# THE TAO OF ATWV: PROBING THE MYSTERIES OF KEYWORD SEARCH PERFORMANCE

*Steven Wegmann, Arlo Faria, Adam Janin, Korbinian Riedhammer, Nelson Morgan*

International Computer Science Institute, Berkeley, CA, USA

## ABSTRACT

In this paper we apply diagnostic analysis to gain a deeper understanding of the performance of the the keyword search system that we have developed for conversational telephone speech in the IARPA Babel program. We summarize the Babel task, its primary performance metric, "actual term weighted value" (ATWV), and our recognition and keyword search systems. Our analysis uses two new oracle ATWV measures, a bootstrap-based ATWV confidence interval, and includes a study of the underpinnings of the large ATWV gains due to system combination. This analysis quantifies the potential ATWV gains from improving the number of true hits and the overall quality of the detection scores in our system's posting lists. It also shows that system combination improves our systems' ATWV via a small increase in the number of true hits in the posting lists.

***Index Terms—*** keyword search, spoken term detection

## 1. INTRODUCTION

In this paper we will describe the strengths and weaknesses of the keyword search (KWS) system that we have developed for conversational telephone speech in the IARPA Babel program [1]. The goal of the IARPA Babel program [1] "is to rapidly develop speech recognition capability for keyword search in a previously unstudied language, working with speech recorded in a variety of conditions with limited amounts of transcription." Our approach to solving this problem has been to develop a relatively straightforward recognition system that requires moderate computational resources. This simplicity also facilitates close integration with our university partners (Columbia, OSU, Northwestern, and UW) and diagnostic analysis both of which drive innovation and future improvements.

Supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense US Army Research Laboratory contract number W911NF-12-C-0014. 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.

| Code | Name | Release Version |
|---|---|---|
| 101 | Cantonese | babel101b-v0.4c |
| 104 | Pashto | babel104b-v0.4bY |
| 105 | Turkish | babel105b-v0.4 |
| 106 | Tagalog | babel106b-v0.2g |
| 107 | Vietnamese | babel107b-v0.7 |

**Table 1**. The Babel languages, identifying codes, and release versions.

The first section of this paper covers background: Section 2.1 describes the salient components of the Babel task (including the performance metric ATWV in Section 2.1.2), while Section 2.1 and Section 2.3 describe our recognition and KWS systems respectively. The heart of the paper is Section 3 which describes the novel analysis that we have undertaken to gain a deeper understanding how our KWS is performing and where the largest potential improvements are. This analysis includes constructing two new oracle measures in Section 3.1 and a bootstrap-based confidence interval for ATWV in Section 3.2. System combination appears to give much larger gains in ATWV than in the underlying speech recognition accuracy. In Section 3.3 we investigate where these gains are coming from with the larger goal of realizing these gains from a simpler, single system. We wrap-up with a discussion in Section 4.

## 2. BACKGROUND

### 2.1. A summary of the Babel task

In this section we describe the main features of the Babel's performers task. We will concentrate on the features of the task that are applicable outside the Babel program and avoid describing many of the technical aspects of the evaluation protocol.

#### *2.1.1. The data*

The audio data in each language is conversational telephone speech recorded in a variety of environments and handset types. There were 5 languages that program participants

worked with: Cantonese, Pashto, Turkish, Tagalog, and Vietnamese. Each language comes with 80 hours of transcribed training data (FullLP), a pronunciation lexicon that covers the words in the training data, and a 10 hour development test set. For technical reasons the amount of Vietnamese evaluation data is 75 hours, while the rest of the languages have 15 hours. All of the results in this paper report KWS results using the keywords (KW) provided for the evaluation. Also, when we report results on the evaluation data we will be restricting our results to the subsets, called "eval-part1", where the ground truth was released to participants: a 15 hour subset in the case of Vietnamese and 5 hour subsets for the rest of the languages. Finally, Table 1 gives the Babel codes (and data release versions) for the languages, which we will use in tables and graphs.

#### *2.1.2. The KWS performance metric: ATWV*

In this section we describe ATWV, which was developed for the NIST 2006 Spoken Term Detection evaluation [2] and is the primary metric for the Babel program: participants' ATWV must be above 0.30. To score a posting list (a list of putative hits with start, end times, and system detection scores) for a given KW, $kw$, and detection threshold, $\theta$, entries in the list are matched to reference occurrences using an objective function that accounts for both temporal overlap between the reference and posting list occurrences and the detection scores assigned by the system. The probabilities of miss and false alarm errors at detection threshold $\theta$ are computed using

$$P_{\text{Miss}}(kw,\theta) = 1 - N_{\text{Correct}}(kw,\theta)/N_{\text{Ref}}(kw)$$

$$P_{\text{FA}}(kw,\theta) = N_{\text{Spurious}}(kw,\theta)/N_{\text{NT}}(kw)$$

where $N_{\text{Correct}}(kw,\theta)$ is the number of correctly hypothesized posting list entries with detection scores $\geq \theta$, $N_{\text{Spurious}}(kw,\theta)$ is the number of incorrectly hypothesized posting list entries with detection scores $\geq \theta$, $N_{\text{Ref}}(kw)$ is the number of reference occurrences, and $N_{\text{NT}}(kw)$ is the number of non-target trials for $kw$ in the data. The number of non-target trials for a term is related to the total audio stream duration in seconds, $T_{\text{Audio}}$, via

$$N_{\text{NT}}(kw) = T_{\text{Audio}} - N_{\text{Ref}}(kw).$$

For the 2006 STD evaluation NIST defined a single number performance metric called "term weighted value" (TWV) that specifies the trade-off between misses and false alarms. Given the predefined constant $\beta = 999.9$, it is convenient to first define the KW-specific TWV as

$$\text{TWV}(kw,\theta) \equiv 1 - P_{\text{Miss}}(kw,\theta) - \beta P_{\text{FA}}(kw,\theta),$$

then, assuming that there are $K$ KWs and they live in $\mathcal{K}$, the TWV to be the average of the KW-specific TWVs:

$$\text{TWV}(\theta) \equiv \sum_{kw\in\mathcal{K}} \text{TWV}(kw,\theta)/K.$$

Ideally, the system will choose a detection threshold, $\hat{\theta}$, that attains TWV's maximum value. NIST defined *actual term weighted value* (ATWV) to be the value of TWV at the system's chosen detection threshold, $\hat{\theta}$, i.e., ATWV = TWV($\hat{\theta}$).

We wrap up this section with several observations concerning KW-specific TWV (which carry over to TWV and ATWV.) First note that for every KW $kw$ the range of TWV$(kw,\theta)$ is $(-\infty, 1]$. More specifically, TWV$(kw,\theta)$ attains the value 1 if and only if there are no false alarms and no misses; if the posting list is empty, then it takes the value 0; large numbers of false alarms will drive TWV$(kw,\theta)$ towards $-\infty$. Second note that in TWV$(kw,\theta)$ the cost of a false alarm is effectively constant across KWs, $\approx 1/T_{\text{Audio}}$, since in practice $T_{\text{Audio}} >> N_{\text{Ref}}(kw)$, while the cost of a miss is variable and depends on the number of true occurrences of KWs, $= 1/N_{\text{Ref}}(kw)$.

### 2.2. The recognition system

The Kaldi speech recognition toolkit [3], along with the TNet[1] toolkit, were used for recognition and lattice generation. This section follows in most parts [4], but is updated and extended to document recent changes.

We use 13 MFCCs as primary features after cepstral mean subtraction. We extract pitch and probability-of-voicing (PoV) features using a sub-band autocorrelation classification, SAcC [5]. These two features are smoothed, interpolated, and then pasted with the cepstral features to form a 15-dimensional feature vector. While we use $\Delta$ and $\Delta\Delta$s for early systems in the initialization, we use a variant of HLDA features for the final systems. The LDA transformation takes as input a context of 7 spliced static MFCC vectors and is trained using the context dependent states as targets; we project the features down to 30 dimensions. During training, this LDA matrix is composed with global MLLT matrices as well as speaker dependent fMLLR matrices [6].

We also use 30-dimensional tandem bottleneck (BN) features [7] that are obtained using a hierarchical NN [8]. See [4] for more details. For the tandem features, we paste combinations of cepstral, pitch and bottleneck features together to form the tandem feature vector. Note that HLDA is applied to the cepstral part of the features only, while MLLT and fMLLR are applied to the combined feature stream.

The first two decoding passes are done with a standard continuous acoustic model where the emission probabilities of the context dependent states are derived from GMMs associated with each of the (clustered) states. We initialize the mixtures with a single component each, and subsequently allocate more components by splitting components at every training iteration until a pre-determined total number of components is reached. This model has 5k context dependent states and 80k Gaussian components. The third and final

[1] http://speech.fit.vutbr.cz/software/neural-network-trainer-tnet

decoding pass is done with subspace Gaussian mixture models (SGMM) [9]. Here, we use the SGMM2 variant, which extends the conventional SGMMs by two ideas. First, by a "symmetrization" which makes the speaker subspace and phonetic subspace behave the same way [10]. And second, an idea similar to state-clustered tied mixtures that involves sharing Gaussians among fairly small sets of context-dependent states, but applied at the SGMM sub-state level rather than the Gaussian level [11]. This model has 8k states and 50k sub-states derived from 700 Gaussians.

We estimate a 3-gram language model on the training transcripts, and apply Kneser-Ney smoothing and interpolated counts [12, 13].

### 2.3. The KWS system

Our KWS system consists of three steps: i) converting recognition lattices to indexes, ii) searching the indexes for a given KW and constructing a posting list, iii) and setting the KW-specific detection threshold in order to optimize ATWV. These are further described below. Note that the KWS systems described here are entirely word-based, i.e., we are not combining the word-based search with a separate subword-based search in order to handle out-of-vocabulary (OOV) KWs, e.g., as in [14, 15].

i. We use "lattice-tool" from SRILM [12] to convert lattices to word-level indexes. We set the lattice-tool parameter that controls how far apart in time two occurrences of a word have to be in the lattice before we consider them separate entries in the index to be 0.1 sec. We use Nelder-Mead optimization (the downhill simplex algorithm) on the development set to determine the relative weights of the acoustic and language model scores in the conversion of these scores into "lattice posterior probabilities."

ii. For single word KWs the posting list is simply all of the occurrences of the KW sorted by their posterior probabilities. To construct the posting list for a multi-word KW we follow [14]: the individual words are first retrieved from the index in the correct order with respect to their start and end times, but occurrences are discarded when the time gap between adjacent words is more than 0.5 seconds. The surviving occurrences are assigned a detection probability equal to the minimum of the individual word probabilities.

iii. To determine the detection threshold for a given KW we use the distribution of the KWs posting list probabilities to construct an approximation to the true TWV function for that KW. The detection threshold is chosen to maximize this approximation to TWV.

## 3. ANALYSIS OF KWS PERFORMANCE

In this section we will describe the results of three types of analysis that we have used to gain a deeper understanding of how our KWS is performing and where the largest potential improvements are. We compare the performance of our thresholding algorithm to two useful oracle measures in Section 3.1, apply a bootstrap-based confidence interval for ATWV in Section 3.2, and study the effects of system combination in Section 3.3.

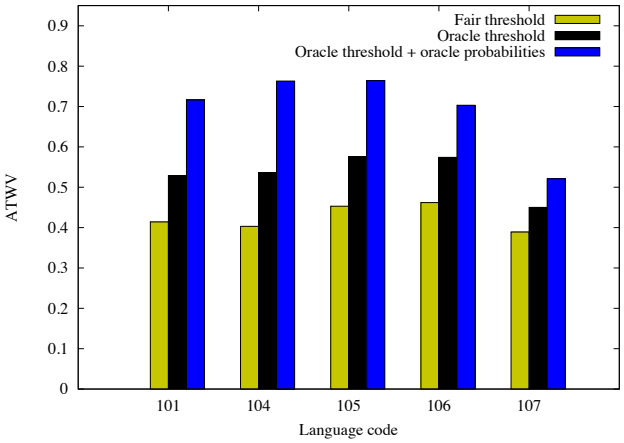


**Fig. 1**. Comparison of KWS performance on development data using fair thresholds to two optimal thresholds (using the language codes in Table 1.)

### 3.1. Two oracle measures

We compare the performance of our thresholding algorithm to two "oracle" measures that use knowledge of ground truth in different ways to set optimal thresholds. The first oracle measure [16] chooses the KW-specific detection thresholds that actually maximizes TWV for every KW. Comparison of our actual performance to this oracle measure gives us an upper bound on how much gain in ATWV we could realize through improvements to our thresholding algorithm.

The second oracle measure first modifies the detection probabilities on the posting lists by setting the probabilities of all of the true hits to 1 and all of the false alarms to 0. Then the KW-specific detection thresholds are chosen just like the first oracle measure, namely to actually maximize the KW-specific TWVs. However, any $\theta \in (0, 1]$ (e.g., $\theta = 0.5$) is optimal due to the way the detection probabilities have been set. Comparison of our actual performance and the previous oracle measure to this measure gives us two types of insight: first, an upper bound on the gain in ATWV that we could realize through improvements to the detection probabilities; and second, an indication of how many true hits make it on to the posting lists since the resulting oracle ATWV is the average fraction of true hits that actually occur in the KW-specific posting lists.

Fig. 1 shows the results of these oracle measures. Note that there are large gaps between performance of the fair and

the two oracle measures across all of the languages. This means that there is opportunity for substantial gains in ATWV via improving the thresholding algorithm. Since the gaps between the performance of the two optimal thresholds appears to be even larger, this suggests that improving the ordering in our posting lists via more accurate detection probabilities may be more important. Note that for all of the languages, except for 107 (Vietnamese), on average 70-76% of the true hits make it onto the posting lists; in Vietnamese only 52% of the true hits make it onto the posting lists. This anomaly is a large source of potential improvement that we are currently studying.

### 3.2. Confidence intervals for ATWV

ATWV can be an extremely unstable performance measure, e.g., small changes in underlying recognition performance may result in large changes to ATWV. This led us to use Efron's bootstrap [17] to estimate confidence intervals for ATWV performance on all of the languages' development data which we applied to several statistical inference problems. First, during system development we used our estimates of ATWV's standard error to assess the significance levels of potential improvements. Second, we used this prior to the evaluation to get an estimate of the range of expected performance on the evaluation data given our results on the development data. Third, we used this post-evaluation as a diagnostic tool by focusing our attention to languages where our actual performance on the evaluation data was worse than expected.

To help understand how we use the bootstrap to obtain a confidence interval for ATWV on a given language, imagine that we have 100 (instead of just a single) 10 hour development test sets. In this imaginary and profligate world we would simply run recognition, indexing, and KWS to get posting lists with detection thresholds on these 100 test sets. Scoring with the corresponding ground truths would result in 100 corresponding ATWVs, from which we would compute the mean and standard error. Providing that these 100 test sets were random, independent draws from the larger population of test sets, this confidence interval would be a good estimate of the range of ATWV on future test sets.

In reality, we have only one collection of KW-specific posting lists for each language (obtained by running recognition, indexing, and KWS on a single development test set). However, the bootstrap enables us to construct 100 KW-specific posting lists, ground truths, and resulting ATWVs from the single collection of KW-specific posting lists and ground truth via simulation using a process called "resampling". Once we have these 100 ATWVs we use them, just as in our imaginary experiment, to estimate the mean and standard error of ATWVs computed from the larger population of potential test sets.

To apply the bootstrap we use each posting list to construct the empirical distribution–a non-parametric approximation to the true population distribution of the KW's posting lists–and we simulate from this empirical distribution to construct pseudo posting lists that approximate lists drawn from the true population distribution (as in our imaginary experiment.) In this context, simulation consists of a random draw with replacement from the posting list. To actually do this, we first use the ground truth to label each entry in the posting lists with whether or not it is a true hit. We also use the ground truth to augment the posting lists with true occurrences of the KWs that didn't make it into the posting lists (each of these extra entries are given detection probability $= 0$ to distinguish them from the original entries that all have detection probabilities $> 0$.) After removing the times, the result is a list of all of the true occurrences of the KWs together with all of our hypothesized occurrences with their detection probabilities and correctness labels. We resample from this list the requisite number of times to get a pseudo posting list with the same length as the original list.

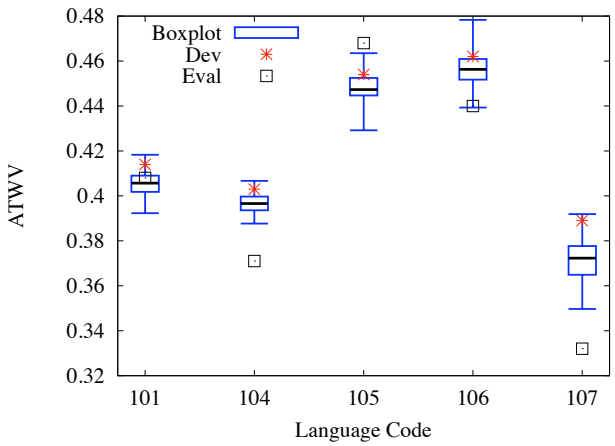


**Fig. 2**. Boxplots of the bootstrap replicates' ATWVs together with development and eval-part1 ATWVs (using the language codes in Table 1.)

Fig. 2 displays boxplots (min, 1st quartile, median, 3rd quartile, max) of the 100 bootstrap replicates' ATWV together with the ATWVs of the real development data and eval-part1 for each of the languages. Note that only the Cantonese (101) eval-part1 performed as expected. This means that for all of the other languages there were properties of eval-part1 that were markedly different from the development data. Here is a brief, preliminary analysis for Tagalog (106) and Vietnamese (107).

On the Tagalog development data none of the occurring KWs were OOV with respect to the recognition lexicon, but on eval-part1 $\approx 10\%$ of the KWs are OOV and all of these KW-specific TWVs are 0; on the $90\%$ of KWs that were invocabulary the ATWV was 0.477 which is in our bootstrap estimated range of performance; hence unexpected OOVs ac-

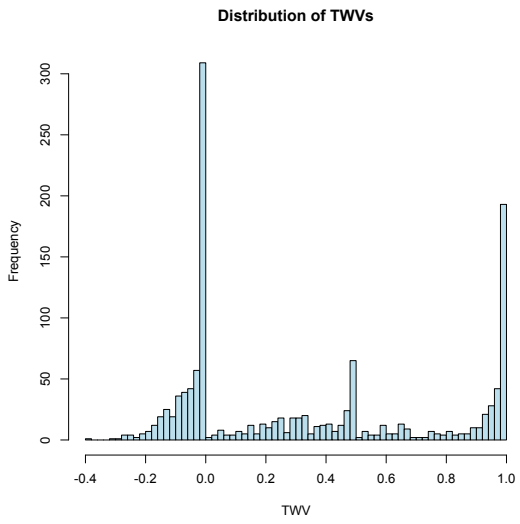


**Fig. 3**. The distribution of Vietnamese KW-specific TWVs.

count for the difference in performance.

For Vietnamese there were also no OOV KWs in the development data and only $1\%$ of the KWs were OOV on eval-part1, so OOVs do not account for the bulk of the difference in performance. Of the 1309 KWs that occur in eval-part1, 418 also occur in the development data and these KWs have very similar ATWV performance on these two sets $\approx 0.36$; this is also in the bootstrap estimated range of performance. In addition $57\%$ of the true hits made onto these KWs' posting lists in both the development data and eval-part1. However, ATWV on the 891 remaining KWs that do not occur in the development data is much lower $0.331$ and outside the estimated performance range. Only $41\%$ of the true hits made it onto the posting lists for these KWs, which strongly suggests that recognition performance was much lower around these KW occurrences. The inherent variability of TWVs may also be the culprit. Fig. 3 displays a histogram of the KW-specific TWVs on the development data. Generally speaking KW-specific TWVs have a broad, multi-modal distribution, and Vietnamese is no exception to this. Note that the mean of this distribution doesn't "mean" very much!

### 3.3. System combination

In this section we shed some light on why system combination can give large gains in ATWV. To merge two systems' KWS results, for each KW we first take the union of the two posting lists without modifying the probabilities, we merge any overlapping entries taking the max of overlapping probabilities, and then compute the KW-specific threshold.

We combined Vietnamese KWS results from an HTK-based system with the Kaldi baseline system described in Section 2.2 and two additional Kaldi-based systems. The HTK-based system [18] is an analog of the continuous baseline Kaldi system (i.e., it does not use SGMMs). The two additional Kaldi-based systems differ only from the baseline in the front-end feature sets that they use. The first variant's feature set uses only the bottleneck features and is labeled "BNO" in Table 2. The second variant, labeled "PLP" in Table 2, has the baseline's MFCC features replaced with PLP [19] features. The second column in Table 2 shows the stand alone performance of the four systems, which are quite similar, while the third column shows the performance gain due to merging search results: each row gives the result of merging all of the systems listed above and including that row. The largest ATWV gain comes from merging the HTK and Kaldi baseline systems ($0.389 \rightarrow 0.425$ or 0.036 absolute), but subsequent merges give further absolute ATWV gains (0.013).

| System | ATWV | Merge ATWV | Unhyped Misses |
|---|---|---|---|
| Kaldi | 0.389 | NA | 1145 |
| HTK | 0.382 | 0.425 | 959 |
| BNO | 0.369 | 0.438 | 866 |
| PLP | 0.396 | 0.451 | 818 |

**Table 2**. Vietnamese ATWV before and after merging.

We will examine the following possible explanations for these gains: 1) Merging increases the number of entries on the posting lists which results in better thresholds. 2) The merging process improves the detection probabilities in the posting lists. 3) Merging cuts down on single system misses.

Misses happen in precisely two ways, namely, when a true hit is on the posting list but its detection probability is lower than the system threshold or when a true hit doesn't make it onto the posting list. We call the latter an "unhyped miss". The fourth column of Table 2 shows that the number of unhyped misses definitely decreases as we merge search results, but that doesn't necessarily mean that we are converting these new hits to detections.

We will look more carefully at the differences between the posting lists produced by the Kaldi baseline and the first merge between the baseline HTK and Kaldi systems. From Table 2 we see that the difference in ATWV between these two systems is 0.036. For each KW's posting list we examine the difference in the KW-specific TWV, the difference in the number of entries in the posting list, which we will call the "hypDiff", and the difference in the number of true hits in the posting list, which we will call the "hitDiff" (hitDiff is independent of whether or not the true hits are correctly detected, i.e., above threshold.) Recall that only 899 KWs occur in the Vietnamese development set. A quarter of these have hypDiff = 0 and the change in ATWV due to merging is 0. Another quarter of the KWs have hypDiff = 1 or 2 and these KWs account for nearly all of the improvement in ATWV: 0.032. Restricting our attention to to the posting lists that have hypDiff = 1 or 2 we find that the difference in ATWV from KWs that

have hitDiff = 0 is 0, but that the difference from KWs that have hitDiff = 1 is 0.032. Thus, in this case nearly all of the improvement due to merging comes from adding just 1 true hit in 1 or 2 extra entries to the posting lists. The improvement is not due to better detection probabilities or thresholding.

## 4. DISCUSSION

To gain a deeper sense of how our KWS system is performing we have used three complimentary forms of diagnostic analysis across the 5 Babel languages. First, we compared our thresholding algorithm's ATWV to two oracle ATWVs that use knowledge of TWV to set optimal thresholds without and with knowledge of the correctness of the posting list entries. Next, we used bootstrap-based estimates of ATWV confidence intervals to compare expected versus actual evaluation performance. Finally, we analyzed the cause of the large gain in ATWV that results from merging two KWS results that perform similarly but arise from different recognition systems. While this may seem obvious in hindsight, this analysis has shown that there are two key opportunities for improving our KWS system: 1) getting more true hits into the posting lists; and 2) getting more accurate detection probabilities for the existing entries in the lists. There is also a smaller opportunity available from potential improvements to the detection threshold algorithm.

Are there ways to make progress in 1) without resorting to brute force methods that result in huge expenditures in CPU cycles and storage, e.g., system combination and gigantic lattices? This was partly the motivation of our system combination analysis and it is an active area in our research. Another active area in our research, which addresses 2), is a novel application of machine learning to compute the posting list probabilities.

## 5. REFERENCES


[1] "IARPA Babel Program - Broad Agency Announcement (BAA)," `http://www.iarpa.gov/Programs/ia/Babel/solicitation_babel.html`, 2011.

[2] J. G. Fiscus, J. Ajot, J. S. Garofolo, and G. Doddingtion, "Results of the 2006 spoken term detection evaluation," in *Proceedings of ACM SIGIR Workshop on Searching Spontaneous Conversational Speech*, 2007, pp. 51–55.

[3] D. Povey, A. Ghoshal, et al., "The Kaldi Speech Recognition Toolkit," in *Proc. ASRU*, 2011.

[4] K. Riedhammer, Van Hai Do, and J. Hieronymus, "A study on LVCSR and keyword search for tagalog," in *Proc. Interspeech*, 2013.

[5] B.S. Lee and D. Ellis, "Noise robust pitch tracking using subband autocorrelation classification (SAcC)," in *Proc. Interspeech*, 2012.

[6] M. J. F. Gales, "Maximum likelihood linear transformations for hmm-based speech recognition," *Computer Speech and Language*, vol. 12, pp. 75–98, 1998.

[7] F. Grezl, M. Karafiat, S. Kontar, and J. Cernocky, "Probabilistic and bottleneck features for lvcsr of meetings," in *Proc. ICASSP*, 2007, pp. 757–760.

[8] F. Valente, J. Vepa, C. Plahl, C. Gollan, H. Hermansky, and R. Schlüter, "Hierarchical neural networks feature extraction for lvcsr system," in *Proc. Interspeech*, 2007, pp. 42–45.

[9] D. Povey, L. Burget, M. Agarwal, P. Akyazi, F. Kai, A. Ghoshal, O. Glembek, N. Goel, M. Karafiát, A. Rastrow, R. Rose, P. Schwarz, and S. Thomas, "The subspace Gaussian mixture model - A structured model for speech recognition," *Computer Speech and Language*, vol. 25, pp. 404–439, 2011.

[10] D. Povey, M. Karafiát, A. Ghoshal, and P. Schwarz, "A symmetrization of the subspace gaussian mixture model," in *Proc. ICASSP*, 2011, pp. 4504–4507.

[11] K. Riedhammer, T. Bocklet, A. Ghoshal, and D. Povey, "Revisiting Semi-Continuous Hidden Markov Models," in *Proc. ICASSP*, 2012, pp. 4721–4724.

[12] A. Stolcke, "SRILM – an extensible language modeling toolkit," in *Proc. ICSLP*, 2002, pp. 901–904.

[13] R. Kneser and H. Ney, "Improved backing-off for n-gram language modeling," in *Proc. ICASSP*, 1995, pp. 181–184.

[14] D. R. H. Miller, M. Kleber, C.-L. Kao, O. Kimball, T. Colthurst, S. A. Lowe, R. M. Schwartz, and H. Gish, "Rapid and accurate spoken term detection,," in *Proc. Interspeech*, 2007, pp. 314–317.

[15] B. Kingsbury, J. Cui, X. Cui, M. J. F. Gales, K. Knill, J. Mamou, L. Mangu, D. Nolan, M. Picheny, B. Ramabhadran, R. Schluter, A. Sethy, and P. C. Woodland, "A high-performance Cantonese keyword search system," in *Proc. ICASSP*, 2013, pp. 8277–8281.

[16] D. Vergyri, I. Shafran, A. Stolcke, R.R. Gadde, M. Akbacak, B. Roark, and W. Wang, "The SRI/OGI 2006 spoken term detection system," in *Proc. Interspeech*, 2007, pp. 2393–2396.

[17] B. Efron, "Bootstrap methods: another look at the jackknife," *Annals of Statistics*, vol. 7, no. 1, pp. 1–26, 1979.

[18] S. Young, G. Evermann, M. Gales, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland, *The HTK book version 3.4*, 2006.

[19] H. Hermansky, "Perceptual linear predictive (PLP) analysis of speech," *JASA*, vol. 87, pp. 1738–1752, 1990.