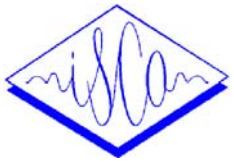

ODYSSEY04 - The Speaker and Language Recognition Workshop
Toledo, Spain
May 31 - June 3, 2004

Modeling NERFs for Speaker Recognition

Sachin Kajarekar[1], Luciana Ferrer[1], Kemal Sönmez[1], Jing Zheng[1], Elizabeth Shriberg[1,2], Andreas Stolcke[1,2]

[1]SRI International, Menlo Park, CA, USA
[2] International Computer Science Institute, Berkeley, CA, USA
{sachin,lferrer,kemal,zj,ees,stolcke}@speech.sri.com

Abstract

We introduce a new type of feature to capture long-range patterns associated with individual speakers or with speaking styles. NERFs, or Nonuniform Extraction Region Features, are defined based on regions of speech that are delimited by various automatically extractable events of interest. There is a wide unexplored space of potentially useful NERFs, but to use them successfully, at least two important challenges must be addressed: (1) methods for coping with inherently missing features, and (2) methods for feature selection from large sets of potentially correlated NERFs. We address the issue of missing features in this paper. We propose three methods for modeling NERFs that cope with missing features. We show that on the 2003 NIST extended-data speaker recognition evaluation task, a NERF system yields an EER of 11.6% alone, and improves the MFCC baseline performance by roughly 15% relative.

1. Introduction

Speaker recognition is the task of recognizing the identity of a speaker from his or her voice. Conventionally, this task is performed using spectral features estimated from a short segment of the waveform (about 10-50 ms) [1]. These features capture the speaker's vocal tract characteristics. However, they fail to capture the stylistic aspects of a talker's speech, and are sensitive to transmission channel characteristics.

Recently, there has been significant research activity in representing longer-term characteristics of a talker's speech, such as his or her choice of words, intonation, and duration patterns [2,3,4]. In addition to capturing vital information about a speaker's unique style, the long-term features are expected to be more robust to variation in the transmission channel than frame-based spectral features. Finally, longer-range features are potentially useful not only for discriminating speakers, but also in characterizing different speaking *styles*.

In this paper, we introduce the nonuniform extraction region features (NERFs) in Section 2. Section 3 describes three different methods of modeling these features. Section 4 describes the modified re-estimation algorithm used in two of these methods. Section 5 describes the experimental setup, and Section 6 gives the results. In Section 7, we present a summary and conclusions.

2. Nonuniform Extraction Region Features

NERFs are defined both by the region from which they are extracted (the NER) and by the type of feature(s) extracted within that region (the Fs). A region refers to a contiguous stretch of speech bounded by automatically extractable events of interest. A region could be bounded, for example, by pauses, by unstressed syllables, by pitch rises or falls, and so on. For defining potential NERs, we consider both what might constitute meaningful or characteristic units at some level of production (for example, a prosodic phrase) and what types of boundary events we can use to automatically delimit those units. The Fs are the features within the NERs. They can be defined to measure, for example, the maximum or mean pitch values, duration patterns, energy contours, and so forth. These features are similar to those used in studies of other unit types, such as utterances and words.

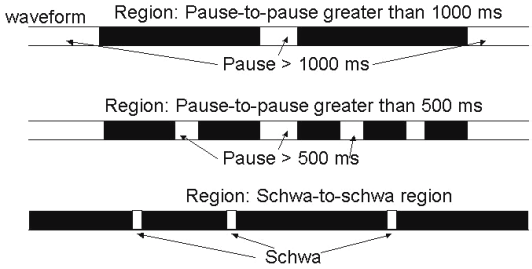

Figure 1 Schematic depiction of different NERs. Note that these regions may not be defined over the entire waveform and are asynchronous with respect to each other.

Figure 1 shows a stylized example of the non-uniform extraction regions (NERs). It shows the same waveform with different regions specified on it. For example, the first type of region is defined as the part of a waveform between two pauses of length greater than 1 s. The second type of region is similar to the first one except that the length of the pause threshold is reduced to 500 ms. The third type of region is defined as bounded by schwas, representing a rough foot-like metrical unit. From the figure, it is clear that all regions are

not defined over the entire waveform (e.g., pause-to-pause greater than 500 ms is not defined over pauses). In addition, they are not synchronous with respect to each other.

One set of features is extracted from each instance of a region; these features are called *NERFs*. There are two issues in modeling NERFs. First, a single NERF may correlate with other features from the same region, with features from previous or following regions of the same type, or even with features from other region types. This means that the NERFs should be modeled in joint region-feature space to capture the correlations. However, as mentioned earlier, these regions may not be defined over the entire waveform and their co-occurrence cannot be modeled based on a simple structure like a hidden Markov model (HMM) or time-frequency multiresolution tiling. The second issue is that some of the NERFs may not be defined in some instances of the region. For example, if a certain region does not contain any voiced frames, then pitch features will be undefined for that region.

Both these issues can be addressed using more flexible structures like graphical models (GMs). This requires significant development, which is being pursued in parallel. In this paper, we describe three ways of modeling these features by using Gaussian mixture models. These models are developed for a single NER, pause-to-pause greater than 500 ms.

3. Modeling Approaches

As mentioned before, potentially useful NERFs may be inherently undefined in some speech intervals. Note that the issue of an "undefined" feature is different from a "missing" feature. A "missing" feature can be estimated from other features but an "undefined" feature cannot be. It leads to a sequence of NERF vectors that do not have all the elements defined at all times, and makes the use of conventional models, like GMM, difficult. In this section, we describe some of our efforts to address this issue. Specifically, we propose three methods, all based on the use of GMMs but treating the problem of the missing features in different ways.

Method 1: Independent Modeling of Undefined Feature Combinations

In this method we independently model each set of feature vectors where the same feature components are undefined. For this, we first label feature vectors based on whether or not an element is present. For example, the label for a vector [0.9 X 1.7] will be 1-X-1 because the first and third features are defined but the second one is not. Similarly, the label for a vector [X X 2.3] will be X-X-1. The data for each label is modeled independently using a GMM. These models are adapted independently to estimate a speaker model. During verification, the models corresponding to the label for each feature vector are used to score the test utterance. The final score for the test utterance is simply the sum of the individual scores normalized by the number of vectors used. The training and evaluation procedures are similar to those used for the cepstral system (see Section 4.2) except that all the GMM parameters are adapted during speaker-model training.

This method is the easiest to implement, but it is effective only under the assumption that the few frequent labels represent most of the data. If a large portion of data has infrequent labels, then it is not efficient to model the corresponding data independently. Methods described next overcome this problem by using a single model for all the feature vectors.

Method 2: Bootstrapped GMM with Undefined Features

In this approach, we model all feature vectors, irrespective of their labels, using a single GMM in a framework that can handle vectors with undefined features. The probability computation and reestimation steps of conventional expectation maximization (EM) training of GMMs are modified to include this *a priori* probability when a feature is not defined. The high-level algorithm is described as follows:

1. Bootstrap a GMM using the feature vectors with all defined elements.
2. Reestimate the GMM parameters using all the data, including vectors with undefined elements.
3. Adapt target models using all data.
4. Perform verification using all data.

The reestimation (Step 2), using vectors with undefined elements, is described in detail in the next section. This method assumes that there is sufficient data with all defined elements to train the boot model. The method described next addresses this limitation.

Method 3: GMM Directly Trained with Undefined Features

This is modification of Method 2, where we eliminate the need to obtain a bootstrap GMM. This method does not require any preprocessing of data to compute labels. The high-level algorithm is described as follows:

1. Estimate single Gaussian model using the whole data, including vectors with undefined elements.
2. Split the Gaussian to create twice the number of Gaussians.
3. Reestimate the Gaussians using all data.
4. If the number of Gaussians is less than the desired number of Gaussians then go to Step 2.

The estimation (Step 1) and reestimation (Step 3) processes using vectors with undefined elements are described in detail in the next section.

4. Estimating GMM Using Undefined Features

To estimate a GMM using undefined features, we modified two steps. First, the step where the probability of the vector with respect to the model is computed. An EM derivation

using this modified probability equation gave the second modification in the reestimation equations. We describe the modified probability estimation first, followed by the model reestimation.

The probability of a feature vector with undefined elements, modeled using a GMM, is estimated as follows,

$$p_i(\bar{x}_t) = \prod_{k=1}^{N} \begin{cases} \left(\frac{1}{s_i^k \sqrt{2p}} e^{-\frac{1}{2}\left(\frac{x_t^k - m_i^k}{s_i^k}\right)^2} \right) P(i,k), & x_t^k \text{ is defined} \\ 1 - P(i,k), & \text{otherwise,} \end{cases} \quad (1)$$

where $\bar{x}_t$ is a feature vector at time t and i is the Gaussian index. Since the data is modeled using diagonal covariances, the probability computation is interpreted as a product of k per-element probabilities. If the element is defined then the probability is the likelihood (term in []) multiplied by the prior ($P(i,k)$) that the feature is defined for the Gaussian. Otherwise the probability is the prior that the feature is undefined for the Gaussian ($1 - P(i,k)$). The new model has four parameters (w, μ, σ, P). If we maximize the probability of the data computed using the above equation then the modified EM equations are

$$\Pr(i/\bar{x}_t) = \frac{w_i p_i(\bar{x}_t)}{\sum_{j=1}^{M} w_i p_i(\bar{x}_t)} \quad (2)$$

$$w_i = \frac{\sum_{t=1}^{T} \Pr(i/\bar{x}_t)}{T} \quad (3)$$

$$E_i(x^k) = \frac{\sum_{t=1}^{T} d(k,t) x_t^k \Pr(i/\bar{x}_t)}{\sum_{t=1}^{T} d(k,t) \Pr(i/\bar{x}_t)} \quad (4)$$

$$E_i\left[(x^k)^2\right] = \frac{\sum_{t=1}^{T} d(k,t) (x_t^k)^2 \Pr(i/\bar{x}_t)}{\sum_{t=1}^{T} d(k,t) \Pr(i/\bar{x}_t)} \quad (5)$$

$$P(k,i) = \frac{\sum_{t=1}^{T} d(k,t) \Pr(i/\bar{x}_t)}{\sum_{t=1}^{T} \Pr(i/\bar{x}_t)} \quad (6)$$

Equation (2) calculates the posterior probability of each Gaussian for a given vector. Equation (3) estimates the mixture weight of each Gaussian. Note that these equations are the same as the ones used in standard EM. The equations (4, 5 and 6) that reestimate the model parameters are similar to the ones in normal EM, with two differences. First, the reestimation is performed independently for each feature element. Second, only the known feature values (and their Gaussian posteriors) are used in per-element reestimation. We define $\delta(k,t)$ as 1 if the k^{th} element of t^{th} vector is defined and 0 otherwise to specify this modification.

Note that these equations are the same as in the original EM equations if all features are always defined. That is, if $\delta(k,t) = 1$, for all k and t, then these equations fall back to the original EM reestimation equations.

In Method 2, the above equations are used to reestimate the model, which was booted the modeled trained using all-defined features. Therefore, EM iterations are initialized with the boot model parameters and $P(i,k)$ are initialized with $P(k)$, which is the prior for the feature (k) being defined. This prior is computed over the background model data.

In Method 3, the above equations are used to estimate the single Gaussian model and to reestimate the models obtained by splitting this model. A single Gaussian model is booted using zero mean and unit variance per element and $P(i,k)$ are initialized with $P(k)$.

5. Experimental Setup

We present results obtained with the proposed modeling approaches on only one region type, the pause-to-pause regions with a pause threshold of 500 ms. A subset of NERFs is described in Table 1. For the chosen pause-to-pause region type, we have 32 features, which means that we have to model 2^{32} possible defined/undefined pattern labels when the first modeling method is implemented. Fortunately, the three most frequent labels represent 86% of the data. Our initial experiments showed that modeling the data labeled by these labels gave the best performance. Here, the data for the most common label (50%) is modeled using 64 Gaussians, the data for the second most common label (31%) is modeled using 32 Gaussians, and the data for the third most common label (5%) is modeled using 8 Gaussians.

The scores obtained from different NERF systems are then combined with a state-of-the-art cepstral-based system (Section 5.2) to show how much independent information the NERF system provides with respect to the baseline system.

5.1. Task

The modeling methods described were evaluated on NIST 2003 extended-data speaker recognition task. This is a detection task that uses data from Switchboard-II phases 2 and 3 databases. The task comprises telephone speech with about 1500 speaker models and 23,000 test trials. Each speaker model is trained using approximately 16 minutes of speech (8 conversation sides) and each test is performed using approximately 2 minutes of speech (1 conversation side).

For the evaluation, the data was divided into 10 nonoverlapping splits. All the splits have similar amounts of training and test data. Systems are not allowed to use data from the split that is being evaluated.

Table 1 Subset of NERFs estimated from pause-to-pause region

Feature Name	Feature description
MEAN_STY_F0_LOG	Log of the mean stylized f0 in the region
MAX_STY_F0_LOG	Log of the maximum stylized f0 in the region
AVE_Z_PHONE_DUR	Average of the normalized (by mean and variance) phone duration in the region
AVE_N_PHONE_DUR	Average of the normalized (by mean only) phone duration in the region
AVE_N_VOWEL_DUR	Average of the normalized (by mean only) vowel duration in the region
REGION_LENGTH_LOG	Log of the region length
AVE_Z_VOWEL_DUR	Average of the normalized (by mean and variance) vowel duration in the region
NUM_V_FRAMES_NORM_LENGTH	Number of voiced frames in the region normalized by the length of the region
STY_F0_RANGE	Range (log of the ratio between max and min values) of stylized f0 in the region
MAX_NEG_SLOPE_F0_LOG	Log of the maximum negative slope of the stylized f0 in the region
NUM_NODES_F0_LOG	Log of the number of nodes in the stylized f0
NUM_R_FRAMES_NORM_V	Number of rising frames in the stylized f0 normalized by the number of voiced frames
MAX_NEG_SLOPE_ENERGY_LOG	Log of the maximum negative slope of the stylized energy in the region

Performance of a system is measured using false acceptance and false rejection errors. In this paper, we compare different results using equal error rate (EER), the point on the detection error tradeoff curve at which the number of false acceptances equals the number of false rejections.

5.2. Cepstral-Based System

Our baseline system uses 13 Mel frequency cepstral coefficients (MFCCs), which are normalized using cepstral mean removal, and which are concatenated with delta and delta-delta features. The distribution of features is modeled in the Gaussian mixture model (GMM) and universal background model (UBM) framework. We use a 2048-component GMM as the background model. It is trained using gender- and handset-balanced data from Switchboard-II phase 1 corpus. Speaker models are created from the background model using maximum a-posteriori adaptation (MAP), in which only the means of Gaussians are adapted. Verification is performed using the log-likelihood ratio between the corresponding speaker model and the background model. During adaptation and verification, features are normalized for variation due to different handsets [7]. This system results in an EER of 2.30%.

5.3. Combination Method

The baseline and the NERF systems are combined at score level by using a neural network combiner (LNKnet [5]). The combiner is trained using 10-fold cross-validation. In each fold, the combiner is trained using nine splits and tested on the remaining split. The combiner is trained using two classes: true speaker and impostor. In the combination experiments, the class priors are adjusted to 10 and 1, respectively.

6. Results

Table 2 shows the performance of NERF systems using different modeling approaches. First, the table shows our NERF baseline, which is a GMM trained using only the all-defined feature vectors. This gives EER=15.0%. When feature vectors with unknowns are added to the system (row 2a), the performance improves to 11.57%. In this system, the three most frequent unknown combinations (around 86% of the data) are modeled separately (Method 1). This performance degrades when all the combinations for 99% of the data are used in the system (Row 2b). The degradation occurs because additional unknown combinations do not have sufficient data to model them independently.

This hypothesis can be supported by results of systems trained using Method 2 (Rows 3a and 3b). Judging from the comparison of Rows 2a and 3a, where the same amount of data was used, modeling the three most frequent combinations separately give similar performance. This performance improves slightly (Row 3b) with the addition of features for less-frequent unknown combinations. Thus the second method of training NERFs overcomes the short comings of the

first method by modeling the combinations together in a single model.

Finally, results show that Method 3 (single GMM trained using all the data without booting) performs better than the NERF baseline, but worse than systems trained using Methods 1 and 2. This shows that booting is important for training single GMM. However, note that this method can be used as a first system when training on a novel dataset because it does not require any preprocessing of the data.

Table 2 Performance of pause-to-pause region features

NERF System		Amount of data	%EER
1	Single GMM, all known elements	32%	15.0
2a	Separate GMMs	86%	11.57
2b	Separate GMMs	99%	12.40
3a	Single GMM, retrained EM	86%	11.87
3b	Single GMM, retrained EM	100%	11.57
4	Single GMM, complete EM	100%	13.57

Table 3 shows performance of the combination of baseline and different NERF systems. Results show that NERF systems trained using all the three methods give 14-17% improvement (significant at 95% confidence level) in EER after combination. Their performance is also better than the combination of the GMM baseline with the NERF baseline.

Table 3 Performance of the combination of NERF systems with Baseline, Baseline + n, where n refers to the system in row n from Table 2.

System Combination	%EER
Baseline	2.30
Baseline + 1	2.07
Baseline + 2a	1.94
Baseline + 3a	1.90
Baseline + 3b	1.94
Baseline + 4	2.00

7. Summary and Conclusions

We have presented nonuniform extraction region features (NERFs), which model long-term patterns associated with the speaking style of individual speakers. We described the modeling issues and proposed three methods to model these features in the existing framework of GMM.

The first method assumes that features with different combinations of unknowns must be modeled separately. Therefore, features for different unknown combinations are collected and modeled using different GMMs. This assumption is based on the hypothesis that correlations among features might be different when some features are missing, for example, NERFs from voiced and unvoiced regions where the pitch features will be defined or undefined respectively. This method requires preprocessing of the data to create labels for feature vectors, but the modeling requires no changes to the GMM framework.

The drawback of the first method is that if a data set had a lot of unknown combinations with few feature vectors per combination, then the modeling would not be efficient. To overcome this problem, we propose a second method. Here, a single GMM is trained using all data. This model is booted from data with all defined feature elements. This method models averaged correlations among features when some of them are missing. However, it is more efficient in modeling infrequent unknown combinations.

A minor drawback of the second method is that it assumes that there is sufficient data with all-known vectors to boot the model. We investigate a third method where a single model is trained using all data without booting. First, a single Gaussian model is estimated from the data. Then, split-and-retrain iterations are used to derive the GMM. This approach does not require any preprocessing of the data, but it does not give any improvement over the second method.

Results show that systems based on these methods give significant improvements over a NERF baseline, which uses only all-known feature vectors. They also give a significant reduction in EER when combined with a state-of-the-art baseline system.

This paper lays the groundwork for modeling NERFs. We consider the performance of NERFs independently and in combination with the baseline as very promising. In future work, we will add more NERFs and model joint region-feature correlations, and we will also explore other issues, such as feature selection of NERFs.

8. Acknowledgment

This work was funded by a DoD KDD award via NSF IRI-9619921. The views herein are those of the authors and do not reflect the views of the funding agencies.

9. References

[1] D. Reynolds, T. Quatieri, and R. Dunn, "Speaker Verification Using Adapted Mixture Models," *Digital Signal Processing*, vol. 10, pp.181-202 (2000).

[2] 2001 JHU Summer Workshop Report, SuperSID: Exploiting high-level information for high-performance speaker recognition, http://www.clsp.jhu.edu/ws2002/groups/supersid/supersid-final.pdf

[3] D. Reynolds, W. Andrews, J. Campbell, J. Navratil, B. Peskin, A. Adami, Q. Jin, D. Klusacek, J. Abramson, R.

Mihaescu, J. Godfrey, D. Jones, and B. Xiang, "The SuperSID Project: Exploiting High-level Information for High-accuracy Speaker Recognition," in *Proc. IEEE ICASSP* (Geneva), 2003

[4] S. Kajarekar, L. Ferrer, A. Venkataraman, K. Sonmez, E. Shriberg, A. Stolcke, H. Bratt, and R. R. Gadde, "Speaker Recognition Using Prosodic and Lexical Features," in *Proc. IEEE ASRU* (St. Thomas, VI), pp. 19-24, December 2003.

[5] LNKNet, MIT Lincoln Laboratory. http://www.ll.mit.edu/IST/lnknet/

[6] K. Sonmez, E. Shriberg, L. Heck, M. Weintraub, "Modeling Dynamic Prosodic Variation for Speaker Verification", *Proc. Intl. Conf. on Spoken Language Processing*, vol. 7, pp. 3189-3192, Sydney, Australia (1998).

[7] D. A. Reynolds, "Channel Robust Speaker Verification via Channel Mapping", *Proc. IEEE ICASSP*, vol. 2, pp. 53-56, Hong Kong (2003).