PHONETIC SPEAKER RECOGNITION USING MAXIMUM-LIKELIHOOD BINARY-DECISION TREE MODELS

Jiří Navrátil[1] *Qin Jin*[2] *Walter D. Andrews*[3] *Joseph P. Campbell*[4]

[1]IBM T.J. Watson Research Center, Yorktown Heights, NY 10598, jiri@us.ibm.com
[2]Carnegie Mellon University, NSH 2602J, Pittsburgh, PA 15213, qjin@cs.cmu.edu
[3]Department of Defense, Ft. Meade, MD 20755, waltandrews@ieee.org
[4]MIT Lincoln Laboratory, Lexington, MA 02420, jpc@ll.mit.edu

ABSTRACT

Recent work in phonetic speaker recognition has shown that modeling phone sequences using n-grams is a viable and effective approach to speaker recognition, primarily aiming at capturing speaker-dependent pronunciation and also word usage. This paper describes a method involving binary-tree-structured statistical models for extending the phonetic context beyond that of standard n-grams (particularly bigrams) by exploiting statistical dependencies within a longer sequence window without exponentially increasing the model complexity, as is the case with n-grams. Two ways of dealing with data sparsity are also studied; namely, model adaptation and a recursive bottom-up smoothing of symbol distributions. Results obtained under a variety of experimental conditions using the NIST 2001 Speaker Recognition Extended Data Task indicate consistent improvements in equal-error rate performance as compared to standard bigram models. The described approach confirms the relevance of long phonetic context in phonetic speaker recognition and represents an intermediate stage between short phone context and word-level modeling without the need for any lexical knowledge, which suggests its language independence.

1. INTRODUCTION

In recent years, the research area of automatic speaker recognition has seen an increased interest in utilizing sources of high-level speaker discriminative information in order to complement widely and successfully used frame-by-frame approaches exploiting only short-time acoustic information from the speech signal. Motivated by the work of Doddington [3] on modeling idiolectal differences among speakers by means of word n-grams, Andrews et al. [1] investigated n-gram modeling of phonetic units in sequences automatically obtained from multiple phone recognizers for speaker verification. The results of this work indicate that such models effectively capture speaker characteristics complementary to the short-time acoustic information that relate particularly to speaker-specific pronunciation of words as well as word idiolect. Due to the fact that the phone recognizers do not apply any constraints on the search space during decoding (such as grammars or pronunciation baseforms), pronunciation differences propagate through the decoder and are reflected in variations of phones and their ordering. Viewed statistically, for example in terms of n-gram probabilities, these variations can be observed as varying statistical dependencies between phone tokens in the sequence and offer themselves for speaker-dependent modeling. Performance presented in [1] suggest that the dependencies indeed carry a substantial speaker-dependent component.

Phonetic speaker modeling using n-grams, however, comes with a burden: in order to capture dependencies within a reasonably long time window, the model order needs to be chosen correspondingly high, incurring an exponential growth in the number of parameters. This leaves three solutions: 1) provide sufficiently large amounts of training data for each speaker, 2) decrease the model order, or 3) use smoothing techniques. Smoothing techniques have been extensively used in n-gram modeling; however, the model order in practice is still limited to 2 (i.e., trigrams) or 1 (bigrams). Another weakness of the n-gram model is its rigid structure, i.e., the way contexts (or histories) of the modeled tokens are partitioned. For example, while certain phones preceding a token may belong to a common category and hence would ideally be members of the same (history) partition, the n-gram assigns separate partitions for these alone because of their different labeling.

In this paper, we introduce a binary-tree modeling approach applied to phonetic speaker recognition that allows for exploiting dependencies from longer contexts than that of typical n-grams while keeping the number of free parameters under control. This binary-decision tree structure is optimized using a maximum-likelihood training criterion and provides flexible context clustering. Tree structured models were successfully applied in language and speech recognition previously [6, 2]. To deal with limited training data and robustness issues, we also introduce an adaptation step in creating the tree models as well as a recursive smoothing technique.

2. BASELINE SYSTEM

N-grams are a standard language modeling technique that approximates the probability of occurrence of a spoken utterance A represented by a sequence of tokens (in our case decoded phones) $a_1, ..., a_T$ up to the $(N-1)$-th order. Thus, bigram models $(N = 2)$ imply the assumption that the probability occurrence of a token depends solely on the immediately preceding word. Due to the fact that the n-gram model complexity increases exponentially with the order, we restrict our considerations to bigrams and trigrams for both the speaker and the background models.

The baseline system is a basic log-likelihood ratio detector. Five language- and gender-dependent open-loop phonetic recognizers are used to generate multiple language phone sequences that represent multiple views of the input speech signal [1]. Phonetic speaker recognition is performed in three steps. First, the five phone recognizers process the test speech utterance to produce multiple phonetic sequences. Then, the test sequence from each phone recognizer is compared to the hypothesized speaker model and a speaker-independent Universal Background Phone Model (UBPM), corresponding to the appropriate phone recognizer [1, 4]. Finally, the scores from the hypothesized speaker models and the UBPM are combined to form log-

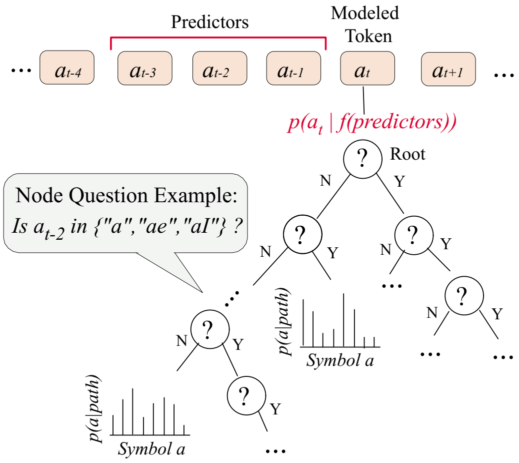
Predictors
Modeled Token
... a_{t-4} a_{t-3} a_{t-2} a_{t-1} a_t a_{t+1} ...
$p(a_t \mid f(predictors))$
Root
N
Y
Node Question Example:
Is a_{t-2} in {"a","ae","aI"} ?
p(a|path)
Symbol a
p(a|path)
Symbol a

Figure 1. An example structure of a binary tree model

likelihood ratio (LLR) scores, again corresponding to each phone recognizer. The five LLR scores are then fused together producing a single weighted score. The LLR score of A, given a hypothesized speaker model M and a single phone recognizer, is calculated as

$$LLR(A|M) = \log P(A|M) - \log P(A|UBPM) \quad (1)$$

3. BINARY-DECISION TREE MODELING

Let us consider a token sequence $a_1, ..., a_T$ representing a decoded utterance of a speaker and a particular token a_t in that sequence. The quality of a model with respect to a_t is measured by its power in predicting a_t from a certain context – represented by a set of predictor variables X. These may be chosen according to some prior knowledge and are typically selected to be the N time slots preceding a_t; i.e., $a_{t-N}, ..., a_{t-1}$. We now seek a model with a good overall quality in predicting individual tokens from their respective contexts. For this purpose, we apply binary-decision trees (BT) which provide a versatile and flexible structure. Figure 1 illustrates the function of such a model consisting of nonterminal nodes associated with a binary question leading to either of two child nodes and terminal nodes (leaves) that contain symbol distributions. Certain selected N time-slots of the phone sequence are denoted *predictors*, $X_1, ..., X_N$ and are taken into account in the binary questions. The probability of a token a_t given its context can be obtained from the BT model by successively using appropriate predictor values to answer the binary questions at each node until a leaf node with a symbol distribution is reached, as exemplified in Figure 1. Obviously, for a given sequence the predictor values determine the path through the tree structure and thus effectively determine the distribution to be used for a_t. Hereby a *variable* context clustering is easily achieved by including multiple predictor values into the subsets at each node.

To determine the tree structure and parameters, some applications, such as acoustic context modeling in speech recognition [7], are motivated by linguistically based schemes designed by an expert. Because the BT structure can vary from speaker to speaker and no straightforward rules can be determined for phone sequences a priori for speaker modeling, a fully data-driven BT building algorithm appears necessary. We seek to create a speaker model with the objective of attaining a high average prediction power which is expressed by means of average prediction entropy of the BT leaf distributions. Here, low entropy, e.g. predicting unique symbols, corresponds to the desirable prediction property and vice versa (e.g. predicting all symbols with same probability). Defining the entropy for a distribution of a symbol set $\mathcal{A}$ at a leaf l as

$$H_l = -\sum_{s_i \in \mathcal{A}} P_l(s_i) \log_2 P_l(s_i) \quad (2)$$

the average BT prediction entropy is then

$$\overline{H} = \sum_l P_l \cdot H_l \quad (3)$$

with P_l denoting the prior probability of visiting the leaf l, and $P_l(s)$ the probability of observing a symbol s at that leaf. The measure (3) is to be minimized in the course of building the BT model. During this process the probabilities $P_l(s_i)$ and P_l are not known and have to be replaced by estimates, $\hat{P}_l(s_i)$ and $\hat{P}_l$, obtained from a training sample $a_t, ..., a_T$. Assuming a BT model structure with certain parameters leads to partitioning of the training data into L leaves, each containing a data partition α_l, then the sample distribution estimates are calculated from counts as follows:

$$\hat{P}_l(s_i) = \frac{\#(s_i|\alpha_l)}{|\alpha_l|} \quad (4)$$

$$\hat{P}_l = \frac{|\alpha_l|}{\sum_{l=1}^{L} |\alpha_l|} \quad (5)$$

with $\#(s_i|\alpha_l)$ being the a_i count at leaf l, and $|\alpha_l|$ the total symbol count at l. On the other hand, the average training data likelihood given a BT model can be computed as follows:

$$\mathcal{L} = \frac{1}{T} \sum_{t=1}^{T} \log_2 P(a_t|BT) \quad (6)$$

$$= \sum_{l=1}^{L} \hat{P}_l \sum_{s_i \in \mathcal{A}} \hat{P}_l(s_i) \log_2 P_l(s_i) \quad (7)$$

Thus, by replacing $P_l(s_i)$ with the estimate (4), the measure (3) and (7) are in a relationship

$$\mathcal{L} = -H_l \quad (8)$$

Hence, building the BT model so as to minimize the overall prediction entropy identically maximizes the likelihood of the training data.

The remaining problem of finding an optimum tree structure and the corresponding node questions is solved by applying a greedy search algorithm at each node, combined with a recursive procedure for creating tree nodes. To limit the otherwise extensive search space, we restrict the binary questions to be elementary expressions involving a single predictor, rather than allowing for composite expressions. Stated in principal steps the tree building algorithm is as follows:

1. Let c be the current node of the tree. Initially c is the root.
2. For each predictor variable $X_i (i = 1, ..., N)$ find the subset $\mathbf{S}_i^c$ which minimizes the average conditional entropy of the symbol distribution Y at node c

$$\overline{H}_c(Y \mid \text{“}X_i \in \mathbf{S}_i^c\text{?”})$$
$$= -P(X_i \in \mathbf{S}_i^c \mid c) \sum_{s_j \in \mathcal{A}} P(s_j \mid c, X_i \in \mathbf{S}_i^c) \times$$

$$\times \log_2 P(s_j \mid c, X_i \in \mathbf{S}_i^c)$$

$$-P(X_i \notin \mathbf{S}_i^c \mid c) \sum_{s_j \in \mathcal{A}} P(s_j \mid c, X_i \notin \mathbf{S}_i^c) \times$$

$$\times \log_2 P(s_j \mid c, X_i \notin \mathbf{S}_i^c). \qquad (9)$$

where $\mathbf{S}_i^c$ denotes a subset of phones at node c.

3. Determine which of the N questions derived in Step 2 leads to the lowest entropy. Let this be question k, i.e.,

$$k = \arg\min_i \overline{H}_c(Y \mid \text{“}X_i \in \mathbf{S}_i^c\text{?”})$$

4. The reduction in entropy at node c due to question k is

$$R_c(k) = H_c(Y) - \overline{H}_c(Y \mid \text{“}X_k \in \mathbf{S}_k^c\text{?”}),$$

where

$$H_c(Y) = -\sum_{s_j \in A} P(s_j \mid c) \cdot \log_2 P(s_j \mid c).$$

If this reduction is “significant,” store question k, create two descendant nodes, c_1 and c_2, pass the data corresponding to the conditions $X_k \in \mathbf{S}_k^c$ and $X_k \notin \mathbf{S}_k^c$, and repeat Steps 2-4 for each of the new nodes separately.

Simply stated, the algorithm seeks a data split at each node such that the average entropy of the two data subsets due to that split significantly reduces the entropy of total data before the split. The entropy reduction is considered significant relative to some threshold effectively determining the size of the tree model to be grown.

In order to determine the phone subset $\mathbf{S}_i^c$ in Step 2 the following greedy algorithm was applied:

1. Let $\mathbf{S}$ be empty.
2. Insert into $\mathbf{S}$ the phone $a \in \mathcal{A}$ ($\mathcal{A}$ being the phonetic set) which leads to the greatest reduction in the average conditional entropy (9). If no $a \in \mathcal{A}$ leads to a reduction, make no insertion.
3. Delete from $\mathbf{S}$ any member a, if so doing leads to a reduction in the average conditional entropy.
4. If any insertions or deletions were made to $\mathbf{S}$, return to Step 2.

In addition to the significance criterion in Step 4 of the tree building we implement an occupancy constraint. Systematic data sparseness may occur with too low significance thresholds due to the recursive partitioning, thus leading to poor entropy estimates and consequently to overtraining. The occupancy constraint is applied in evaluating each potential split during the search, discarding split hypotheses not fitting this constraint. Furthermore, in order to prevent overtraining by modeling training data particularities, we apply cross-evaluation in Step 4 of the tree growing using a separate held-out set when computing the reduction $R_c(k)$.

3.1. Data Sparseness Issues

Applying the leaf occupancy constraint causes the BT models to grow adaptively with respect to not only the intrinsic data properties, but also the data set size. The latter may become a problem with small training data amounts for which the growing process may terminate with only a few leaf nodes, resulting in extremely coarse models.

Furthermore, even in sufficiently large training sets, sparseness of symbols *in certain contexts* may exist. In the following, we describe two approaches to mitigate these problems.

3.1.1. Leaf Adaptation

In case of sparse training data for an individual speaker, a speaker-independent (SI) BT model built from sufficient amounts of data can provide a robust tree structure (i.e., the nonterminal nodes) as a fixed basis for creating the speaker model by adaptation. Herein, the speaker training set is partitioned according to the fixed structure and the leaf distributions are updated using the new partitions. Let $Y_0 = \{\hat{P}_l(s_j)\}_{s_j \in \mathcal{A}}$ denote a leaf distribution estimate of the SI model, $\#(s_j|\alpha_l)$ the count of s_j tokens in the leaf partition α_l, and $|\alpha_l|$ the leaf token count of the speaker data. The updated leaf distribution $Y_1 = \{\hat{P}_l(s_j)'\}_{s_j \in \mathcal{A}}$ is then calculated as a linear interpolation

$$\hat{P'}_l(s_j) = \left[b_j \frac{\#(s_j|\alpha_l)}{|\alpha_l|} + (1 - b_j)\hat{P}_l(s_j) \right] / D \qquad (10)$$

with

$$b_j = \frac{\#(s_j|\alpha_l)}{\#(s_j|\alpha_l) + r} \qquad (11)$$

where D normalizes the adapted values to probabilities, and r is an empirical value controling the strength of the update. Such a BT model retains the context resolution of the SI model, while describing the speaker-specific statistics. This training scheme is particularly effective when the SI model is used at the same time as the background model during the likelihood ratio test in which symbols with too low an observation count in certain contexts nearly cancel out due to the identical tree structure.

3.1.2. Bottom-Up Recursive Smoothing

Despite sufficient token counts in a leaf overall, individual symbols with unreliable estimates may still exist. A symbolwise back-off or smoothing scheme with one or several reliable estimates may be beneficial. The BT framework offers a simple way of finding such estimates, namely by backing-off to the parent distribution of a leaf. Each parent distribution is a pool of both child distributions and therefore is more likely to contain more observations of a given symbol. The back-off process can be repeated recursively bottom up until either enough observation mass is collected or the root node is reached. We suggest the following recursive smoothing algorithm for calculating the probability of a symbol $a_t = s_j$ given its context $X = \{a_{t-N}, ..., a_{t-1}\}$:

1. Find the leaf l using X. Set a node variable $c = l$.
2. Calculate symbol probability $\hat{P}_{smooth}(s_j) = b_j \hat{P}_c(s_j) + (1 - b_j)\hat{P}_{par(c)}(s_j)$ where $\hat{P}_{par(c)}(s_j)$ is obtained by repeating Step 2 with $c := par(c)$ recursively until $c = root$.

Again, a linear interpolation scheme is used, whereby $par(c)$ denotes the parent node of c, and r is as in (11).

4. EXPERIMENTS

4.1. Database

The Extended Data Paradigm used in the framework of the NIST 2001 Speaker Recognition Evaluation was adopted in our experimental setup. As described in [8], the Extended Data Task comprises of the complete Switchboard-I telephone-speech corpus partitioned into six splits to evaluate the performance in a fashion similar to cross-validation. Furthermore, five different training conditions with data amounts consisting of 1, 2, 4, 8 and 16 conversation sides (each of nominal length of 2.5 minutes) were considered.

	# Training Conversations				
	16	8	4	2	1
Avg # of leaves	50	29	4	1	1

Table 1. Average speaker tree size for variable training amounts (no adaptation)

4.2. Baseline Performance

The baseline bigram system was implemented using the CMU Statistical Language Modeling toolkit (CMU-SLM). A smoothing scheme by Katz [5], which combines Good-Turing discounting with back-off, was used with a discounting threshold of 7. Two UBPMs were created from splits 1-3 and 4-6, each used in evaluation of the respective excluded partition sets. Furthermore, we include trigram baseline results obtained using a joint-probability system with pruning described in [1]. In this system, all trigrams with an observation count lower than 500 were excluded from the scoring. The final performance results of both systems were obtained by pooling all six splits and combining the five decoder-dependent streams with uniform weights as described in Section 2.

4.3. Binary Trees System

The speaker BT models were examined in three configurations: 1) Models with no smoothing, 2) with Bottom-Up Recursive Smoothing (BURS), and 3) Adapted from a background (BG) model with BURS. The BG BT model was created in the same fashion as for n-grams described in Section 4.2. The significance threshold was set such that the BG BT possessed on the order of 200-400 leaves. The same threshold along with an occupancy constraint of $5\cdot|\mathcal{A}| \approx 250$ produced unadapted BT models with an average of 30 leaves for 8-conversations training. In unadapted speaker models, the occupancy constraint appeared to be active in almost all split decisions, as opposed to the redundancy reduction which tended to be more active in building the BG model with large data amounts. Table 1 shows the average model size (leaf count) for the five training conditions with no adaptation. Lack of context resolution becomes obvious for 4 or fewer training conversations due to the occupancy constraint, compared to, for example, a bigram context resolution of 45 (for $|\mathcal{A}| = 45$). The value of the adaptation constant r in (11) seemed not critical in the range $(0.5, 16)$ and was set to 4 in all experiments, based on a small data subset.

The maximum number of predictors N considered in the training was set to four. Most of the BT models tended to use up to three preceding predictors, namely X_1, X_2, X_3 in such a way that X_1 (i.e. immediately preceding) tended to be chosen in splits earlier in the treee growing procedure to split the data set, followed by X_2 and then X_3 chosen deeper for more detailed split decisions.

Figure 2 compares the performance of BT models with and without adaptation and the bigram and trigram baselines in terms of the equal-error rate (EER) across training conditions. A considerable improvement in BT performance with adaptation can be seen for training conditions 4, 2, and 1, in which the resolution of unadapted models is insufficient (see Table 1). With 8 and 16 conversations, the BT models are able to further improve upon the trigrams due to their extended context length and more flexible structure.

5. CONCLUSION

Binary-tree models represent a step towards flexible context structuring and extension in phonetic speaker recognition, consistently outperforming standard smoothed bigrams as well as trigrams. Our experiments show that the problems of data sparseness in speaker model training can be addressed effectively by applying principles of adaptation and smoothing for which the BT models offer a suitable basis. Using smoothing and adaptation, a relative reduction in EER ranging between 10-60% compared to the best n-gram system was achieved across the different training conditions.

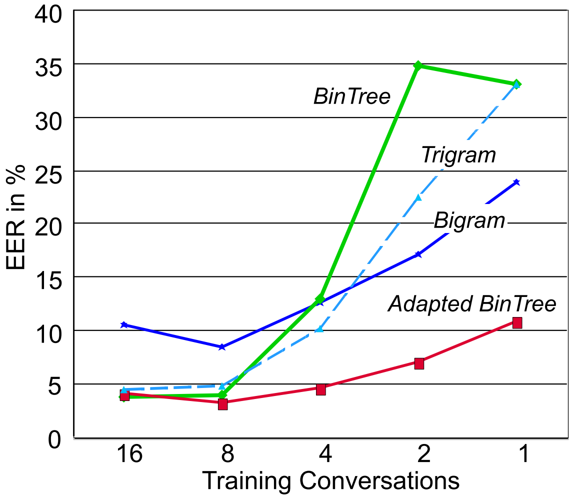

Figure 2. The EER performance of the 5-tokenizer system using BT models with and without adaptation and n-grams

REFERENCES

[1] W. Andrews, M. Kohler, J. Campbell, J. Godfrey, and J. Hernandez-Cordero. Gender-dependent phonetic refraction for speaker recognition. In *Proc. of the ICASSP*, Orlando, FL, May 2002. IEEE.

[2] L.R. Bahl, P.F. Brown, P.V. DeSouza, and R.L. Mercer. A tree-based statistical language model for natural language speech recognition. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 37(7):1001–8, July 1989.

[3] G. Doddington. Speaker recognition based on idiolectal differences between speakers. In *Proc. of the EUROSPEECH*, pages 2521–4, Aalborg, Denmark, September 2001.

[4] Q. Jin, J. Navrátil, D. Reynolds, J. Campbell, W. Andrews, and J. Abramson. Combining cross-stream and time dimensions in phonetic speaker recognition. ICASSP'03, to appear.

[5] S.M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *Trans. on Acoustics, Speech, and Signal Processing*, 35(3):400–401, 1987.

[6] J. Navrátil. Spoken language recognition - a step towards multilinguality in speech processing. *IEEE Trans. Audio and Speech Processing*, 9(6):678–85, September 2001.

[7] L. Polymenakos, P. Olsen, D. Kanevsky, R.A. Gopinath, P.S. Gopalakrishnan, and S.S. Chen. Transcription of broadcast news - some recent improvements to IBM's LVCSR system. In *Proc. of the ICASSP*, volume 2, pages 901–4, Seattle, May 1998.

[8] D. Reynolds et al. The SuperSID project: Exploiting high-level information for high-accuracy speaker recognition. ICASSP'03, to appear.