interspeech 2008 incorporating SST 08

Applications of Virtual-Evidence based Speech Recognizer Training

Amarnag Subramanya and Jeff Bilmes

SSLI Lab, Dept. of Electical Engineering, University of Washington, Seattle, USA.

asubram@u.washington.edu, bilmes@ee.washington.edu

Abstract

We present two applications of our previously proposed virtual-evidence (VE) based speech recognizer training algorithm [1, 2]. The first relates to two-pass training where segmentations obtained during the first pass are used as VE to train the subsequent pass. We use the TIMIT phone and SVitchboard continuous speech recognition tasks to demonstrate the benefits of using VE based training in two-pass systems. The second application involves making use of functions that can incorporate prior domain knowledge to generate VE-scores. Here, in the case of TIMIT phone recognition, we show that using the proposed function to generate VE-scores results in about 6% relative error rate reduction over the baseline.

Index Terms: Dynamic Bayesian Networks, Speech Recognition, Virtual-Evidence.

10.21437/Interspeech.2008-635

1. Introduction

Most state-of-the-art speech recognizers are trained using sequence-labeled (SL) data, i.e., a training set in which only the sequence of words is revealed [3]. In [1, 2], we proposed a new technique which involves annotating only a *small* region for every word/phone (depending on whether the task is word or phone recognition) in every utterance in the training set (in most cases the region is a single unit of time). This approach, henceforth referred to as partially-labeled (PL), is different from the two classical ways of annotating speech data, namely, (a) the above mentioned SL case and (b) the fully-labeled (FL) approach where the exact start and end times of every word/phone is known. Note that training using FL data does not mean "Viterbi Training" since under FL the within-word (or within-phone) segmentations are unknown.

Training with PL-data involves using the notion of *virtual-evidence* (VE) in dynamic Bayesian Networks (DBN) [6]. The VE framework is a convenient way of expressing uncertainty associated with the evidence. Our work in [1] and [2] showed the benefits of training using PL-data for phone recognition and large vocabulary continuous speech recognition (LVCSR) respectively. The advantages of training using PL-data can be summarized as follows – (a) it is easier to annotate and less error-prone compared to FL-data, (b) requires smaller amounts of memory and compute in comparison to training using SL-data, and most importantly, (c) leads to much improved performance over both FL-and-SL cases. Clearly, given FL-data, one can generate PL-data and subsequently SL-data. In both [1, 2], PL-data was generated by taking FL-data and dropping labels around transition points of words/phones (section 4). Given an utterance, sequence annotation takes least amount of time, while FL is the most expensive and PL lies between SL and FL [2].

This work was supported by an ONR MURI grant, No. N000140510388 and a Microsoft Research Fellowship.

In this paper we consider two extensions to training using PL-data. The first relates to the idea of *multi-pass training*. In many cases, speech recognizers are trained in multiple passes with segmentations from an initial pass used to train the subsequent pass [4]. In particular, we consider the case of two-pass training. We show how the VE framework may be used to improve the benefits of two-pass training for the TIMIT and SVitchboard (SVB) [11] tasks. The other idea is related to making use of a certain parameterized class of functions to generate *VE-scores* which encode the uncertainty associated with the evidence. Previous work [1, 2] used relatively simple ways of generating the VE-scores. In this paper, we consider a more powerful approach that incorporates prior domain knowledge. We show that using these functions to generate the VE-scores can provide further improvements over the performance of PL-systems in [1, 2].

2. Partially-Labeled Data

We briefly describe our approach to partially-labeling speech training data for word recognition. Note that the same arguments hold in the case of data annotation for phone recognition. Consider the utterance in figure 1 where t_1, t_4, t_7, t_{10} are the start times of words "what", "was", "the" and "other", while the end times are t_4, t_7, t_{10} and t_{13} respectively [1]. In the FL case, we have access to the start and end times of each of the words (in addition to the word identities). In practice, it is difficult (in most cases impossible) to label the *exact* start and end times of the words due to a number of inherent properties of human speech such as co-articulation [2]. In PL-data, only a small region of every word is annotated. For example, in the case of the word "was" in figure 1, the interval $[t_5, t_6]$ is annotated while $[t_8, t_9]$ is annotated as belonging to the word "the" and the interval $[t_6, t_8]$ is left unlabeled. Given the above PL-data, it is clear that in the interval $[t_6, t_8]$, the speaker uttered either "was" or "the". In other words, for every frame (see section 4) in the interval $[t_6, t_8]$, we have a distribution [2] with some non zero mass on words "the" and "was", while all other words get zero mass. In the labeled regions, all the probability mass lies on the labeled word (i.e. a delta distribution). Clearly FL-data is a special case of PL-data. The most interesting case of PL-data occurs when only a single instant is annotated for every word in the utterance. Note that PL-data does not suffer from the same drawbacks as FL-data as we are not insisting that the entire word be labeled. For more information see [1, 2].

We would like to point out that our PL approach is different from the setting where one trains a recognizer using a set of utterances that are completely labeled (usually in SL format)

[1] In this paper, time t is discrete. However, for brevity of the exposition here we assume it to be continuous.

[2] Does not have to be a probability measure, any unsigned measure will do.

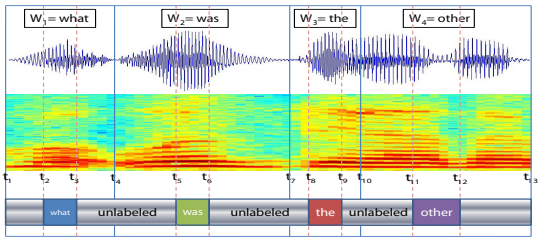

Figure 1: Partially-Labeled (PL) Data

augmented by another set of utterances that are completely unlabeled [5]. While the general VE-based training framework may be extended to such situations, it is not the focus of our work here. Next we describe how PL-data may be used as VE to train a recognizer.

2.1. Virtual-Evidence

Consider a DBN over n random variables (rv) $\mathcal{X} = \{X_1, \ldots, X_n\}$. Given evidence $X_1 = \bar{x}_1$, the joint distribution reduces to $p(\bar{x}_1, \ldots, x_n)$ (no longer a function of x_1). The same behavior can be simulated by introducing a rv $\mathcal{V} \notin \mathcal{X}$ and setting $p(\mathcal{V} = 1|X_1) = \delta(X_1 = \bar{x}_1)$ [3] . It can be shown that $p(\mathcal{V} = 1, x_1, \ldots, x_n) = p(\bar{x}_1, \ldots, x_n)$. This is commonly referred to as *specific evidence* and is the most common way of injecting evidence into a model. The notion of VE arises when evidence suggests multiple possible assignments to a particular rv. In the above example the evidence might suggest, by means of scores, that two values of X_1 are possible. Here the scores are some measure of the likelihood of each value. We would like to integrate such evidence into the rest of the model to make inferences. One way of achieving this is by setting $p(\mathcal{V} = 1|X_1) \propto f(x_1)$. Here $f(.)$ is the function that generates the scores. In relation to the example in figure 1, if $\mathcal{W}_t$ is the rv representing the spoken word at time t, for $t \in [t_6, t_8]$, we set $p(\mathcal{V}_t = 1|\mathcal{W}_t =$ "was"$) \propto h(t)$, $p(\mathcal{V}_t = 1|\mathcal{W}_t =$ "the"$) \propto l(t)$, $h(t), l(t) \geq 0$. Here $h(t)$ and $l(t)$ represent our confidence in whether $\mathcal{W}_t =$ "was" or $\mathcal{W}_t =$ "the". We only need to define the above conditionals up to a constant of proportionality as only the ratio of VE-scores affects the posteriors [6]. Note that $p(\mathcal{V} = 1|\mathcal{W}_t = w_1) > p(\mathcal{V} = 1|\mathcal{W}_t = w_2)$ does not imply $p(\mathcal{W}_t = w_1|\text{evidence}) > p(\mathcal{W}_t = w_2|\text{evidence})$ (where $p(\mathcal{W}_t|\text{evidence})$ is the posterior).

There are a number of ways of setting $h(t)$ and $l(t)$. The simplest approach is $h(t) = l(t) = \gamma > 0$, where γ is some constant (the exact value of γ does not matter since only ratios count). We refer to this as *UNIF-PL-data*. This encodes the fact that in the unlabeled regions we are uniformly weighing the words on either side. UNIF-PL-data has been shown to work for both TIMIT phone recognition [1] and LVCSR [2]. Yet another setting could take into account the fact that for times closer to t_6, one is more confident that $\mathcal{W}_t =$ "was", while for times closer to t_8, we have a higher weight on $\mathcal{W}_t =$ "the" [1]. One of the goals of this paper is to explore other methods of generating VE-scores. In particular, we propose a family of parametric functions that can be used to generate a variety of VE-scores. For more information on VE see [6, 7].

3. Generating VE-scores

Consider a general PL-data scenario in which an unlabeled region starts at time t_s and ends at t_e. Let $\mathcal{W}_t = w_1$, for $t = t_s - 1$ and $\mathcal{W}_t = w_2$ for $t = t_e + 1$. With this setting, we define for

[3] $\delta(x = y)$ is 1 when x equals y and 0 otherwise.

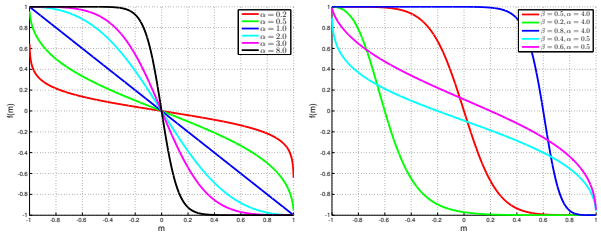
Figure 2: First figure shows $f(m)$ for various values of α with $\beta = 0.5$. Second figure shows $f(m)$ for different values of β and α. $\eta = 1$ for both figures.

every $t \in [t_s, t_e]$ (note that t is discrete)

$$\ln \frac{p(\mathcal{V}_t = 1|\mathcal{W}_t = w_1)}{p(\mathcal{V}_t = 1|\mathcal{W}_t = w_2)} \triangleq f\left(\frac{2(t - t_s)}{t_e - t_s} - 1\right) \quad (1)$$

where $f(m)$ is defined in the interval $[-1, 1]$ as

$$f(m) = \eta \frac{g(m)^\alpha - 1}{g(m)^\alpha + 1}, \quad g(m) = \left(\frac{m+1}{2}\right)^{\frac{1}{\log_2 \beta}} - 1. \quad (2)$$

Here $f(m)$ is continuous over $[-1, 1]$ and we are sampling a scaled and shifted version of $f(m)$ to obtain the VE-scores in equation 1. Its parameters are, (a) $\alpha > 0$ controls the shape of the curve, (b) $0 < \beta < 1$ controls the zero-crossing-point of $f(m)$ which is given by $2\beta - 1$, and, (c) $\eta > 0$ is the strength parameter. The first plot in figure 2 shows $f(m)$ for various values of α. Setting $\alpha = 1, \beta = 0.5$ results in a line, while $\alpha < 1$ leads to smoothly varying versions of $f(m)$ (closer to UNIF-PL-data), and values of $\alpha > 1$ result in sharper transitions (closer to FL-data). When $f(m) = 0$, the VE-scores for w_1 and w_2 are equal and so the zero-crossing-point determines when the weight shifts from w_1 to w_2. The second plot in figure 2 shows $f(m)$ for various values of β. Setting $\beta = 0.5$ ensures that $f(0) = 0$. This is useful in cases where w_1 and w_2 are of similar lengths and the labeled regions lie on or around the center of the words. Figure 2 also shows that setting $\beta < 0.5$ results in an early transition while values of $\beta > 0.5$ delay the transition. This is useful in cases where words (or phones) of dissimilar lengths occur next to each other (e.g. a consonant followed by a vowel or vice-versa). Note that $f(m)$ allows us to vary the degree of uncertainty in various ways, setting $\alpha < 1$ allows us to be uncertain throughout the unlabeled region, while $\alpha > 1$ makes the scores more certain. Also, setting $\eta = 0$ leads to UNIF-PL-data. Thus the UNIF-PL VE-score generation process is a special case of the above proposed function.

4. Baseline Description

The TIMIT baseline phone recognition system was trained making use of all phone-level segmentations, i.e., FL-data [8]. In order to extract the acoustic observations (frames), the speech signal was pre-emphasized (coefficient = 0.97) and then windowed using a Hamming window of size 20ms at 100Hz. We then extracted MFCC's from these windowed features. Each phone is modeled using a 3 states context-independent (CI) hidden Markov model (HMM) with up to 64 Gaussians per state. We follow the standard practice of building models for 48 phones and then mapping them down to 39 phones for scoring purposes [8]. Figure 1 in [1] shows the training graph. We test on the NIST Core test set [9]. In addition we used a 50 speaker development set different from the training and test sets for parameter tuning. All phone recognition results reported in this

paper were obtained by computing the string edit (Levenshtein) distance between the hypothesis and the reference.

SVB is a subset of Switchboard I [10] chosen to give a small, closed vocabulary [11]. This allows one to experiment on spontaneous continuous speech, but with less computational complexity and experiment turn-around time than true large vocabulary recognition. We use the SVB 500 word task in this paper. The baseline systems are HMMs implemented using DBNs [12] using left-to-right state clustered within-word triphones with three states and 32 Gaussians per state. Figure 1 in [13] shows the training graph. The observations are 13 dimensional PLPs normalized on a per conversation side basis along with their deltas and double-deltas. The A, B, and C folds were used for training, the D_short fold was the development set, and E was used as the evaluation set (see [11] for a definition of these folds). As in [13], the baseline here is trained using FL-data (word segmentations are known and fixed) [14].

For both tasks, we used a bigram language model (LM) for decoding (phone bigram in case of TIMIT). While using a trigram can lead to improved performance in SVB, we decided to use a bigram to make our results comparable to previously published work on SVB. In TIMIT, while context-dependent (CD) models have been shown to perform better, we use CI models for a rapid experiment turn-around time. Further, application to SVB clearly demonstrates the viability of our algorithm to CD systems. All systems in this paper were trained until 2% convergence using the expectation-maximization (EM) algorithm. For each task, the number of Gaussians per state, LM scale and word-insertion-penalty (WIP) were optimized over the dev set. All models in this paper were implemented using the Graphical Models Toolkit (GMTK) [15].

Next we describe how we generated PL-data. As mentioned in section 1, one way to generate PL-data is to drop labels of frames in FL-data [1, 2]. As word/phone transition points are regions of high uncertainty (due to co-articulation and other phone-boundary ambiguities), we start by dropping labels of frames around transitions. Note that we are not dropping frames but only the labels attached to frames. For example, in figure 1, consider the case when the length of all the unlabeled regions is zero, and then starting at transition points, they spread into words on either side causing an increasing number of frames to be unlabeled. If the total number of frames in the training set is N_T, and we drop labels for N frames, the percentage of unused segmentation data is defined as $\mathcal{U} = \frac{N}{N_T} \times 100$. As $\mathcal{U}$ increases, we are using smaller amounts of segmentation data. Also, $\mathcal{N}$ denotes the number of unlabeled frames per word/phone. In cases where the length of word/phone was $L < \mathcal{N}$ frames, we only dropped labels for $L - 1$ frames.

We refer to the baseline system trained using FL-data as *FL-Baseline*. We also trained systems using the exact same procedure as FL-Baseline, but replacing FL-data by the sequence labels (*SL-Baseline*). Given PL-data generated as described above, we built a *UNIF-PL-Baseline* system making use of uniform VE-scores, i.e. UNIF-PL-data. The number of labels dropped per word/phone was optimized separately for each task on the dev set. We found that $\mathcal{N} = 8$ frames/phone and $\mathcal{N} = 36$ frames/word gave the best results on TIMIT and SVB respectively. The results of these systems are shown in table 1. Systems trained on FL-data outperform those trained on SL-data. Also PL-data systems yield the best performance [1, 2]. Note that while FL-Baseline is the system that compares with previous work in TIMIT [8] and SVB [13], we suffix systems trained on SL-and-PL-data also by "Baseline" to highlight the fact that they use a modified version of the annotations used by the FL-Baseline. Further, in the following, we compare against the results of all these systems.

System	TIMIT		SVB	
	Dev	Eval	Dev	Eval
FL-Baseline	30.6	33.3	48.7	52.1
SL-Baseline	42.3	40.3	51.4	53.5
UNIF-PL-Baseline	29.1	32.5	47.0	51.2

Table 1: Baseline Results. Results for TIMIT are in phone error rate (PER), SVB results are in word error rate (WER).

FA Using	Training Set	TIMIT		SVB	
		Dev	Eval	Dev	Eval
FL-Baseline	FA-FL	28.0	31.9	46.8	50.9
	FA-UNIF-PL	27.8	31.6	46.5	50.9
SL-Baseline	FA-FL	32.2	35.7	49.2	51.9
	FA-UNIF-PL	28.6	**32.3**	47.2	**51.2**

Table 2: Two-pass training results. The first column shows which system was used to generate the segmentations. Results for TIMIT are in PER, SVB results are in WER.

5. Results

5.1. Two-pass Training

Sequence transcriptions were force-aligned with the audio using the appropriate FL-and-SL-Baseline systems resulting in two sets of forced-alignments (FA) each for TIMIT and SVB. We used these FAs to generate phone-segmentations and word-segmentations for TIMIT and SVB respectively (within-phone and within-word segmentations were dropped). The performance of the systems trained using the above segmentations are shown in table 2 against rows marked *FA-FL*. These segmentations constitute FL-data and we generated PL-data using the procedure described in section 4 with $\mathcal{N} = 8$ frames/phone and $\mathcal{N} = 36$ frames/word for TIMIT and SVB respectively (same as UNIF-PL-Baseline). We trained systems using the above PL-data with uniform VE-scores and the resulting error rates are in table 2 denoted by *FA-UNIF-PL*. The results show the benefits of two-pass training. It can be be seen that PL-systems are able to improve on the performance of the corresponding second-pass FL-system. We believe this is because the small labeled regions for every word/phone used in the case of PL-systems are more likely to be accurate in comparison to the full-segmentations used by FL-systems. Table 2 also suggests that one can start with sequence labels, and use our proposed PL approach to out-perform a single-pass system trained using FL-data. While two-pass systems are computationally more expensive than single-pass systems, this is more than offset as the annotator time spent in sequence-labeling is very small in comparison to full-annotation [2].

5.2. Generalized Function

Table 3 shows the results of using the proposed function to generate VE-scores (denoted by *G-PL*) for TIMIT phone recognition. In each case, we varied the number of unlabeled frames per phone. The row 1^* is the result of using PL-data where every phone had one labeled frame which corresponds to $\mathcal{U} = 84.7\%$ [4]. The column UNIF-PL shows results of systems trained on UNIF-PL-data (similar to [1]). In G-PL, for all values of $\mathcal{N}$ except 1^*, we ran a grid search over the following sets: $\alpha \in \{0.2, 0.5, 0.8, 1, 2, 8\}$, $\eta \in \{1000, 100000\}$ and

[4] Average phone length in TIMIT is about 7 frames.

$\mathcal{N}$	$\mathcal{U}$	UNIF-PL		(α, β, η)	G-PL	
		Dev	Eval		Dev	Eval
4	42.6	29.1	32.6	(2,0.25,10000)	27.9	31.9
8	65.3	29.1	32.5	(8,0.25,1000)	28.0	31.4
12	74.8	30.1	32.9	(1,0.25, 1000)	27.6	31.6
16	78.2	31.1	33.8	(1,0.5,100000)	27.9	31.9
1^*	84.7	32.0	34.2	(4.1,0.38,28623.5)	28.3	32.0

Table 3: TIMIT Phone Recognition Results (in PER). $\mathcal{N}$ is the number of labels dropped for each phone, $\mathcal{U}$ is the percentage of un-used segmentation data (section 5.2). 1^* corresponds to the case where there was only one labeled frame per phone.

$\beta \in \{0.25, 0.5, 0.75\}$ optimizing the parameters over the dev set. In the case of 1^*, as we require $0 < \beta < 1$, we used the *complex* method [16] (variation of downhill-simplex for constrained problems) to obtain the optimal parameters. Note that while the complex method may be used to find the optimal parameters for all $\mathcal{N}$, it is used for only $\mathcal{N} = 1^*$ as we found that it did not yield significant improvements over grid search which gave 28.6% and 32.2% on the dev and eval sets respectively for $\mathcal{N} = 1^*$. The results show that the proposed G-PL approach provides significant improvements over UNIF-PL. In particular, in the case of $\mathcal{N} = 1^*$, using G-PL to generate VE-scores results in an absolute improvement of **2.2%** over UNIF-PL (significant at the 0.02 level according to difference of proportions significance test). Note that even though 1^* uses only 15.3% of the labels used by FL-Baseline, it is **1.3%** better than FL-Baseline. Even in cases where one has access to FL-data, training using PL-data (by dropping labels around transitions) can lead to improved results. We note that this is similar to a phenomenon that sometimes arises in the active learning setting where using fewer labels actually improves performance [17]. For example, when $\mathcal{N} = 8$, the G-PL case is about **2%** better than FL-Baseline (significant at the 0.05 level).

While table 3 suggests that larger values of α are preferred, parameter combinations with $\alpha < 1$ gave similar (but slightly worse) results. For example, $\mathcal{N} = 8$, $(\alpha, \beta, \eta) = (0.5, 0.5, 1000)$ got PERs of 28.2% and 31.7% on the dev and eval sets respectively. In general, no single combination of parameters gave good performance for all values of $\mathcal{N}$. This was because different parameter combinations had varying effects on each of the phone classes – setting $\alpha < 1$ and $\alpha > 1$ improved vowel and consonant recognition performance respectively. To validate the above, we ran an experiment where the VE-scores were generated based on the classes of phones of either side of the unlabeled regions. The TIMIT phone set was collapsed in to two classes – vowels and consonants, and different combinations of the parameters was used for the four different types of transitions. Once again these parameters were optimized on the dev set using the complex method. The constraints were phonetically motivated (e.g., we expect vowel-to-vowel transitions to be fairly smooth, consonant-to-consonant transitions to be sharper in comparison, etc.). The combination of parameters shown in table 4 gave a PER of **28.1%** and **31.8%** on the dev and eval sets respectively for the $\mathcal{N} = 1^*$ case. The optimal values of these class-based parameters match our aforementioned expectations based on phonetic knowledge. Further, as vowels tend to have higher durations than consonants, vowel-consonant transitions are delayed ($\beta = 0.58$), while consonant-vowel transitions are early in nature ($\beta = 0.48$).

first/second class	vowel	consonant
vowel	(0.19, 0.5, 9403.2)	(4.1, 0.58, 4506.1)
consonant	(3.6,0.48,4322.1)	(6.2, 0.5, 8292.1)

Table 4: Combinations of (α, β, η) for different types of transitions. This gave PER of **28.1%** and **31.8%** for TIMIT dev and eval sets respectively in the $\mathcal{N} = 1^*$ case.

6. Discussion

We have presented two applications of our VE-based training approach. In the case of multi-pass training we have shown how one can use SL-data to obtain similar performance as systems trained on FL-data. It is interesting to note that segmentations generated using a first-pass system result in such significant improvements. We conjecture that as the EM algorithm is prone to local-optima, the segmentations are helping avoid local-optima. We also proposed a class of parameterized functions that can incorporate prior domain knowledge in to the VE-score generation process. While in this paper, we used grid-search based methods to obtain the optimal parameters, it is possible to learn them jointly with the other parameters in the DBN – an avenue we will explore in our future work.

7. References

[1] A. Subramanya and J. Bilmes, "Virtual Evidence for training Speech Recognizers using partially labeled data," in *Human Language Technologies Conference (HLT-NAACL)*, 2007.

[2] A. Subramanya, C. Bartels, J. Bilmes and P. Nguyen, "Uncertainty in training Large Vocabulary Speech Recognizers," in *Proc. of ASRU*, 2007.

[3] G. Evermann, H. Y. Chan and M.J.F. Gales, "Training LVCSR systems on thousands of hours of data," in *Proc. of ICASSP*, 2000.

[4] A. Stolcke et al., "The SRI March 2000 Hub-5 conversational speech transcription system," in *Proc. of NIST Speech Transcription Workshop*, 2000.

[5] L. Lamel, J. Gauvian and G. Adda, "Lightly supervised and unsupervised acoustic model training," in *Computers, Speech and Language*, 2002.

[6] J. Bilmes, "On Soft Evidence in Bayesian networks," Tech. Rep. UWEETR-2004-0016, Univ. of Washington, Dept. of EE, 2004.

[7] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann Publishers, Inc., 1988.

[8] K. F. Lee and H. Hon, "Speaker independent phone recognition using hidden markov models," *IEEE Transactions on Acoustics, Speech and Signal Processing*, 37(11), 1989.

[9] J. Glass, J. Chang and M. McCandless, "A Probabilistic Framework for Feature-based Speech Recognition," in *Proc. of ICSLP*, 1996.

[10] J. Godfrey, E. Holliman and J. McDaniel, "Switchboard: Telephone speech corpus for research and development," in *Proc. of ICASSP*, 1992, vol. 1, pp. 517–520.

[11] S. King, C. Bartels and J. Bilmes, "SVitchboard: Small-vocabulary tasks from switchboard," in *Proc. of the European Conf. on Speech Communication and Technology*, 2005.

[12] K. Livescu et al., "Articulatory feature-based methods for acoustic and audio-visual speech recognition: Summary from the 2006 JHU summer workshop," in *Proc. of ICASSP*, 2007.

[13] C. Bartels and J. Bilmes, "Use of Syllable Nuclei Locations to Improve ASR," in *Proc. of IEEE ASRU*, 2007.

[14] N. Deshmukh et al., "Re-segmentation of switchboard," in *Proc. of ICSLP*, Sydney, Australia, November 1998, pp. 1543–1546.

[15] J. Bilmes, *GMTK: The Graphical Models Toolkit*, 2002.

[16] M. Box, "A New Method of Constrained Optimization and a comparison with Other Methods," in *Computer Journal*, 1965.

[17] A. Bordes, S. Ertekin, J. Weston and L. Bottou, "Fast kernel classifiers with online and active learning" in *Proc. of JMLR*, 6, 2005.