Annual Report for Period: 09/2006 - 08/2007
Submitted on: 08/12/2007
Principal Investigator: Picone, Joseph .
Award ID: 0414450
Organization: Mississippi State Univ
Title:
Nonlinear Statistical Modeling of Speech

Project Participants

Senior Personnel

Name: Picone, Joseph
Worked for more than 160 Hours: Yes
Contribution to Project:

Name: Lazarou, Georgios
Worked for more than 160 Hours: Yes
Contribution to Project:
Professor Lazarou has assumed supervision of the students while Dr. Picone is on a sabbatical.

Post-doc

Graduate Student

Name: Patil, Sanjay
Worked for more than 160 Hours: Yes
Contribution to Project:
Mr. Patil is leading our research into Lyapunov exponent estimation and particle filtering.

Name: Raghavan, Sridhar
Worked for more than 160 Hours: Yes
Contribution to Project:
Mr. Raghavan is responsible for development of the baseline speaker recognition system that will be used to assess our research. He is also improving our Support Vector Machine and Relevance Vector Machine models that are used as alternative statistical estimators. These will provide important comparison points for the nonlinear estimators.

Name: Prasad, Saurabh
Worked for more than 160 Hours: No
Contribution to Project:
Mr. Prasad joined the project in Fall'2005, and is leading our research on Kalman filters and unscented Kalman filters. These are being used in our particle filter implementation.

Name: Pannuri, Madhulika
Worked for more than 160 Hours: No
Contribution to Project:
Ms. Pannuri joined the project in Fall'2005 as an entry-level MS student. She is currently providing programming support to the project as she develops her background in this research.

Name: Srinivasan, Sundararajan
Worked for more than 160 Hours: No
Contribution to Project:
Mr. Srinivasan joined the project in Fall'2005, and is developing more robust ways to estimate Lyapunov exponents, as well as providing programming support while he gets up to speed on the research.

Name: May, Daniel
Worked for more than 160 Hours: Yes

Contribution to Project:

Daniel May is now a graduate student pursuing his MS in Computer Engineering. He has been developing new statistical models for features based on nonlinear measures.

==========

Daniel May joined the project in Fall'2005 as a senior undergraduate. He is assisting the release of our software, including Java applets developed on an NSF REU extension to this project.

Name: Ma, Tao

Worked for more than 160 Hours: Yes

Contribution to Project:

Speech researcher - initially responsible for maintaining and upgrading our experimental infrastructure

Undergraduate Student

Name: Irwin, Ryan

Worked for more than 160 Hours: Yes

Contribution to Project:

Ryan Irwin joined the project in Summer'2005 as an undergraduate working on the NSF REU extension. He has extended our Java-based pattern recognition applet to include Kalman and particle filtering.

Name: Holland, Wesley

Worked for more than 160 Hours: Yes

Contribution to Project:

Wesley Holland joined the project in Summer'2005 as an undergraduate working on the NSF REU extension. He has developed software that allows our speech recognition system to support XML grammar formats. This includes software to manipulate and transform context-free and regular grammars.

Technician, Programmer

Other Participant

Research Experience for Undergraduates

Organizational Partners

Department of Defense

DoD is a partial sponsor of this work. I am also currently working on a sabbatical with the group that sponsored the work. We have a close working relationship with them and transfer software from MS State to DoD on an as-needed basis. We try to address DoD needs that arise that are relevent to the project. Two examples of such work are confidence measures and lattice rescoring. We delivered some Perl tools to DoD for these computations, and also released these as part of our public domain software.

Other Collaborators or Contacts

We support a number of collaborators through the distribution of our public domain software.

Activities and Findings

Research and Education Activities: (See PDF version submitted by PI at the end of the report)

Findings:

Convergence of Kalman filtering, particle filtering, and other such iterative algorithms is very sensitive to prior knowledge. This is typically

why these techniques don't work well for robust speech processing applications.

Our initial SVM-based speaker recognition system provides a small improvement over the GMM baseline, a finding that is consistent with what others in the community have found.

Our attempts at adding several nonlinear features independently have not yet provided an improvement in performance. We have certified our implementations against previously published work, so our results conflict with previously published results. We continue to explore and analyze these experiments.

Training and Development:

Several of our students have improved their technical writing skills through publications and documentation related to this project.

All students are learning a strict software engineering process we use in our lab, and have increased their knowledge considerably.

All graduate students have received special training through a graduate level course in Natural Language Processing that we were able to offer in conjunction with this project.

Outreach Activities:

We have hosted several open houses for a local high school that specializes in mathematics and sciences: The Mississippi School for Mathematics and Science. It is one of the best math and science schools in the country and attracts the best students from all over Mississippi.

We also presented a lecture at the Math Club at the same institution. This presentation emphasized the value of math in the engineering disciplines.

Journal Publications

S. Prasad, S. Srinivasan, M. Pannuri, G. Lazarou and J. Picone, "Nonlinear Dynamical Invariants for Speech Recognition", International Conference on Spoken Language Processing, p. 2518, vol. 1, (2006). Published,

S. Prasad, S. Srinivasan and J. Picone, "Reconstructed Phase Space of a Vector Time Series", 14th European Signal Processing Conference, p. 1, vol. 1, (2006). Accepted, but declined trip because the lead author dropped out,

S. Srinivasan, S. Prasad, S. Patil, G. Lazarou and J. Picone, "Estimation of Lyapunov Spectra From a Time Series", Proceedings of IEEE SoutheastCon, p. 192, vol. 1, (2006). Published,

S. Patil, S. Srinivasan, S. Prasad, R. Irwin, G. Lazarou and J. Picone, "Sequential State-Space Filters for Speech Enhancement", Proceedings of IEEE SoutheastCon
, p. 240, vol. 1, (2006). Published,

Books or Other One-time Publications

J. Picone, "Risk Minimization Approaches in Speech Recognition", (2006). Book, Submitted
Editor(s): Idea Group Inc., USA
Collection: Kernel Methods in bioengineering, communications, and signal processing
Bibliography: G. Camps-Valls, J.L. Rojo-Álvarez, and M. Martínez-Ramón, "Kernel Methods in bioengineering, communications, and signal processing," The Idea Group, Inc., Hershey Pennsylvania, USA

Web/Internet Site

URL(s):

http://www.ece.msstate.edu/research/isip/project/nsf_nonlinear

Description:

We always maintain a web site for every project we execute. This web site includes software, data, publications, etc., related to the project.

Other Specific Products

Product Type:

Teaching aids

Product Description:

A tutorial on particle filtering.

Sharing Information:

This work is disseminated via a URL:

http://www.cavs.msstate.edu/hse/ies/whats_new/2005_06/

Product Type:

Teaching aids

Product Description:

We have developed a number of tutorials on key core technologies for this project.

Sharing Information:

These tutorials are available at:

http://www.cavs.msstate.edu/hse/ies/projects/nsf_nonlinear/doc/

Product Type:

Software (or netware)

Product Description:

We have augmented our pattern recognition applet to include three time series analysis techniques: linear prediction, Kalman filtering, and Particle filtering.

Sharing Information:

The applet is available at:

http://www.cavs.msstate.edu/hse/ies/projects/speech/software/demonstrations/applets/util/pattern_recognition/current/index.html

Product Type:

Software (or netware)

Product Description:

We have developed baseline implementations of several techniques for estimating nonlinearities in a signal. These are provided in MATLAB and used as reference implementations for our C++ code.

Sharing Information:

This software is located at:

http://www.cavs.msstate.edu/hse/ies/projects/nsf_nonlinear/downloads/software/matlab/

Contributions

Contributions within Discipline:

We have replicated previously published work on particle filtering, Kalman filtering, and Lyapunov exponent estimation. We have applied these techniques to more comprehensive speech databases as a first step in characterizing their performance on a large-scale application.

We have provided reference implementations of Lyapunov exponents, Correlation Dimenions, and the Embedding Dimension for use in speech recognition experiments.

We have evaluated these features on both speaker recognition and speech recognition tasks.

Contributions to Other Disciplines:
We have made software available as part of our public domain system, and also released a number of tutorials on our web site.

Contributions to Human Resource Development:
We have introduced two undergraduate students to speech research. Both are showing great promise and plan to pursue graduate studies.

A third undergraduate will enter graduate school in Spring'2006 and continue working on this project.

Contributions to Resources for Research and Education:
A project web site has been developed and maintained.

Contributions Beyond Science and Engineering:

Special Requirements

Special reporting requirements: None
Change in Objectives or Scope: None
Unobligated funds: $ 0.00

Animal, Human Subjects, Biohazards: None

Categories for which nothing is reported:

Contributions: To Any Beyond Science and Engineering

09/30/06 — 08/31/07: RESEARCH ACTIVITIES

The primary goal of this project is to develop novel nonlinear modeling techniques for speech and speaker recognition systems. While there have been tremendous advances in speech processing systems brought about by predominantly linear models, for the past several years these improvements have been marginal and have reached a point of stagnation. To build futuristic recognition machines that achieve and even go beyond the capabilities of the human auditory system, it behooves speech engineers to look beyond the linear modeling paradigm. However, two questions arise when considering nonlinear models for speech: 1) do we really need a nonlinear model; i.e., how do we know that nonlinearities are present in speech signals? 2) how to build tractable nonlinear models for speech?

Earlier in this project, we have shown a significant presence of nonlinearities in speech signals and their potential in broad phone classification. This answers the first question. Encouraged by this, we have concentrated our efforts to make use of the nonlinearities in a tractable way, and channeled our research in three different directions. First, we have continued to work with the nonlinear invariants which we had earlier used to prove the presence of nonlinear effects in speech. We now model these invariant features using conventional HMM techniques for speech recognition. In our second route to attain our goal, we use a nonlinear MixAR (Mixture of Autoregressive) model that uses the conventional wisdom of linear models along with novel nonlinear modeling capabilities, for speaker recognition. Thirdly, we consider the use of Linear Dynamical Models (LDM) as a convenient starting point, and hope to replace it's linear filter part with a nonlinear one.

A. Continuous Speech Recognition with Nonlinear Dynamical Invariants

Last year, we estimated statistical models for various sustained phones using nonlinear dynamical invariants and computed the KL divergence between these models. These are called invariants because they remain constant under transformations of the phase space as long as these transformations are smooth. We were able to show that these invariants contained a high level of discriminative information between different phone classes. Since these invariants describe the nonlinear content of speech, they can be combined with traditional linear acoustic information to produce a more accurate acoustic model.

This year, we combine these nonlinear invariants with the traditional MFCC feature vector in order to model the nonlinear acoustic information contained within speech. We test this new feature vector with data recorded in a clean environment as well as data with significant additive noise. The goal is to produce acoustic models that are more robust to channel variations and noise conditions.

A.1 Review of Nonlinear Dynamical Invariants

It is difficult to extract nonlinear properties from a time-series representation of an observable from a non-linear system using traditional Fourier-based techniques. An alternate representation of the system, the phase space representation, is required [1]. This represents the temporal evolution of the system's states. The path created by this evolution of states is called the system's trajectory, and the set of points in the state space that are accumulated in the limit as $t \to \infty$ is called the attractor of the system [1] [2] [3]. In order to estimate the system's phase space representation from the observed time-series, we use a method called Phase Space Reconstruction. This method uses a technique called embedding in order to construct the phase space. There are two common types of embedding. The simplest form of embedding is time-delay embedding which uses time-delayed copies of the original scalar time series as components of the reconstructed phase space (RPS). This form of the RPS is represented by:

$$X = \begin{pmatrix} x_0 & x_\tau & \cdots & x_{(m-1)\tau} \\ x_1 & x_{1+\tau} & \cdots & x_{1+(m-1)\tau} \\ x_2 & x_{2+\tau} & \cdots & x_{2+(m-1)\tau} \\ \vdots & & \vdots & \end{pmatrix}$$

where m is the embedding dimension and τ is the embedding time delay. Each row in the matrix X is a point in the RPS [1].

The second type of embedding is called Singular Value Decomposition (SVD). This method consists of two steps: First, a dimensionality, or window size, is chosen and the scalar time series is embedded into this higher dimensional space using time-delay embedding and a delay of one sample. Next, the embedded matrix is reduced in dimensionality by a linear transformation. We use SVD for embedding because it is the preferred embedding technique for noise-corrupted data [4].

Our experiments have focused on three specific nonlinear dynamical invariants: Lyapunov exponents, fractal dimension, and Kolmogorov-Sinai entropy. These invariants are described below.

A.1.1 Lyapunov Exponent

Lyapunov exponents measure the separation over time of trajectories with infinitesimally close initial points [1]. Suppose a system's evolution function is defined by *f*. We need to analyze:

$$\Delta x(t) \approx \Delta x(0) \frac{d}{dx}(f^N)x(0)$$

where $x(0)$ is the initial point. To quantify this separation, we assume that the growth rate over time of the separation of trajectories is exponential. From this assumption, we define the exponents, λ_i as:

$$\lambda_i = \lim_{n\to\infty} \frac{1}{n} \ln(\mathrm{eig_i} \prod_{\mathrm{p=0}}^{\mathrm{n}} \mathrm{J(p)})$$

where J is the Jacobian of the system as the point p traverses the attractor. The Lyapunov exponent is computed by applying the above calculation to the points on the reconstructed attractor, and averaging this separation over the entire attractor. Literally, Lyapunov exponents provide some idea of the amount of chaos in a system.

A.1.2 Fractal Dimension

A fractal is defined as an object that is self-similar, in geometrical structure for example, at various resolutions [5] [6]. Fractal dimension is used to describe the fractal structure and the fractal dimension of an attractor can be estimated using a technique called correlation dimension. This technique uses the power-law relation between the correlation integral of an attractor and the neighborhood radius of the analysis hyper-sphere to provide an estimate of the fractal dimension:

$$D = \lim_{N\to\infty} \lim_{\varepsilon\to 0} \frac{\partial \ln C(\varepsilon)}{\partial \ln \varepsilon},$$

where $C(\varepsilon)$, the correlation integral is defined as:

$$C(\varepsilon) = \frac{2}{N*(N-1)} \sum_{i=1}^{N} \sum_{j=i+1}^{N} \Theta(\varepsilon - \left\| \vec{x}_i - \vec{x}_j \right\|),$$

where $\vec{x}$ is a point on the attractor (which has N such points). The correlation integral is essentially a measure of the average number of points within a neighborhood of radius ε over the entire attractor. To avoid temporal correlations in the time series from producing an underestimated dimension, we use Theiler's correction for estimating the correlation integral.

A.1.3 Kolmogorov-Sinai Entropy

Entropy is a well-known measure used to quantify the amount of disorder in a system [7], and has also been associated with the amount of information stored in general probability distributions. Numerically, the Kolmogorov-Sinai entropy can be estimated as the second order Renyi entropy (K_2) and, like correlation dimension, can be related to the correlation integral of the reconstructed attractor [5] by:

$$C_d(\varepsilon) \sim \lim_{\substack{\varepsilon \to 0 \\ d \to \infty}} \varepsilon^D \exp(-\tau\, d\, K_2),$$

where D is the fractal dimension of the system's attractor, d is the embedding dimension and τ is the time-delay used for attractor reconstruction. This results in the relation

$$K_2 \sim \frac{1}{\tau} \lim_{\substack{\varepsilon \to 0 \\ d \to \infty}} \ln \frac{C_d(\varepsilon)}{C_{d+1}(\varepsilon)}$$

In practical situations, however, the values of ε and d are restricted by the resolution of the attractor and the length of the observed time series.

A.2 Phonetic Classification Experiments

Last year, we successfully showed that nonlinear dynamical invariants can discriminate between different phone classes. We demonstrated this using a small database of sustained phonemes, extracting invariants from each phone, estimating a GMM, and measuring the KL-Divergence between these models. The results suggested that these invariants contain useful information about the non-linear characteristics of speech that is not present in linear acoustic features.

This year, we decided to investigate whether or not this nonlinear information could be combined with traditional linear acoustic data, such as MFCCs, to better model speech signals. Our overall goal was to show that we could use dynamical invariants to improve the recognition performance of a continuous speech recognition task. As a first step in our investigation, we ran a set of closed-loop phonetic classifications experiments on Wall Street Journal (WSJ) large-vocabulary continuous speech recognition corpus. The purpose of these experiments was to show that the addition of dynamical invariants to the standard MFCC feature vector would improve the classification of signal frames as their corresponding phones within continuous speech.

The WSJ corpus consists of high-quality recordings of speech read from newspaper articles appearing in the Wall Street Journal. The data set used for these experiments consists of 7,138 16 kHz-sampled utterances from 83 speakers totaling 14 hours of speech. Standard MFCC features were extracted from each utterance using a 10ms frame and a 25ms overlapping window. The feature vector consists of 13

features: absolute energy and 12 MFCCs. Since these initial experiments attempt to classify each frame independently, the first and second derivatives are not used as features.

The three nonlinear dynamical invariants described previously are also extracted from the data. Last year, our pilot experiments found the optimal parameter values for each of the invariant computation algorithms for speech data. These were found using the sustained phone speech corpus described earlier. Using these parameters, we compute the invariants and append each to the standard 13-dimensional MFCC feature vector to form three new feature vectors with 14 elements each. This results in a total of four feature vectors: one containing the 13 MFCCs which we will use as a baseline, and three containing a combination of MFCCs and invariants.

Using time-aligned phonetic transcriptions of the corpus, we estimate a 16-mixture GMM for each of the 40 phones contained in the lexicon. The time-alignments for the transcriptions were obtained via a forced alignment method using our public-domain speech recognition toolkit. Using a closed-loop experimental apparatus, we used the trained GMMs to classify each frame of the training data as one of the 40 phonemes. The results in Table 1 below show the average relative classification improvement of each MFCC/invariant combination compared to the MFCC baseline results.

Figure 1 shows examples of the relative improvement for each phone class using each of the three different feature combinations.

	Correlation Dimension	Lyapunov Exponent	Correlation Entropy
Affricates	10.3%	2.9%	3.9%
Stops	3.6%	4.5%	4.2%
Fricatives	-2.2%	-0.6%	-1.1%
Nasals	-1.5%	1.9%	0.2%
Glides	-0.7%	-0.1%	0.2%
Vowels	0.4%	0.4%	1.1%

Table 1 - Relative Phone Classification Improvement

All three feature combinations result in classification improvements for affricates and stops, suggesting a significant amount of useful nonlinear information present in these phone types. Fricatives, however, suffered a drop in classification accuracy. This is most likely attributed to the unvoiced subset of fricatives (s, sh, f, th) since they are highly chaotic. This results in invariant values with wide variance which do not seem to contribute to the linear acoustic information. Nasal and glide classification accuracy showed little or no improvement. Vowels showed a slight improvement for all feature combinations.

The addition of the Lyapunov exponent resulted in a classification accuracy improvement for affricates, strops, and nasals. Correlation entropy resulted in a fairly consistent improvement for all phone classes except for fricatives, suggesting that it may contain the most speech-relative nonlinear acoustic information. Correlation dimension resulted in significant classification improvements for affricates and stops, but suffered a decrease in classification accuracy for fricatives, nasals, glides. In general, however, the relative improvements are much higher than accuracy decreases. This provides sufficient justification for a set of continuous speech recognition experiments using these feature combinations.

A.3 Continuous Speech Recognition Experiments

For our speech recognition experiments, we use a corpus developed by the Aurora Working Group. This corpus is based on the DARPA Wall Street Journal large vocabulary recognition corpus described in the previous section and consists of utterances recorded in clean conditions as well as a variety of additive noise conditions with random SNRs between 5 and 15 dB. More specific details about this corpus can be found in [7]. Experiments were performed on both the clean and noisy test sets.

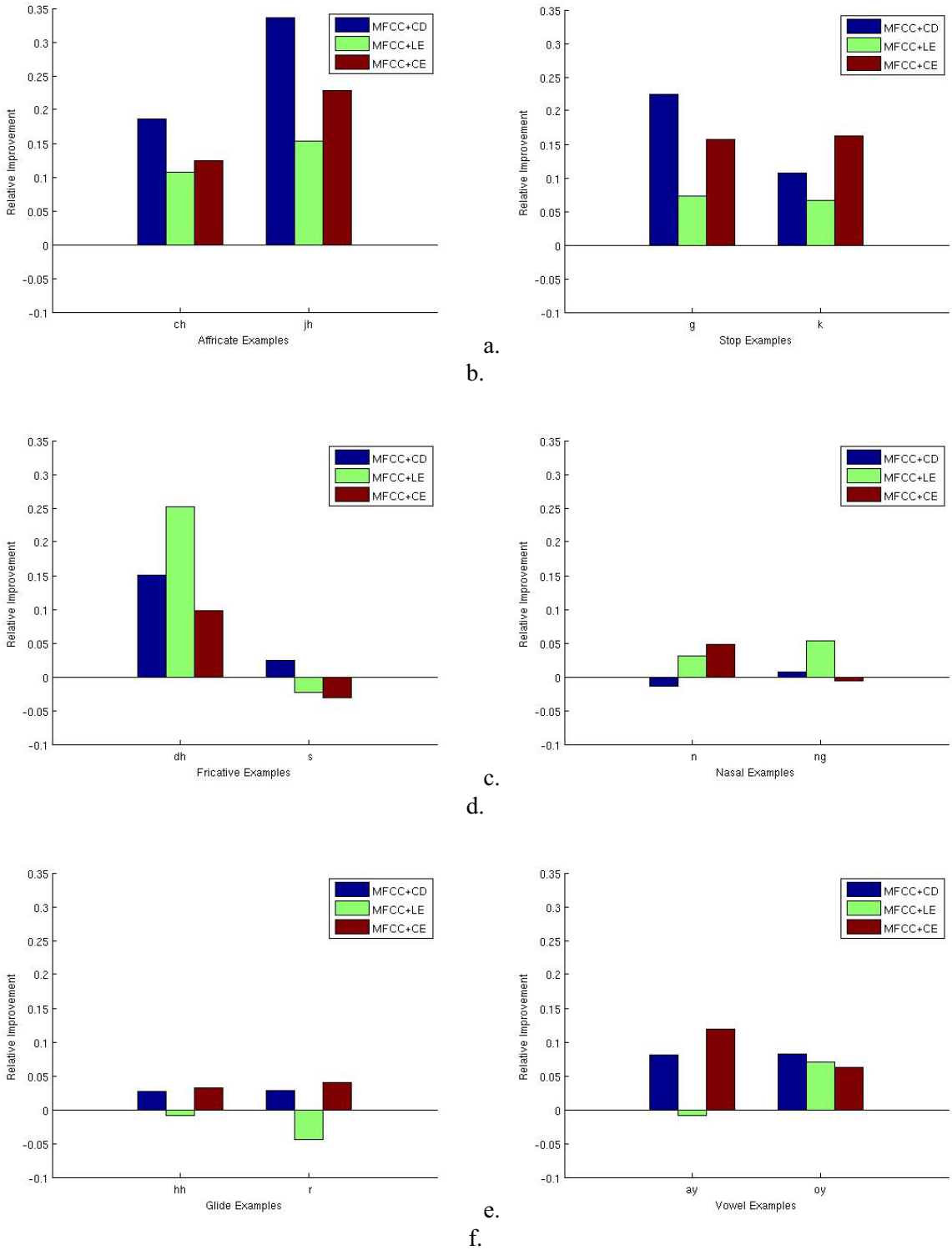

Figure 1 - Examples of relative improvement for select phones in different classes: Affricates (a), Stops (b), Fricatives (c), Nasals (d), Glides (e), and Vowels (f).

These experiments were performed using the a speech recognition system developed by our group that we refer to as the prototype system. This system uses context-dependent hidden-Markov models to model

acoustics and N-gram language models with backoff probabilities [7]. The acoustic models for these experiments were trained using the clean Aurora training set. Using features extracted from this data, a set of context dependent triphone models were estimated. A standard 3-state left-to-right topology with self-loops on each states was used for the models. Each state consists of an underlying 4-mixture GMM. The following sections present the results obtained on both the clean and noisy test sets.

A.3.1 Clean Speech Data

Table 2 contains the speech recognition results for the clean test set using the different feature combinations.

	WER (%)	Relative Improvement (%)
MFCCs	13.5	--
MFCCs+CD	12.2	9.6
MFCCs+LE	12.5	7.4
MFCCs+CE	12.0	11.1
MFCCs+All	12.8	5.2

Table 2 - Performance of Feature Combinations on Clean Test Set

All feature combinations resulted in a WER improvement over the baseline MFCC results. In general, the results reflect the findings in the phonetic classification experiments in A.2. The most significant improvement resulted from the addition of correlation entropy. Correlation dimension shows the second best improvement, followed by lyapunov exponents. An experiment combining MFCCs with all three invariants was also performed, but did not show as much of a WER improvement as any of the three individual invariants. This suggests the presence of a large amount of overlapping information across the three invariants.

A.3.2 Noisy Speech Data

The Aurora corpus consists of six different test sets with different additive noise conditions that are likely to be encountered in real applications. The noise conditions include airport noise, random babble, car noise, restaurant noise, street noise, and train noise. The purpose of this set of experiments is to determine whether or not these invariants, when added to MFCCs, are more robust to mismatches between the training and test data. The models used for these experiments are the same that were used for those in A.3.1 which were trained using clean training data. Table 3 contains the results of the speech recognition experiments.

Surprisingly, this first set of experiments suggest that the addition of an invariant damages the WER in most cases. The only exceptions to this occur with the correlation entropy invariant in the case of airport, car, and train noise, and even in these cases, the WER improvement is small. These initial results contradict our original theory that the addition of these invariants would result in more robust models. This theory was based on the fact that invariants remain constant under smooth transformations of the phase space, and that changes in channel conditions result in such a transformation. Additional experiments are required in order to fully understand these results.

	WER (%)					
	Airport	**Babble**	**Car**	**Restaurant**	**Street**	**Train**
MFCCs	53.0	55.9	57.3	53.4	61.5	66.1
MFCCs+CD	57.1	59.1	65.8	55.7	66.3	69.6
MFCCs+LE	56.8	60.8	60.5	58.0	66.7	69.0
MFCCs+CE	52.8	56.8	58.8	52.7	63.1	65.7
MFCCs+All	58.6	63.3	72.5	60.6	70.8	72.5

Table 3 - Performance of Feature Combinations on Noisy Test Sets

A.4 Future Directions

Until now, we have run experiments with feature vectors consisting of the traditional 39 dimensional feature vector (12 MFCCs, absolute energy, first and second derivatives), combined with invariants. In order better symmetrize the feature vector, we will run experiments which also include the first and second derivative of the invariants. The hope is that the invariant derivatives contain additional acoustic data.

We were able to show in A.3.1 that invariants can improve the WER for the clean test set of the Aurora corpus. This is a strong indication that these nonlinear invariants contain additional information not present in traditional linear acoustics such as MFCCs.

A deeper investigation is required in order to understand the noisy test data results in Table 3. The significant increase in WER after the addition of only a single feature was unexpected. The distributions of these invariants for the different noise conditions need to be analyzed and closely compared to the corresponding distributions for the clean data. We will also determine whether we can use post-processing techniques to refine the nonlinear features and remove some of the effects introduced by noise.

B. MixAR Modeling of Speech Signals for Speaker Recognition

It has been a long standing tradition of the speech processing community to view the process of speech production in the form of a source-filter model [9]. This is based on the idea that speech can be thought of as the output of a source signal originating at or below the vocal cords filtered through the vocal tract. Almost all the models developed use this as the underlying principle in one way or the other, though the way the two components – the source and the filter – of this paradigm are treated, varies greatly.

The most popular theme until now for the filter component (which is the more important of the two for recognition tasks) has been the linear autoregressive (AR) model [9]. This model is motivated by the fact that there exists significant correlation between consecutive speech samples, and hence each sample to a large degree should be predictable from knowledge of the past few samples. The linear AR model crystallizes this idea in its simplest form - predict each sample from a weighted sum of the past p (called prediction order) samples and expressed mathematically as:

$$\hat{X}(t) = \sum_{i=1}^{p} a(i) X(t-i)$$

The weights $\alpha(i)$ are called linear predictive coefficients (LPC's), and they are indicative of the type of sounds they represent. Even the perceptually more relevant Mel-Frequency Cepstral Coefficients

(MFCCs), which have been shown to possess a high degree of discriminability that is useful in speech and speaker recognition, can also been viewed as variations on the theme of LPCs.

However, recent studies on the properties of speech signal show the presence of a significant amount of nonlinearity that may have bearing on the perception of speech. Past experience in our own group in this project has taught us that the nonlinear effects alone may be strong enough even to discriminate between different classes of sounds. On the other hand, linear models have led us this far in the history of speech processing, that it would be unwise to abandon it altogether. Perhaps, our best hopes for a comprehensive speech model that encompasses both linear and nonlinear effects lie with those that capture the nonlinear effects sitting on top of existing linear models. Mixture of autoregressive models (MixAR) does exactly this.

To the best of our knowledge, we are the first to apply MixAR [10] as a nonlinear speech signal model for speaker recognition. On a more fundamental level, unlike previous works that have used it only as a predictive model, we demonstrate MixAR as a novel pattern classifier. In addition, we also liberate the model from the constraint in the original formultaion that the orders for gate function and prediction filter are to be the same. In this way, another closely related model called the Mixture Autoregressive (MAR) model [11] is realized as a special case of MixAR model, with gate order zero.

B.1 MixAR: A Nonlinear Time-Series Model

The linear AR model was described above. A mixture of autoregressive (MixAR) model [10] is comprised of a mixture of several linear AR models, all of which are tied together probabilistically. A time series following a MixAR($m;d$) process is described mathematically by the equation:

$$X(t)=\begin{cases} a_1^T X_{t-p}^{t-1}+\mu_1+\varepsilon_t^{(1)} & w.p. \quad g_1(X_{t-p}^{t-1};\theta_g) \\ a_2^T X_{t-p}^{t-1}+\mu_2+\varepsilon_t^{(2)} & w.p. \quad g_2(X_{t-p}^{t-1};\theta_g) \\ \vdots & \\ a_m^T X_{t-p}^{t-1}+\mu_m+\varepsilon_t^{(m)} & w.p. \quad g_m(X_{t-p}^{t-1};\theta_g) \end{cases}$$

where

X_{t-p}^{t-1} : lag vector $(X_{\text{t-1}},\ldots,X_{\text{t-p}})$

$\varepsilon_t^{(j)}$: iid Gaussian process with mean zero and variance σ_{j}^2

a_j : ARpredictor coefficients for component j

μ_j : mean of component j

g_j : gate function for component j

θ_g : gate parameters

The $g_j(.;\theta_g)$ add up to one, and can be interpreted as a probability distribution over the m different AR models. Each AR component of the model can be thought of as a local AR structure, and the component's role at a specific point in the time series is dictated by the gate function.

As in [10] we choose multinomial logits from the neural networks literature for the gating functions. This is described as:

$$g_j(X;\theta_g) = \frac{\exp(r_j^T X + s_j)}{\sum_{i=1}^{m} \exp(r_i^T X + s_i)}$$

where $\theta_g = [r_1^T, s_1, \ldots, r_m^T, s_m]^T$ is the set of gate parameters.

Thus a full set of parameters, θ, needed to specify a MixAR model is:

$$\theta = [a_1^T, \mu_1, \sigma_1, \ldots, a_m^T, \mu_m, \sigma_{m,} \theta_g^T]^T$$

In the above, for purposes of clarity, we have shown only the constrained original formulation of MixAR, with both the gate and prediction orders equal. But our implementation does follow this constraint. With a gate order of zero, this degenerates to a mixture autoregressive model (MAR).

For a probabilistic model to be useful in practice, certain theoretical regularity conditions like the existence and uniqueness of the probability measure are desirable, and these are also established for MixAR in [10]. This ensures, "for example that we can expect to find consistent parameter estimates, and also imply central limit theorems for estimators" [10].

A significant role is played by the gates in setting MixAR model apart from conventional linear AR models. The relative contributions of each AR component of MixAR are dictated by the gates, which in turn are functions of data. Hence, these gates achieve a data-dependent probabilistic mixing of individual AR models and act as sources of nonlinearity. However, it is to be noted that even when gate order is zero and MixAR degenerates to MAR, the model is still nonlinear.

B.2 MixAR Model Parameter Estimation

Given a set of parameters of a statistical model, the likelihood function of some data provides us with information about how likely it is that the data could have come from a probability distribution described by the model. A higher value for likelihood indicates a better match between the distribution of data and the model. Hence it is desirable to obtain a set of parameters for which this likelihood function of data is maximized. The popular method of estimating parameters of a statistical model by maximizing likelihood of given data is called the maximum likelihood estimation (MLE) [10]. It should be noted that in most cases MLE does not directly maximize the likelihood, but instead maximizes the log-likelihood which is typically easier to handle. Since the log function increases monotonically on its domain, the two solutions are equivalent.

Other than a few specific cases, MLE for most statistical models, as that for MixAR, may not exist as closed-form solutions. For these models, expectation-maximization (EM) algorithm is a popular estimation method [13]. This algorithm is an iterative procedure that starts from an prior assumed initial set of parameters and repeatedly runs an E-step (expectation) followed by an M-step (maximization). The iterations are stopped once convergence of likelihood is reached. Fortunately, EM algorithm produces monotonic increase in the likelihood and hence, is guaranteed to converge at least to a local maximum.

For a MixAR model, the log-likelihood of data is given by:

$$Q(X \mid \theta) = \sum_{t=d+1}^{N} \sum_{j=1}^{m} \tau_{t,j} \log(g_j(X_{t-d}^{t-1};\theta_g)\, n(X_t; a_j^T X_{t-d}^{t-1} + \mu_j, \sigma_j))$$

where $n(.;\mu,\sigma)$ is the Gaussian density function; $\tau_{t,j}$ is the relative contribution of component j to describe sample X_t, and is described by the equation:

$$\tau_{t,j} = \frac{g_j(X_{t-d}^{t-1};\theta_g)\, n(X_t; a_j^T X_{t-d}^{t-1} + \mu_j, \sigma_j)}{\sum_{i=1}^{m} g_i(X_{t-d}^{t-1};\theta_g)\, n(X_t; a_i^T X_{t-d}^{t-1} + \mu_i, \sigma_i)}$$

The E- and M- steps for MixAR MLE consist of the following [10][14]:

1. Expectation Step:

   Compute $\tau_{t,j}^k$ as in previous eqn. for the k^{th} iteration

2. Maximization Step: (Re-estimation)

$$\sigma_j^{k+1} = \left(\frac{1}{\sum_{t=p+1}^{N} \tau_{t,j}^k} \sum_{t=p+1}^{N} \tau_{t,j}^k (X_t - a_j^k X_{t-d}^{t-1} - \mu_j^k)^2 \right)^{1/2}$$

$$[(a_j^{k+1})^T, \mu_j^{k+1}]^T = (R_j^k)^{-1} c_j^k$$

$$\theta_g^{k+1} = \theta_g^k + \alpha \frac{\partial Q(\theta)}{\partial \theta_g}\Big|_{\theta=\theta^k}$$

   where, α is a fixed scalar,

$$R_j^k = \sum_{t=d+1}^{N} \tau_{t,j}^k \chi_{t-1} \chi_{t-1}^T, \quad c_j^k = \sum_{t=d+1}^{N} \tau_{t,j}^k \chi_{t-1} X_t$$

   and χ_{t-1} given by $\chi_{t-1} = [(X_{t-d}^{t-1})^T, 1]^T$.

Note that the maximization step for re-estimation of gate parameters does not have a closed-form expression, and needs to use a Newton-Raphson-like gradient ascent method. With this constraint, this algorithm actually comes under a class of more generalized form of EM called the generalized-EM (GEM).

B.3 Some Implementation Issues

Nonlinear techniques are notoriously sensitive to initial conditions and convergence problems. Below we discuss several issues that have been investigated to improve the robustness of this technology.

B.3.1 EM Training: Initialization

To initialize the EM iterations, we need to initialize the MixAR parameters with reasonable values, otherwise the algorithm could converge to a very low local maximum. For this, we split the first window

of data into m sub-windows, and use the LP coefficients, mean and the prediction variance from each sub-window to initialize one component. Also, the gate parameters are all assigned equal weights by assigning a value of zero.

B.3.2 EM Training: Accumulating Statistics

The basic implementation of the MixAR EM described above can lead to excessively large memory requirements. This is because each input sample of the signal is associated with an internal state vector $\tau_{t,j}^{k}$. When training with speech signals that could often be several minutes long, this becomes impractical. To circumvent this obstacle, data is split into smaller sized windows (typically a few seconds), the required statistics are computed for each window, and then the statistics are accumulated over all the windows. Except for marginal differences, the models trained this way were identical with that on the full data for all the data files we tested this on. Hence, we used this windowed method for performing EM.

B.3.3 EM Training: Highly Unlikely Components

With some initial conditions, it is possible that during EM training one or more components of the MixAR model can have very low likelihood that they practically play no role in modeling data. Effectively this means that only a fewer number of mixtures is used, and the rest are degenerate. To solve this issue we investigated the following method:

Check for the log-likelihood of data for each component separately. If for some component this falls below a threshold, reinitialize this component by perturbing the parameters of the component with the highest likelihood, and assigning half its likelihood to both the components. Repeat this for all the components whose likelihood is lesser than the threshold. Re-run iterations with these new initial parameters, until all components have likelihoods greater than the threshold.

One problem with this is that after every such reinitialization the algorithm might take several more iterations to converge, and this problem is compounded by the possibility of more than one reinitialization.

B.3.4 Testing: Scoring

For all our experiments, we first use the simplest possible scoring method – comparison of log likelihoods (LL) of data given model. For fixed data, we assign it to the class whose MixAR has the largest LL.

While this LL scoring method is simple, it is sometimes desirable to use some form of normalization to take into account of the varying degrees to which MixAR models fit data from different classes. We are currently working on developing more robust scoring methods.

B.4 MixAR as a Pattern Classifier: Example 2-way Classification

To demonstrate the potential of MixAR modeling in pattern classification, we constructed a simple example. A 2-class problem was set up by synthesizing two threshold autoregressive (TAR) signals [15] to act as both train and test data.

Model I:

Actual Model: $X_t = \begin{cases} 0.8X_{t-1} + 1 + \varepsilon_t & X_{t-1} < 0 \\ -0.5X_{t-1} - 0.6 + \varepsilon_t & X_{t-1} \geq 0 \end{cases}$ with $\varepsilon_t \sim N(0,0.02)$.

Estimated MixAR model parameters:
$$a_1 = 0.8482, \mu_1 = 1.2200, \sigma_1 = 0.0237$$
$$a_2 = -0.4870, \mu_2 = -0.7094, \sigma_2 = 0.0233$$
$$\theta_g = [314.111, 557.312, -239.726, 213.646]^T$$

Model II:

Actual Model: $X_t = \begin{cases} 0.5X_{t-1} + 0.8 + \varepsilon_t & X_{t-1} < 0 \\ -0.7X_{t-1} - 0.2 + \varepsilon_t & X_{t-1} \geq 0 \end{cases}$ with $\varepsilon_t \sim N(0,0.04)$.

Estimated MixAR model parameters:
$$a_1 = 0.5262, \mu_1 = 1.0948, \sigma_1 = 0.0552$$
$$a_2 = -0.6721, \mu_2 = -0.2889, \sigma_2 = 0.0527$$
$$\theta_g = [-39.4071, -9.1644, 35.7413, 3.0548]^T$$

We can see that the estimated values are close to the actual ones and this also serves as a sanity check to our implementation. Even the apparent relatively large differences in the standard deviation between the actual and estimated are only due to amplitude normalization.

Next, with the estimated MixAR models, the log-likelihood score for all the four combinations of models and test signals were evaluated and are shown in Table 3.

From this table it is clear that MixAR modeling can be used to separate signals from these two different models and hence demonstrates its potential application in pattern classification.

B.5 MixAR Modeling for Speaker Recognition

A statistical model used on a any novel application would require the model to undergo some modifications and tuning to best suit the task at hand. Hence we considered it prudent to try using MixAR modeling in a very simple speaker recognition task first. To this end, we start experimentation with a sustained-phone database.

B.5.1 On Sustained-Phone Database

The sustained phone database recorded in our lab consists of 7 speakers uttering 8 phones 4 seconds long each. A speaker database was extracted from this by combining 2.9 seconds near the beginning of all the phones for training and the next 2.9 seconds for testing, for each speaker.

Table 3 – Log-likelihood scores for the 2-class problem

	Test Signal from	
	Model I	Model II
Param. I	2.33	-34.29
Param. II	-10.88	1.5

The next step was to check for convergence of the EM algorithm with number of iterations. This is achieved by a plot of the log-likelihood of data with the MixAR model trained after the end of each iteration and iteration number. The plot for one speaker for different values of parameter α is shown in Figure 2.

From this it can be deduced that a mid-range value of $\alpha = 5.0$ shows best convergence performance and at this value, MixAR EM algorithm converges around 5 iterations. We chose to run 15 iterations for each training. The variations with parameter delta were only marginal, and we fixed a value of 0.001 after some experimentation.

A popular tool used for studying performance of a speaker recognition system is the detection error tradeoff (DET) plot [16]. Two kinds of errors that plague speaker recognition systems are false alarms (an impostor is detected as the correct speaker) and missed detections (the correct speaker is rejected as an impostor). The DET curve is obtained by varying a threshold parameter and computing the number of false-alarms and missed detections at each value of the threshold. The closer the curve is towards the origin better the recognition performance is.

The DET curve for the sustained phone database with MixAR(8, 10) is given in Fugyre 3. While there is still room for improvement, this preliminary investigation shows that MixAR model holds much promise for speaker recognition.

B.5.2 On Continuous Speech: NIST-2001 database

Encouraged by our results with MixAR modeling on sustained phone database, we next turned to the popular NIST-2001 development database [16]. This contains 60 train speakers and 78 test speakers, and several utterances in the test database were recorded in different noisy conditions. Again to keep things as simple as possible in our initial stages, we decided to use the training part only for forming both the train and test data. For each speaker utterance, the first 65 seconds were used for training and the next 25 seconds for testing.

The speaker recognition performance for MixAR(8,10) with our modified NIST-2001 database is shown in Figure 4. We used the same set of algorithmic parameters for EM as for the experiment on sustained phone database. Here again, though the performance of the current MixAR system cannot compete with those from the state of the art systems, it is encouraging enough to warrant further investigations into MixAR modeling of speech.

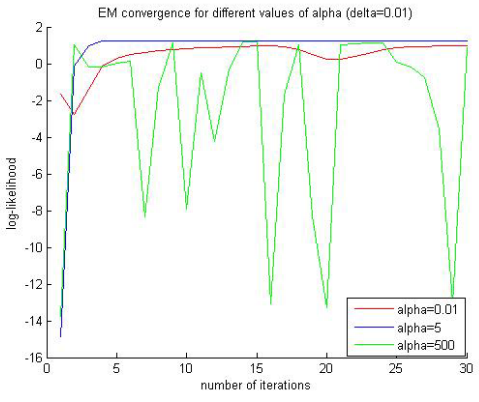

Figure 2 – EM convergence results for one speaker from sustained-phone database for different values of α for MixAR(8, 10)

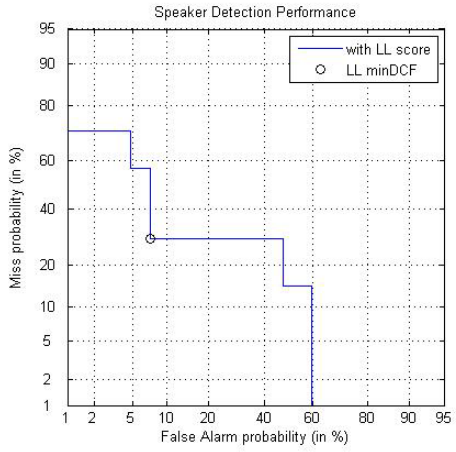

Figure 3 – DET curve for the sustained-phone database for MixAR(8; 10)

B.6 Future Directions

While we have the basic implementation of MixAR model, there are various improvements that need to be done to tune it optimally for speech. The first step along these lines is to experiment more thoroughly with the algorithmic parameters of MixAR EM for ensuring better and faster convergence. One model parameter in particular that we would like to experiment with is the gate order. It would be interesting to observe how the performance varies with increasing gate orders, starting from zero (this corresponds to a MAR model).

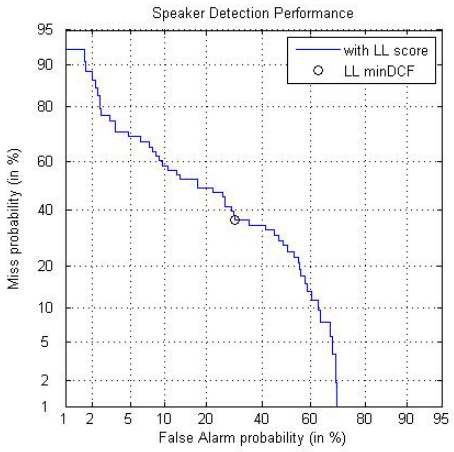

Figure 4 – DET curve for speaker recognition for modified NIST 2001 database with MixAR(8; 10)

Secondly, a better algorithm needs to be developed for dealing with components that show very low likelihoods during EM procedure. Moreover, investigations with different initialization strategies are needed to be done to check the sensitivity of the MixAR EM algorithm on speech to initial conditions.

On the testing side, more robust scoring methods must be researched upon. In addition, the effects of silence frames on both EM training and also testing have to be studied and appropriate modifications made to minimize them. Finally, the resilience to noise on the speaker recognition performance needs to be studied.

Our experience so far with MixAR suggests that after the current major concerns are addressed, MixAR can serve as a noise-robust speech model very useful in speaker recognition applications.

C. Linear Dynamic Models for Speech Recognition

Linear Dynamic Model (LDM) is a recently emerged and promising technique for speech recognition. The fundamental idea of an LDM is to describe a linear dynamic system as underlying states and observables, with an observation equation to link the internal states to the observables, and a state equation to capture the time-evolution of the states. Suppose y_t is p-dimension observation vector and x_t is q-dimension internal state vector [18][19], the LDM equations are specified as follows:

$$y_t = Hx_t + \varepsilon_t \qquad \varepsilon_t \sim N(v, C)$$
$$x_t = Fx_{t-1} + \eta_t \qquad \eta_t \sim N(w, D)$$

$$x_1 \sim N(\pi, \Lambda)$$

In these equations, H and F are linear transformation matrices; ε_t and η_t are uncorrelated white Gaussian noise to drive the linear stochastic system; sequence of observation y_t and sequence of underlying states x_t are finite dimensional and following multivariate Gaussian distribution for every specific time t. The first equation maps the output observation to the internal states. The second equation is the autogressive state process which describes how state evolves from one time-frame to the next.

In comparison to the Hidden Markov Model, in which Gaussian mixtures model the output distribution and spatial correlations present in speech parameters are frequently ignored through the use of diagonal covariance matrices, LDMs incorporates the dynamic evolution of hidden states for a segment or phoneme. On a fundamental level, we first demonstrate LDMs as a novel pattern classifier for speech recognition. After that, our sustained-phone database was chosen to run phonetic speech recognition experiments on, for which we got promising results. In the sustained-phone speech recognition experiments, LDMs shows good performance on phoneme classification especially for fricatives. As we increased the training dataset, we noticed significant performance improvements as shown by the confusion matrix. Our experience until now with LDMs suggest that we could expect LDMs to be an acoustic model competitive with the current state of the art HMM systems.

C.1 State Inference

The Kalman filter and Rauch-Tung-Striebel (RTS) smoother can be used to infer underlying system states given an N-length observation sequence y_t and model parameters of a LDM. Filtering provides an estimate of the state distribution at time t given all the observation up to and including that time, and smoothing gives a corresponding estimate of the underlying state conditioned on the entire observation [19][21][22]. All the necessary recursions are summarized below.

$$\hat{X}_{t/t} = \hat{X}_{t/t-1} + K_t e_t$$
$$\hat{X}_{t/t-1} = F\hat{X}_{t-1/t-1} + w$$
$$e_t = y_t - \hat{y}_t = y_t - v - H\hat{x}_{t/t-1}$$
$$K_t = \sum_{t/t-1} H^T \sum_{e_t}^{-1}$$
$$\sum_{e_t} = H \sum_{t/t-1} H^T + C$$
$$\sum_{t/t} = \sum_{t/t-1} - K_t \sum_{e_t} K_t^T$$
$$\sum_{t,t-1/t} = (I - K_t H) F \sum_{t-1,t-1}$$
$$\sum_{t/t-1} = F \sum_{t-1/t-1} F^T + D$$

For the smoothing part, we are using the fixed interval RTS smoother to compute the required statistics once all data has been observed. RTS smoother adds a backward pass that follows the standard Kalman filter forward recursion. In addition, in both the forward and the backward pass, we need some additional recursions for the computation of the cross-covariance [18][23]. All the necessary recursions are summarized below.

$$\hat{X}_{t-1/N} = \hat{X}_{t-1/t-1} + A_t(\hat{X}_{t/N} - \hat{X}_{t/t-1})$$
$$\sum_{t-1/N} = \sum_{t-1/t-1} + A_t(\sum_{t/N} - \sum_{t/t-1})A_t^T$$
$$A_t = \sum_{t-1/t-1} F^T \sum_{t/t-1}^{-1}$$
$$\sum_{t,t-1/N} = \sum_{t,t-1/t} + (\sum_{t/N} - \sum_{t/t}) \sum_{t/t}^{-1} \sum_{t,t-1/t}$$

Results are summarized in Figures 5 and 6.

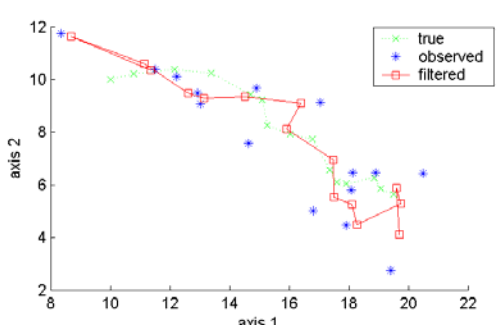

Figure 5 – State inference

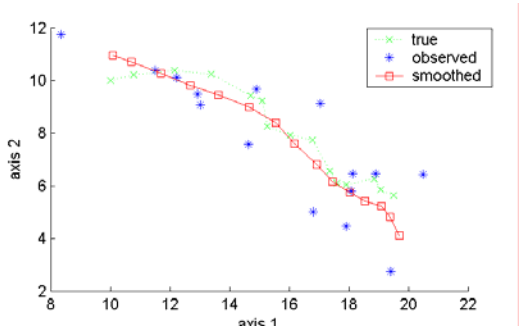

Figure 6 – State inference with RTS smoother

C.2 Parameter Estimation

For a specific LDM, the hidden nature of the state makes the parameter estimation complicated. The LDM training procedure is unsupervised learning process so that the model needs to describe the unconditional probability density of the observations. In 1982 Shumway & Stoffer provides the solution for transformation matrix H and Digalakis, Rohlicek & Ostendorf give the estimate solution for all the LDM parameters [19][26]. Assuming that there are no constrains on the structure of the transformation matrices, the parameter estimation is as follows:

$$\left[\hat{H}\ \hat{v}\right] = \left[\sum_{t=1}^{N} y_t x_t^T \ \sum_{t=1}^{N} y_t\right] \begin{bmatrix} \sum_{t=1}^{N} x_t x_t^T & \sum_{t=1}^{N} x_t \\ \sum_{t=1}^{N} x_t^T & 1 \end{bmatrix}^{-1}$$

$$\hat{C} = \frac{1}{N}\sum_{t=1}^{N} y_t y_t^T - \frac{1}{N}\sum_{t=1}^{N} y_t x_t^T \hat{H}^T - \frac{1}{N}\sum_{t=1}^{N} y_t v^T$$

$$\left[\hat{F}\ \hat{w}\right] = \left[\sum_{t=2}^{N} x_t x_{t-1}^T \quad \sum_{t=2}^{N} x_t\right] \begin{bmatrix} \sum_{t=2}^{N} x_{t-1} x_{t-1}^T & \sum_{t=2}^{N} x_{t-1} \\ \sum_{t=2}^{N} x_{t-1}^T & 1 \end{bmatrix}^{-1}$$

$$\hat{D} = \frac{1}{N-1}\sum_{t=2}^{N} x_t x_t^T - \frac{1}{N-1}\sum_{t=2}^{N} x_t x_{t-1}^T \hat{F}^T - \frac{1}{N-1}\sum_{t=2}^{1} x_t w^T$$

$$\hat{\pi} = x_1$$

$$\hat{\Lambda} = x_1 x_1^T - x_1 \pi^T$$

C.3 Implementation Issues

Below we describe issues we are facing with the EM convergence and robustness.

C.3.1 Expectation-Maximization (EM)

During an LDM training procedure, the expectation-maximization (EM) algorithm is used for finding the maximum likelihood estimates of parameters for a specific word or phone, where the model depends on unobserved latent variables [24]. EM alternates between performing an expectation (E) step, which computes an expectation of the likelihood by including the latent variables as if they were observed, and maximization (M) step, which computes the maximum likelihood estimates of the parameters by maximizing the expected likelihood found on the E step. The parameters found on the M step are then used to begin another E step, and the process is repeated [19][25].

Digalakis et al. (1992) provides the EM algorithm and how it can be used in speech recognition [18]. The E step algorithm consists of computing the conditional expectations of the complete-data sufficient statistics for standard ML parameter estimation. Therefore, the E step involves computing the expectations conditioned on observation and model parameters. The RTS smoother as described before can be used to compute the complete-data estimates of the state statistics $X_{t|N}$, $\sum_{t|N}$, and $\sum_{t,t-1|N}$. EM for LDMs then consists of evaluating the ML parameter estimates replacing x_t, $x_t x_t^T$, and $x_t x_{t-1}^T$ with their expectations [19][27].

$$E\left[x_t / y, \theta^{(i)}\right] = \hat{x}_{t/N}$$
$$E\left[x_t x_t^T / y, \theta^{(i)} g\right] = \sum_{t/N} + \hat{x}_{t/N} \hat{x}_{t/N}^T$$
$$E\left[x_t x_{t-1}^T / y, \theta^{(i)}\right] = \sum_{t,t-1/N} + \hat{x}_{t/N} \hat{x}_{t-1/N}^T$$

C.3.2 EM Convergence

In order to apply LDM as an acoustic model for speech classification, first we need to train LDM model for each phone or word. The EM training process is one of the most important procedures. Theoretically, the EM algorithm is supposed to converge as with additional iterations. The LDM model parameters can be regarded as statistically close enough to the real values once the EM recursion converges.

Our first LDM prototype is a synthetic prototype of 2-dimensional observation and 4-dimensional internal state built under Matlab. In the convergence test, we keep increasing the times of EM recursions and calculate the likelihood value of trained model after each recursion. We found that the EM likelihood curve increases quickly and will stabilize.

However, in real life the EM trained model parameters do not always converge. This is acceptable as long as the EM training converges to a point sufficiently close. We explored different dimensions of the internal state from 1 to 10 to optimize the dimension. For this specific model, 7-dimensional state performs best and has good stability as we can see from Figures 7 and 8.

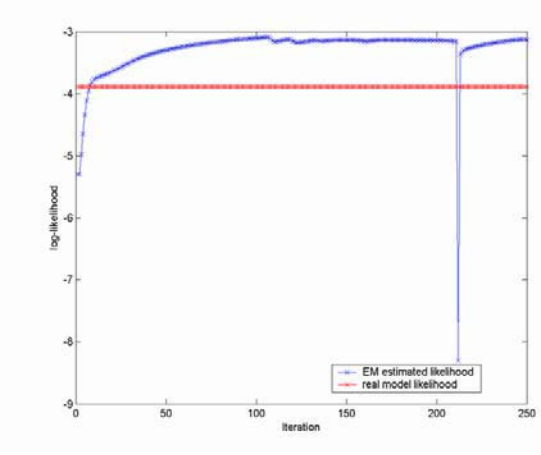

Figure 7 – EM log-likelihood evolution.

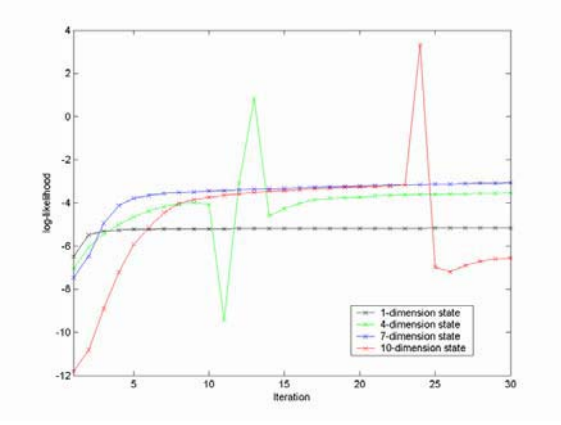

Figure 8 – EM evolution as a function of the dimension of the internal state.

C.3.3 Towards Improving Robustness

Through the course of the project, we have investigated quite a few ways to improve the robustness of the EM training process. One major problem is that matrix computation result might lead to a singular matrix after several matrix inversion steps due to computational issues. The EM likelihood curve then goes negative and is nolonger stable. In this case, we check and add a small diagonal matrix before matrix inversing and the "singular matrix" problem is solved.

Another problem is that linear transforming matrix F might get too big and negatively impacts LDM system stability as we increase the number of EM recursions. Such behavior may not be apparent over small numbers of frames, but it becomes especially important in the situation where the state is not reset between models. Here singular value decomposition (SVD) is introduced to constrain $|F| < 1$ which prohibit the exponential growth of LDM state evolution [19].

After this modification, our LDM training was quite robust and we were able to train models from any of our training data.

C.4 Two-class Synthetic Signal Classification

In order to gain insight into the effectiveness of LDM for pattern classification and speech recognition, we first explored a 2-class synthetic signal classification experiment. In this pilot experiment, we constructed 2 LDM models with a 1-dimensional observation and a 1-dimensional internal state. The parameters of Model 1 are F = [0.3], H = [0.7], C = [0.07], D = [0.06], v = [0], w = [0], pi = [5], lambda = [0.2]; parameters of Model 2 are F = [0.7], H = [0.3], C = [0.03], D = [0.02], v = [0], w = [0], pi = [7], lambda = [0.1]. We created observation vectors from these two models and ran 50 EM iterations to train the models. In the end we tested using the same observation vectors to see if LDM can classify these two signals. Though it appears to be a simple task, it might be one of the best ways to evaluate LDM and the implementation itself is a challenging task.

Table 4 shows the combinations of log-likelihood scores of two test signals and two LDM models. We can see from Figure 9 that the confusion matrix is diagonal which proves good performance. After these experiments, we tried a 2-dimensional observation and 4-dimensional hidden state model and got very similar result. It is clear that LDM modeling can be used to separate signals from these two different models and we found that LDM has a great potential for efficiency.

Table 4 – Log-likelihood scores for the 2-class problem

	Test Signal from	
	Model 1	Model 2
Param. 1	-0.2493	-1.3965
Param. 2	-6.3240	0.3499

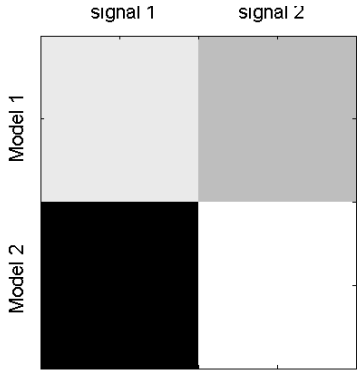

Figure 9 – 2-class confusion matrix

C.5 Sustained 8-phone Classification

Real speech signals differ with synthetic signals and usually additional work will be needed when we apply pattern recognition models to real speech. Following the synthetic signal classification experiments, we were studied LDM using our sustained phone database. The sustained phone database is composed of 7 speakers (4 male and 3 female) with eight phones recorded for each speaker. Each speaker produced sustained sounds, which is 4 seconds long for three vowels (/aa/, /ae/, /eh/), two nasals (/m/, /n/) and three fricatives (/f/, /sh/, /z/) at a sampling rate of 16,000 Hz. Standard 13-MFCC features were extracted and used for both training and testing.

Our first experiment was to train LDM phone models using speaker [sap] and use another speaker [mc] to test. We specified LDM models as 20-dimensional internal state for both training and testing. For this experiment, three fricatives (/f/, /sh/, /z/) were recognized but three vowels (/aa/, /ae/, /eh/) and two nasals (/m/, /n/) were not. The recognition accuracy was 3/8 or 37.5%. It is reasonable because we only trained LDM models using one utterance.

Next, we increased the training dataset into 5 speakers [cm], [if], [mp], [sap], [sp] and used another speaker [mc] to test. After adding 4 speakers to the trained LDM models, we observed a performance enhancement. Three fricatives (/f/, /sh/, /z/) were again recognized. For vowels, it succeeded in classifying /aa/. /ae/ was confused with /m/ and /n/ but the likelihood scores were close. /eh/ was confused with /ae/ but the likelihood values were fairly close. /m/ was confused with /n/ but the likelihood values were also close. The recognition accuracy was 4/8 or 50%. We could expect better performance if we increased the size of the training dataset.

Finally, we trained phone models using 5 speakers [cm], [if], [mp], [sap], [sp] and chose one of the speakers [sap] to test. In this preliminary experiment, the test phones were contained in the training database, so we expected very good performance. The confusion matrix is shown in Figure 10. The accuracy is summarized in Table 5. The preliminary experiments are promising, and we are now turning our attention to larger, more realistic training databases.

Table 5 – Results for the 8-class problem

Training	Testing	Correction Rate
speaker [sap]	speaker [mc]	37.5%
speakers [sap], [cm], [if], [mp], [sp]	speaker [mc]	50.0%
speakers [sap], [cm], [if], [mp], [sp]	speaker [sap]	87.5%

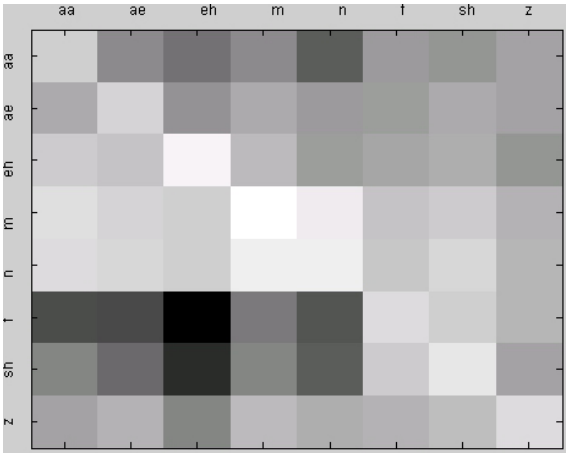

Figure 10 – 8-phone confusion matrix.

Our pilot experiments have demonstrated that LDM can be a novel pattern classifier and has very good potential for speech recognition. Until now, we only ran experiments on a sustained phone database. In order to investigate LDM for large vocabulary continuous speech recognition and speaker recognition, we will run experiments on corpora like the NIST speaker recognition corpus that we have been using in previous experiments.

D. References

[1] J. P. Eckmann and D. Ruelle, "Ergodic Theory of Chaos and Strange Attractors," *Reviews of Modern Physics*, vol. 57, pp. 617-656, July 1985.

[2] A.H. Nayfeh, B. Balachandran, *Applied Nonlinear Dynamics*, Wiley Interscience, New York, New York, USA, 1995.

[3] H. Abarbanel, *Analysis of Observed Chaotic Data*, New York: Springer-Verlag, 1996.

[4] M. Banbrook, "Nonlinear analysis of speech from a synthesis perspective," *PhD Dissertation,* The University of Edinburgh, Edinburgh, UK, 1996.

[5] H. Kantz and T. Schreiber, *Nonlinear Time Series Analysis*, Cambridge University Press, UK, 2003.

[6] B. Mandelbrot, *The Fractal Geometry of Nature*, W. H. Freeman & Co, 1982.

[7] T. M. Cover and J. A. Thomas , *Elements of Information Theory*, Wiley, New York, 1991.

[8] N. Parihar, "Performance Analysis of Advanced Front Ends on the Aurora Large Vocabulary Evaluation," *MS Thesis*, Mississippi State University, Starkville, MS, 2003.

[9] X. Huang, A. Acero and H. W. Hon, *Spoken Language Processing: A Guide to Theory, Algorithm and System Development*, Prentice Hall, NJ, 2001.

[10] A. Zeevi, R. Meir and R. Adler, "Non-linear models for time series using mixtures of autoregressive models," *Technical report*, Technion and Stanford University, 2000.

[11] C.S. Wong and W.K. Li, "On a Mixture Autoregressive Model," *Journal of Royal Statistical Society*, vol. B62, pp. 91-115, 2000.

[12] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*, Prentice Hall, NJ, 1993.

[13] A. P. Dempster, N. M. Laird and D. B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of Royal Statistical Society*, vol. B39, pp. 1-38, 1977.

[14] M. Jordan and R. Jacobs, "Hierarchical Mixture of Experts and the EM algorithm," *Neural Computation*, vol. 6, pp 181-214, 1994.

[15] H. Tong, *Nonlinear Time Series: A Dynamical Systems Approach*, Oxford University Press, Oxford, 1990.

[16] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki, "The DET Curve in Assessment of Detection Task Performance," *Proceedings of Eurospeech Conference,* Rhodes, Greece, pp. 1895-1898, Sep 1997.

[17] "NIST 2001 Speaker Recognition Evaluation Plan," *http://www.nist.gov/speech/tests/spk/2001/doc/index.htm*.

[18] V. Digalakis, "Segment-based stochastic models of spectral dynamics for continuous speech recognition," *Ph.D. dissertation*, Boston University Graduate School, Boston, MA, 1992.

[19] J. Frankel, "Linear dynamic models for automatic speech recognition," *Ph.D. dissertation*, The Centre for Speech Technology Research, University of Edinburgh, Edinburgh, UK, 2003.

[20] A. Rosti and M. Gales, "Generalized linear Gaussian models," Cambridge University Engineering, *Technical Report*, CUED/F-INFENG/TR.420, 2001.

[21] R. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, pp. 35–44, March 1960.

[22] S. Roweis and Z. Ghahramani, "A unifying review of linear Gaussian models." *Neural Computation*, vol. 11, no. 2, 1999

[23] H. E. Rauch, "Solutions to the linear smoothing problem," *IEEE transactions on Automatic Control*, vol. 8, pp. 371–372, 1963.

[24] J. Bridle, L. Deng, J. Picone, H. Richards, J. Ma, T. Kamm, M. Schuster, S. Pike, and R. Reagan, "An investigation of segmental hidden dynamic models of speech coarticulation for automatic speech recognition," *Workshop on Language Engineering, Center for Language and Speech Processing at Johns Hopkins University, Tech. Rep.*, 1998.

[25] J. Picone, S. Pike, R. Regan, T. Kamm, J. Bridle, L. Deng, Z. Ma, H. Richards, and M. Schuster, "Initial evaluation of hidden dynamic models on conversational speech," *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, Phoenix, AZ, 1999, pp. 109–112.

[26] V. Digalakis, J. Rohlicek, and M. Ostendorf, "ML estimation of a stochastic linear system with the EM algorithm and its application to speech recognition," *IEEE Trans. Speech and Audio Processing*, vol. 1, no. 4, pp. 431–442, October 1993.

[27] K. Iso, "Speech recognition using dynamical model of speech production," *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, Minneapolis, MN, 1993, pp. 283–286.