# Calibration of Confidence Measures in Speech Recognition

Dong Yu, *Senior Member, IEEE*, Jinyu Li, *Member, IEEE*, Li Deng, *Fellow, IEEE*

*Abstract*—**Most of the speech recognition applications in use today rely heavily on confidence measure for making optimal decisions. In this work, we aim to answer the question: what can be done to improve the quality of confidence measure if we have no access to the internals of speech recognition engines? The answer provided in this paper is a post-processing step called confidence calibration, which can be viewed as a special adaptation technique applied to confidence measure. We report three confidence calibration methods that have been developed in this work: the maximum entropy model with distribution constraints, the artificial neural network, and the deep belief network. We compare these approaches and demonstrate the importance of key features exploited: the generic confidence-score, the application-dependent word distribution, and the rule coverage ratio. We demonstrate the effectiveness of confidence calibration on a variety of tasks with significant normalized cross entropy increasement and equal error rate reduction.**

***Index Terms*— confidence calibration, confidence measure, maximum entropy, distribution constraint, word distribution, deep belief networks**

## I. Introduction

Automatic speech recognition (ASR) technology has been widely deployed in applications including spoken dialog systems, voice mail (VM) transcription, and voice search [2][3]. Even though the ASR accuracy has been greatly improved over the past three decades, errors are still inevitable, especially under the noisy conditions [1]. For this reason, most speech applications today rely heavily on a computable scalar quantity, called confidence measure, to select optimal dialog strategies or to inform users what can be trusted and what cannot. The quality of the confidence measure is thus one of the critical factors in determining success or failure of speech applications.

Depending on the nature of a specific speech application, one or two types of confidence measures may be used. The word confidence measure (WCM) estimates the likelihood a word is correctly recognized. The semantic confidence measure (SCM), on the other hand, measures how likely the semantic information is correctly extracted from an utterance. For example, in the VM transcription application, SCM is essential for the keyword slots such as the phone number to call back and WCM is important for the general message to be transcribed. In the spoken dialog and voice search (VS) applications, SCM is more meaningful since the goal of these applications is to extract the semantic information (e.g., date/time, departure and destination cities, and business names) from users' responses.

Note that SCM has substantially different characteristics from WCM, and requires distinct treatment primarily because the same semantic information can be delivered in different ways. For instance, number 1234 may be expressed as "one thousand two hundred and thirty four" or "twelve thirty four". In addition, it is not necessary to recognize all the words correctly to obtain the correct semantic information. For example, there will be no semantic error when November seventh is misrecognized as November seven and vice versa. This is especially true when irrelevant or redundant words, such as *ma'am* in "*yes ma'am*" and *ah* in "*ah yes*", are misrecognized, or filtered out (e.g., using a garbage model [4][5]).

Numerous techniques have been developed to improve the quality of the confidence measures; see [6] for a survey. Briefly, these prior techniques can be classified into three categories. In the first category, a two-class (*true* or *false*) classifier is built based on features (e.g., acoustic and language model scores) obtained from the ASR engine and the classifier's likelihood output is used as the confidence measure. The classification models reported in the literature include the linear discriminant function [7][8], generalized linear model [9][10], Gaussian mixture classifier [11], neural network [12][13][49], decision tree [14][15], boosting [16], and maximum entropy model [17]. The techniques in the second category take the posterior probability of a word (or semantic slot) given the acoustic signal as the confidence measure. This posterior probability is typically estimated from the ASR lattices [18][19][20][21] or N-best lists [20][22]. These techniques require some special handling when the lattice is not sufficiently rich but do not require an additional parametric model to estimate the confidence score. The third category of techniques treats the confidence estimation problem as an utterance verification problem. These techniques use the likelihood ratio between the null hypothesis (e.g., the word is

Manuscript received July 26, 2010.

Some material contained in this paper has been presented at ICASSP 2010 [46][47]

D. Yu is with Microsoft Research, One Microsoft Way, Redmond, WA 98052, USA. phone: 425-707-9282; fax: 425-706-7329 (attn: dongyu); e-mail: dongyu@microsoft.com.

J. Li is with Microsoft Corporation, One Microsoft Way, Redmond, WA 98052, USA. e-mail: jinyli@microsoft.com.

L. Deng is with Microsoft Research, One Microsoft Way, Redmond, WA 98052, USA. e-mail: deng@microsoft.com.

correct) and the alternative hypothesis (e.g., the word is incorrect) as the confidence measure [8][23][24]. Discussions on the pros and cons of all the three categories of techniques can be found in [6]. Note that the parametric techniques in the first and third categories often outperform the non-parametric techniques in the second categories. This is because the parametric techniques can always include the posterior probability as one of the information sources and thus improve upon it.

Whichever parametric technique is used, the confidence measure is typically provided by the ASR engine and trained on a generic dataset. It is thus a black box to the speech application developers. Using a generic training set can provide good average out-of-box performance across a variety of applications. However, this is obviously not optimal since the data used to train the confidence measure may differ vastly from the real data observed in a specific speech application. The disparity can be due to different language models used and different environments in which the applications are deployed. In addition, having the confidence model inside the ASR engine makes it difficult to exploit application-specific features such as the distribution of the words (see Section IV). These application-specific features are either external to the ASR engine or cannot be reliably estimated from the generic training set.

Currently, only a limited number of companies and institutions have the capability and resources to build real-time large vocabulary continuous ASR engines. Most speech application developers have no access to the internals of the engines and cannot modify the confidence estimation algorithms built in. Thus, they often have no choice but rely on the confidence measure provided by the engines. This situation can be painful for the speech application developers, especially when a poor confidence model or feature set is used in the ASR engine or when the model parameters are not well tuned.

In this paper we aim at answering the following question: what can be done to improve the quality of the confidence measures if we have no access to the internals of the ASR engines? This problem has become increasingly important recently since more speech applications are built by application developers who know nothing about the ASR engines. The solution provided in this paper is a technique which we call confidence calibration. It is a post-processing step that tunes the confidence measure for each specific application using a small amount of transcribed calibration data collected under real usage scenarios. To show why confidence calibration would help, let us consider a simple speech application that only recognizes "yes" and "no". Let us further assume "yes" is correctly recognized 98% of time and it consists of 80% of the responses, and "no" is correctly recognized 90% of time and it consists of 20% of the responses. In this case, a confidence score of 0.5 for "yes" from the ASR engine may obviously mean differently from the same score for "no". Thus an adjusted (calibrated) score using this information would help to improve the overall quality of the confidence score if done correctly.

We propose and compare three approaches for confidence calibration: the maximum entropy model (MaxEnt) with distribution constraints (MaxEnt-DC), the conventional artificial neural network (ANN), and the deep belief network (DBN). To the best of our knowledge, this is the first time MaxEnt-DC and DBNs are applied to confidence estimation and calibration. The contribution of this work also includes the discovery of effective yet non-obvious features such as the word distribution information and the rule coverage ratio in improving confidence measures.

We demonstrate that the calibration techniques proposed in this paper work surprisingly well with significant confidence quality improvements over the original confidence measures provided by the ASR engines across different datasets and engines. We show that DBNs typically provide the best calibration result, but is only slightly better than the MaxEnt-DC approach, yet with the highest computational cost.

The quality of the confidence measure in this paper is evaluated using the normalized cross entropy (NCE) [50], the equal error rate (EER), and the detection error trade-off (DET) curve [26]. We provide their definitions below.

The NCE is defined as

$$\mathrm{NCE} = \frac{H_{base} - H_{cond}}{H_{base}}, \quad (1)$$

where

$$H_{cond} = -\sum_{i=1}^{N} log\big(c_i\delta(y_i = 1) + (1 - c_i)\delta(y_i = 0)\big), \quad (2)$$

and

$$H_{base} = -nlog\left(\frac{n}{N}\right) - (N - n)log\left(1 - \frac{n}{N}\right). \quad (3)$$

Here we assume we have a set of $N$ confidence scores and the associated class labels $\{(c_i \in [0,1], y_i \in \{0,1\}) \mid i = 1, \cdots, N\}$, where $y_i = 1$ if the word is correct and $y_i = 0$ otherwise. In (2) and (3) $\delta(x) = 1$ if $x$ is true and $\delta(x) = 0$ otherwise, and $n$ is the number of samples whose $y_i = 1$. The higher the NCE is, the better the confidence quality.

EER is the error rate when the operating threshold for the accept/reject decision is adjusted such that the probability of false acceptance and that of false rejection become equal. The lower the EER is, the better the confidence quality. The DET curve describes the behavior over different operating points. The crossing of the DET curve with the $(0,0) - (1,1)$ diagonal line gives the EER. The closer the DET curve is to the origin $(0,0)$, the better the confidence quality.

A perfect confidence measure is the one that always outputs one when the label is *true* and zero otherwise. Under this condition the EER equals zero, NCE equals one, and the DET curve shrinks to the single point of $(0,0)$. Note that these criteria measure different aspects of the confidence scores although they are somewhat related. In particular, NCE measures how close the confidence is related to the probability that the output is true. On the other hand, EER and DET indicate how well the confidence score is in separating the true and false outputs with a single value and a curve, respectively, when a decision needs to be made to accept or reject the hypothesis. For example, two confidence measures can have

the same value of EER but very different NCE values, as we will see in Section VI. For many speech applications, EER and DET are more important than NCE since speech application developers typically care about how the confidence scores can be used to reduce costs (e.g., time to task completion or dissatisfaction rate). When EER and DET are the same, one then prefers the confidence measure with higher NCE.

Please note that when applied to a specific application, the criterion can be different. For example, in the directory assistance [3] application, the goal is to maximize the profit. A correctly routed call can reduce the human cost and hence increase the profit. In contrast, an incorrectly routed call may reduce the caller satisfaction rate and thus reduce the profit. The total profit, in this example, would be

$$profit = g \cdot n_+ - c \cdot n_- , \quad (4)$$

where $g$ is the gain if the call is correctly routed, $c$ is the cost if the call is misrouted, and $n_+$ and $n_-$ are the number of calls routed correctly and incorrectly, respectively. $c$ is typically 10 times larger than $g$. No cost or profit is incurred if the call is not routed automatically but directed to a human attendant. The optimal operation point depends on the DET curve and the actual values of the gain and cost.

The rest of the paper is organized as follows. In Section II we review the MaxEnt model with distribution constraints (MaxEnt-DC). We also describe the specific treatment needed for both the continuous and the multi-valued nominal features as required for confidence calibration. In Section III we introduce DBNs and explain its training procedure. In Sections IV and V, we illustrate the application-specific features that have been proven to be effective in improving the quality of the WCM and the SCM, respectively. We evaluate the proposed techniques on several datasets in Section VI and conclude the paper in Section VII.

## II. Maximum Entropy Model with Distribution Constraints

The MaxEnt model with moment constraints (MaxEnt-MC) [27] is a popular discriminative model that has been successfully applied to natural language processing (NLP) [28], speaker identification [29], statistical language modeling [30], text filtering [31], machine translation [32], and confidence estimation [17]. Given an $N$-sample training set $\{(x_n, y_n) \mid n = 1, \cdots, N\}$ and a set of $M$ features $f_i(x, y), i = 1, \cdots, M$ defined on the input $x$ and output $y$, the posterior probability in the MaxEnt-MC model is defined in the log-linear form

$$p(y|x; \boldsymbol{\lambda}) = \frac{1}{Z_{\boldsymbol{\lambda}}(x)} \exp\left(\sum_i \lambda_i f_i(x, y)\right), \quad (5)$$

where $Z_{\boldsymbol{\lambda}}(x) = \sum_y \exp(\sum_i \lambda_i f_i(x, y))$ is the normalization constant to fulfill the probability constraint $\sum_y p(y|x) = 1$. The parameters $\lambda_i$ above are optimized to maximize the conditional log-likelihood

$$O(\boldsymbol{\lambda}) = \sum_{n=1}^{N} \log p(y_n|x_n) \quad (6)$$

over the entire training set.

Impressive classification accuracy has been achieved using the MaxEnt-MC model on tasks where binary features are used. However, it was not as successful when continuous features are used. Recently we developed the MaxEnt-DC [25] model and proposed that the information carried in the feature distributions be used to improve the classification performance. This model is a natural extension to the MaxEnt-MC model since the moment constraints are the same as the distribution constraints for binary features.

Binary features, continuous features, and multi-valued nominal features are treated differently in the MaxEnt-DC model. For the binary features, no change is needed since the distribution constraint is the same as the moment constraint. The special treatment for the continuous features is as follows. Each continuous feature $f_i(x, y)$ is expanded to $K$ features, where $K$ can be determined based on the amount of training data available or through a held out set. When $K \geq 4$, the expansion takes the form of

$$f_{ik}(x, y) = a_k\big(f_i(x, y)\big) f_i(x, y), \quad (7)$$

where $a_k(.)$ is a weight function whose definition and calculation method can be found in [25][33] [34]. When $K \leq 3$, the expansion has a simpler form of

$$f_{ik}(x, y) = [f_i(x, y)]^k \quad (8)$$

For the confidence calibration tasks evaluated in this work, we have found that $K \leq 4$ is generally sufficient. If $K = 1$ the MaxEnt-DC model is reduced to the MaxEnt-MC model.

The special treatment for the multi-valued nominal features is as follows. The nominal feature values are first sorted in the descending order of their number of occurrences. The top $J - 1$ nominal values are then mapped into token IDs in $[1, J - 1]$, and all remaining nominal values are mapped into the same token ID $J$, where $J$ is chosen to guarantee the distribution of the nominal features can be reliably estimated and may be tuned based on a held out set. Subsequently each feature $f_i(x, y)$ is expanded to $J$ features

$$f_{ij}(x, y) = \delta(f_i(x, y) = j). \quad (9)$$

In our experiments we have used the following relatively simple way to determine $J - 1$. We set it to be the number of nominal values that has been observed in the training set for at least $\theta$ times where we set $\theta = 20$ in all our experiments. As an example, we have a multi-valued nominal feature that takes values $\{A, B, C, D, E, F, G\}$ and these values have been observed in the training set by A(23), B(96), C(11), D(88), E(43), F(14), and G(45) times, respectively. We now first sort these values in the descending order of the times they are observed; i.e., $B(96)$, $D(88)$, $G(45)$, $E(43)$, $A(23)$, $F(14)$, and $C(11)$. We then set $J = 6$ since only five values are observed more than $\theta = 20$ times. We thus convert this feature into six features $f_{i1}(x, y), \cdots, f_{i6}(x, y)$, out of which only one expanded-feature equals to one and the remaining expanded-features equal to zero. More complicated approaches can be applied, for example, by clustering less frequently observed values. We have not explored further along this direction since this will not affect our main message. Note that since the features are categorical, the MaxEnt-DC model would

be equivalent to the MaxEnt-MC model (where each nominal value is considered a separate feature) if $J$ were to be chosen so that each nominal value has its own token ID (i.e., $\theta = 1$). Based on our experiments setting $\theta = 1$ often performs worse than using some $\theta > 1$. By setting $\theta = 20$, $J$ automatically decreases when fewer calibration data are available and so will less likely cause over fitting. Depending on the size of the data set, $J$ varies between 12 and 133.

After the continuous and multi-valued nominal features are expanded, the posterior probability in the MaxEnt-DC model is evaluated as

$$p(y|x) = \frac{1}{Z_{\boldsymbol{\lambda}}(x)} \exp\left(\sum_{i\in\{binary\}} \lambda_i f_i(x,y) + \sum_{i\in\{continuous\},k} \lambda_{ik} f_{ik}(x,y) + \sum_{i\in\{nominal\},j} \lambda_{ij} f_{ij}(x,y)\right) \quad (10)$$

and the existing training and decoding algorithms [36] [37] [38] [39] as well as the regularization techniques [40] [41] [42] [43] [44] for the MaxEnt-MC model can be directly applied to this higher-dimensional space. In our experiments we have used the RPROP [36] training algorithm and used the L2-norm regularization with the regularization parameter set to 100 in all experiments.

The MaxEnt-DC model has been applied to several tasks in recent past [25] [35] [45]. Consistent improvement over the MaxEnt-MC model has been observed when sufficient training data is available. In this paper we will show that this model is also effective in calibrating the confidence measures. Note that White et al. [17] applied the MaxEnt-*MC* model to confidence measure in speech recognition and observed improved confidence quality over the baseline systems. Our work described in this paper differs substantially from [17] in three ways. First, we use the more general MaxEnt-*DC* model. Second, we exploit the application-specific features, which are essential to improving confidence measure, yet only available at the application level. Third, we target the use of the MaxEnt model at the confidence calibration setting instead of for generic confidence measure. In addition, the work reported in [17] focused on WCM only while we develop and apply our technique for both WCM and SCM measures.

## III. Deep Belief Networks

DBNs are densely connected, directed belief nets with many hidden layers. Inference in DBN is simple and efficient. Each pair of layers is separated into an input visible layer **v** and an output hidden layer $\boldsymbol{h}$ with the relationship

$$v_{j+1} = h_j = \sigma\left(\sum_{i=1}^{V} w_{ij}\, v_i + a_j\right), \quad (11)$$

where $\sigma(x) = 1/\big(1 + exp(x)\big)$, $w_{ij}$ represents the interaction term between input (visible) unit $v_i$ and output (hidden) unit $h_j$, and $a_j$ is the bias terms. The output of the lower layers becomes the input of the upper layers till the final layer, whose output is then transformed into a multinomial distribution using the softmax operation

$$\mathrm{p}(l = k|\mathbf{h};\theta) = \frac{exp(\sum_{i=1}^{H} \lambda_{ik} h_i + a_k)}{Z(\boldsymbol{h})}, \quad (12)$$

where $l = k$ denotes the input been classified into the $k$-th class, and $\lambda_{ik}$ is the weight between $h_i$ at the last layer and the class label $k$. For confidence calibration purposes, $k$ only takes values 0 (false) or 1 (true).

On contrast, learning in DBNs is very difficult due to the existence of many hidden layers. In this paper we adopt the procedure proposed in [56][54][55] for training DBN parameters: train a stack of restricted Boltzmann machines (RBMs) generatively first and then fine-tune all the parameters jointly using the back-propagation algorithm by maximizing the frame-level cross-entropy between the true and the predicted probability distributions over class labels (0 and 1 in our case).

An RBM can be represented as a bipartite graph, where a layer of visible units **v** are connected to a layer of hidden units **h** but without visible-visible or hidden-hidden connections. In the RBMs, the joint distribution $\mathrm{p}(\mathbf{v},\mathbf{h};\theta)$ can be defined as

$$\mathrm{p}(\mathbf{v},\mathbf{h};\theta) = \frac{exp\big(-\mathrm{E}(\mathbf{v},\mathbf{h};\theta)\big)}{Z} \quad (13)$$

over the energy function $\mathrm{E}(\mathbf{v},\mathbf{h};\theta)$, where $\theta$ is the model parameter and $Z = \sum_u \sum_h exp\big(-\mathrm{E}(\mathbf{v},\mathbf{h};\theta)\big)$ is a normalization factor or partition function. The marginal probability that the model assigns to a visible vector **v** follows as

$$\mathrm{p}(\mathbf{v};\theta) = \frac{\sum_{\boldsymbol{h}} exp\big(-\mathrm{E}(\mathbf{v},\mathbf{h};\theta)\big)}{Z}. \quad (14)$$

Note that, the energy functions for different types of units are different. For a Bernoulli (visible)-Bernoulli (hidden) RBM with $V$ visible units and $H$ hidden units, the energy is

$$\mathrm{E}(\mathbf{v},\mathbf{h};\theta) = -\sum_{i=1}^{V}\sum_{j=1}^{H} w_{ij}\, v_i h_j - \sum_{i=1}^{V} b_i v_i - \sum_{j=1}^{H} a_j h_j. \quad (15)$$

It follows directly that the conditional probabilities are

$$p\big(h_j = 1|\mathbf{v};\theta\big) = \sigma\left(\sum_{i=1}^{V} w_{ij}\, v_i + a_j\right), \quad (16)$$

$$p(v_i = 1|\mathbf{h};\theta) = \sigma\left(\sum_{j=1}^{H} w_{ij}\, h_j + b_i\right). \quad (17)$$

The energy for the Gaussian-Bernoulli RBM, in contrast, is

$$\mathrm{E}(\mathbf{v},\mathbf{h};\theta) = -\sum_{i=1}^{V}\sum_{j=1}^{H} w_{ij}\, v_i h_j + \frac{1}{2}\sum_{i=1}^{V}(v_i - b_i)^2 - \sum_{j=1}^{H} a_j h_j. \quad (18)$$

The corresponding conditional probabilities become

$$p\big(h_j = 1|\mathbf{v};\theta\big) = \sigma\left(\sum_{i=1}^{V} w_{ij}\, v_i + a_j\right), \quad (19)$$

$$p(v_i|\mathbf{h};\theta) = N\left(v_i; \sum_{j=1}^{H} w_{ij}\, h_j + b_i, 1\right). \quad (20)$$

where $N(x; m, 1)$ is a Gaussian distribution with mean $m$ and

variance one. Gaussian-Bernoulli RBMs can be used to convert real-valued stochastic variables to binary stochastic variables which can then be further processed using the Bernoulli-Bernoulli RBMs.

In the RBMs the weights are updated following the gradient of the log likelihood $\log p(\mathbf{v};\theta)$ as [54]:

$$\Delta w_{ij} = \langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model}, \quad (21)$$

where $\langle v_i h_j \rangle_{data}$ is the expectation observed in the training set and $\langle v_i h_j \rangle_{model}$ is that same expectation under the distribution defined by the model. Note that $\langle v_i h_j \rangle_{model}$ is extremely expensive to compute exactly. Thus, the contrastive divergence (CD) approximation to the gradient is used where $\langle v_i h_j \rangle_{model}$ is replaced by running the Gibbs sampler initialized at the data for one full step [55].

As we will see in Sections IV and V, both real- and binary-valued features will be used in the confidence calibration procedure. This requires the use of a mixed first layer of units where both Gaussian and Bernoulli units exist. This, unfortunately, turns out to be very unstable during the training process even if we carefully adjust the learning rate for different types of units. This issue was resolved by using only Bernoulli-Bernoulli RBMs after noticing that all features used in our models are bimodal within the range of $[0,1]$.

## IV. Features for the Word Confidence Calibration

Speech application developers have no access to the engine's internal information. Hence the information available to the confidence calibration module is just the recognized word sequence and the associated confidence scores

$$\left\{ \boldsymbol{x}_{n,t}^{w} = \begin{bmatrix} w_{n,t} \\ c_{n,t} \end{bmatrix} \middle| t = 1, \cdots, T_n \right\} \quad (22)$$

from the ASR engine, where $w_{n,t}$ is the $t$-th recognized word in the $n$-th utterance and $c_{n,t}$ is the associated confidence score. The goal of the word confidence calibration is to derive a better confidence measure $c_{n,t}^{'} = p\left(y_{n,t}^{w} \middle| \boldsymbol{x}_{n,t}^{w}; \boldsymbol{\lambda}\right)$ for each word $w_{n,t}$. To distinguish between these two confidence measures, we call the confidence measures before and after the calibration the *generic* and *calibrated* confidence measures, respectively. To learn the calibration model, we need a labeled training (calibration) set that informs whether each recognized word is correct (true) or not (false).

The key to the success of confidence calibration is to identify the effective features from $\boldsymbol{x}_{n,t}^{w}$. The obvious feature for word $w_{n,t}$ is the generic confidence measure $c_{n,t}$. However, using this feature alone does not provide additional information and thus cannot improve the EER as we will see in Section VI. After some additional analysis, it is not difficult to suggest the use of the adjacent words' confidence scores $c_{n,t-1}$ and $c_{n,t+1}$ since an error in adjacent words can affect the central word. Unfortunately, using the adjacent confidence scores helps only by a small margin as will be demonstrated in Section VI.

The non-obvious but highly effective feature was discovered in this work when we notice that the word distribution for different applications is often vastly different. This difference is quantitatively shown in TABLE I, where the top ten words and their frequencies in VM transcription and command and control (C&C) datasets are displayed. The non-uniform distribution contains valuable information and that information can be naturally exploited using the MaxEnt-DC model, ANN, and DBNs but not MaxEnt-MC model. By exploiting the word distribution information, the confidence calibration tool can treat different words differently to achieve better overall confidence quality. We will show in Section VI that the word distribution is the most important source of information in improving the WCM.

TABLE I
Top 10 Words and Their Frequencies in the Voice Mail Transcription and Command and Control Datasets

| VM | | | C&C | | |
|---|---|---|---|---|---|
| **word** | **count** | **percentage** | **word** | **count** | **percentage** |
| i | 463 | 3.03% | three | 716 | 4.81% |
| you | 451 | 2.95% | two | 714 | 4.80% |
| to | 446 | 2.92% | five | 713 | 4.79% |
| the | 376 | 2.46% | one | 691 | 4.64% |
| and | 369 | 2.42% | seven | 651 | 4.38% |
| uh | 356 | 2.33% | eight | 638 | 4.29% |
| a | 302 | 1.98% | six | 627 | 4.21% |
| um | 287 | 1.88% | four | 625 | 4.20% |
| that | 215 | 1.41% | nine | 616 | 4.14% |
| is | 213 | 1.39% | zero | 485 | 3.26% |

To use these features we construct the feature vector for the $t$-th recognized word in the $n$-th utterance as

$$\boldsymbol{v}_{n,t}^{w} = \left[\boldsymbol{w}_{n,t-1}, c_{n,t-1}, \boldsymbol{w}_{n,t}, c_{n,t}, \boldsymbol{w}_{n,t+1}, c_{n,t+1},\right]^{T} \quad (23)$$

and

$$\boldsymbol{v}_{n,t}^{w} = \left[\boldsymbol{w}_{n,t}, c_{n,t}\right]^{T} \quad (24)$$

with and without using information from adjacent words, respectively. In (23) and (24) $\boldsymbol{w}_{n,t}$ is a vector representation of word $w_{n,t}$ using the approach explained in Section II to handle the multi-valued nominal features, and $[\cdot]^T$ is the transpose of $[\cdot]$. Note that $c_{n,t}$ is a continuous feature and needs to be expanded according to (7) or (8) when using the MaxEnt-DC model.

## V. Features for the Semantic Confidence Calibration

In addition to the recognized words $w_{n,t}$ and the corresponding generic word confidence scores $c_{n,t}$, speech application developers also have access to the generic semantic confidence score $c_n^s$ of the $n$-th trial (utterance) from the ASR engine to calibrate the SCM. In other words, we have the observation vector of

$$\boldsymbol{x}_n^s = \langle c_n^s, \begin{bmatrix} w_{n,1} \\ c_{n,1} \end{bmatrix}, \begin{bmatrix} w_{n,2} \\ c_{n,2} \end{bmatrix}, \cdots, \begin{bmatrix} w_{n,T_n} \\ c_{n,T_n} \end{bmatrix} \rangle. \quad (25)$$

The goal of the SCM calibration is to derive a better semantic confidence score $c_n^{s\prime} = p(y_n^s | \boldsymbol{x}_n^s; \boldsymbol{\lambda})$ for each trial by post-processing $\boldsymbol{x}_n^s$.

From our previous discussion we know that the distribution of the generic WCM and the recognized words carry valuable information. This information can also be exploited to improve the SCM. However, $T_n$, the total number of words recognized

in each trial, can be different, while the MaxEnt-DC model, ANN and DBN all require a fixed number of input features. Using the intuition that whether the semantic information extracted is correct or not is determined primarily by the least confident keywords, we sort the keyword confidence scores in the ascending order, keep only the top $M$ keyword confidence scores and the associated keywords, and discard *garbage* words that are not associated with the semantic slot. Our experiments indicate that $M = 2$ and $M = 3$ perform similarly and are optimal for most tasks although the average number of keywords in these tasks varies from one to seven. Denoting the top $M$ sorted words and confidence scores as

$$\begin{bmatrix}\bar{w}_{n,1}\\ \bar{c}_{n,1}\end{bmatrix}, \begin{bmatrix}\bar{w}_{n,2}\\ \bar{c}_{n,2}\end{bmatrix}, \cdots, \begin{bmatrix}\bar{w}_{n,M}\\ \bar{c}_{n,M}\end{bmatrix} \quad (26)$$

we construct the features for the $n$-th utterance as

$$\boldsymbol{v}_n^s = \left[c_n^s, \bar{\boldsymbol{w}}_{n,1}, \bar{c}_{n,1}, \bar{\boldsymbol{w}}_{n,2}, \bar{c}_{n,2}, \cdots, \bar{\boldsymbol{w}}_{n,M}, \bar{c}_{n,M}\right]^T, \quad (27)$$

Here, again, $\bar{\boldsymbol{w}}_{n,i}$ is the vector representation of word $\bar{w}_{n,i}$ , and $c_n^s$ and $\bar{c}_{n,i}$ are real-valued features that need to be expanded when using the MaxEnt-DC model.

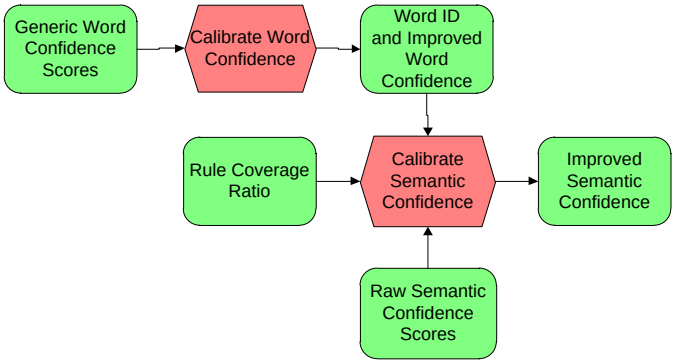


Fig. 1. The procedure to calibrate the semantic confidence

We can significantly improve the SCM using the above features for calibration. However, further improvements can be obtained by adding a less obvious feature: the rule coverage ratio (RCR) defined as

$$r_n = \frac{\#\text{ words within the rule slot}}{\text{total }\#\text{ of recognized words}}. \quad (28)$$

This feature is only available when a garbage model (e.g., the N-gram based filler model [5]) is used in the grammar so that the grammar has the form of <garbage><rule><garbage>. The RCR is the ratio of the number of words associated with the rule slot and the total number of words (including garbage words) recognized. The reason RCR can be helpful is that when many words are outside of the rule slot, chances are that the acoustic environment is bad (e.g., with side talking) or the speech is more casual. By including RCR, the feature vector becomes

$$\boldsymbol{v}_n^s = \left[c_n^s, \bar{\boldsymbol{w}}_{n,1}, \bar{c}_{n,1}, \bar{\boldsymbol{w}}_{n,2}, \bar{c}_{n,2}, \cdots, \bar{\boldsymbol{w}}_{n,M}, \bar{c}_{n,M}, r_n\right]^T, \quad (29)$$

In the formulation of (26) we can use the generic word confidence scores obtained from the ASR engine directly. However, a more natural and effective way is to use the calibrated word confidence scores. The whole procedure of semantic calibration is illustrated in Fig. 1 . Note that if dialogs are involved, some features described in [53] can also be used to further improve the quality of SCM. However, our experiments reported below show that once the word distribution and the RCR features are used, adding other features only provides small further improvements on the tasks we have tested.

## VI. Experimental Evaluation

To evaluate and compare the effectiveness of the confidence calibration techniques we just described, we have conducted a series of experiments on several datasets collected under real usage scenario using two different ASR engines. In this section we describe these experiments and compare the quality of the calibrated confidence measures using different features we described in Sections IV and V. We show that we can significantly improve the confidence measures using the word distribution and RCR over the generic confidence measure from the ASR engines used in different versions of Bing search for mobile (earlier versions named live search for mobile) [58].

Each dataset in the experiments were split into calibration (training), development, and test sets by the log time so that test set contains the most recent data collected and the training set earliest. The generic confidence measures were obtained directly from the ASR engines E1 and E2. Both of the engines were trained on a large generic training set including the data collected under many different applications. Engine E1 used a Gaussian mixture model classifier trained discriminatively. Engine E2 used an ANN based classifier.

The features used in Engine E1 to produce the generic confidence scores are:

- the normalized (by subtracting the best senone likelihood and then dividing it by the duration) acoustic model (AM) score;
- the normalized background model score;
- the normalized noise score;
- the normalized language model (LM) score;
- the normalized duration;
- the normalized LM perplexity;
- the LM fanout;
- the active senones;
- the active channels;
- the score difference between the first and second hypotheses;
- the number of n-best; and
- the number of nodes, arcs, and bytes in the lattice.

In addition to these basic features, Engine E2 also used features from the adjacent words, the average AM score, and the posterior probability estimated from the lattice.

The AM scores measure how well the acoustic data matches the grammar and acoustic model, the unconstrained speech-like sounds, and noise. Features taken from the LM are included to help the classifier adapt to different recognition grammars. All other features above measure the state of the recognition process itself. For instance, how hard the recognizer had to work to produce the result (active senones and channels), how

many entries are in the lattice, and the size (in nodes, arcs, and bytes) of the recognition result. Although the generic confidence score generated by the Engine E1 is not as good as that generated by Engine E2, it is better than the posterior probability based approach mainly because the lattice is not sufficiently rich due to aggressive pruning.

In all the results presented below, the best configuration is always determined based on NCE on the development set. The best configuration is then applied to the test set to calculate metrics. For the MaxEnt-DC approach, we run experiments with all continuous features expanded to one to four features and pick the best configuration. For the ANN approach, we run experiments with one hidden layer since more hidden layers actually performed worse on the development and test sets. The number of hidden units takes the values of 30, 50, 100, and 200. Since the weights are initialized randomly, we run five experiments for each configuration and pick the best one on the development set. The configuration with 50 units typically wins over. For the DBN approach, we run experiments with one to four hidden layers and each hidden layer has 50, 100, 150, and 200 units. For each configuration we also run five experiments and pick the best model based on the development set performance. The best system typically is the one with three hidden layers and 100 hidden units each. Note that the results from the ANN and DBN have larger variance than that from the MaxEnt-DC model (which is convex). This is mainly due to the fact that the former is not convex and random initialization can lead to different model parameters and performance.

MaxEnt-DC, ANN, and DBN are three representative approaches we compare in this paper. MaxEnt-DC is simple and effective for the task, and DBN has shown to be very powerful. We have also tested the confidence calibration technique using conditional random field (CRF) with distribution constraints. Results using CRF-DC is not presented in this paper because they are very close to the MaxEnt-DC results but achieved with much higher computational complexity. We did not try support vector machine (SVM) or decision tree (DT) since we do not expect a significantly better result over the methods we have tested to be obtained. In addition, to make the SVM and DT outputs look like confidence scores additional steps need to be taken to convert the score to the range of [0 1].

### *A. Word Confidence Calibration*

The performance of the word confidence calibration has been evaluated on many datasets and similar gains have been obtained. In this paper we use two datasets - a voice mail (VM) transcription dataset and a command and control (C&C) dataset to demonstrate the effectiveness of the proposed approaches. TABLE II summarizes the number of utterances and words and word error rate (WER) obtained from a speaker-independent ASR Engine E1 in the training (calibration), development, and test sets for each dataset. The VM transcription is a large vocabulary task with vocabulary size 120K and the C&C is a middle vocabulary task with vocabulary size 2K. Both datasets were collected under real usage scenarios and contain both clean and noisy data. The ASR engines E1 and E2 support using application specific language models and vocabularies. The LMs used for the C&C task are probabilistic context-free grammars (CFG) and each dialog turn uses a different LM. The LM used for the VM task is a class-based n-gram model with reference to the CFGs that recognize numbers, date time, and personalized name list, etc. The perplexity of the C&C task on the calibration set varies from 3 to over 100 depending on the dialog turn. The perplexity of the VM task on the calibration set is 87.

TABLE II
SUMMARY OF DATASETS FOR WORD CONFIDENCE CALIBRATION

| | VM | | | C&C | | |
|---|---|---|---|---|---|---|
| | # utterances | # words | WER | # utterances | # words | WER |
| train | 352 | 15K(4 hrs) | 28% | 4381 | 15K(4 hrs) | 8% |
| dev | 368 | 15K(4 hrs) | 27% | 4391 | 15K(4 hrs) | 8% |
| test | 371 | 15K(4 hrs) | 28% | 4371 | 15K(4 hrs) | 8% |

TABLE III and TABLE IV compare the word confidence calibration performance in NCE and EER with and without using information from the adjacent words and word distributions on the VM and C&C datasets, respectively. In these tables, each setting is denoted as ±W±C where W means word distribution, C means the context (adjacent word) information, and + sign and - sign indicate the information is and is not used respectively. As explained in Section II, we assign a unique token ID for words that occur more than $\theta = 20$ times in the training set and assign the same token ID $J$ to all other words. This yields 133 and 109 word tokens (i.e., $J$=133 and 109) in the VM and C&C calibration models respectively. In other words, each word in the VM and C&C tasks is represented as a 133-dim and 109-dim vector, respectively, when constructing features in eqs. (23) and (24).

TABLE III
WORD CONFIDENCE QUALITY COMPARISON USING DIFFERENT FEATURES AND APPROACHES ON THE VOICE MAIL DATASET

| Features | MaxEnt-DC | | ANN | | DBNs | |
|---|---|---|---|---|---|---|
| | NCE | EER% | NCE | EER% | NCE | EER% |
| No Calibration | -0.264 | 33.8 | -0.264 | 33.8 | -0.264 | 33.8 |
| -W-C | 0.104 | 33.8 | 0.099 | 33.8 | 0.104 | 33.8 |
| -W+C | 0.132 | 31.9 | 0.130 | 31.6 | 0.130 | 31.8 |
| +W-C | 0.232 | 27.3 | 0.229 | 27.2 | 0.238 | 27.1 |
| +W+C | 0.250 | 26.1 | 0.243 | 25.6 | 0.255 | 26.1 |

+W and +C indicate the word distribution and the context (adjacent word) information are used, respectively.

TABLE IV
WORD CONFIDENCE QUALITY COMPARISON USING DIFFERENT FEATURES AND APPROACHES ON THE COMMAND & CONTROL DATASET

| Features | MaxEnt-DC | | ANN | | DBNs | |
|---|---|---|---|---|---|---|
| | NCE | EER% | NCE | EER% | NCE | EER% |
| No Calibration | -0.455 | 32.7 | -0.455 | 32.7 | -0.455 | 32.7 |
| -W-C | 0.105 | 32.7 | 0.085 | 32.7 | 0.092 | 32.7 |
| -W+C | 0.129 | 30.2 | 0.097 | 32.1 | 0.100 | 32.2 |
| +W-C | 0.190 | 23.1 | 0.183 | 23.7 | 0.215 | 22.8 |
| +W+C | 0.209 | 21.2 | 0.169 | 23.0 | 0.212 | 22.2 |

+W and +C indicate the word distribution and the context (adjacent word) information are used, respectively.

From TABLE III and TABLE IV we observe that when only the generic word confidence score (i.e., the setting -W-C) is

used as the feature, no EER reduction is obtained. However, we can improve NCE from -0.264 and -0.455 to around 0.1 on the VM and C&C test sets respectively no matter which approach is used. This indicates that NCE and EER, although both are important, measure different aspects of the confidence scores. This improvement can be more clearly seen in Fig. 2 and Fig. 3, where the relationship between the WCM and the accurate rate for the VM and the C&C datasets are displayed. Ideally, we would expect the curve to be a diagonal line from (0,0) to (1,1) so that a confidence score of *x* indicates that the prediction is correct with probability of *x*. It is clear from Fig. 2 and Fig. 3 that the curve obtained using the -W-C setting aligns better to the diagonal line than the generic score retrieved directly from the ASR engine even though the EER is the same. Note that to increase NCE, the lowest confidence score value is increased to the [0.4, 0.5] and [0.5, 0.6] buckets for the VM and C&C datasets, respectively, with the -W-C setting.

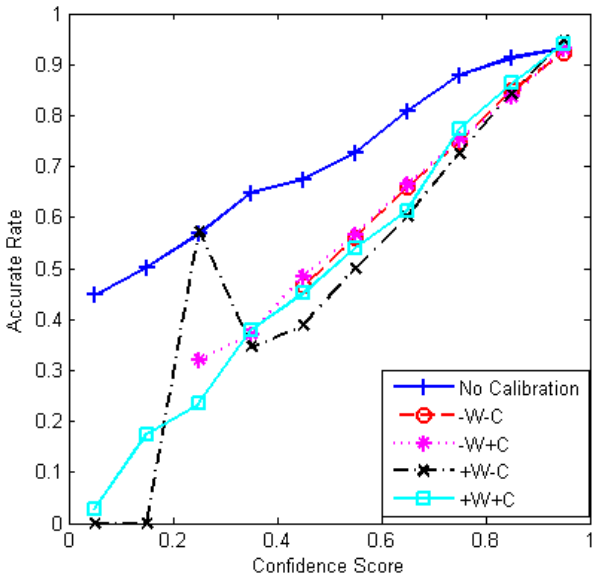


Fig. 2. The relationship between the WCM and the accurate rate for the VM test set where the calibrated results are from the DBNs. The curve is similar when ANN and MaxEnt-DC are used.

From Fig. 4 and Fig. 5, where the quality of the calibrated confidence scores are compared using the DET curves, we can observe that the DET curve with the -W-C setting overlap with the one without calibration. This is an indication that the quality of the confidence is not improved from the decision point of view, which is the most important aspect of the confidence measure for the speech application developers. Note that approaches such as piece-wise linear mapping [48] can also improve NCE but cannot improve EER and the DET curves since exploiting additional features is difficult using these techniques. If no additional feature is used (i.e., the –W-C setting), the piece-wise linear mapping approach can improve NCE to 0.089 with EER and DET unchanged. Due to page limit we only displayed the curves for the VM dataset with the DBN approach, and curves for the C&C dataset with the MaxEnt-DC approach. However, these curves are representative and similar curves can be observed using other approaches we proposed and compared in this paper.

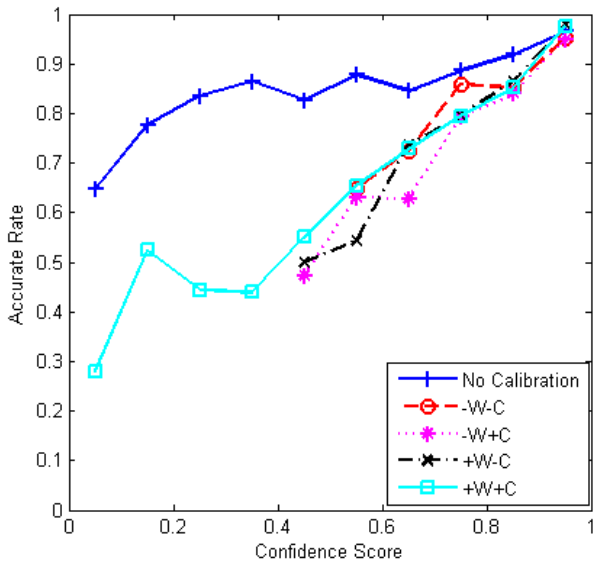


Fig. 3. The relationship between the WCM and the accurate rate for the C&C test set where the calibrated results are from the MaxEnt-DC model. The curve is similar when ANN and DBNs are used.

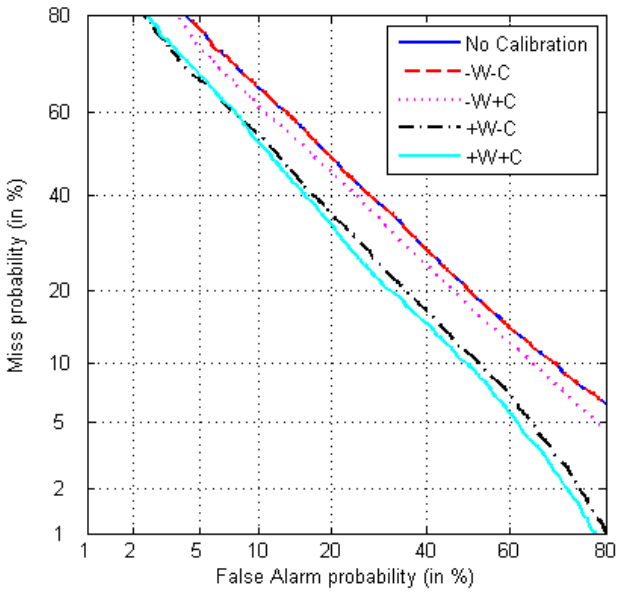


Fig. 4. Comparison of different settings (features) using the DET curves on the VM test set where the calibrated confidence scores are generated with DBNs.

We can slightly improve the quality of the calibrated confidence when the information from the adjacent words are used as shown in TABLE II, TABLE III, Fig. 4, and Fig. 5. However, the gain is very small, e.g., EER improves from 33.8% to 31.8% and NCE improves from 0.104 to 0.130 on the VM test set when the DBN approach is used. The biggest gain is from using the word distribution features. As can be seen from the tables and figures that the +W+C setting outperforms the -W+C setting with the improvement of the NCE from 0.130 to 0.255 and the EER from 31.8% to 26.1% on the VM dataset using the DBN approach, and the NCE from 0.129 to 0.209 and EER from 30.2% to 21.2% on the C&C dataset using the MaxEnt-DC approach. The gain can be clearly observed from

the big gap between the dotted pink line and the solid cyan line in Fig. 4 and Fig. 5. This behavior can also be observed from Fig. 2 and Fig. 3 by noticing that the calibrated confidence scores under the +W+C setting (the solid cyan line) now covers the full [0, 1] range while still aligning reasonably well with the diagonal line.

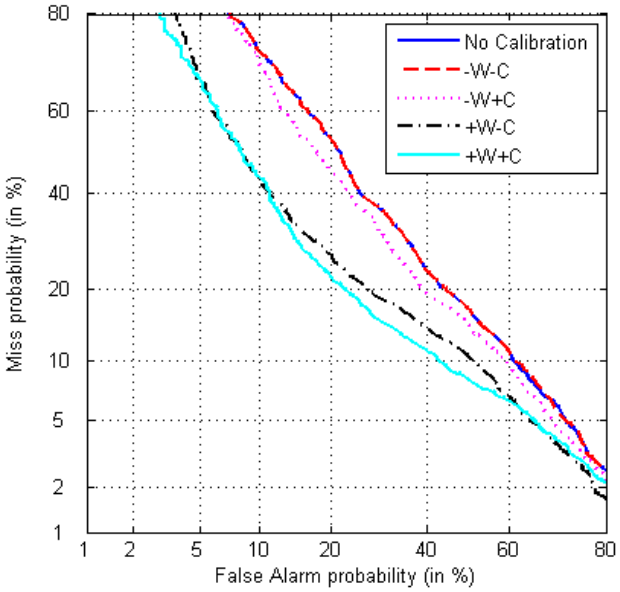


Fig. 5. Comparison of different settings (features) using the DET curves on the C&C test set where the calibrated confidence scores are generated with the MaxEnt-DC model.

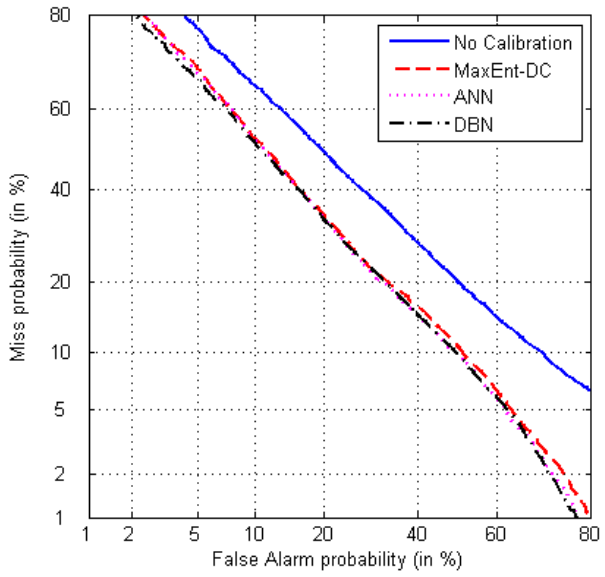


Fig. 6. Comparison of different approaches using the DET curves on the VM test set when both the word distribution and context information is used.

The performance of different calibration approaches can be compared using the DET curves shown in Fig. 6 and Fig. 7, where both the word distribution and context information are used. From Fig. 6 we can see that MaxEnt-DC, ANN, and DBN approaches perform similarly on the VM dataset, although ANN slightly underperforms other approaches if we look closer. However we can see clearly that MaxEnt-DC and DBN have similar performance. The same conclusion also holds when NCE is used as the criterion as shown in TABLE II and TABLE III. For example, using MaxEnt-DC and DBN we can achieve 0.209 and 0.212 NCE respectively while only 0.169 NCE is obtained using ANN.

Please note that the calibrated confidence measure can be further calibrated using the same features and techniques. However, the gain obtained with the second calibration step is typically small and no significant gain can be observed with the third calibration step. For example, on the VM task with +W+C setting and MaxEnt-DC model, the second step calibration only brings NCE from 0.250 to 0.261 and EER from 26.1% to 26.2% since the same information has been well exploited in the first step calibration.

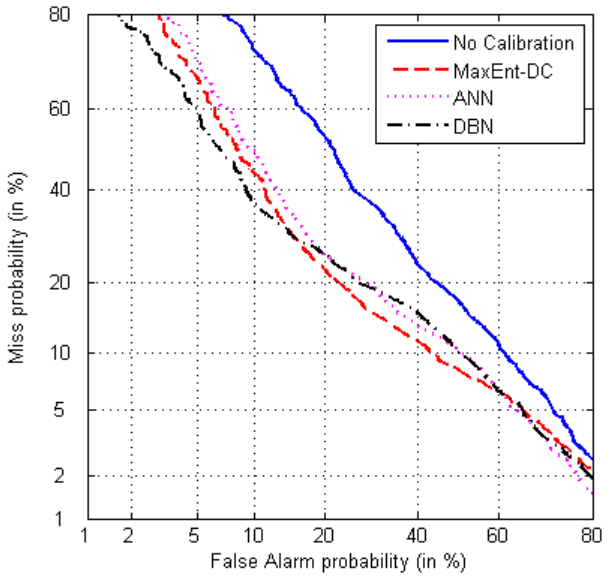


Fig. 7. Comparison of different approaches using the DET curves on the C&C test set when both the word distribution and context information is used.

TABLE V
WORD CONFIDENCE CALIBRATION RESULTS ON THE COMMAND AND CONTROL TASK WITH DIFFERENT CALIBRATION SET SIZE WHERE WORD COUNT THRESHOLD IS SET TO 20 AND BOTH WORD DISTRIBUTION AND CONTEXT INFORMATION ARE USED

| Settings | C&C | | | |
|---|---|---|---|---|
| | # words | J | NCE | EER (%) |
| **No Calibration** | 0K | 0 | -0.455 | 32.7 |
| **+W+C** | 2K (0.5 hr) | 19 | 0.133 | 30.6 |
| **+W+C** | 4K (1 hr) | 34 | 0.164 | 27.7 |
| **+W+C** | 7.5K (2 hrs) | 55 | 0.183 | 24.1 |
| **+W+C** | 15K (4 hrs) | 109 | 0.209 | 21.2 |

In TABLE V and Fig. 8 we compare the word confidence calibration results on the C&C dataset using the MaxEnt-DC approach but with calibration sets of different sizes (words). It is clear that some improvement can be obtained even with only 2K words of calibration data and the quality of the confidence measure continues to improve as the size of the calibration set increases. The same curve for the VM task is shown in Fig. 9. To obtain these results we have fixed $\theta = 20$ and so the

number of tokens $J$ increases automatically as more calibration data is available. By tuning $\theta$ better improvements can be achieved, especially when fewer calibration data are available, but the main trend remains.

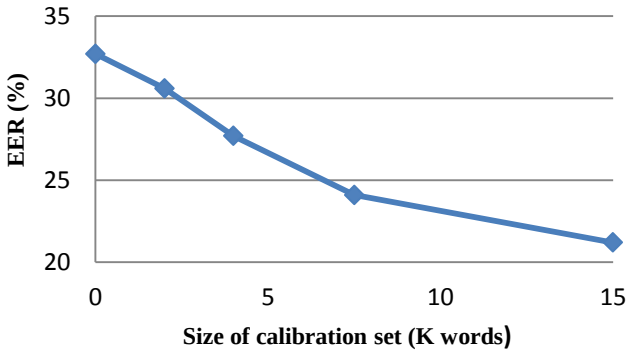


Fig. 8. The EER on the C&C reduces as the size of the calibration set increases. The results are obtained with the MaxEnt-DC approach using both the word distribution and context information.

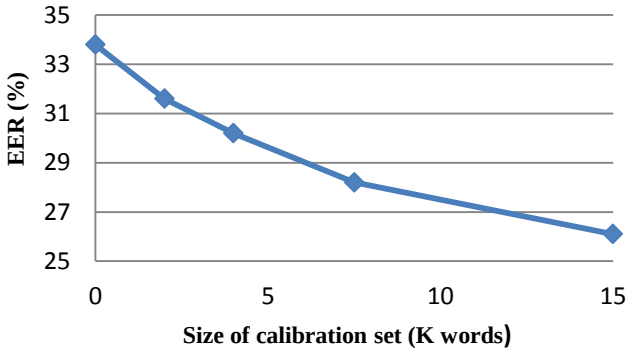


Fig. 9. The EER on the VM task reduces as the size of the calibration set increases. The results are obtained with the MaxEnt-DC approach using both the word distribution and context information.

TABLE VI
WORD CONFIDENCE QUALITY COMPARISON WITH MATCHED AND MISMATCHED CALIBRATION SET ON THE VOICE MAIL DATASET

| Features | MaxEnt-DC | | ANN | | DBNs | |
|---|---|---|---|---|---|---|
| | NCE | EER% | NCE | EER% | NCE | EER% |
| **No Calibration** | -0.264 | 33.8 | -0.264 | 33.8 | -0.264 | 33.8 |
| **Mismatched** | 0.204 | 28.4 | 0.198 | 28.7 | 0.209 | 28.3 |
| **Matched** | 0.250 | 26.1 | 0.243 | 25.6 | 0.255 | 26.1 |

TABLE VI demonstrates the calibration performance with matched and mismatched calibration sets using the +W+C setting. The mismatched VM calibration set was not collected under the real usage scenario. Instead, it was collected under the controlled data collection sessions and thus the environment and vocabulary can be very different. For example, the top ten most frequent words in the mismatched calibration set are *you*, *I*, *to*, *and*, *the*, *a*, *that*, *is*, *in*, and *it*, which are different from the list shown in TABLE I. To do a fair comparison we used the same calibration set size for both cases. We can see from TABLE VI that although mismatched calibration set was used, significant quality boost can still be obtained from confidence calibration although the gain is not as big as that achievable when the matched calibration set is used.

### B. *Semantic Confidence Calibration*

To better understand the property of our calibration technique, we have also conducted experiments on the important voice search (VS) dataset collected under the real usage scenario and have run experiments on both Engine E1 and Engine E2. As we point out earlier Engine E2 generates significantly better generic confidence measures than Engine E1 and is the best engine we have access to from the confidence point of view. TABLE VII summaries the information of voice search dataset. The vocabulary size for this task is 120K. The word error rate is 28.2% and 26.7% on the test set for the Engine E1 and E2 respectively. The LM perplexity of the VS task is 137 for the calibration set.

TABLE VII
SUMMARY OF THE VOICE SEARCH DATASET

| | # utterances | # words | Sem acc E1 | Sem acc E2 |
|---|---|---|---|---|
| **train** | 44K (33 hrs) | 120K | 64.7% | 65.3% |
| **dev** | 22K (16 hrs) | 60K | 64.7% | 65.3% |
| **test** | 22K (16 hrs) | 60K | 64.7% | 65.3% |

TABLE VIII
SEMANTIC CONFIDENCE QUALITY COMPARISON WITH AND WITHOUT THE KEYWORD COVERAGE INFORMATION ON THE VOICE SEARCH DATASET

| Engine E1 | MaxEnt-DC | | ANN | | DBNs | |
|---|---|---|---|---|---|---|
| | NCE | EER% | NCE | EER% | NCE | EER% |
| **No Calibration** | 0.549 | 11.2 | 0.549 | 11.2 | 0.549 | 11.2 |
| **+W-RCR** | 0.700 | 9.3 | 0.695 | 9.4 | 0.713 | 9.4 |
| **+W+RCR** | 0.755 | 7.5 | 0.702 | 7.7 | 0.757 | 7.6 |

+W and +RCR indicate the word distribution and the rule coverage ratio (RCR) feature are used, respectively.

TABLE IX
SEMANTIC CONFIDENCE QUALITY COMPARISON WITH AND WITHOUT THE KEYWORD COVERAGE INFORMATION ON THE VOICE SEARCH DATASET

| Engine E2 | MaxEnt-DC | | ANN | | DBNs | |
|---|---|---|---|---|---|---|
| | NCE | EER% | NCE | EER% | NCE | EER% |
| **No Calibration** | 0.736 | 8.4 | 0.736 | 8.4 | 0.736 | 8.4 |
| **+W-RCR** | 0.775 | 7.8 | 0.758 | 7.8 | 0.800 | 6.7 |
| **+W+RCR** | 0.799 | 6.4 | 0.762 | 6.7 | 0.802 | 6.5 |

+W and +RCR indicate the word distribution and the rule coverage ratio (RCR) feature are used, respectively.

TABLE VIII and TABLE IX compare the performance of different confidence calibration techniques on the voice search dataset with different features using Engine E1 and E2 respectively. A setting with and without using the calibrated word confidence scores is denoted as ±W where + sign means the feature is used and – sign means it is not used. Similarly, a setting with and without using RCR is denoted as ±RCR. From the tables we observe that both the calibrated word confidence score and RCR contribute to the improvement of the calibrated semantic confidence measures. The improvement is reflected by relative EER reductions of 32%, 16%, and 24%, 12% with and without using RCR over the generic SCM obtained using the engines E1 and E2, respectively, with the MaxEnt-DC approach. Similar gain is obtained using ANN and DBN approaches. The improvements can also be observed from the DET curves in Fig. 10 and Fig. 11. It can also be noticed that MaxEnt-DC only slightly underperforms DBN on the VS dataset and outperforms the ANN approach significantly. Note

that the confidence calibrated using the ANN approach (dash-dotted black line) without using the RCR feature has even worse quality than the one directly from Engine E2 (solid blue line) for a large part of the operation range as shown in Fig. 11. This is another sign that the ANN approach does not perform as well as other approaches on many datasets. The fact that ANN typically performs no better than DBN is well known (e.g., [54]). This is because the ANN weights are typically less well initialized compared to the DBN weights and the ANN typically uses fewer (usually one) hidden layers. We believe ANN underperforms the MaxEnt-DC on many datasets because MaxEnt-DC has better generalization ability. Although not reported in this paper, we have observed similar improvements consistently across a number of other datasets and semantic slots.

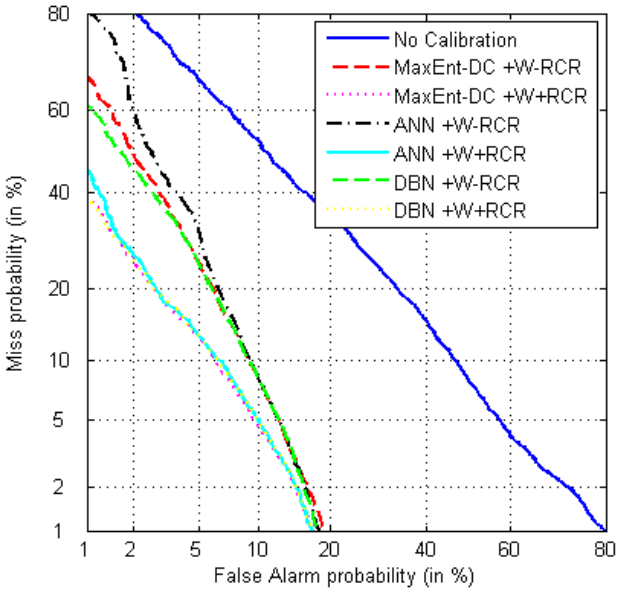


Fig. 10. The DET curve for the voice search dataset using engine E1.

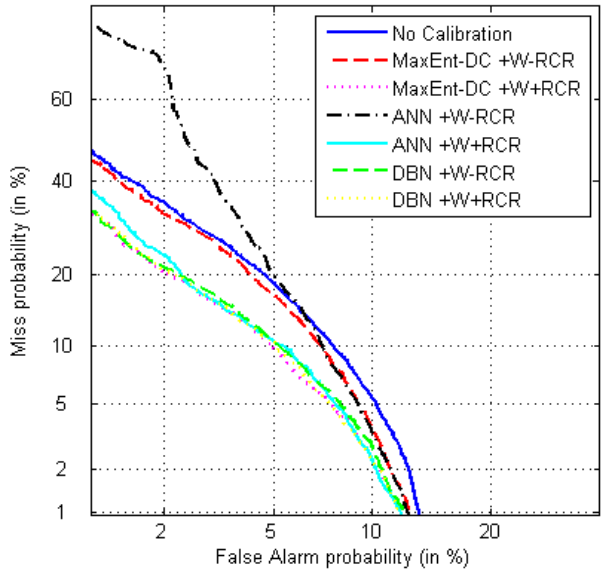


Fig. 11. The DET curve for the voice search dataset using engine E2.

## VII. Discussion and Conclusions

We have described a novel confidence-measure calibration technique based on the MaxEnt-DC model [25], ANN and DBN for improving both WCM and SCM for speech recognition applications. We have shown that by utilizing the information carried within the generic confidence measure, word distribution, and rule coverage ratio we can significantly increase the quality of the confidence measures. This is achieved without accessing any internal knowledge of how the confidence measure is generated in the ASR engines. Our findings above have high practical value to the speech application developers who typically do not have access to the internal information of the ASR engine. We have demonstrated the effectiveness of our approach on several different datasets and two different ASR engines. The significant performance gain as reported in Section VI is attributed both to the novel features we described in the paper and to the calibration algorithms proposed. Among the three techniques compared in this paper, DBNs often provide the best calibration result, but is only slightly better than the MaxEnt-DC approach, with the highest computational cost. MaxEnt-DC is a good compromise between the calibration quality, the implementation cost, and the computational cost, and is recommended for most tasks.

In this study we have used NCE, EER and DET as the confidence quality measures. We would like to point out that the criterion of EER used in this paper only measures the performance at one operation point and so it has well known limitations. The conclusions drawn from using EER alone may not be consistent with those drawn from using other criteria. Practitioners should select the criteria that best fit their need including recall/precision/F-measure that have not been discussed in this paper. In addition, many other features, esp. those that are specific to the application, may be developed and exploited to further improve the confidence quality. We leave this for the future work.

Finally we would like to point out that getting enough calibration data is not an issue nowadays. For example, at the early stage of the speech application development, we can release the service to a small portion of the users and obtain thousands of utterances per month easily. Once the service is fully deployed we can collect hundreds or even thousands of hours of speech data per month. By using the newly collected data, we can enable the feedback loop and thus continually improve the performance of the confidence measures.

## Acknowledgment

We would like to thank Michael Levit, Wei Zhang, Pavan Karnam, and Nikko Strom at Microsoft Corporation for their assistance in preparing experimental data and designing experiments. Thanks also go to Drs. Yifan Gong, Jian Wu, Shizhen Wang and Alex Acero at Microsoft Corporation, Prof. Chin-Hui Lee at Georgia Institute of Technology, and Dr. Bin Ma at Institute for Infocomm Research ($I^2R$), Singapore for valuable discussions.

## REFERENCES


[1] Y.-F. Gong, "Speech recognition in noisy environments: A survey", *Speech Communication*. 16, 261–291, 1995.

[2] Y.-Y. Wang, D. Yu, Y.-C. Ju, and A. Acero. "An Introduction to Voice Search," *IEEE Signal Processing Magazine*, vol. 25, no. 3, pp. 28-38, May 2008.

[3] D. Yu, Y.-C. Ju, Y.-Y. Wang, G. Zweig, A. Acero. "Automated directory assistance system - From theory to practice," in Proc. *Interspeech*, pp. 2709-2712, 2007.

[4] J. Wilpon, L. Rabiner, and C.-H., Lee, "Automatic recognition of keywords in unconstrained speech using hidden Markov models", *IEEE Trans. ASSP* vol. 38, pp. 1870-1878, 1990.

[5] D. Yu, Y.-C. Ju, Y.-Y. Wang, A. Acero. "N-gram based filler model for robust grammar authoring", in Proc. *ICASSP*, vol. I, pp. 565-568, 2006.

[6] H. Jiang, "Confidence measures for speech recognition: a survey," *Speech Communication*, vol. 45, no. 4, pp. 455-470, Apr. 2005.

[7] R.A. Sukkar, "Rejection for connected digit recognition based on GPD segmental discrimination", in Proc. *ICASSP*, pp. I-393–I-396, 1994.

[8] R.A. Sukkar, C.-H. Lee, "Vocabulary independent discriminative utterance verification for nonkeyword rejection in subword based speech recognition", *IEEE Trans. Speech Audio Process*. Vol. 4, no. 6, pp. 420–429, 1996.

[9] L. Gillick, Y. Ito, J. Young, "A probabilistic approach to confidence estimation and evaluation", in Proc. *ICASSP*, pp. 879–882, 1997.

[10] M. Siu, H. Gish, H., "Evaluation of word confidence for speech recognition systems", *Computer Speech Language*, vol.13, pp. 299–319, 1999.

[11] B. Chigier, "Rejection and keyword spotting algorithms for a directory assistance city name recognition application", in Proc. *ICASSP*, pp. II-93–II-96, 1992.

[12] L. Mathan, L., Miclet, "Rejection of extraneous input in speech recognition applications, using multi-layer perceptrons and the trace of HMMs", in Proc *ICASSP*, pp. 93–96, 1991.

[13] M. Weintraub, F. Beaufays, Z. Rivlin, Y. Konig, A. Stolcke, "Neural-network based measures of confidence for word recognition", in Proc. *ICASSP*, pp. 887–890, 1997.

[14] E. Eide, H. Gish, P. Jeanrenaud, A. Mielke, "Understanding and improving speech recognition performance through the use of diagnostic tools", in Proc. *ICASSP*, pp. 221–224, 1995.

[15] C.V. Neti, S. Roukos, E. Eide, "Word-based confidence measures as a guide for stack search in speech recognition", in Proc. *ICASSP*, pp. 883–886, 1997.

[16] P.J. Moreno, B. Logan, B. Raj, "A boosting approach for confidence scoring", in Proc *EuroSpeech*, 2001.

[17] C. White, J. Droppo, A. Acero, and J. Odell, "Maximum entropy confidence estimation for speech recognition," in Proc. *ICASSP*, vol. IV, pp. 809-812, 2007.

[18] T. Kemp, T. Schaaf, "Estimating confidence using word lattices", in Proc. *EuroSpeech*, pp. 827–830, 1997.

[19] F. Wessel, K. Macherey, R. Schluter, "Using word probabilities as confidence measures", in Proc. *ICASSP*, pp. 225–228, 1998.

[20] F. Wessel, K. Macherey, H. Ney., "A comparison of word graph and N-best list based confidence measures", in Proc. *EuroSpeech*, pp. 315–318, 1999.

[21] F. Wessel, R. Schluter, K. Macherey, H. Ney, "Confidence measures for large vocabulary continuous speech recognition", *IEEE Trans. Speech Audio Process*, vol. 9, no. 3, pp. 288–298, 2001.

[22] B. Rueber, B., "Obtaining confidence measures from sentence probabilities", in Proc. *EuroSpeech*, 1997.

[23] R.C. Rose, B.H. Juang, C.H. Lee, "A training procedure for verifying string hypothesis in continuous speech recognition", in Proc. *ICASSP*, pp. 281–284, 1995.

[24] M.G. Rahim, C.-H. Lee, B.-H. Juang, "Discriminative utterance verification for connected digits recognition", *IEEE Trans. on Speech and Audio Processing* vol. 5, no. 3, pp. 266–277, 1997.

[25] D. Yu, L. Deng, and A. Acero, "Using continuous features in the maximum entropy model", *Pattern Recognition Letters*. vol. 30, no. 8, pp.1295-1300, June, 2009.

[26] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki, "The DET curve assessment of detection task performance," in Proc. *EuroSpeech*, vol. 4, pp. 1895-1898, 1997.

[27] S. Guiasu, and A. Shenitzer, "The principle of maximum entropy", The *Mathematical Intelligencer*, vol. 7, no. 1, 1985.

[28] A.L. Berger, S.A. Della Pietra, and V.J. Della Pietra, "A maximum entropy approach to natural language processing", *Computational Linguistics*, vol. 22, pp. 39-71, 1996

[29] C. Ma, P. Nguyen and M. Mahajan, M., "Finding Speaker Identities with a Conditional Maximum Entropy Model", In proc. *ICASSP*, vol. IV, pp. 261-264, 2007.

[30] R. Rosenfeld, "A maximum entropy approach to adaptive statistical language modeling", *Computer Speech and Language*, 10:187-228, 1996.

[31] D. Yu, M. Mahajan, P. Mau, and A. Acero, A., "Maximum entropy based generic filter for language model adaptation," in proc. *ICASSP* 2005, vol. I, pp. 597-600, 2005.

[32] F. J. Och, and H. Ney, "Discriminative training and maximum entropy models for statistical machine translation", in proc. *ACL*, pp. 295-302, 2002.

[33] D. Yu, L. Deng, Y. Gong, and A. Acero, "A novel framework and training algorithm for variable-parameter hidden Markov models," *IEEE trans. on Audio, Speech, and Language Processing*, vol 17, no. 7, pp. 1348-1360, September 2009.

[34] D. Yu, and L. Deng, "Solving nonlinear estimation problems using Splines," *IEEE Signal Processing Magazine*, vol. 26, no. 4, pp.86-90, July, 2009.

[35] D. Yu, L. Deng, and A. Acero, "Hidden conditional random field with distribution constraints for phonetic classification," in Proc. *Interspeec*, pp. 676-679, 2009.

[36] M. Riedmiller and H. Braun. "A direct adaptive method for faster back-propagation learning: The RPROP algorithm," in Proc. *IEEE ICNN*, vol. 1, pp. 586-591. 1993.

[37] D. van Leeuwen and N. Brümmer. "On calibration of language recognition scores," in Proc. *IEEE Odyssey: The Speaker and Language Recognition Workshop*, 2006.

[38] R. Malouf, "A comparison of algorithms for maximum entropy parameter estimation", in proc. *CoNLL*, vol. 2, pp. 1-7, 2002.

[39] J. Nocedal, J., "Updating quasi-Newton matrices with limited storage", *Mathematics of Computation*, vol. 35, pp. 773-782, 1980.

[40] S. F. Chen, and R. Rosenfeld, "A Gaussian prior for smoothing maximum entropy models", In Technical Report CMU-CS-99-108, *Carnegie Mellon University*, 1999.

[41] S. F. Chen, and R. Rosenfeld, "A survey of smoothing techniques for ME models. IEEE Transactions on Speech and Audio Processing, vol. 8, no. 1, pp. 37-50, 2000.

[42] J. Goodman, "Exponential priors for maximum entropy models", in Proc. *HLT-NAACL*. pp. 305-311, 2004.

[43] J. Kazama, J., "Improving maximum entropy natural language processing by uncertainty-aware extensions and unsupervised learning", Ph.D. thesis, *University of Tokyo*, 2004.

[44] J. Kazama, and J. Tsujii, "Maximum entropy models with inequality constraints: A case study on text categorization", *Machine Learning*, vol. 60, no. 1-3 , pp. 159 – 194, 2005.

[45] D. Yu, S. Wang, Z. Karam, L. Deng, "Language recognition using deep-structured conditional random fields", in Proc. *ICASSP* 2010.

[46] D. Yu, S. Wang, J. Li, L. Deng, "Word confidence calibration using a maximum entropy model with constraints on confidence and word distributions," in Proc. *ICASSP* 2010.

[47] D. Yu, L. Deng, "Semantic confidence calibration for spoken dialog applications", in Proc. *ICASSP* 2010.

[48] G. Evermann, and P. Woodland, "Large vocabulary decoding and confidence estimation using word posterior probabilities", In Proc. *ICASSP* 2000.

[49] A. Stolcke, H. Bratt, J. Butzberger, H. Franco, V. R. Rao Gadde, M. Plauché, C. Richey, E. Shriberg, K. Sonmez, and J. Zheng, F. Weng, "The SRI March 2000 Hub-5 conversational speech transcription system", In Proc. *Speech Transcription Workshop*, 2000.

[50] http://www.icsi.berkeley.edu/Speech/docs/sctk-1.2/sclite.htm

[51] J. Xue and Y. Zhao, "Random forests-based confidence annotation using novel features from confusion network," In Proc. *ICASSP*, vol. I, pp. 1149-1152, 2006.

[52] D. Hillard and M. Ostendorf, "Compensating Forword Posterior Estimation Bias in Confusion Networks", In Proc. *ICASSP*, vol. I, pp. 1153-1156, 2006.

[53] R. Sarikaya, Y. Gao, M. Picheny, and H. Erdogan, "Semantic confidence measurement for spoken dialog systems," *IEEE trans. on Audio, Speech, and Language Processing*, vol. 13, no. 4, pp. 534-545, 2005.

[54] G. Hinton, S. Osindero and Y. Teh, "A Fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, 2006, pp. 1527-1554.

[55] G. Hinton, and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313. no. 5786, 2006, pp. 504 – 507.

[56] A.-R. Mohamed, G. Dahl, G. Hinton, "Deep Belief Networks for Phone Recognition," in Proc. *NIPS Workshop*, Dec. 2009.

[57] A.-R. Mohamed, D. Yu, and L. Deng, "Investigation of full-sequence training of deep belief networks for speech recognition," in Proc. *Interspeech* 2010.

[58] A. Acero, N. Bernstein, R. Chambers, Y. Ju, X. Li, J. Odell, P. Nguyen, O. Scholtz, and G. Zweig, "Live search for mobile: Web services by voice on the cellphone," in Proc. *ICASSP*, 2008, pp. 5256–5259.