comparable to using Pashto alone which has the same amount of shared phonemes with Tamil.

As described in section 2.2, other have also made use of the different phoneme sets by combining them. Although we compare the different phoneme sets, our system uses only the phonemes from the target language. We do not use the phonemes from the other languages explicitly. They were only used implicitly during the network training as target states while fine-tuning the network.

6. CONCLUSION AND OUTLOOK

In this work we have examined the use of multilingual DBNFs for Tamil speech recognition on the BABEL task. We have performed experiments to give insight into three questions: a) which is the best technique for training the multilingual DBNFs, b) is it more important to increase the total amount of training data or to vary the number of languages in DBNF training, c) which is the best selection of languages for multilingual training.

The experiments show that using multilingual data at all stages of our DBNFs (pre-training, fine-tuning, shifting stage) gives the best performance. Also, the total amount of training data is not as important as the variety of the languages in the multilingual training dataset. Some experiments suggest that adding too much data might from a certain point on decrease system performance again.

With respect to selecting suitable languages we compared the strategy of selecting those languages that give the best improvements when combined individually with the target language against selecting those that give the worst. Results show that selecting the best performing languages seems to be a reasonable strategy.

With respect to the question of how to select suitable languages, more experiments need to be performed. We have identified the amount of shared phonemes as a first indicator to predict the performance of the resulting system. But our work has also shown that additional metrics are required. Our goal is therefore to examine more strategies and try to find good strategies that are computationally in-expensive.

7. REFERENCES

[1] Sebastian Stüker, Markus Müller, Quoc Bao Nguyen, and Alex Waibel, "Training time reduction and performance improvements from multilingual techniques on the babel ASR task," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014.

[2] Quoc Bao Nguyen, Jonas Gehring, Markus Müller, Sebastian Stüker, and Alex Waibel, "Multilingual shift-

ing deep bottleneck features for low-resource ASR," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 5607–5611.

[3] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh, "A Fast Learning Algorithm for Deep Belief Nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, July 2006.

[4] Dong Yu and Michael L. Seltzer, "Improved bottleneck features using pretrained deep neural networks," in *INTERSPEECH*, 2011, pp. 237–240.

[5] L. Mangu, Hong-Kwang Kuo, S. Chu, B. Kingsbury, G. Saon, Hagen Soltau, and F. Biadsy, "The IBM 2011 GALE Arabic Speech Transcription System," in *Proceedings of the ASRU*, Waikoloa, HI, USA, December 2011.

[6] T.N. Sainath, B. Kingsbury, and B. Ramabhadran, "Auto-Encoder Bottleneck Features Using Deep Belief Networks," in *Proceedings of the ICASSP*, Kyoto, Japan, March 2012.

[7] Jonas Gehring, Yajie Miao, Florian Metze, and Alex Waibel, "Extracting Deep Bottleneck Features Using Stacked Auto-Encoders," in *Proceedings of the ICASSP*, Vancouver, Canada, May 2013.

[8] Pawel Swietojanski, Arnab Ghoshal, and Steve Renals, "Unsupervised cross-lingual knowledge transfer in DNN-based LVCSR," in *Proceedings of the Spoken Language Technology Workshop (SLT), 2012 IEEE*. IEEE, 2012, pp. 246–251, IEEE.

[9] Ngoc Thang Vu, Wojtek Breiter, Florian Metze, and Tanja Schultz, "Initialization Schemes for Multilayer Perceptron Training and their Impact on ASR Performance using Multilingual Data," in *Proceedings of the INTERSPEECH*, Portland, Oregon, September 2012.

[10] T. Schultz and A. Waibel, "Language Independent and Language Adaptive Acoustic Modeling for Speech Recognition," *Speech Communication*, vol. 35, no. 1-2, pp. 31–51, August 2001.

[11] Karel Vesely, Martin Karafiat, Frantisek Grezl, Milos Janda, and Ekaterina Egorova, "The language-independent bottleneck features," in *Proceedings of the Spoken Language Technology Workshop (SLT), 2012 IEEE*. 2012, pp. 336–341, IEEE.

[12] G. Heigold, V. Vanhoucke, A. Senior, P. Nguyen, M. Ranzato, M. Devin, and J. Dean, "Multilingual Acoustic Models Using Distributed Deep Neural Networks," in *Proceedings of the ICASSP*, Vancouver, Canada, May 2013.

[13] Stefano Scanzio, Pietro Laface, Luciano Fissore, Roberto Gemello, and Franco Mana, "On the use of a multilingual neural network front-end," in *Proceedings of the Interspeech*, 2008, pp. 2711–2714.

[14] Arnab Ghoshal, Pawel Swietojanski, and Steve Renals, "Multilingual training of Deep-Neural networks," in *Proceedings of the ICASSP*, Vancouver, Canada, 2013.

[15] Samuel Thomas, Michael L Seltzer, Kenneth Church, and Hynek Hermansky, "Deep neural network features and semi-supervised training for low resource speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 6704–6708.

[16] KM Knill, Mark JF Gales, Shakti P Rath, Philip C Woodland, Chao Zhang, and S-X Zhang, "Investigation of multilingual deep neural networks for spoken term detection," in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*. IEEE, 2013, pp. 138–143.

[17] Frantisek Grézl and Martin Karafiát, "Adapting multilingual neural network hierarchy to a new language," in *Spoken Language Technologies For Under-resourced Languages (SLTU), 2014 4th International Workshop on*, 2014, pp. 39–45.

[18] Kornel Laskowski, Mattias Heldner, and Jens Edlund, "The Fundamental Frequency Variation Spectrum," in *Proceedings of the 21st Swedish Phonetics Conference (Fonetik 2008)*, Gothenburg, Sweden, June 2008, pp. 29–32.

[19] Kjell Schubert, "Grundfrequenzverfolgung und deren Anwendung in der Spracherkennung," M.S. thesis, Universität Karlsruhe (TH), Germany, 1999, In German.

[20] František Grézl, Martin Karafiát, Stanislav Kontár, and J Cernocky, "Probabilistic and bottle-neck features for LVCSR of meetings," in *Proceedings of the Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*. IEEE, 2007, pp. V–757 – IV–760.

[21] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K.J. Lang, "Phoneme recognition using time-delay neural networks," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 37, no. 3, pp. 328–339, Mar 1989.

[22] Monika Woszczyna, N. Aoki-Waibel, Finn Dag Buø, Noah Coccaro, Keiko Horiguchi, Thomas Kemp, Alon Lavie, Arthur McNair, Thomas Polzin, Ivica Rogina, Carolyn Rose, Tanja Schultz, Bernhard Suhm, M. Tomita, and Alex Waibel, "Janus 93: Towards spontaneous speech translation," in *International Conference*

on Acoustics, Speech, and Signal Processing 1994, Adelaide, Australia, 1994.

[23] Hagen Soltau, Florian Metze, Christian Fugen, and Alex Waibel, "A one-pass decoder based on polymorphic linguistic context assignment," in *Automatic Speech Recognition and Understanding, 2001. ASRU'01. IEEE Workshop on*. IEEE, 2001, pp. 214–217.

[24] David RH Miller, Michael Kleber, Chia-Lin Kao, Owen Kimball, Thomas Colthurst, Stephen A Lowe, Richard M Schwartz, and Herbert Gish, "Rapid and accurate spoken term detection.," in *INTERSPEECH*, 2007, pp. 314–317.