A KEYWORD SEARCH SYSTEM USING OPEN SOURCE SOFTWARE

Jan Trmal[1], *Guoguo Chen*[1], *Dan Povey*[1], *Sanjeev Khudanpur*[1],
Pegah Ghahremani[1], *Xiaohui Zhang*[1], *Vimal Manohar*[1], *Chunxi Liu*[1]
Aren Jansen[1], *Dietrich Klakow*[2], *David Yarowsky*[1], *Florian Metze*[3]

[1]Johns Hopkins University, Baltimore, MD, USA
[2]Saarland University, Saarbrücken, Germany
[3]Carnegie Mellon University, Pittsburgh, PA, USA
radical@clsp.jhu.edu

ABSTRACT

Provides an overview of a speech-to-text (STT) and keyword search (KWS) system architecture build primarily on the top of the Kaldi toolkit and expands on a few highlights. The system was developed as a part of the research efforts of the Radical team while participating in the IARPA Babel program. Our aim was to develop a general system pipeline which could be easily and rapidly deployed in any language, independently on the language script and phonological and linguistic features of the language.

***Index Terms*—** Kaldi, spoken term detection, keyword search, speech recognition, deep neural networks, pitch, IARPA BABEL, OpenKWS

1. BACKGROUND

The IARPA BABEL program aims to achieve the capability to rapidly develop speech-to-text (STT) and keyword search (KWS) systems in new languages with limited linguistic resources—transcribed speech, pronunciation lexicon and matched text—with emphasis on conversational speech.

The four BABEL program participants were evaluated by NIST via two benchmark tests: on five *development languages* and on a *surprise language* revealed only at the beginning of the evaluation period. The development languages were Assamese, Bengali, Haitian Creole, Lao and Zulu, and the surprise language was Tamil. Eight additional teams worldwide participated in the surprise language evaluation.

The primary 2014 evaluation was on KWS performance using systems trained on an IARPA-provided *limited language pack* (LimitedLP) containing 10 hours of transcribed speech, a dictionary that covered words in the transcripts, 70 hours of un-transcribed speech for unsupervised training, and 10 hours of transcribed speech for development-testing. A secondary evaluation was on KWS performance using a *full language pack* (FullLP), in which transcripts and dictionary entries were provided for an additional 50 of the 70 hours of un-transcribed speech: total 60 hours transcribed.[1]

The test data provided by NIST contained 15 hours of speech for each development language, 75 hours for the surprise language, and a list of ca 3000 keywords for each language. The primary KWS evaluation metric was *actual term weighted value* (ATWV), and the BABEL program goal for 2014 was to attain an ATWV of 0.30 in the LimitedLP training condition on all six languages.

This paper describes the system submitted to NIST by the JHU Kaldi team. It is expected to interest readers because the submitted system attained all the program goals, enabling the RADICAL team to achieve third place worldwide, and because 9 of the top 10 participants in the NIST evaluation used Kaldi components/recipes[2] in their submitted system.

2. JHU KALDI SYSTEMS OVERVIEW

The Kaldi KWS system is comprised of LVCSR based lattice generation followed by OpenFST based indexing and keyword search. LVCSR systems based on four different acoustic models are used to decode and index the speech:

1. A subspace Gaussian mixture model (**SGMM**) of the type described in [1], trained discriminatively via

This work was supported by NSF IIS award No 0963898, DARPA BOLT contract No HR0011-12-C-0015, and IARPA BABEL contract No W911NF-12-C-0015. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. 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 NSF, DARPA, IARPA, DoD/ARL or the U.S. Government.

[1]The exact corpus identifiers are
Assamese, `IARPA-babel102b-v0.4`;
Bengali, `IARPA-babel103b-v0.3`;
Haitian Creole, `IARPA-babel201b-v0.2b`;
Lao, `IARPA-babel203b-v3.1a`;
Tamil, `IARPA-babel204b-v1.1b`;
Zulu, `IARPA-babel206b-v0.1e`.

[2]Available via `http://kaldi.sourceforge.net/`.

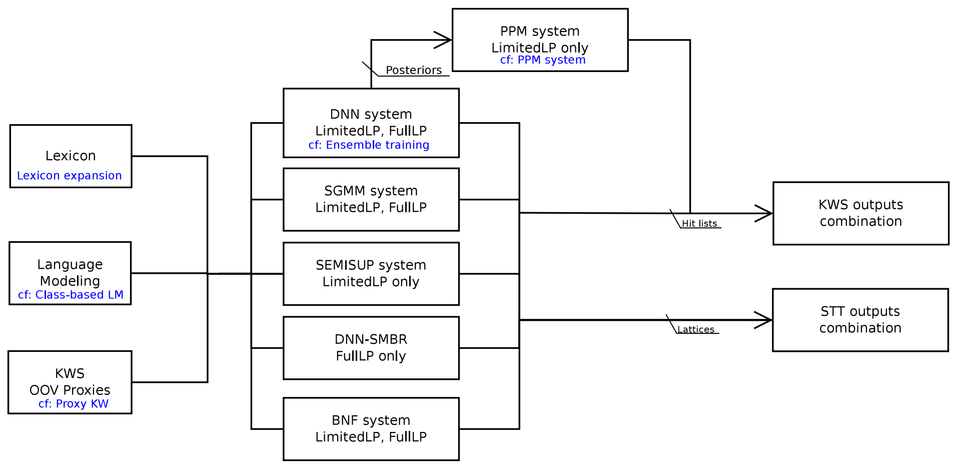

Fig. 1. Schematic diagram the JHU Kaldi systems described in Section 2 (with some novel components highlighted).

boosted MMI [2].

2. A deep neural network (**DNN**) with p-norm activation, as described in [3]. For the LimitedLP condition, an novel ensemble training method, describe below, provides improved performance.
3. A model trained on bottleneck features (**BNF**) extracted from a DNN. The 42-dim bottleneck features are used in a tandem SGMM system, again trained discriminatively via BMMI.

4a. A BNF model with semi-supervised training on 50 hours of un-transcribed speech (**BNF-SEMISUP**). The automatic transcripts were obtained using the LimitedLP SGMM and DNN models. BNF-SEMISUP was used *only* in the LimitedLP training condition.

4b. A "sequence-trained" deep neural network, trained using a state-level minimum Bayes risk (**DNN-SMBR**) criterion. Sequence training did not improve DNN performance in the LimitedLP condition. DNN-SMBR was hence used *only* in the FullLP training condition

All LVCSR systems use one of two pronunciation lexicons: the *base lexicon* for the appropriate (LimitedLP/FullLP) training condition, or an *expanded lexicon* generated as described in Section 4.1. Language models for all LVCSR systems are estimated from the appropriate *training transcripts only*, using the SRILM tools. This results in a total of $4\times2\times1 = 8$ STT decoding runs each in the LimitedLP and FullLP conditions for each language.

The Kaldi KWS pipeline is based on lattice-indexing as described in [4], and utilizes as its input the *exact lattice generation* method described in [5]. Two additional strategies are used to handle out of vocabulary (OOV) keywords:

1. One is to search for phonetically similar in-vocabulary words (i.e. *proxy keywords*) for each OOV keyword, as described in [6]. The novelty in the 2014 system beyond [6] is that due to the vastly increased vocabulary when using an expanded lexicon, proxy-based search has the opportunity to be more effective, but straightforward search becomes computationally prohibitive. Several optimizations, including lazy composition, breaking down the search into several steps, and intermediate pruning have been implemented to reduce the memory footprint and run time of the FSTs.
2. The other is a novel Poisson point process model (PPM), as described in [7]. This method bypasses most of the STT modules, using only the DNN outputs as its input, and is agnostic to the keyword being OOV.

To obtain the final submission to NIST, outputs from various individual systems are combined. For STT, lattice-level combination of 4 to 8 STT systems is performed, while for the KWS task, the ranked lists of several systems and KWS search-strategies are combined, as detailed in Section 6.

Code and scripts used for almost all results reported here are available via `svn://svn.code.sf.net/p/kaldi/code/trunk/`. Scripts under `egs/babel/s5b` were used to build the JHU Kaldi systems, and by other participants who submitted systems to NIST.

3. JHU KALDI INNOVATIONS IN 2014

A few notable innovations in the 2014 JHU Kaldi systems relative to the 2013 release (which was also made available via SourceForge under `egs/babel/s5`) are as follows.

1. All JHU Kaldi systems now use pitch and probability-of-voicing features along with PLPs. Their extraction procedure is described in [8], and their inclusion improves STT and KWS performance on all languages and in both training conditions.
2. All DNNs now use units with the novel p-norm activation function described in [3]. This results in modest but consistent (1%-2% absolute) improvement in

Language	LimitedLP WER		LimitedLP MTWV	
	normal	ensemble	normal	ensemble
Tamil	76.9%	75.7%	0.193	0.212
Assamese	65.2%	63.8%	0.224	0.241
Bengali	67.6%	66.2%	0.231	0.241
Zulu	70.1%	68.7%	0.249	0.268

Table 1. Performance of *normal versus ensemble training* of a DNN. STT and KWS results are on the development sets of four representative languages, and use the NIST keywords.

Language	FullLP WER		FullLP MTWV	
	DNN	DNN-SMBR	DNN	DNN-SMBR
Tamil	68.4%	67.4%	0.375	0.414
Assamese	53.4%	52.8%	0.421	0.456
Bengali	56.5%	56.4%	0.431	0.453
Zulu	60.3%	59.7%	0.460	0.473

Table 2. Performance of *DNN versus DNN-SMBR* (sequence) training. STT and KWS results are on the development sets of four representative languages, and use the NIST keywords.

STT performance over the previous $tanh$-based DNNs across all languages and training conditions.

3. A previously unpublished innovation in the systems of Section 2 is *ensemble training of DNNs*. We found that averaging the outputs of 4 independently trained DNNs (differing only in their random initialization but trained on the same data) results in a ca 1.5% reduction in STT errors. In order not to have to use 4 DNNs at test-time, we modified the training objective of the 4 networks, whereby the desired output for each input is a *mixture* of the original 1-of-N training target and the *averaged output of the 4 networks* for that input. By thus nudging the outputs of the 4 DNNs towards each other, we are able to use only one of them at test-time and obtain comparable gains, as demonstrated in Table 1.
4. BNF training has now been rewritten in C++, making it faster and more accurate in terms of STT errors than the previous Theano-based implementation. The basic design is unchanged: 42-dim bottleneck features are extracted form the DNN and appended to fMLLR features from a PLP-based system, three such augmented frames are spliced together and reduced to 60-dim via LDA, and an SGMM system is trained on the resulting features. The speed-up comes largely from the ability to parallelize DNN training on multiple GPUs. The consistent ca 0.5% reduction in STT errors is harder to attribute to any one cause.
5. Semi-supervised acoustic model training has now been implemented and is used for all languages in the LimitedLP condition. The implementation is similar to that of [9]. Specifically, unlabeled speech is decoded using the SGMM and DNN systems, and their outputs are combined. We use the resulting 1-best output as supervision for BNF training. Only frames whose state-level label has a posterior probability above an empirically determined threshold of 0.35 are used.
6. Another previously unpublished innovation is *automatic, syllable-based lexicon expansion*, as described in Section 4.1. Using a base lexicon with syllable boundaries marked in the pronunciations, we stochastically generate new syllable sequences, which we then treat as *pronunciations of unseen words* in that language. An orthographic form (i.e. a word-form) is generated for each such new pronunciation using a "reverse" G2P system trained on the base lexicon. About 2 million such "word"+pronunciation entries are generated for each language. The language model treats them as unseen vocabulary items for purposes of probability assignment. The impact of this massive lexicon expansion is language-dependent. e.g. it makes no difference in Assamese and Bengali, but significantly improves ATWV for Zulu in the Limited LP condition. Its impact is more pronounced if a language model with data-driven word classes is used (cf. Section 4.2).

4. LEXICON CREATION & LANGUAGE MODELING

We use the SRILM tools to build language models from the training transcripts. Several n-gram models with different smoothing methods and count cutoffs are built. The one with the lowest perplexity on the development data is retained — typically a Good-Turing 3-gram in the LimitedLP condition.

IARPA provided lexicons are used in all systems, with syllabic stress or tone converted into a "tag" attached to each phoneme in that syllable. Another tag indicates whether a phoneme is word-initial, word-final, etc. Questions concerning these tags are permitted during triphone clustering.

In addition to phonemes in the IARPA-provided lexicon[3], four special phonemes are introduced: `silence`, `noise`, `vocalized-noise` and `unknown-word`. The first two are self-explanatory. The vocalized-noise phoneme models coughs, laughter, etc. while the unknown-word phoneme models out-of-vocabulary speech, such as unintelligible words and un-transcribed foreign words, etc.

4.1. Lexicon Expansion to Enable OOV Keyword Search

We developed a novel syllable-based lexicon expansion method, which is described next. The main idea is to automatically generate millions of distinct lexicon entries whose pronunciations are phonotactically plausible in that language. An OOV (key)word in the test speech will then have a good

[3]For some languages, such as Vietnamese in the 2013 NIST evaluation and Zulu in 2014, the IARPA-provided lexicon systematically re-labels a phoneme as one of two or more variants based on context. We found in such cases that it is beneficial to collapse such variants back into a single phoneme and let the data-driven triphone clustering step decide whether multiple variants are warranted.

chance of begin decoded as a similar-sounding lexicon entry, obviating the need for a separate phonetic decoding pass or a separate subword index for OOV search. The word-lattices may be searched directly for the OOV keyword, with the proxy-based method of [6] to mitigate differences between the correct spelling (of the keyword) and the spelling generated during this automatic lexicon expansion.

We first use the IARPA lexicon to estimate an n-gram "language model" for syllable sequences that constitute words in the language; this requires a syllabified lexicon. Each pronunciation in the lexicon is treated as a "sentence" and the syllables that constitute the pronunciation are treated as atomic "words," so that the syllable inventory becomes the "vocabulary" of this "language model." Once this statistical language model has been estimated, it is used generatively to simulate new "sentences" in the language: each simulated "sentence" is the syllabic pronunciation of a potential word.

We discard syllable sequences that already exist in the IARPA lexicon, retaining only OOV syllable sequences. We also discard sequences comprised of very few phonemes. Up to 2 million of the remainder, sorted by their syllabic "language model" scores, are selected for addition to the lexicon.

The last step is to generate an orthographic form for each selected syllable sequence. For this we resort to standard G2P techniques *in reverse*: we treat each phoneme on the pronunciation-side of the lexicon as a single orthographic character (grapheme), and each orthographic character on the word-side of the lexicon as a phoneme. We train a Sequitur G2P system [10] using the IARPA lexicon in reverse, as described above. We refer to it as the P2G system to remind readers that its input is a phoneme sequence (instead of a grapheme sequence), and its output is a sequence of characters (instead of phonemes). Once trained, the P2G system accepts each selected syllable sequence, viewed as a phoneme sequence, and generates the needed orthographic form.

Since these orthographic forms are not seen in the language model training text, they are inserted in to the language model as unseen unigrams, and are assigned the unigram probability of the unseen word (times the probability of their pronunciation under the syllabic language model).

For the NIST evaluation, there were two versions of each Kaldi decoding run described in Section 2, one with the base lexicon and one with the expanded lexicon described above (cf Figure 1). On the development data, the expanded lexicons provided some improvement in ATWV for some languages (e.g. Zulu), especially when used in conjunction with the proxy-based KWS method for OOV keywords, and negligible gain for other languages. We saw no degradation from their use in any condition on the development data.

However, NIST reported a ($\approx$ 0.2% WER) degradation in STT performance for languages where we saw negligible gains, while the languages that improved on the development data continued to do so on evaluation data. We expect that the degradation may be alleviated by tuning the total language model probability assigned to the new lexical entries.

Language	Lexicon	LM	WER	ATWV
Zulu	basic	Word 3-gram	69.8%	0.26
	expanded	Word 3-gram	69.5%	0.27
	expanded	Word+Class LM	68.5%	0.32
Tamil	basic	Word 3-gram	75.7%	0.21
	basic	Word+Class LM	75.3%	0.23
	expanded	Word 3-gram	75.7%	0.20
	expanded	Word+Class LM	75.7%	0.25

Table 3. Performance of the LimitedLP DNN system with a *basic v/s expanded lexicon* and a *basic v/s class-based LM* on the respective development sets using NIST keywords.

4.2. Orthographic-Class Based Language Modeling

A shortcoming of the massive lexicon expansion of Section 4.1 is the arbitrary assignment of language model (LM) probabilities to the new words. Class-based LMs, especially those based on syntax or semantic word classes, are a good way to selectively assign different probabilities in different contexts to an otherwise indistinguishable set of unseen words. Our investigations in this direction are described next.

A major hurdle in the limited resource setting is that neither data-driven techniques (e.g. Brown clustering [11]) nor knowledge-based ones are feasible for creating word classes. Furthermore, "words" resulting from the automatic expansion are not guaranteed to be real words in the language. We therefore resort to simple, spelling-based clustering methods.

We created three such clusterings, estimated a class-based LM for each clustering, and linearly interpolated them with the baseline 3-gram LM and 2 other LMs:

1. a class-based LM, using the first three characters;
2. a class-based LM, using the first six characters;
3. a class-based LM, using the last three characters;
4. a skip bigram LM;
5. a word 3-gram LM whose absolute discounting parameters depend on the count level via a rational function.

Models 1-5 were implemented using Saarland University's LSVLM toolkit. To map the resulting LMs to ARPA format, an artificial corpus of 30 million tokens was sampled using model 5. A trigram tree was constructed and probabilities of models 1-5 where written to the leafs of that tree.

This method is still under development/evaluation, but it already seems from the preliminary results in Table 3 on two languages (Zulu and Tamil) that the interpolated class-based LM provides modest STT improvement, and somewhat more significant KWS improvement in both languages. For Tamil, the model 2 had the largest contribution for all experiments. We note that the sampling/pooling steps in converting the LSVLM to ARPA format must be performed carefully.

Finally, to obtain the results in Table 3, we only rescored lattices generated by the DNN system (cf Section 2) using

Language	ATWV		
	PPM	DNN	DNN+PPM
Assamese	0.11	0.30	0.33
Bengali	0.09	0.28	0.31
Haitian Creole	0.15	0.35	0.38
Lao	0.15	0.37	0.40
Zulu	0.14	0.28	0.33
Tamil	0.08	0.25	0.27

Table 4. Performance of *PPM-based versus word-based KWS systems* built on a LimitedLP DNN system, and KWS system combination, on the evaluation set using the NIST keywords.

the interpolated LM. Incorporating the new LM into first-pass decoding is likely to lead to further improvements.

5. POISSON POINT PROCESS MODELS FOR KWS

The point process model (PPM) for keyword search is a whole-word, event-based acoustic modeling and phonetic search technique [7, 12]. The PPM represents keywords as a set of time-inhomogeneous Poisson point processes, one process per phone. Therefore, if a PPM can be constructed for a keyword, and the speech is indexed with corresponding phonetic "events," there is no OOV problem. We use either dictionary or G2P-based pronunciations to seed the keyword PPM, and the per-frame posterior probabilities generated by our p-norm DNN to construct the phonetic event index. Indexing is approximately $2\times$faster than real-time, and the matching (search) is optimized so that it is extremely fast ($\approx$ 400,000$\times$ real time). Each detection is assigned a PPM likelihood. The outstanding issue is the normalization of this likelihood across keywords to enable the setting of a global detection threshold. The performance of PPM itself is usually on the par with other phonetic search systems but it combines really well with the word-based systems, as shown in Table 4.

6. SYSTEM COMBINATION FOR STT AND KWS

Our final submissions to NIST employ combination of several systems depicted in Figure 1 and described below.

6.1. System Combination for Speech to Text

The only system combination method used for the STT submission is the minimum Bayes risk (MBR) decoding method described in [13], which we view as a systematic way to perform confusion network combination (CNC) [14]. Note that it is nontrivial to perform MBR decoding when the vocabularies of the systems are vastly different. We therefore combine the STT outputs via MBR decoding[4] of the 4 systems that use the base lexicon (cf Section 2), and separately the 4 that use the expanded lexicon. Table 5 shows a typical, modest reduction in STT errors from system combination.

[4]A system-specific offset determined empirically is applied to the language model weight for each system during decoding.

6.2. System Combination for Keyword Search

System combination for KWS is a basic merging, for each keyword, of the ranked lists produced by the component KWS systems. Putative hits are *aligned* across systems based on proximity/overlap of time-spans, and the lattice posterior probabilities[5] of aligned putative hit are averaged across the systems. If a putative hit does not appear in a system's list, that system is assumed to have assigned it zero probability.

Specifically, if a putative hit has scores $\{s_1, s_2, \ldots, s_N\}$ in the ranked lists of N independent KWS systems, where some of the s_n's may be 0, the combined score of the hit is defined to be

$$s_{\text{avg}} = \left(\frac{1}{N} \sum_{n=1}^{N} w_n s_n^p \right)^{\frac{1}{p}}, \quad (1)$$

where p and the weights w_n are determined empirically, and are typically found to be around $p = 0.5$ and $w_n = 1$. The ranked list after KWS system combination therefore is the union of the individual ranked lists sorted by s_{avg}.

Table 5 shows typical improvements from KWS system combination for the 8 word-indexed systems described in Section 2 and further combination with the PPM system described in Section 5.

7. NIST EVALUATION RESULTS

The *primary development language submissions* of the JHU Kaldi team in both the FullLP and the LimitedLP conditions were combinations of 4 to 9 systems as described above. The primary STT system was a combination of 4 STT systems with expanded lexicons, as described in Section 6.1, while the primary KWS system was a combination of 8 word-indexed systems with the PPM system, as described in Section 6.2.

The *primary surprise language submission* was a combination of the PPM system with 5 word-indexed KWS systems, each derived from an STT system with an expanded lexicon. Two of these STT systems entailed lattice rescoring with the interpolated class-based LM (cf Section 4.2). STT system combination was not performed for the surprise language due to some computational limitations.

Table 6 reports the official NIST evaluation of the primary STT and KWS systems, demonstrating that the ambitious BABEL goal of 0.30 ATWV in the LimitedLP condition is attainable in all five development languages and in the surprise language using the JHU Kaldi tools. Performance of other systems (that also used these open source tools) that were submitted to NIST further attests to the quality of the tools.

[5]Averaging the lattice posteriors (without further normalization) was adequate when combining various Kaldi KWS systems. Combining further with non-Kaldi systems may benefit from normalizing scores within each ranked list before merging.

system	expanded lexicon		basic lexicon	
	WER	ATWV	WER	ATWV
DNN	64.2%	0.293	64.0%	0.303
PLP	65.9%	0.249	66.0%	0.243
BNF	63.4%	0.270	63.4%	0.265
BNF-SEMISUP	61.3%	0.277	61.5%	0.279
4-way combination	60.7%	0.343	60.6%	0.342
PPM	—	0.108	—	0.108
8-way combination	0.353 ATWV			
8-way + PPM	0.375 ATWV			

Table 5. Performance of *STT and KWS system combination* for Assamese on evaluation data, using NIST keywords.

Language	LimitedLP		FullLP	
	WER	ATWV	WER	ATWV
Assamese	60.6%	0.375	50.9%	0.532
Bengali	62.1%	0.355	52.8%	0.514
Haitian Creole	57.2%	0.433	48.1%	0.578
Lao	54.7%	0.437	45.0%	0.581
Zulu	67.1%	0.380	58.6%	0.484
Tamil	—	0.313	—	0.457

Table 6. Official evaluation of STT and KWS performance of the JHU Kaldi system on NIST data using NIST keywords.

8. CONCLUSION

We have described the design and implementation of state-of-the-art STT and KWS systems using the Kaldi open source tools, and outlined some innovations and capabilities we have recently added to these tools. The STT performance is on par with the best systems, and the KWS performance is respectable. We hope that this information will enable further improvement and/or fruitful deployment of the tools.

9. REFERENCES

[1] D. Povey, M. Karafiat, A. Ghoshal, and P. Schwarz, “A symmetrization of the Subspace Gaussian Mixture Model,” in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, May 2011, pp. 4504–4507.

[2] D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah, “Boosted MMI for Model and Feature-space Discriminative Training,” in *Acoustics, Speech and Signal Processing (ICASSP), 2008 IEEE International Conference on*, March 2008, pp. 4057–4060.

[3] X. Zhang, J. Trmal, D. Povey, and S. Khudanpur, “Improving Deep Neural Network Acoustic Models Using Generalized Maxout Networks,” in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, Florence, Italy, 2014.

[4] D. Can and M. Saraclar, “Lattice Indexing for Spoken Term Detection,” *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 19, no. 8, pp. 2338–2347, Nov 2011.

[5] D. Povey, M. Hannemann, G. Boulianne, L. Burget, A. Ghoshal, M. Janda, M. Karafiat, S. Kombrink, P. Motlicek, Yanmin Qian, K. Riedhammer, K. Vesely, and Ngoc Thang Vu, “Generating Exact Lattices in the WFST framework,” in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, March 2012, pp. 4213–4216.

[6] G. Chen, O. Yilmaz, J. Trmal, D. Povey, and S. Khudanpur, “Using Proxies for OOV Leywords in the Keyword Search Task,” in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, Dec 2013, pp. 416–421.

[7] A. Jansen and P. Niyogi, “Point Process Models for Spotting Keywords in Continuous Speech,” *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 17, no. 8, pp. 1457–1470, Nov 2009.

[8] P. Ghahremani, B. BabaAli, D. Povey, K. Riedhammer, J. Trmal, and S. Khudanpur, “A Pitch Extraction Algorithm Tuned for Automatic Speech Recognition,” *Proceeding of Int. Conf. ICASSP 2014*, 2014.

[9] K. Veselý, M. Hannemann, and L. Burget, “Semi-supervised Training of Deep Neural Networks,” in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*. 2013, pp. 267–272, IEEE Signal Processing Society.

[10] M. Bisani and H. Ney, “Joint-sequence Models for Grapheme-to-phoneme Conversion,” *Speech Communication*, vol. 50, no. 5, pp. 434 – 451, 2008.

[11] P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. Della Pietra, and J. C. Lai, “Class-based N-gram Models of Natural Language,” *Comput. Linguist.*, vol. 18, no. 4, pp. 467–479, Dec. 1992.

[12] K. Kintzley, A. Jansen, and H. Hermansky, “Featherweight Phonetic Keyword Search for Conversational Speech,” in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, 2014.

[13] H. Xu, D. Povey, L. Mangu, and J. Zhu, “Minimum Bayes Risk Decoding and System Combination Based on a Recursion for Edit Distance,” *Computer Speech & Language*, vol. 25, no. 4, pp. 802 – 828, 2011.

[14] G. Evermann and P. C. Woodland, “Posterior Probability Decoding, Confidence Estimation and System Combination,” in *Proc. Speech Transcription Workshop*. Baltimore, 2000, vol. 27.