THE 2016 BBN GEORGIAN TELEPHONE SPEECH KEYWORD SPOTTING SYSTEM

Tanel Alumäe[*†], *Damianos Karakos, William Hartmann, Roger Hsiao, Le Zhang*
Long Nguyen, Stavros Tsakalidis, Richard Schwartz

Raytheon BBN Technologies, Cambridge, MA, USA
{talumae, dkarakos, whartmann, whsiao, lzhang, ln, stavros, schwartz}@bbn.com

ABSTRACT

In this paper we describe the 2016 BBN conversational telephone speech keyword spotting system; the culmination of four years of research and development under the IARPA Babel program. The system was constructed in response to the NIST Open Keyword Search (OpenKWS) evaluation of 2016. We present our technological breakthroughs in building top-performing keyword spotting processing systems for new languages, in the face of limited transcribed speech, noisy conditions, and limited system build time of one week.

***Index Terms*—** keyword spotting, under-resourced languages, multilingual training, subword systems

1. INTRODUCTION

The IARPA Babel program recently completed its fourth and final year. The principal objective of Babel was to develop a keyword search (KWS) system that delivers high accuracy for any new language, in the face of very limited transcribed speech, noisy acoustic and channel conditions, and limited system build time of one week.

In this paper, we present a top-performing keyword spotting system for conversational telephone speech that BBN constructed in response to the NIST Open Keyword Search Evaluation (OpenKWS) evaluation of 2016. The 2016 system was the culmination of 4 years of research and development under the IARPA Babel program.

The system contains—beyond the technology introduced in the first three years of the program—several, recently developed, novel methods that significantly improve speech-to-text (STT) and KWS performance. We incorporated data augmentation for improving the speaker and environmental variability of the data. We assessed the efficacy of various recently developed neural network (NN) models. We developed various methods to improve the performance of NN based acoustic models for low resource languages, including joint alignment. We addressed multilingual NN training from imbalanced data sources. We further improved our subword models by modifying our automatic syllabification algorithm. The next sections describe, to the extent possible, the components of our top-performing keyword spotting system.

[*]We would like to thank our BABELON team partners for their contributions: Brno University of Technology helped to apply the WPE algorithm. North-West University worked on the syllabification part. We would also like to thank the members of the BBN Speech and Language group for useful discussions. This work was supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense US Army Research Laboratory contract number W911NF-12-C-0013. 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.

[†]Now at Tallinn University of Technology, Estonia

2. DATA

The data for the IARPA Babel program consists of conversational telephone speech from 25 languages. Over the course of the program, the amount of in-language data available for the target language has varied between 3 and 80 hours of transcribed speech. Speech is collected in real-life usage scenarios and includes mainly conversational telephone speech, recorded under different conditions, such as mobile phone conversation made on the street. Most of the languages also contain a small amount of data collected using a distant microphone. Each year a set of languages are given as development languages for the purpose of research and testing, before a surprise language is given for final evaluation. For the fourth and final year, Georgian was selected as the surprise language.

We participated in the primary evaluation — full language pack (FLP) — consisting of approximately 40 hours of transcribed speech and 40 hours of untranscribed speech. Instead of a pronunciation lexicon, IARPA provides a Language Specification Peculiarities (LSP) document that details the common letter to IPA mappings. While Georgian consists of many dialects, they were merged into two groups (Eastern and Western) in the LSP. Given the broad classification of dialects provided for the training data, we did not explore any dialect-specific approaches.

The primary evaluation metric for Babel is actual term-weighted value (ATWV), though we also report WER results. ATWV is a keyword spotting metric. A score is generated for each individual keyword and then averaged together to obtain the final result. While each keyword is treated equally, that is not true of each individual detection; it is more important to recognizer any instance of a rare keyword than of a common keyword. All false alarms are treated equally. For more detailed information about the metric, see [1].

3. SPEECH RECOGNITION

We use BBN's *Sage* STT platform [2] for building the system. Sage integrates technologies from multiple sources, each of which has a particular strength. In Sage, we combine proprietary sources, such as BBN's Byblos [3], with open source toolkits, such as Kaldi [4] and CNTK [5]. Sage also includes a cross-toolkit FST recognizer that supports models built using the various component technologies, and software supporting keyword search from Byblos [6].

3.1. Pronunciation dictionary

The pronunciation dictionary was directly obtained from the letter-to-phone mappings defined in the LSP.

3.2. Multilingual bottleneck features

Language	#Hours	Release
Amharic	40	IARPA-babel307b-v1.0b
Assamese	80	IARPA-babel102b-v0.4
Bengali	80	IARPA-babel103b-v0.3
Cantonese	100	IARPA-babel101b-v0.4c
Cebuano	40	IARPA-babel301b-v2.0b
Dholuo	40	IARPA-babel403b-v1.0b
Guarani	40	IARPA-babel305b-v1.0a
Haitian	80	IARPA-babel201b-v0.2b
Igbo	40	IARPA-babel306b-v2.0c
Javanese	40	IARPA-babel402b-v1.0b
Kazakh	40	IARPA-babel302b-v1.0a
Kurdish	40	IARPA-babel205b-v1.0a
Lao	85	IARPA-babel203b-v3.1a
Lithuanian	40	IARPA-babel304b-v1.0b
Mongolian	40	IARPA-babel401b-v2.0b
Pashto	87	IARPA-babel104b-v0.4bY
Swahili	47	IARPA-babel202b-v1.0d
Tamil	50	IARPA-babel204b-v1.1b
Tagalog	86	IARPA-babel106b-v0.2g
Telugu	40	IARPA-babel303b-v1.0a
Tok Pisin	40	IARPA-babel207b-v1.0b
Turkish	88	IARPA-babel105b-v0.4
Vietnamese	85	IARPA-babel107b-v0.7
Levantine Arabic	180	LDC2006S29, LDC2006T07
English (random subset of Fisher)	265	LDC2004S13, LDC2004T19, LDC2005S13, LDC2005T19
Mandarin (HKUST, CallHome, CallFriend)	260	LDC2005S15, LDC2005T32, LDC96S34, LDC96T16, LDC96L15, LDC98S69, LDC98T26, LDC96L15
Spanish (Fisher, CallHome, CallFriend)	235	LDC2010S01, LDC2010T04, LDC96S35, LDC96T17, LDC98S70, LDC98T27

Table 1. Languages used for multilingual training, together with the training data sizes and release identificators.

Almost all acoustic models in our combined system use fMLLR-transformed multilingual bottleneck (BN) features [7]. The feature extractor is trained using 23 Babel languages and four non-Babel languages (see Table 1). One the Babel languages (Zulu) was excluded from the multilingual pool to be used as a development language for testing the porting of multilingual models. Total duration of the training data is about 2300 hours. Training data is artificially doubled using speed and noise augmentation [8]. We use 32 dimensional filterbank features, combined with 3 Kaldi pitch features as input to the BN model. The input features are stacked to accumulate a temporal context of 11 frames. All hidden layers use the p-norm non-linearity [9]. The BN network has three hidden layers before and one after the BN layer. The BN p-norm layer has an input dimensionality of 800 and output dimensionality of 80. Other hidden layers have 5000 input units and 500 output units. The output layer is a block-softmax [7] with approximately 4500 context-dependent targets per language. The BN model was trained before the Georgian training data became available which demonstrates the usefulness of multilingual features for rapid porting to a new language. The training data was scaled using language specific weights, in order to equalize the contribution of each language [10]. The BN extractor was ported to Georgian by first replacing the block-softmax with the Georgian-specific softmax, training the softmax layer while keeping the rest of the model fixed, and finally training the whole model, using a 10 times smaller learning rate than the original [11].

The BN features are used for GMM-based speaker-adapted training (SAT), and the fMLLR-transformed BN features are used as input to the acoustic models.

Typically about 15% of the BABEL development and evaluation data includes speech recorded using a distant microphone. We dereverberated the microphone data with a weighted prediction error (WPE) algorithm [12]. Based on our experience with BABEL development languages, this can improve the WER of distant microphone data by around 3% absolute.

3.3. Acoustic models

We explored a set of diverse acoustic models in our combined system: a simple feed-forward DNN, LSTM and BLSTM models, and finally time-delay neural network (TDNN) and BLSTM based "chain" models. Those models are trained on multilingual features. We also train a DNN, LSTM and BLSTM model using monolingual BN features and a CNN using filterbank features. All models are trained on one copy of clean and one additional copy of speed and noise perturbed data. A PLP-based SAT GMM-HMM system was first trained from flat start to obtain initial alignments for the DNN and CNN models. LSTM, BLSTM and chain models were trained using alignments generated with the DNN model.

DNN. For this model, we use a temporal context of 13 stacked frames (-10, $-5..5$, 10), it has six 2048-dimensional hidden layers with the sigmoid activation function, and the output layer has approximately 4500 tied-state targets. Unsupervised RBM-based pre-training is used to initialize the weights of the hidden layers. This is followed by four epochs of cross-entropy training and five epochs of SMBR-based sequence discriminative training.

MultiDNN. The multilingual DNN acoustic model [10] is trained on the same set of 27 languages as the multilingual BN feature extractor. Similarly to the simple DNN, MultiDNN uses 13 stacked fMLLR-transformed features as input. The fMLLR transforms are estimated independently for each language. The MultiDNN parameters are initialized using RBM-based unsupervised pre-training over a subset of the multilingual data. The architecture of MultiDNN is similar to the simple DNN, with one additional dimension-reducing p-norm non-linearity with 3500 input units and 350 output units before the softmax layer. Since the block-softmax layer of a multilingual DNN-AM is very large (around 130 000 in our experiments), this additional p-norm layer greatly reduces the number of parameters of the hidden-to-softmax layer and thus the whole multilingual DNN-AM and makes DNN training much faster. The strategy of porting the MultiDNN model to Georgian is similar to porting of the BN feature extractor: first, we replace the block-softmax output layer of the MultiDNN with a softmax for Georgian, train it for two epochs, and finally train the whole DNN, using a smaller learning rate. Finally, five epochs of SMBR-based sequence discriminative training are performed.

LSTM and BLSTM acoustic models are based on the work in [13], where each LSTM or BLSTM layer consists of peephole connections and a recurrent projection layer. Our LSTM and BLSTM models each consist of four hidden layers. The first two layers are fully connected layers like a regular DNN and the last two layers are LSTM/BLSTM layers. The fully connected layers have

2048 sigmoid units and they are initialized with RBM. For the LSTM model, the LSTM layer has 1024 memory cells and the recurrent projection layer would project the output to 512 dimensions. For the BLSTM model, each BLSTM layer has two directions: the forward direction and the backward direction. Each direction is a regular LSTM with 512 memory cells and 300 dimensional projected output. The LSTM and BLSTM layers are initialized randomly except for the bias of the forget gate, which is initialized as a vector of ones according to [14]. The LSTM and BLSTM are first trained with cross entropy training followed by SMBR training. The initial alignment for cross entropy training comes from the DNN model. However, once we train an initial LSTM and BLSTM, we realign the training set with a joint model of DNN, LSTM and BLSTM with equal weights similar to the work in [15]. The new alignment is then used to retrain the BLSTM. During our initial investigation, we found the LSTM and BLSTM generally perform well in terms of WER. However, the LSTM based models tend to be too sharp that the lattices generated from these models have suboptimal KWS performance. To improve KWS performance, the objective functions used in the cross entropy and SMBR training are multi-task functions. The objective functions contain an extra term, which is the cross entropy of the DNN output and the LSTM/BLSTM output. The term is weighted by a scalar, 0.25, which is determined empirically. By doing so, the LSTM and BLSTM models could learn from the DNN model to prevent the models being too sharp. We find that this technique can improve both STT and KWS performance for LSTM and BLSTM models. Based on the same idea, we improved our linear least squares based adaptation (LLS) proposed in [16], such that, instead of using a hypothesis for adaptation, we could adapt to a NN output directly.

ChainTDNN and ChainBLSTM. We also train TDNN and BLSTM based "chain" models that were recently implemented in Kaldi [17]. Chain models use many techniques that make them different from traditional DNNs and (B)LSTMs, suchs as CTC-like lattice-free maximum mutual information (LF-MMI) criterion based training without the need for frame-level cross-entropy pre-training, the use of 3-fold reduced frame rate and a different HMM topology. Therefore, the chain models provide diversification to our system combination which is known to bring substantial benefits. The TDNN-based chain model has six 450-dimensional hidden layers with ReLU activation, and the splicing indices per layer are -1,0,1 0 -3,0,3 0 -3,0,3 0 -6,-3,0 0. We reduced the width of splicing slightly compared to the Kaldi recipes since we use BN features that are already based on spliced filterbank features. Cross-entropy regularization, output l_2 regularization and the leaky HMM method are applied during training, similarly to the appropriate Kaldi recipes. We found slight improvements (around 0.5 in WER) from using DNN-generated lattices as input to the training, as opposed to GMM-generated lattices. Around 2000 left-biphone dependent states are used as TDNN targets. The ChainTDNN model is trained using four epochs of LF-MMI and five epochs of SMBR. The BLSTM-based chain model has three BLSTM layers each with 650 units, and 160-dimensional recurrent and non-recurrent projections. It is trained using only five epochs of LF-MMI.

CNN model was trained using filterbank features. The general structure of the model is similar to the one described in [18] — two convolutions layers followed by four fully connected layers. The first convolutional layer uses 128 filters of size 9 in the filterbank dimension. Note that we only a one-dimensional convolution; our tests using two-dimensional convolution did not improve performance. The second layer doubles the filters to 256 and reduces the filter size to 4. In between the two convolution layers there is also a 3x3 max pooling layer. We also use pitch features, but they do not pass through the convolution layers. Instead they are fed to two smaller fully connected layers in parallel. The output of the smaller pitch network is concatenated with the output of the two convolution layers before being passed to four fully connected layers with 2048 hidden nodes each.

Total lexicon size	#Words from web data	*OOV*	*WER*	*ATWV*
37k	0	8.7	40.7	0.700
87k	50k	5.0	38.0	0.726
137k	100k	3.9	37.5	0.731
187k	150k	3.3	37.1	0.734
237k	200k	2.9	36.8	0.735
537k	500k	1.8	36.6	0.739

Table 2. Comparision of development set OOV rates between training and web-enhanced lexicons with 50k to 500k new words added.

3.4. Language Modeling

We improve the language model (LM) using web text documents contributed by BBN and Columbia University under the Babel program. The BBN part of the web text is obtained through an automatic procedure that queries a search engine with query terms similar to those found in the training transcripts[19]. We use the Bing Search API in our experiments, although the algorithm is search engine agnostic. We post-process the downloaded web text using the XenC tool[20], which selects high-confidence text based on the cross-entropy between training data and the provided web text. Of the 328 million words from the web text, 36 million words (11%) are retained for language modeling after the filtering.

In addition, we increase the size of the decoding lexicon by adding new words selected from the web text using frequency ranking. This step, often called lexicon expansion, can substantially reduce the out-of-vocabulary (OOV) rates. We observed big gains from using the LM enhanced with web data. Table 2 lists the OOV rates with WER and ATWV results, with increasingly large LM lexicons, when using a DNN with multilingual features as the AM. The ATWV improvement from using web texts in the LM was between 2.6 and 3.9 points—one of the largest that we have seen across all the BABEL languages.

In order to balance OOV rate reduction and runtime memory requirements, we used the LM with 150k new words in most of our final decoding runs. To provide more diversity, some decodings were run with a 500k LM. The LMs were constructed by interpolating the standard trigram LM estimated from the different training corpora, using weights optimized on the development set. The resulting LMs were pruned before converting to a FST and being used in decoding.

3.4.1. Subword-based models

In addition to whole words, we also ran decoding using subword units. Subword units offer the benefit of diversity when combining multiple acoustic models and tokenizations, as well as improved detection for OOV keywords [21]. We used automatically-generated syllable-like units [22] as subwords. Moreover, a variant of the syllabification algorithm was used this year [23]) with the goal of increasing the diversity of the subword units as much as possible.

The process of generating and using the subword units mostly followed the one described in [21, 24]. Note that decoding involves subword units was well as their *compounded* variants. E.g., if a word is segmented into subwords A B C, the pseudo-words A, B, C, AB, BC, ABC are added to the lexicon. The keyword spotting pipeline follows the steps mentioned in the next section. Both IV and OOV keywords are detected as sequences of subword units, using fuzzy matching as an additional mode of search.

4. KEYWORD SEARCH

The keyword search pipeline contains the following steps:

1. Decoder lattices are converted into posterior lattices, where the score of each arc is set to the posterior probability of following a path that contains the arc.
2. Confusion networks [25], and time-quantized lattices [26] are generated from the posterior lattices.
3. Phonetic confusion networks are generated from phone-transformed posterior lattices, as described in [27].
4. Whole-unit search for IV keywords is performed on confusion networks and time-quantized lattices from step 2 above.
5. Approximate match search for OOV keywords is performed on confusion networks, time-quantized lattices and phonetic confusion networks. Two methods are used for approximate match: (i) fuzzy-phonetic search of [28] and proxy keywords [29]. As mentioned in [26], these two methods are complementary and give gains when combined.
6. A total of 5-6 hit lists are generated from each one of the above techniques. These hit lists are subsequently *normalized* based on the linear fit method [30].
7. The normalized hit lists are then merged together (each one of the resulting hits is assigned a vector of scores from the individual search methods) and then a single score is computed through a linear combination of the scores (this is done separately for IV and OOV keywords, as the strength of each search technique depends on the keyword type). The weights are trained using Powell's method [31].
8. The scores of the two resulting hit lists (IV, OOV) are transformed using the "probability of correct", so that they resemble posteriors [31].
9. A decision threshold is chosen for each hit list based on the Dev set. Finally, the two hit lists (IV, OOV) are merged together and a single threshold is computed using the "exponential normalization" formula of [31].

5. RESULTS

Table 3 presents the speech-to-text (STT) and KWS results of the individual systems on development data. In order to reduce the number of final systems and to increase the quality of the single decoding runs, several systems actually perform joint decoding using multiple acoustic models. Three of the systems use pseudo-syllables as LM units. We didn't convert the STT output of the syllable-based systems to word-based output, as during the development phase we found that these were not competitive with the outputs of the word-based systems. However, it can be seen that the best KWS results were actually obtained from the pseudo-syllable based systems.

Tables 4 and 5 present system combination results on development and evaluation data in terms of STT and KWS performance. STT results were obtained by running ROVER [32] on the 1-best outputs of the word-based decoding runs. KWS results were combined using the hitlist combination technique [33]. The order of adding individual systems to the combination was determined heuristically, by selecting models with good performance while also trying to maximize the diversity of acoustic and language models. It is perhaps surprising that the absolute improvement from system combination on evaluation data is larger for WER (3.9%) than for ATWV (1.8%).

Features	*Model(s)*	*WER*	*ATWV*
	Words as LM units		
Multi	BLSTM	35.9	0.731
Multi	ChainTDNN	36.1	0.749
Multi	ChainBLSTM	35.1	0.743
Mono	DNN+CNN+LSTM+BLSTM	36.4	0.732
Multi	DNN+MultiDNN+LSTM+BLSTM	**34.4**	0.754
	Pseudo-syllables as LM units		
Multi	BLSTM		0.736
Mono	DNN+CNN+LSTM+BLSTM		0.730
Multi	DNN+MultiDNN+BLSTM+LSTM		**0.759**

Table 3. WER and ATWV results of the individual decoding runs on dev data, including joint decodings with multiple models.

Systems	*Dev*	*Eval*
Multi/DNN+MultiDNN+LSTM+BLSTM	34.4	32.1
+ Multi/ChainBLSTM	33.2	29.2
+ Mono/DNN+CNN+LSTM+BLSTM	32.8	28.8
+ Multi/BLSTM	32.4	28.4
+ Multi/ChainTDNN	32.2	28.2

Table 4. WER results on development and evaluation data, using a ROVER combination from an increasing number of systems.

6. CONCLUSION

This paper described the BBN conversational telephone speech KWS system for Georgian, built under the IARPA BABEL program for the 2016 NIST OpenKWS evaluation. Our system combines monolingual and multilingual acoustic features, various neural network based acoustic models with word and pseudo-syllable based language models. The combined system achieved a KWS performance of 0.873 in ATWV and a WER of 28.2%.

Systems	*Dev*	*Eval*
Multi/DNN+MultiDNN+BLSTM+LSTM/syll	0.759	0.855
+ Multi/ChainTDNN	0.770	0.865
+ Multi/DNN+MultiDNN+LSTM+BLSTM	0.771	0.867
+ Mono/DNN+CNN+LSTM+BLSTM	0.775	0.868
+ Multi/BLSTM/syll	0.778	0.871
+ Multi/ChainBLSTM	0.783	0.872
+ Mono/DNN+CNN+LSTM+BLSTM/syll	0.784	0.873
+ Multi/BLSTM	0.784	0.873

Table 5. ATWV results on development and evaluation data, using a combination of hitlists from an increasing number of systems.

7. REFERENCES

[1] S. Wegmann, A. Faria, A. Janin, K. Riedhammer, and N. Morgan, "The tao of ATWV: Probing the mysteries of keyword search performance," in *ASRU*, 2013.

[2] R. Hsiao, R. Meermeier, T. Ng, Z. Huang, M. Jordan, E. Kan, T. Alumäe, J. Silovsky, W Hartmann, F. Keith, O. Lang, M. Siu, and O. Kimball, "Sage: The new BBN speech processing platform," in *Interspeech*, 2016.

[3] S. Tsakalidis, R. Hsiao, D. Karakos, T. Ng, S. Ranjan, G. Saikumar, L. Zhang, L. Nyugen, R. Schwartz, and J. Makhoul, "The 2013 BBN Vietnamese telephone speech keyword spotting system," in *ICASSP*, 2014.

[4] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The Kaldi speech recognition toolkit," in *ASRU*, 2011.

[5] D. Yu, A. Eversole, M. Seltzer, K. Yao, B. Guenter, O. Kuchaiev, F. Seide, H. Wang, J. Droppo, Z. Huang, Y. Zhang, G. Zweig, C. Rossbach, J. Currey, J. Gao, A. May, A. Stolcke, and M. Slaney, "An introduction to computational networks and the computational network toolkit," Tech. Rep., Microsoft Research, 2014.

[6] T. Ng, R. Hsiao, L. Zhang, D. Karakos, S. H. Mallidi, M. Karafiát, K. Vesely, I. Szoke, B. Zhang, L. Nyugen, and R. Schwartz, "Progress in the BBN keyword search system for the DARPA RATS program," in *Interspeech*, 2014, pp. 959–962.

[7] S. Scanzio, P. Laface, L. Fissore, R. Gemello, and F. Mana, "On the use of a multilingual neural network front-end," in *Interspeech*, 2008.

[8] W. Hartmann, T. Ng, R. Hsiao, S. Tsakalidis, and R. Schwartz, "Two-stage data augmentation for low-resourced speech recognition," in *Interspeech*, 2016.

[9] X. Zhang, J. Trmal, D. Povey, and S. Khudanpur, "Improving deep neural network acoustic models using generalized maxout networks," in *ICASSP*, 2014, pp. 215–219.

[10] T. Alumäe, S. Tsakalidis, and R. Schwartz, "Improved multilingual training of stacked neural network acoustic models for low resource languages," in *Interspeech*, 2016.

[11] F. Grézl and M. Karafiát, "Bottle-neck feature extraction structures for multilingual training and porting," in *SLTU*, 2016.

[12] T. Nakatani, T. Yoshioka, K. Kinoshita, M. Miyoshi, and B.-H. Juang, "Blind speech dereverberation with multi-channel linear prediction based on short time Fourier transform representation," in *ICASSP*, 2008, pp. 85–88.

[13] H. Sak, A. W. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," in *Interspeech*, 2014.

[14] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: continual prediction with lstm," in *ICANN*, 1999, vol. 2, pp. 850–855 vol.2.

[15] G. Saon, T. Sercu, S. Rennie, and H.-K. J. Kuo, "The IBM 2016 English Conversational Telephone Speech Recognition System," in *Interspeech*, 2016.

[16] R. Hsiao, T. Ng, S. Tsakalidis, L. Nguyen, and R. Schwartz, "Unsupervised Adaptation for Deep Neural Network using Linear Least Square Method," in *Interspeech*, 2015.

[17] D. Povey, V. Peddinti, D. Galvez, P. Ghahrmani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely sequence-trained neural networks for ASR based on lattice-free MMI," in *Interspeech*, 2016.

[18] T. Sainath, A. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," in *ICASSP*, May 2013, pp. 8614–8618.

[19] L. Zhang, D. Karakos, W. Hartmann, R. Hsiao, R. Schwartz, and S. Tsakalidis, "Enhancing low resource keyword spotting with automatically retrieved web documents," in *Interspeech*, 2015.

[20] A. Rousseau, "Xenc: An open-source tool for data selection in natural language processing," *The Prague Bulletin of Mathematical Linguistics*, , no. 100, pp. 73–82, 2013.

[21] D. Karakos and R. Schwartz, "Subword and phonetic search for detecting out-of-vocabulary keywords," in *Interspeech*, 2014.

[22] M. Davel, E. Barnard, C. van Heerden, W. Hartmann, D. Karakos, R. Schwartz, and S. Tsakalidis, "Exploring minimal pronunciation modeling for low resource languages," in *Interspeech*, 2015.

[23] C. van Heerden et al., "Constructing sub-word units for spoken term detection," in *ICASSP*, 2017, submitted.

[24] I. Bulyko, J. Herrero, C. Mihelich, and O. Kimball, "Subword speech recognition for detection of unseen words," in *Interspeech*, 2012.

[25] L. Mangu, E. Brill, and A. Stolcke, "Finding consensus in speech recognition: word error minimization and other applications of confusion networks," *Computer Speech and Language*, vol. 14, no. 4, pp. 373–400, 2000.

[26] D. Karakos and R. Schwartz, "Combination of search techniques for improved spotting of OOV keywords," in *ICASSP*, 2015.

[27] I. Bulyko, O. Kimball, M.-H. Siu, J. Herrero, and D. Blum, "Detection of unseen words in conversational Mandarin," in *ICASSP*, Kyoto, Japan, Mar 2012.

[28] I. Bulyko, J. Herrero, C. Mihelich, and O. Kimball, "Subword speech recognition for detection of unseen words," in *Interspeech*, Portland, Oregon, Sep 2012.

[29] G. Chen, O. Yilmaz, J. Trmal, D. Povey, and S. Khudanpur, "Using proxies for OOV keywords in the keyword search task," in *ASRU*, 2013.

[30] D. Karakos, I. Bulyko, R. Schwartz, S. Tsakalidis, L. Nguyen, and J. Makhoul, "Normalization of phonetic keyword search scores," in *ICASSP*, Florence, Italy, 2014.

[31] D. Karakos, R. Schwartz, S. Tsakalidis, L. Zhang, S. Ranjan, T. Ng, R. Hsiao, G. Saikumar, I. Bulyko, L. Nguyen, J. Makhoul, F. Grezl, M. Hannemann, M. Karafiat, I. Szoke, K. Vesely, L. Lamel, and V.-B. Le, "Score normalization and system combination for improved keyword spotting," in *ASRU*, Olomouc, Czech Republic, 2013.

[32] J. G Fiscus, "A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover)," in *ASRU*, 1997.

[33] D. Karakos, R. Schwartz, S. Tsakalidis, L. Zhang, S. Ranjan, T. Ng, R. Hsiao, G. Saikumar, I. Bulyko, L. Nguyen, et al., "Score normalization and system combination for improved keyword spotting," in *ASRU*, 2013.