Final Report for Period: 09/2008 - 02/2009 **Submitted on:** 05/31/2009

Principal Investigator: Picone, Joseph . **Award ID:** 0414450

Organization: Mississippi State Univ

Submitted By:

Picone, Joseph - Principal Investigator

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: Yes

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: Yes

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 was a partial sponsor of this work by providing a supplement for the first year of the award. Dr. Picone traveled to DoD for a one-year Intergovernmental Personnel Action (IPA) starting January 1, 2005, to strengthen this collaboration. Dr. Georgios Lazarou, with whom Dr. Picone closely collaborates, became the project PI once Dr. Picone began his IPA.

As his responsibilities managing human language technology (HLT) research and development grew in his first year at DoD, the IPA was extended two more years, allowing him an opportunity to have a significant impact on HLT within DoD. Dr. Lazarou continued to be the project PI during this time. Dr. Picone continued to provide consulting as needed to Dr. Lazarou while he was at DoD.

Dr. Picone eventually returned to MS State on January 1, 2008, and continued his collaborations with Dr. Lazarou on this (and other) projects. Dr. Lazarou was forced to leave MS State in early 2007 due to the university's denial of his promotion and tenure application. He has remained active in the project even though he is no longer employed by MS State.

Throughout the project we have maintained a close working relationship with DoD and have transfered software at various points in this project. We try to address DoD needs that arise that are relevant 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.

In 2008, we developed a phone-spotting system to support some experiments with nonlinear modeling (the work on probabilistic MAR models). We have put this system into the public domain, making it one of the first phonetic keyword search systems that is available as open source software. The motivation for releasing this software was based, in part, on DoD's interest in this technology.

We also developed a keyword search system in summer of 2008 to serve as a baseline system for DoD's evaluation of spoken term detection technology. Our attempts to apply nonlinear statistical modeling of speech to speech processing problems of significant scale have utilized many of the software components used to build these systems. DoD has access to all this technology as part of our standard software releases.

Other Collaborators or Contacts

Our research group has a long tradition of supporting a public domain speech and signal processing toolkit that includes all of the software developed in this project. We actively support a number of users of this software through our open source distribution. The research conducted within this project, including the various system configurations, experiments, and underlying libraries, have been integrated into our standard software release.

Activities and Findings

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

The primary goal of this project was to develop novel nonlinear modeling techniques for speech and speaker recognition systems. There were three significant outcomes from this project. First, we demonstrated a statistically significant improvement in speech recognition performance by augmenting the traditional speeech recognition feature vector with features derived from estimates of the degree of nonlinearity in the speech signal. Second, we demonstrated that a new acoustic modeling technique based on a nonlinear mixture of autoregressive models can provide comparable performance to traditional approaches with a significant reduction in the number of parameters in the model. Third, in work that is still in its preliminary stages, we demonstrated modest improvements in performance on limited tasks using a linear dynamic model. These findings, along with a number of other attempts to introduce nonlinear statistical models into a traditional hidden Markov model-based speech processing approach, are described in this report.

The report is attached.

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.

Demonstrated a small improvement in feature analysis using a combination of standard features and nonlinear features.

Demonstrated a small, but consistent improvement on a pilot database for a new acoustic modeling approach based on a probabilistic mixed autoregressive model.

Demonstrated a small, but consistent improvement in speech recognition performance using a linear dynamic model.

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.

Introduced the students to a Java-based interactive development environment, Eclipse, that is rapidly becoming an industry-standard environment.

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 (MSMS). 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 two seminars at the MSMS Math Club that focused on human language technology and briefly mentioned this research project. These presentations emphasized the value of math in the engineering disciplines. The talks are available at:

J. Picone, 'Can You Say Hamburger in 6,000 Languages?' The Mississippi School For Mathematics and Science, October 02, 2008. (http://www.isip.piconepress.com/publications/seminars/external/2008/msms/)

J. Picone, 'National Security By The Numb3rs,' The Mississippi School For Mathematics and Science, August 31, 2006. (http://www.isip.piconepress.com/publications/seminars/external/2006/msms/)

Feedback from the students was very positive.

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. , vol. , (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,

S. Srinivasan, T. Ma, D. May, G. Lazarou and J. Picone, "Nonlinear Mixture Autoregressive Hidden Markov Models For Speech Recognition", Proceedings of INTERSPEECH, p. , vol. 1, (2008). Published,

D. May, T. Ma, S. Srinivasan, G. Lazarou and J. Picone, "Continuous Speech Recognition Using Nonlinear Dynamic Invariants", Proceedings of INTERSPEECH, p. , vol. , (2008). Rejected, but available from our project web site,

T. Ma, S. Srinivasan, D. May, G. Lazarou and J. Picone, "Robust Speech Recognition Using Linear Dynamic Models", Proceedings of INTERSPEECH, p. , vol. , (2008). Rejected, but available from our project web site,

S. Raghavan, G. Lazarou and J. Picone, "Speaker Verification Using Support Vector Machines", Proceedings of IEEE SoutheastCon, p. 188, vol. 1, (2006). Published,

T. Ma, S. Srinivasan, D. May, G. Lazarou and J. Picone, "Robust Speech Recognition Using Linear Dynamic Models", IEEE Signal Processing Letters, p. , vol. , (2009). Submitted,

S. Srinivasan, T. Ma, D. May, G. Lazarou and J. Picone, "Nonlinear Statistical Modeling of Speech", 29th International Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering (MaxEnt 2009), p. , vol. , (2009). Accepted,

S. Srinivasan, T. Ma, D. May, G. Lazarou and J. Picone, "A Nonlinear Mixture Autoregressive Model for Speaker Recognition", International Conference on Spoken Language Processing (INTERSPEECH), p. , vol. , (2009). Submitted,

Books or Other One-time Publications

J. Picone, A. Ganapathiraju and J. Hamaker, "Applications of Kernel Theory to Speech Recognition", (2007). Book Chapter, Published
Editor(s): Idea Group Inc., USA
Collection: Kernel Methods in Bioengineering, Communications, and Signal Processing
Bibliography: J. Picone, et al., "Applications of Kernel Theory to Speech Recognition," in G. Gustavo, et al., (Eds.), "Kernel Methods in Bioengineering, Communications and Image Processing,"

J. Picone, "Statistical Optimization in Speech Recognition", (2009). research monograph, under development
Bibliography: under development

D. May, "Nonlinear Dynamic Invariants For Continuous Speech Recognition", (2008). Thesis, Published
Bibliography: D. May, Nonlinear Dynamic Invariants For Continuous Speech Recognition, M.S. Thesis, Department of Electrical and Computer Engineering, Mississippi State University, May 2008.

S. Srinivasan, "Nonlinear Mixture Autoregressive Hidden Markov Models For Speech Recognition", (2010). Thesis, under development
Bibliography: MS State Dissertation

T. Ma, "Robust Speech Recognition Using Linear Dynamic Models", (2010). Thesis, under development
Bibliography: MS State dissertation

Web/Internet Site

URL(s):
http://www.isip.piconepress.com/projects/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.isip.piconepress.com/whats_new/archives/2005/2005_06_nonlinear/

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.isip.piconepress.com/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.isip.piconepress.com/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.isip.piconepress.com/projects/nsf_nonlinear/downloads/software/matlab/

Contributions

Contributions within Discipline:

(1) We combined traditional MFCCs with nonlinear dynamic invariants to produce a more robust feature vector for speech processing (specifically speaker verification, speaker identification and speech recognition). This new feature vector exploits the underlying nonlinear dynamic properties that traditional linear techniques fail to capture. We performed a set of phoneme classification experiments using these new features and saw a maximum relative improvement of 10.3% for certain phoneme types. Evaluations of the Aurora-4 continuous speech recognition corpus show a maximum relative increase of 11.1% for the clean evaluation set. However, an average relative decrease of 7.6% was observed for the data sets containing noise.

(2) We proposed the use of Linear Dynamic Models (LDMs) as an alternative to Hidden Markov Models (HMMs) for robust speech recognition in noisy environments. In speech recognition, HMMs typically assume a diagonal covariance matrix where correlations between feature vectors for adjacent frames are ignored. LDMs use a state space-like formulation that explicitly models the evolution of hidden states using an autoregressive process. This smoothed trajectory model allows the system to better track the speech dynamics in noisy environments. We demonstrate that LDMs provide a 4.9% relative improvement on the Aurora-4 clean evaluation set, and a 6.5% relative improvement on the noisy evaluation set.

(3) Gaussian mixture models (GMMs) are a very successful method for modeling the distribution of speaker features. In this approach, the dynamics of the speech spectrum are typically encapsulated in the feature vector through the use of derivatives. This model is limited by the assumption that the dynamics of speech features are linear and can be modeled with static features and their derivatives. In this paper, a nonlinear mixture autoregressive model (MixAR) is used to model speaker features. Experiments show that MixAR performs better than a GMM when the signal contains strong evidence of nonlinear behavior. On the 2001 NIST Speaker Recognition Evaluation Corpus, MixAR is shown to lower the equal error rate by 10.6% relative and uses significantly fewer parameters than GMM.

(4) 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, and found they provide no significant improvements in performance. Therefore, we did not pursue particle filtering and Kalman filtering.

(5) We have provided reference implementations of Lyapunov exponents, correlation dimension, and the embedding dimension for use in speech recognition experiments. We have evaluated these features on both speaker recognition and speech recognition tasks. We performed a detailed analysis and optimization of several key parameters associated with these features so that they can be of general use to speech processing systems.

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. The software was developed and released in a manner that makes it useful for general signal processing.

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. The first, Ryan Irwin, is at Virginia Tech pursuing a Ph.D. in communications. The second student, Wesley Holland, received a graduate fellowship and completed his M.S. degree at Mississippi State University in a different area.

A third student, Daniel May, who began working on the project as an undergraduate, continued into graduate school on this project and successfully defended M.S. thesis in May 2008. He began employment at SAIC in April 2009 and will hopefully be working on human language technology projects once his security clearance is completed.

Contributions to Resources for Research and Education:

A project web site with numerous tutorial materials has been developed and maintained. Many of these materials are used in various undergraduate and graduate courses in signal processing.

For example, our popular Java applet on pattern recognition:

http://www.isip.piconepress.com/projects/speech/software/demonstrations/applets/util/pattern_recognition/current/index.html

was extended to include Kalman filtering and particle filtering.

Contributions Beyond Science and Engineering:

Though we have engaged and supported a number of companies on related technology, such as our software toolkit, we have not directly engaged them on nonlinear statistical modeling. We are currently looking mainly at more traditional maximum likelihood methods.

Conference Proceedings

Categories for which nothing is reported:

Any Conference

FINAL REPORT: RESEARCH ACTIVITIES

The primary goal of this project was to develop novel nonlinear modeling techniques for speech and speaker recognition systems. There were three significant outcomes from this project. First, we demonstrated a statistically significant improvement in speech recognition performance by augmenting the traditional speeech recognition feature vector with features derived from estimates of the degree of nonlinearity in the speech signal. Second, we demonstrated that a new acoustic modeling technique based on a nonlinear mixture of autoregressive models can provide comparable performance to traditional approaches with a significant reduction in the number of parameters in the model. Third, in work that is still in its preliminary stages, we demonstrated modest improvements in performance on limited tasks using a linear dynamic model. These findings, along with a number of other attempts to introduce nonlinear statistical models into a traditional hidden Markov model-based speech processing approach, are described in this report.

A. Historical Perspectives on Acoustic Modeling in Speech Recognition

Statistical or machine-learning techniques, such as Hidden Markov models (HMMs) and Gaussian mixture models (GMMs), have dominated the signal processing and pattern recognition literature for the past 25 years. However, such approaches are prone to overfitting and have problems with generalization. For example, delivering high performance on previously unseen noise conditions remains an elusive goal. A lack of robustness to previously unseen conditions is a major impediment to the success of human lnaguage technology in many important application spaces, particularly those in the Department of Defense. In this final report, we will review our attempts to advance state of the art by applying principles of nonlinear statistical modeling to acoustic modeling in speech recognition and speaker verification.

A typical pattern recognition approach to speech recognition based on a Bayesian model [1] is shown in Figure 1. The acoustic front-end and the acoustic model, which are the focus of this research, are used to compute the maximum likelihood contribution of the overall posterior probability. The language model is used to compute prior probabilities and is not the subject of this research. However, the techniques described in this report can be applied to language modeling research, something we hope to investigate in future research. The search component ties all these models together by searching a very large space for the most probable sequence of words (or symbols), and is also not the subject of this research.

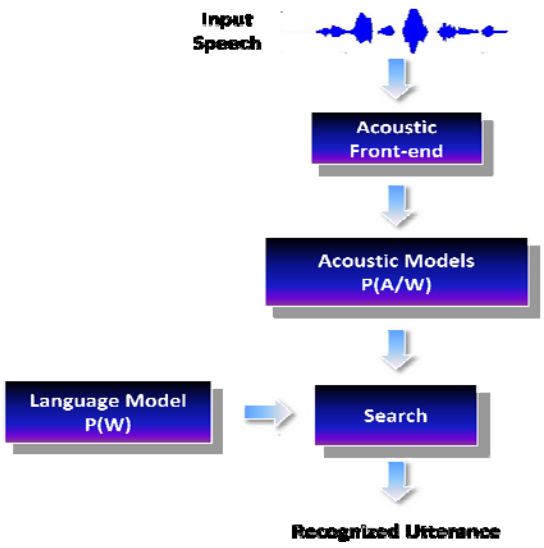

Figure 1. An overview of a typical pattern recognition approach to speech recognition. In this project, we have focused on the acoustic front-end and acoustic modeling.

There are many aspects of the speech problem that make it much more difficult than most pattern recognition problems. First and foremost, there is the issue of segmentation – the begin and end times of units are unknown. This causes an exponential growth in the search space because every word that can begin a sentence or phrase must be hypothesized every frame. Naturally, suboptimal search techniques must be employed to deal with the problem. Accurate segmentation is intimately related to the performance of a system – recognition errors are usually the result of poor segmentation. Modern statistical approachses to speech recognition can be viewed as very powerful utterance detection algorithms, because

differentiation between speech and non-speech is ultimately performed as part of the recognition process.

Second, speakers often delete or poorly articulated sounds when speaking casually, as described in Figure 2. High performance speech recognition requires the use of some form of probabilistic model that takes into account the possibility of missing phonemes. Often this is done through a combination of a lexicon containing multiple pronunciations for each word [2] and a finite state transducer that represents all anticipated pronunciations for the word.

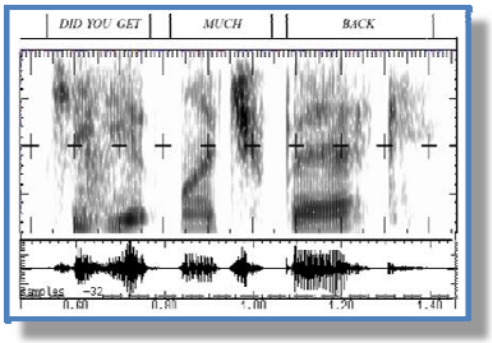

Figure 2. Common phrases, such as "Did you get" are often significantly reduced in conversational speech. Approximately 12% of the expected phonemes and 1% of the expected syllables are deleted in conversational speech. Pronunciation models must account for such ommissions, and routinely do this as part of the Bayesian approach.

A third aspect of the speech problem that must be addressed by any algorithmic approach is the inherent ambiguity of features. This is depicted in Figure 3. Though early studies described the ability to separate vowels based on formant frequency locations for data collected under controlled conditions [3], in conversational speech the raw feature measurements are highly ambiguous, leading to high error rates when classification is done based soley on feature measurements. The classical approach to such problems is to use conditional probabilities where measurements are conditioned on adjacent temporal events and linguistic or domain knowledge, thereby reducing the overlap between features. Modern approaches to speech recognition essentially exploit this simple principle at many levels of the information hierarchy. For example, derivatives of features are used to add additional information about the change in the spectrum. These derivatives are computed over large segments of acoustic data, often 100 ms or more (11 frames at 10 ms per frame). This provides significantly more acoustic context for each feature vector, and produces more discrimination between broad phonetic classes (e.g., vowels, which are steady-state sounds vs. consonsants which are dynamic sounds). This approach will form an important contrasting condition for our nonlinear models.

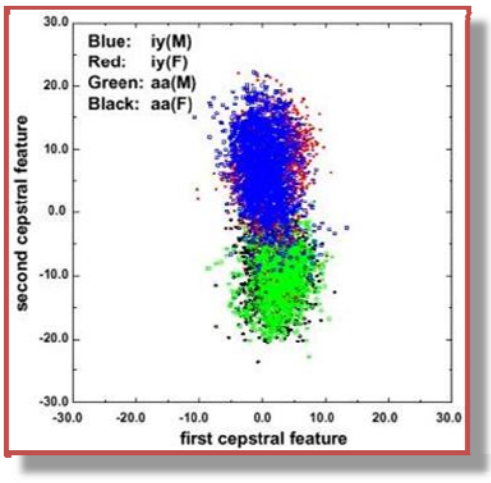

Figure 3. Features used in speech recognition systems exhibit significant overlap in the feature space. Disambiguating these classes requires the use of additional acoustic context.

There are many other forms of such acoustic normalization that play into state of the art systems, including vocal tract length normalization [4] and mean and variance normalization [5]. At a high level, these approaches can be regarded as providing long-term acoustic context based on speaker and channel characteristics.

Similarly, the use of a context-dependent phone as an acoustic model is another approach to adding additional acoustic context. Context-dependent phones model the current phone as a function of the preceding and following phones. The most common phone model is a triphone, which uses a left and right context of one phone. Context-dependent phones significantly increase the complexity of a recogntion system because the inventory of approximately 40 to 60 phones required for a given language is expanded to typically about 10,000

acoustic phone models to capture the most frequently occuring triphones.

Finally, a similar approach at the language model level is the application of the N-gram language model, which is a statistical language model that captures information about sequences of N words. Bigram (N = 2) and trigram (N=3) models [6] are most common in the early stages of the recognition process, though much higher-order models are used in subsequent rescoring passes.

It is important to understand that any posterior probability computed at the frame level in a typical speech recognition system is ultimately conditioned on a combination of all of these contexts: N-gram language models, pronunciation models, context-dependent acoustic models and derivatives of features. Those these techniques provide moderate improvements in performance, and account for many of the advances in speech recognition performance over the past 20 years, these techniques also increase the system's dependency on training data. Not only do these approaches require vast amounts of training data, but more importantly, they tend to make the recognizer more sensitive to mismatches between the training and evaluation data. (Conversely, training across large, diverse sets of data tends to make the models less powerful for a given application.) Improving robustness to this mismatch is a central goal of this work.

B. Motivations for Nonlinear Statistical Models

There is a fundamental problem with any frame-synchronous system based on Bayesian principles: each frame of data contributes to the overall likelihood score. When the system encounters sections of speech that do not match the model well, large negative log-likelihoods are generated, and these tend to dominate the overall utterance likelihood score. An heuristic workaround for this is to limit the amount any single frame can contribute to the overall likelihood. However, we seek a more fundamental solution that avoids the need to accurately decode each frame of data.

Ideally, in a process similar to that which humans use, scores for detecting phonemes should be dominated by the sections of the model for which there is high confidence, and the sections for which matches are poor, should be ignored or discounted. One could argue this is similar to using a confidence score, but as will be seen shortly, our approach is radically different.

The original inspiration for this work was based on a revolutionary device referred to as a phase-locked loop (PLL) [7]. One example of such a device is shown in . A PLL was one of the first truly nonlinear devices of its type to make a large impact within the electronics community on detection problems. It is able to lock on to the frequency or phase of a signal in a remarkably robust manner. It is robust to changes in the operating environment or signal conditions, ambient noise, and does not require sequential decoding of the signal.

Its properties are deeply rooted in nonlinear control systems theory. It was one of the first nonlinear devices to be extensively studied, and helped progress the field of nonlinear system theory. A PLL's asymptotic behavior can only be approximated using state space theory. Our

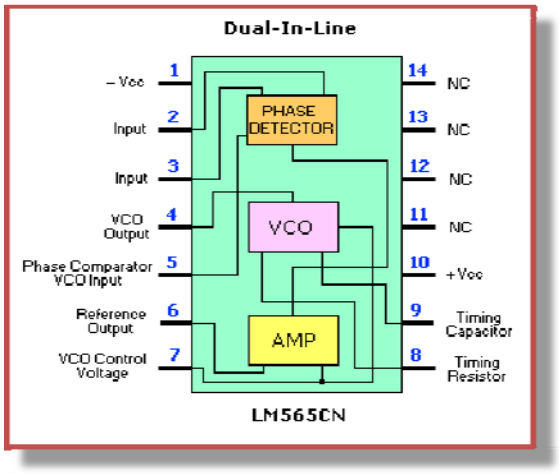

Figure 4. A phased-locked loop is a nonlinear device that is robust to unexplained variations in the input signal. Over time it synchronizes with the input signal without the need for extensive offline training [8].

initial goal was to build phone detectors that functioned much like PLLs.

PPLs are interesting in the context of speech recognition because they do not require extensive offline training. They are an instantiation of an adaptative system that uses feedback to minimize an error signal representating the difference in phase between the input and the reference signal. They are very inexpensive and have been shown to be remarkably robust to noise and other common problems speech processing systems face.

Figure 5. A strange attractor is a set of points or region which bounds the long-term, or steady-state behavior of a chaotic system. Systems can have multiple strange attractors, and the initial conditions determine which strange attractor is reached.

A second desirable property of nonlinear systems that is relevant to this project is strange attraction [9]. A strange attractor is a set of points or region which bounds the long-term, or steady-state behavior of a chaotic system. Systems can have multiple strange attractors, and the initial conditions determine which strange attractor is reached. Nonlinear systems are able to attain a number of behaviors from the same system model through this property. In contrast, conventional Gaussian mixture modeling must enumerate each mode or state of the system separately, increasing complexity and making the model more fragile to unseen analysis conditions.

We had hoped to exploit the property of strange attraction in two ways. First, we sought a model in which the phonetic targets, or hidden states, could be implemented as a strange attractor. Second, we had hoped to collapse context-dependent phone models for a given phone into a single phone model in which the attractors represented the context-dependent realizations of these sounds. Hence, our goal was to solve two fundamental problems with one model: robust frame synchcronous decoding and context-dependency. We had hoped this would result in a model more robust to unseen training conditions, and a model that required fewer parameters. We refer to these two attributes as robustness and parsimony.

An overarching challenge in this project has been dealing with the nonstationarity of the speech signal. We have explored several models that, if given ample amounts of data, appear capable of modeling nonlinar behavior. The parameter estimates of these models converge over time scales of seconds. However, the speech signal varies on the order of 10 to 30 msec. Some phonemes crucial to good recognition and verification performance last only a few tens of milliseconds, on the order of a few frames of data. We have yet to find success at estimating parameters of these nonlinear models over such short durations of speech. In such cases, one only has a few hundred samples with which to compute the model parameters. Worse, coarticulation phenomena often impinge on these samples, making it very difficult to estimate phonemes with short duration. This is one of the great challenges of speech processing – estimating model parameters from a signal that is evolving on very short time scales. We even attempted to oversample the signal in an effort to increase the amount of data, but even this approach suffers from some well-known drawbacks.

One of the reasons we shifted the focus of the original proposal from recognition to verification was to provide a longer time epoch over which parameters could be estimated. Unfortunately, though in verification you have access to a long speech utterance, the fact that the signal model is continually evolving during this time makes it difficult to separate out speaker variations from phoneme variations. Nonlinear models attempting to directly estimate long-term suprasegmental parameters related to the speaker's identity also did not perform well.

Therefore, we ultimately focused on approaches that blend well with the traditional, piecewise linear approximation that has been so successful in speech recognition. We also attempted to leverage as much of the hidden Markov model infrastructure as possible. We focused on three approaches: (1) an extension of the standard speech recognition feature vector that includes estimates of the nonlinear nature of the signal, (2) a nonlinear mixture autoregressive model and (3) a linear dynamic model. A summary of our work in these areas is described below.

C. Continuous Speech Recognition with Nonlinear Dynamic Invariants

The information in this section is covered in greater detail in D. May's M.S. thesis [10] and several related conference publications. Here we summarize the major findings.

For the past several decades, acoustic modeling for speech recognition has been based on the source-filter model and one-dimensional wave propagation in the vocal tract. The signal processing techniques that parameterize acoustic speech data into features operate primarily in the signal's frequency domain. This approach models the vocal tract as a linear filter and captures the lower-order characteristics of the speech production process. Recent theoretical and experimental evidence has suggested the existence of nonlinear characteristics in different types of speech and that that these characteristics contain significant information about speech production. While the traditional linear representation of speech has shown to be a reasonable means of acoustic modeling, it fails to capture this higher-order information of the acoustic dynamic system [10]-[12].

Dynamic systems can be represented by phase space models, where the states of the system evolve in accordance with a deterministic evolution function, and the measurement function maps the states to the observables. The path traced by the system's states as they evolve over time is referred to as a *trajectory*. An *attractor* is defined as the set of points in the state space that are accumulated in the limit as $t \rightarrow \infty$. *Invariants* of a system's attractor are measures that quantify the topological or geometrical properties of the attractor and do not change under smooth transformations of the space. These smooth transformations include coordinate transformations such as phase space reconstruction of the observed time series [13].

Dynamic invariants are a natural choice for characterizing the system that generated the observable. These measures have been previously studied in the context of analysis and synthesis research [13][14] and more recently in the context of speech recognition [15]. Our work began with a thorough analysis of these invariants and their ability to discriminate between different types of speech signals [16]. Using a small database of elongated pronunciations of phones, we measured the between-class separation in a feature space comprised of these invariants and found that they were capable of discriminating between sustained phones. In Figure 6, we show the phase space trajectories of several phonemes. We were encouraged by the relative smoothness of the trajectories for vowels, and the randomness exhibited for sibilants. General properties of these trajectories should be relatively invariant to changes in the acoustic channel or speaker, a very desirable property of a potential feature for speech processing.

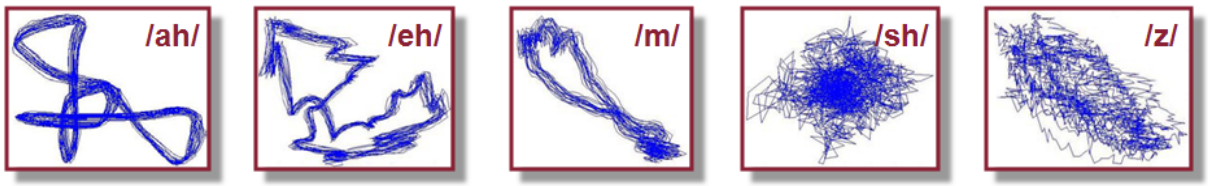

Figure 6. The phase space trajectories of several phonemes. Our goal was to compute a measure of the complexity of the phase space and to use this measure to improve discrimination. For example, a measure of trajectory smoothness should be relatively invariant to changes in acoustic channel or speaker.

The majority of our work on this project focused on our analysis of three standard dynamic invariants that are based on properties of the phase space: Lyapunov exponents, fractal dimension, and Kolmogorov entropy. Lyapunov exponents [17] associated with a trajectory provide a measure of the average rates of convergence and divergence of nearby trajectories. Fractal dimension [18] is a measure that quantifies the number of degrees of freedom and the extent of self-similarity in the attractor's structure. Kolmogorov entropy [18] defined over a state-space, measures the rate of information loss or gain over the trajectory. These measures search for a signature of chaos in the observed time series. Since these measures quantify the structure of the underlying nonlinear dynamic system, they are prime candidates for feature extraction of a signal with strong nonlinearities. The motivation behind studying such invariants from a signal processing perspective is to capture the relevant nonlinear dynamic information from the time series – something that is ignored in conventional spectral-based analysis.

Recent work has shown that the combination of fractal dimension with Mel-frequency cepstral coefficients (MFCCs) improves recognition performance for speech contaminated with noise [19]. This provides sufficient motivation for an investigation into additional dynamic invariants. We combined the three invariants mentioned above with the traditional MFCCs to create a new feature vector that exploits both the linear acoustic model and the nonlinear dynamic information of the signal. We used this new feature vector to evaluate the Aurora-4 large vocabulary evaluation corpus and compare the recognition accuracy to a system using only MFCCs.

C.1 Nonlinear Dynamic Invariants

Nonlinear systems can best be represented by their phase space which defines every possible state of the system. The dimensions of the phase space correspond to the system's dynamic variables, and each point in the space corresponds to a unique state of the system. To characterize the structure of the underlying strange attractor from an observed time series, it is necessary to reconstruct a phase space from the time series. This reconstructed phase space captures the structure of the original system's attractor (the true state-space that generated the observable). The process of reconstructing the system's attractor is commonly referred to as embedding.

The simplest method to embed scalar data is the method of delays. In this method, the pseudo phase-space is reconstructed from a scalar time series, by using delayed copies of the original time series as components of the RPS. It involves sliding a window of length m through the data to form a series of vectors, stacked row-wise in the matrix. Each row of this matrix is a point in the reconstructed phase-space. Letting $\{x_i\}$ represent the time series, the reconstructed phase space (RPS) is represented as:

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

where m is the embedding dimension and τ is the embedding delay. Taken's theorem [17] provides a suitable value for the embedding dimension, m. The first minima of the auto-mutual information versus delay plot of the time series is a safe choice for embedding delay [17].

We experimented with a number of techniques for smoothing or postprocessing the phase space. Our best overall results were obtained using SVD embedding [10].

C.1.1 Lyapunov Exponents

The analysis of separation in time of two trajectories with infinitely close initial points is measured by Lyapunov exponents [17]. For a system whose evolution function is defined by a function *f*, we need to analyze

$$\Delta x(t) \approx \Delta x(0) \frac{d}{dx}(f^N)x(0) \ . \tag{2}$$

To quantify this separation, we assume that the rate of growth (or decay) of the separation between the trajectories is exponential in time. Hence we define the exponents, λ_i as

$$\lambda_i = \lim_{n\to\infty} \frac{1}{n} \ln(\mathrm{eig_i} \prod_{\mathrm{p}=0}^{\mathrm{n}} \mathrm{J(p)}) \ , \tag{3}$$

where, J is the Jacobian of the system as the point p moves around the attractor. These exponents are invariant characteristics of the system and are called Lyapunov exponents, and are calculating by applying (3) to points on the reconstructed attractor. The exponents read from a reconstructed attractor measure the rate of separation of nearby trajectories averaged over the entire attractor and quantify the level of chaos present in the attractor. Attractors corresponding to chaotic systems will generally have high Lyapunov exponents while the exponents from more stable, periodic systems will have lower exponents. Through experimentation, it was found that an embedding dimension of 5 since the Lyapunov spectra converge at 5 over a range of embedding dimensions. A more detailed explanation of this and other parameter values can be found in [16].

C.1.2 Fractal Dimension

Some geometrical objects have a characteristic called self-similarity. An object is characterized as self-similar if a close-up examination of the object reveals that it is composed of smaller versions of itself. Self-similarity in a geometrical structure can be quantified and defines the degree to which it occupies a space. This value is called fractal dimension.

Correlation dimension [18] is a popular choice for numerically estimating the fractal dimension of an attractor. The power-law relation between the correlation integral of an attractor and the neighborhood radius of the analysis hyper-sphere can be used to provide an estimate of the fractal dimension:

$$D = \lim_{N\to\infty} \lim_{\varepsilon\to 0} \frac{\partial \ln C(\varepsilon)}{\partial \ln \varepsilon} \ , \tag{4}$$

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

$$C(\varepsilon) = \frac{2}{N*(N-1)} \sum_{i=1}^{N} \sum_{j=i+1}^{N} \Theta(\varepsilon - \left\| \vec{x}_i - \vec{x}_j \right\|) , \tag{5}$$

where $\vec{x}$ is one of *N* points on the attractor. The correlation integral is essentially a measure of the number of points within a neighborhood of radius ε averaged 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 [18].

C.1.3 Kolmogorov Entropy

Entropy is a well known measure used to quantify the amount of disorder in a system. It has also been associated with the amount of information stored in general probability distributions. Numerically, the Kolmogorov entropy can be estimated as the second order Renyi entropy (K_2) and can be related to the correlation integral of the reconstructed attractor [18] as:

$$C_d(\varepsilon) \sim \lim_{\substack{\varepsilon \to 0 \\ d \to \infty}} \varepsilon^D \exp(-\tau\, d\, K_2), \tag{6}$$

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 leads to the relation

$$K_2 \sim \frac{1}{\tau} \lim_{\substack{\varepsilon \to 0 \\ d \to \infty}} \ln \frac{C_d(\varepsilon)}{C_{d+1}(\varepsilon)}, \tag{7}$$

In practice, the values of ε and d are restricted by the resolution of the attractor and the length of the time series. We found that an embedding dimension 15 gives consistent estimations of Kolmogorov entropy [16].

C.2 Phoneme Classification Experiments

We combined the traditional 39 dimensional MFCC feature vector (consisting of 12 MFCCs, absolute energy, and their first and second derivatives) with nonlinear dynamic invariants and evaluate this combination on the Wall Street Journal derived Aurora-4 large vocabulary evaluation corpus. This corpus represents a well-established LVCSR benchmark and constitutes a balanced trade-off between computational resources and complexity. Also, the limited 5,000 word vocabulary makes this corpus conducive to acoustic modeling research. The subset of the corpus used for our experiments is divided into a training set and seven evaluation sets. The training set consists of 7,138 utterances from 83 speakers totaling 14 hours of speech. The evaluation sets consist of one clean set, and six sets consisting of various levels of digitally-added noise. Each evaluation set consists of 330 utterances from 8 different speakers. All utterances are sampled at 16 kHz.

In an effort to determine whether or not the combination of these invariants with MFCCs is able to better model continuous speech, we perform a set of preliminary phoneme classification experiments. Using automatic, time-aligned phonetic transcriptions of the clean corpus data, we match segments of the continuous speech to 40 phonemes. For each of the feature combinations, a 16-mixture GMM is estimated for every phoneme. Using the same data, we then classify each of the signal frames as one of the 40 phonemes. Table 1 summarizes the relative difference in classification accuracy between the baseline MFCC feature vector and the MFCC/Invariant combination feature vector. Figure 7 illustrates relative improvements for several individual phonemes.

Table 1. Average relative phoneme classification improvements using MFCC/Invariant combination.

	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%

In Table 1, we see that the average relative classification accuracy increases significantly for affricates and stops, with the most

dramatic increase for affricates using the correlation dimension invariant where we get an increase of 10.3%. Stops show a fairly consistent increase for all three invariants. The use of the correlation entropy invariant resulted in an improvement for all phoneme types except for fricatives. Many of the phoneme types saw little or no improvements, and although some suffered a decrease in accuracy, these decreases are minimal.

Figure 7 illustrates some of the results seen in Table 1 by showing the relative classification improvement for several individual phones. The relative improvements for affricates and stops are high for each of the invariants while the nasal phonemes saw little or no improvements. These results are encouraging. The accuracy improvements in these low-level phoneme recognition experiments suggest that we will likely see accuracy increases in continuous speech recognition experiments.

C.3 Recognition Experiments

Our preliminary experiments provided strong support that the addition of these nonlinear invariants the standard MFCC feature vector will improve the accuracy of speech recognition tasks. We next present two sets of continuous speech recognition experiments, each using acoustic models trained from the clean training set mentioned in the previous section. The first set evaluates the noise-free test set using each of the different MFCC/invariant feature vector combinations. The results of these experiments are outlined in Table 2. The purpose of these experiments was to determine whether these new feature vectors would improve recognition performance for an evaluation set with environmental conditions that match those of the training set. The second set of experiments evaluates seven different test sets, each with varying levels and types of additive noise that would be encountered in the following environments: an airport, random babble, a vehicle, a restaurant, the street, and on a train. The results of these experiments are outlined in Table 3. The purpose of this second set was to determine whether or not

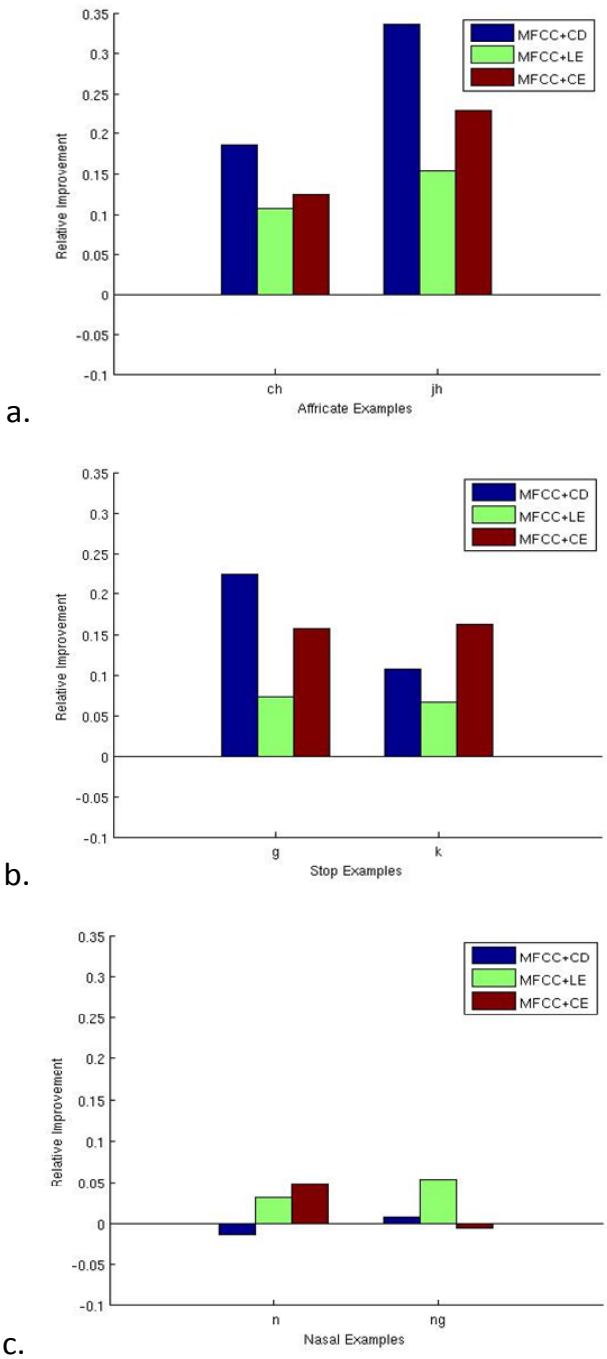

Figure 7. Relative improvements in recognition accuracy for several phonemes for an extended MFCC feature vector that includes nonlinear invariants (Affricates (a), Stops (b), Glides (c)).

Table 2. Continuous Speech Recognition Results for Clean Evaluation Data (no additive noise) and the Relative Improvement vs. the Baseline MFCCs.

	WER (%)	Improvement (%)
Baseline	13.5	--
Correlation Dimens. (CD)	12.2	9.6
Lyapunov Exponent (LE)	12.5	7.4
Correlation Entropy (CE)	12.0	11.1
All Invariants	12.8	5.2

Table 3. Continuous Speech Recognition Results for Noisy Evaluation Data.

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

these nonlinear invariants improve the robustness of the acoustic models to noise conditions that are unseen in the training data.

All experiments use the ISIP prototype system developed at Mississippi State University. This open-source speech recognition system uses HMMs to model acoustics and a trigram backoff language model. The models trained for these experiments were cross-word context dependent HMMs with underlying 4-mixture Gaussians.

The recognition results for the clean test set were encouraging. Each of the MFCC/invariant feature combinations resulted in a significant recognition performance increases over the baseline MFCC experiments. Correlation entropy resulted in the largest relative improvement of 11.1%. This reflected the results in the previous section where we saw a relatively consistent improvement in phoneme accuracy for correlation entropy. While combining all three of the invariants resulted in an improvement over the baseline, this improvement was not as significant as each of the invariants by themselves. This seems to suggest that the new features contributed a certain level of overlapping information.

The recognition results for the noisy test sets were less encouraging as each experiment resulted in a performance decrease compared the baseline. These results contradict our theory that the addition of invariants would result in a feature vector that is more robust to noisy conditions unseen in the training set. Though we pursued some work involving low-pass filtering of the trajectories to enhance the algorithms' robustness to noise, this work did not produce any significant improvements in performance. Hence, we shifted our attention to acoustic modeling.

D. Mixture Autoregressive Modeling of Speech Signals

Gaussian mixture models are a very successful method for modeling the output distribution of a state in a hidden Markov model (HMM). However, this approach is limited by the assumption that the dynamics of speech features are linear and can be modeled with static features and their derivatives. It is well-known that speech production is, in fact, a nonlinear process. In this work, a class of nonlinear mixture autoregressive models (MixAR) were used to model state output distributions in a conventional HMM-based speech recognition system. These models can handle both static and dynamic features. We applied this model to two problems: speaker independent speech recognition and speaker identification.

D.1 Motivation

Over the past decade there has been a great deal of interest in overcoming the barrier imposed by the assumption of linearity in speech signals. Early in the history of speech processing, linear modeling of speech became the de facto standard due to several reasons. First, the linearity assumption is the simplest possible model. A system is said to be linear if the output is proportional to the input and the

superposition principle holds. This makes interpretations of the model simple and also it is easy obtain insights into the model. Furthermore, such models are computationally simple and within the computational reach of modern computers. Moreover, in spite of its simplicity, linear modeling provides remarkably good performance in speech processing. Notwithstanding these advantages, we have now reached a threshold in speech research where this linearity assumption is an impediment to further advancements in performance.

In the case of nonlinear systems, the superposition principle no longer holds true and this has a variety of implications. First, the output of a nonlinear model is no longer constrained to be on the same hyperplane as the inputs, i.e., output need not be just a weighted sum of inputs. Secondly, the nonlinearity lends itself to modeling cycles, periods, nonlinear attractors, and invariant properties of the associated with the attractors that are common in natural systems. Thridly, the use of nonlinear techniques can lead to entirely new insights into the structure of the data to be modeled. This is seen especially with topological methods like fractals, nonlinear manifold learning, and, more recent topological invariants like Betti numbers from point clouds.

The foremost argument in favor of nonlinear modeling approaches is that natural systems are never linear. In speech preception too, there is without doubt, some degree of nonlinearity. For example, doubling of the amplitude of a signal is not perceived as twice as loud. Perceived loudness is a logarithmic function of amplitude. Any hope for modeling natural phenomenon, including speech, lies ultimately in our ability to understand and apply nonlinear models.

To understand where the linearity assumption arises in speech or speaker recognition, we need to take a closer look at the speech features employed at the front-end and also at the statistical models used for recognition. The most common features used in recognition are Mel-Frequency Cepstral Coefficients (MFCCs) [1]. These are derived from speech samples mainly through linear transforms (such as the Discrete Fourier Transform). Though these features use nonlinear warping of the frequency axis that is designed to mimic the auditory response of the human ear, this is a very simplistic model of nonlinearity. More sophisticated features, such as those based on Perceptual Linear Prediction [1], incorporate more knowledge of the human auditory system, but still do not account for phenomena such as chaos and strange attractors. Therefore, in this work we focused on modeling the evolution of the MFCC feature stream as a nonlinear process rather than direct modeling of the time series.

Conventional statistical modeling of MFCCs employ Gaussian Mixture Models (GMM) of state output probabilities in a Hidden Markov Model (HMM) [1], as shown in Figure 8. In this paradigm, each state of an HMM represents a phoneme (or some other abstract sub-phonemic unit of a stable segment of speech), and the corresponding MFCCs are modeled by a mixture of Gaussians random vectors. Transitions between states in the HMM correspond to movement from one phoneme to another. Transition probabilities historically do not play a significant role in speech recognition. Speaker identification systems tend to use large numbers of Gaussian mixtures, often in the thousands, while speeech recognition systems often use 64 to 128 mixtures per state.

One main drawback of applying this model for MFCCs is that the use of GMM enforces the assumption of time-independence [1] – that the output at each time frame is independent of the previous one. This is clearly known to be false – natural speech is much more gradual and smooth, and the same is also true of its

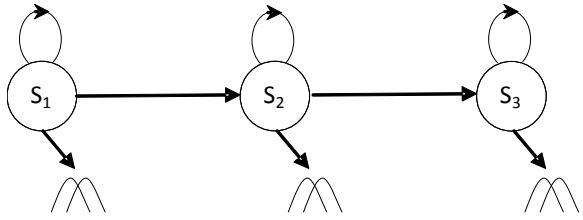

Figure 8. An overview of the Gaussian Mixture Model (GMM) approach to speech modeling based on hidden Markov Models (HMMs).

MFCC representation. To make up for this incorrect assumption, a convenient fix that is typically employed is to use derivative features in addition to the static ones.

In speech recognition tasks, this linear derivative modeling of MFCC dynamics improves the performance significantly [1]. For speaker recognition tasks, however, this approach typically does not improve performance [20]. This latter scenario is contrary to what we would expect if the derivative features were a sufficient representation of speech dynamics, prompting questions such as "What's wrong with linear derivative features?" and "How else can we model the MFCC dynamics?"

To address the first question about the insufficiency of linear derivative features for modeling speech MFCC dynamics, it is important to assess the importance of nonlinear effects in speech signals. We reason that if significant nonlinearities are found to be present in speech time series, then these would also appear in the dynamics of the MFCC representation of speech, making the use of linear derivatives for representing speech dynamics ill-founded. This begs the question: "How much nonlinearity is present in speech signals and does it have any bearing on the recognition problem?"

On the subject of nonlinearity in speech signals, there have been several attempts at measuring it's amount and it's application in recognition [10]-[16]. In attempting to employ information about nonlinear dynamics for recognition, most of these approaches are based on computation of nonlinear invariants as described previously. The techniques have failed to produce improvements in robustness. We conjecture that there are two major reasons for this:

1) Estimation algorithms for nonlinear invariants are sensitive to many practical issues such as finite amounts of data and the time-varying nature of the speech signal.
2) Invariant features only measure the degree of nonlinearity. Two signals having very different dynamics but the same amount of nonlinearity will have identical invariant features. For example, most periodic-like signals, including all voiced vowels, would have Lyapunov exponent values about zero, and cannot be distinguished based on this alone.

From the above discussion we see that what we need is a way to directly model the nonlinearity rather than just the degree of nonlinearity.

This leads us to a second question: how to model the nonlinear dynamics of MFCC features? One possibility is to try using nonlinear autoregression of each MFCC feature, for example a polynomial model, or a Taylor series. However, these models typically have so many more parameters to estimate that it may be difficult to get reliable estimates. Furthermore, it would be desirable to have a new model that has obvious parallels to GMM so that we may build on past experience and also integrate the new model into the HMM framework. Unfortunately, polynomial and Taylor series models are far removed from this ideal.

In retrospect, what we desire is a statistical model for speech MFCCs that:

1) accounts for past dependence explicitly (rather than using extra dynamic features);
2) models the actual nonlinear evolution (instead of quantifying only the amount of nonlinearity);
3) is a weighted mixture of simpler models (just as GMM is a weighted mixture of Gaussian models).

A mixture of autoregressive models fits these objectives particularly well [22]. It is a mixture model where each component consists of a simple linear autoregressive filter and a mean. Each autoregressive component uses a weighted sum of past samples to predict the present sample. The components are weighted probabilistically, and this probabilisitc mixing of linear AR processes lends itself to nonlinear evolution modeling.

D.2 Mixture of Autoregressive Models

There are several kinds of models that fall under the same name of mixture of autoregressive models [21]-[26]. Of these, the most general is the MixAR model [22]. Other mixture of autoregressive models can be derived from this general model under special conditions. The general MixAR process is defined by:

$$x_n = \begin{cases} a_{1,0} + \sum_{i=1}^{p_1} a_{1,i} x_{n-i} + \varepsilon_1[n] & \text{w.p. } g_1(x_{n-1},\ldots,x_{n-p_2}) \\ a_{2,0} + \sum_{i=1}^{p_1} a_{2,i} x_{n-i} + \varepsilon_2[n] & \text{w.p. } g_2(x_{n-1},\ldots,x_{n-p_2}) \\ \vdots & \vdots \\ a_{m,0} + \sum_{i=1}^{p_1} a_{m,i} x_{n-i} + \varepsilon_m[n] & \text{w.p. } g_m(x_{n-1},\ldots,x_{n-p_2}) \end{cases}, \tag{8}$$

where,

ε_i: zero-mean Gaussian random process with a variance of σ_j^2

w.p.: with probability

p_1: prediction order

p_2: gate order

$\{a_{i,j}\}$: linear predictor coefficients for component i

$a_{i,0}$: mean for component i

g_i: gate functionfor component i, assigns probability to each mixture component based on the previous p_2 samples.

The only requirement for the gate functions is that their values sum to *1* at each sample instant.

A convenient and popular functional form for the gate function is:

$$g_j(x_{n-1},\ldots,x_{n-p_2}) = \frac{\exp(\sum_{i=1}^{p_2} A_{k,i} x_{n-i} + A_{j,0})}{\sum_{k=1}^{m} \exp(\sum_{i=1}^{p_2} A_{k,i} x_{n-i} + A_{k,0})}, \tag{9}$$

where $\{A_{k,i}\}$ are the gate parameters for j^{th} mixture component.

It is apparent that an m-mixture MixAR process is the weighted sum of m Gaussian autoregressive processes, with time-dependent weights dependent on previous samples. Here we have generalized the model in [21] by decoupling the gate order and the prediction order. In its previous formulation both these orders were constrained to be equal. Since there is no reason for forcing this constraint, we consider the values for these two orders to be distinct and independent. This allows us to test for the contributions from time-dependency of gate components and AR components, individually and also in conjunction. For this, we make four kinds of assumptions to derive four types of models, as shown in Table 4.

The theoretical utility of identifying these four types of models from the general model is that it povides a unified view of these models. Taking this approach for mixture autoregressive models, we can see the effects of the various assumptions independently. In addition, the practical advantage in distinguishing these models from the general MixAR model is that the training procedures for models vary depending on the assumptions of gate and prediction orders. When the gate order is 0 (Types 1 and 3), the reestimation equations have a closed form expression. When the gate order is non-zero (Types 2 and 4), a gradient descent approach is required. When the prediction order is 0 (Types 1 and 2), we can use the same EM reestimation equations for the mean and variance as for a GMM (Type 1), but when the prediction order is non-zero (Types 3 and 4), we need to resort to a weighted-covariance type approach for estimation of prediction coefficients.

Table 4. The four types of models derived from general a general MixAR model.

MixAR Type	Assumptions	Equivalent Model in Literature
1	$p_1=0, p_2=0$	GMM [1]
2	$p_1=0, p_2>0$	Mixture of Experts [26]
3	$p_1>0, p_2=0$	MAR [22]
4	$p_1>0, p_2>0$	MixAR (general model) [22]

One property of MixAR that is of particular relevance here is the ability to model nonlinearities in a time series. Though the individual component AR processes are linear, the probabilistic mixing of these AR processes constitutes a nonlinear model. In a GMM, the distribution remains invariant to the past samples due to the static nature of the model. For MixAR, the conditional distribution given past data varies with time. This model is capable of modeling both the conditional means and variances. Thus, MixAR can model time series that evolve nonlinearly. This property becomes important in speech processing in the light of recent work on nonlinear processing of speech [13][14]. Some other properties of MAR including conditions required for the process to be stationary are derived in [22].

We have integrated the MAR model into the HMM framework by replacing the GMM output probabilities with that of MAR. This is illustrated in Figure 9.

D.3 Relationship to Other Mixture Auoregressive Models

The MixAR model is related to a family of models found both in statistical and speech literature. The general MixAR model has been applied to two benchmark time series – sunspots and Canadian lynx trapping data – for prediction applications[22], and shown to be superior to linear models. MAR model was shown to be superior to linear models on two real data prediction – IBM stock prices and Canadian lynx data [22]. The Mixture of Experts model has remained an integral part of several neural networks

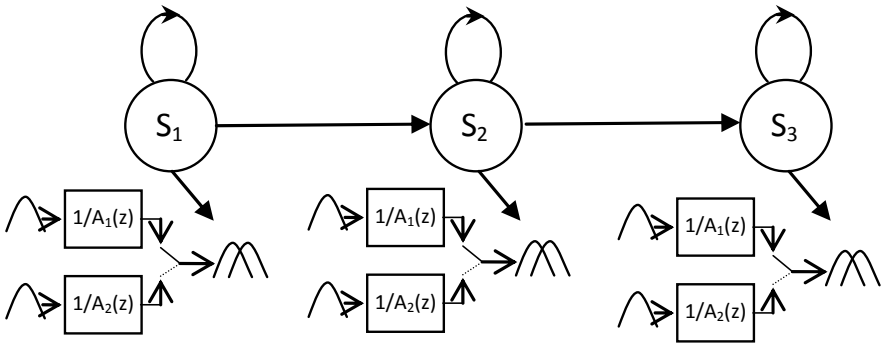

Figure 9. An overview of the MAR-HMM approach.

applications related to soft-threshold partitioning of the feature space [26].

There have also been applications of some variants of autoregressive models with HMMs in speech processing. All of these can be derived as special cases of MAR-HMM. One of the first applications of autoregressive HMMs in speech processing assumed an autoregressive (AR) model for each state, so that the short term correlations in the speech signal and known linguistic properties of sound combinations could be modeled by the state transitions [25]. This model was effectively a single-mixture component MAR-HMM.

The next major advance was the introduction of mixture autoregressive HMMs in [23]. This work applied a weighted mixture of AR filters to model observations at each state. While this appears to be very similar to the MAR-HMM developed in this paper, this approach had two major shortcomings. The model in [23] assumed that all AR components had the same variance, and that each was zero mean. This is equivalent to constraining the MAR model to have zero means and equal variance. In this respect the MAR-HMM considered in our work is more general than the autoregressive models previously applied to speech.

A variant of the original AR-HMM, using switching autoregressive process was considered in [24]. In this approach, the signal correlations during HMM state transitions were also modeled by the switching process. However, this model again was restricted to a single component AR, and thus it too is equivalent to a single-component MAR. Moreover, these variants of AR-HMMs considered only scalar speech time series as observations. Our extensions to vector time series are crucial to application of these models to speech recognition.

D.4 Parameter Estimation Using The Expectation Maximization Algorithm

Similar to GMM training, maximum likelihood estimates for MAR parameters can be calculated using the Expectation Maximization (EM) algorithm [27]. During the E-step, the probability (expectation) that each sample was generated from each of the mixture components from the current model is computed. During the M-step, the weight, mean, and, covariance parameters are updated to maximize the overall data likelihood. These two steps are then run iteratively until convergence of the likelihood is achieved.

Given the orders, p_1 and p_2, the parameter set for each of the m components of a MAR model consists of predictor coefficients (including the mean), the error variance, and the gate parameters:

$$\theta_l = \{a_{l,0}, a_{l,1}, \ldots, a_{l,p_1}, \sigma_l, A_{l,0}, A_{l,1}, \ldots, A_{l,p_2}\} \quad l = 1, \ldots, m, \tag{10}$$

We use an alternative approach using the Q-function to derive the EM update equations [28]. Since direct maximization of the likelihood (or log likilihood) is difficult for this problem, we resort to the use of the auxillary Q-function. It is known from Jensen's inequality that updating parameters to maximize the Q-function is equivalent to maximizing likelihood. Hence this approach is justified.

The auxillary function is defined as [28]:

$$Q(\Theta, \Theta') = E\left[\log(p(X, Y \mid \Theta)) \mid X, \Theta'\right], \tag{11}$$

where, Θ is the updated parameter set, Θ' is the current parameter values, X is the set of training data samples, and Y is the set of hidden states. At any instant, the sample could have arisen from any of

the components of the mixture and the hidden state refers to the specific outcome that a sample arose from a particular mixture. Summing over the marginal distribution, we get:

$$Q(\Theta,\Theta') = \sum_{y \in Y} \log(L(\Theta \mid X,l))p(y \mid X,\Theta'), \tag{12}$$

where the summation is over the set of all possible hidden state sequences, Y, and L is the overall likelihood of data given the model. With the gate function defined as:

$$g_j(x_{n-1},\ldots,x_{n-p_2}) = \frac{\exp(\sum_{i=1}^{p_2} A_{k,i}x_{n-i} + A_{j,0})}{\sum_{k=1}^{m}\exp(\sum_{i=1}^{p_2} A_{k,i}x_{n-i} + A_{k,0})}, \tag{13}$$

and the Gaussian AR probabilities are defined as:

$$n_l(x[n] \mid \theta) \propto \frac{1}{\sigma_l} e^{\frac{-1}{2\sigma_l^2}(x[n] - a_{l,0} - \sum_{i=1}^{m} a_{l,i}\, x[n-i])^2}, \tag{14}$$

the Q function can be expanded using the independence assumption:

$$Q(\Theta,\Theta') = \sum_{l=1}^{m}\sum_{n=p+1}^{N}\begin{pmatrix}\log\left(g_l(x_{n-1},\ldots,x_{n-p})n_l(x_n) \mid \Theta\right)\\ * p(l \mid x_n, x_{n-1},\ldots,\Theta')\end{pmatrix}, \tag{15}$$

Here $p(l,\cdot)$ is the probability that the hidden state at sample instant n is l. Denoting this with the more common notation of γ, we have:

$$\gamma_l[n] = p(l \mid x_n, x_{n-1,}\ldots) = \frac{g_l(x_{n-1},\ldots,x_{n-p})n_l(x_n)}{\sum_{k=1}^{m} g_k(x_{n-1},\ldots,x_{n-p})n_k(x_n)}. \tag{16}$$

Using the identity log(AB)=log(A)+log(B), we can partition the Q-function into the constituent contributions from the gates and the Gaussian AR components as follows:

$$\begin{aligned} Q(\Theta,\Theta') = &\sum_{l=1}^{m}\sum_{n=p+1}^{N}\log\left(g_l(x_{n-1},\ldots,x_{n-p} \mid \Theta)\right)\gamma_l[n] \\ &+ \sum_{l=1}^{m}\sum_{n=p+1}^{N}\log\left(n_l(x_n \mid \Theta)\right)\gamma_l[n] \end{aligned}. \tag{17}$$

From our expression for the Gaussian AR probability, we have:

$$\log\left(n_l(x_n \mid \Theta)\right) = const - \log(\hat{\sigma}_l) - \frac{(x_n - \hat{a}_{l,0} - \sum_{i=1}^{p} \hat{a}_{l,i}\, x_{n-i})^2}{2\hat{\sigma}_l^2}. \tag{18}$$

This can also be written using vectorial notation as:

$$\log\left(n_l(x_n \mid \Theta)\right) = const - \log(\hat{\sigma}_l) - \frac{\left(x_n - X_{n-1}^T \begin{bmatrix} \hat{a}_{l,0} \\ \vdots \\ \hat{a}_{l,p_1} \end{bmatrix}\right)^2}{2\hat{\sigma}_l^2}. \tag{19}$$

where

$$X_{n-1} = [1\, x_{n-1}\, x_{n-2} \cdots x_{n-p}]^T \quad .$$

Maximizing *Q* w.r.t. variance, we differentiate w.r.t. to σ and setting it to zero, obtaining:

$$\sum_{n=p+1}^{N} \frac{\gamma_l[n]}{\hat{\sigma}_l} = \sum_{n=p+1}^{N} \frac{\left(x_n - \hat{a}_{l,0} - \sum_{i=1}^{p} \hat{a}_{l,i}\, x_{n-i}\right)^2 \gamma_l[n]}{\hat{\sigma}_l^3}. \tag{20}$$

Solving:

$$\hat{\sigma}_l^2 = \frac{\sum_{n=p+1}^{N} \left(x_n - \hat{a}_{l,0} - \sum_{i=1}^{p} \hat{a}_{l,i}\, x_{n-i}\right)^2 \gamma_l[n]}{\sum_{n=p+1}^{N} \gamma_l[n]}. \tag{21}$$

Differentiating w.r.t. the prediction coefficients, a set of *m* linear equations are obtained:

$$\sum_{n=p+1}^{N} \gamma_l[n] x_n X_{n-1}[i] = \sum_{j=0}^{p} a_{l,j} \sum_{n=p+1}^{N} \gamma_l[n] X_{n-1}[i] X_{n-1}[j]. \tag{22}$$

From these, the solution for the prediction coefficients (including mean) are obtained by the following matrix solution:

$$\left[\hat{a}_{l,0} \cdots \hat{a}_{l,p_1}\right]^T = R_l^{-1} r_l, \tag{23}$$

where,

$$R_l = \sum_{n=p+1}^{N} \gamma_l[n] X_{n-1} X_{n-1}^T \tag{24}$$

$$r_l = \sum_{n=p+1}^{N} \gamma_l[n] X_{n-1} x[n] \ . \tag{25}$$

Unfortunately, there is no closed-form solution for the gate parameter updates. Instead we need to resort to a steepest ascent approach.

Note that there are two additional design parameters involved in this update: α, and a δ parameter for estimating the derivative of Q. The equations for gate parameter update have no closed-form solution, but we only move the parameter values in the direction of an increase in likelihood (with a scale factor α) at each iteration. In this framework, we are under the framework of Generalized EM algorithm (GEM), where instead of maximizing likelihood at each iteration we only gaurantee parameter updates towards increasing likelihood.

To estimate these parameters, we first need an initial guess for these parameters and then we iterate with EM steps to successively refine the estimates. An initialization strategy that we found to work reasonably well was to first train a GMM with the same number of mixtures and then set each component of the MAR model to have the same mean, variance, and weight as the GMM model. These initial parameters can be then refined recursively using expectation and maximization steps.

$$\begin{bmatrix} \hat{A}_{l,0} \\ \vdots \\ \hat{A}_{l,p_2} \end{bmatrix} = \begin{bmatrix} \hat{A}_{l,0} \\ \vdots \\ \hat{A}_{l,p_2} \end{bmatrix} + \alpha \frac{\partial Q}{\partial \hat{A}}. \tag{26}$$

MixAR Types as Special Cases:

If $p_2 = 0$ (Types 1 and 3) we reduce to the familiar weight equations for GMM. In this case we have the closed-form solution for gate parameters:

$$\hat{A}_{l,0} = \log\left(\frac{\sum_{n=p+1}^{N} \gamma_l[n]}{N-p} \right). \tag{27}$$

If $p_1 = 0$ (Types 1 and 2), the AR parameter update equation simplifies to the familiar GMM mean and variance update equation:

$$\hat{a}_{l,0} = \frac{\sum_{n=0}^{N} \gamma_l[n] x[n]}{\sum_{n=0}^{N} \gamma_l[n]} \quad \hat{\sigma}^2_{l,0} = \frac{\sum_{n=0}^{N} \gamma_l[n] (x[n] - \hat{a}_{l,0})^2}{\sum_{n=0}^{N} \gamma_l[n]}. \tag{28}$$

D.5 Pilot Experiments with MAR Model as a Pattern Classifier

To better understand the efficacy of the MAR-HMM model, we evaluated its performance on two simple pattern recognition tasks. The first task represents data with known nonlinearities. The second task is a simple phone classification task.

The MAR-HMM approach, like GMM-HMMs, can perform classification using a maximum likelihood approach. The log likelihood of data given a set of MAR-HMM model parameters is used to score each model and the class with the maximum score is chosen. A two-class classification problem was designed where data are randomly generated randomly according to the following MAR model parameters:

$$\Theta_1 : \begin{cases} p = 1, m = 2, w_1 = 0.4, w_2 = 0.6, \\ a_{1,0} = -1, a_{1,1} = 0.2, a_{2,0} = 1, a_{2,0} = 0.2, \\ \sigma_1 = 0.25, \sigma_2 = 0.2 \end{cases}, \tag{30}$$

$$\Theta_2 : \begin{cases} p = 0, m = 4, \\ w_1 = 0.2, w_2 = 0.2, w_3 = 0.3, w_4 = 0.3 \\ a_{1,0} = -1.03, a_{2,0} = -0.86, \\ a_{3,0} = 1.13, a_{4,0} = 0.98 \\ \sigma_1 = 0.3263, \sigma_2 = 0.2906, \\ \sigma_3 = 0.2598, \sigma_4 = 0.2894 \end{cases}. \tag{31}$$

where Θ_1 and Θ_2 correspond to classes 1 and 2, respectively.

For this example we chose the parameters for class 2 such that the marginal distribution is about the same as that of the first class, but it lacked the dependence on past samples unlike class 1. Hence the data for class 2 follows only a GMM distribution. This was done to demonstrate a case where GMM would be unable to achieve good classification due to its ability to capture the dynamics in the model. The results of these experiments, along with the number of parameters for each model, are shown in Table 5.

In addition to listing accuracy, the numbers of parameters for each model are shown. Since in this case we knew that the distribution can have a maximum of 4 modes, we use only 2- and 4-mixture models. It can be observed that MAR, with just 2 components and 8 parameters can

Table 5. MAR classification (% accuracy) results for synthetic data.

# mixtures	GMM Static only	MAR Static only	GMM static+ Δ+ΔΔ	MAR static+ Δ+ΔΔ
2	47.5 (6)	100.0 (8)	82.5 (14)	100.0 (20)
4	52.5 (12)	100.0 (16)	85.0 (28)	100.0 (40)

achieve 100% classification accuracy using only static features. The GMM approach using only static features is unable to do much better than a random guess strategy since the two classes have similar static marginal distribution. This demonstrates MAR-HMM's ability to learn dynamic information.

With the inclusion of delta coefficients, the GMM performance increases significantly, but even in this case it achieves only 85% accuracy with 28 parameters. Though delta features capture some amount of dynamic information in the features, it is still only a linear approximation, and we cannot capture their nonlinear evolution with just GMMs. From the above, it is clear that at least some dynamic information is better modeled using MAR-HMM.

D.6 Pilot Experiments with MixAR models as a Pattern Classifier

To better understand the efficacy of the MixAR model, we evaluated its performance on two pattern classification tasks. The first task represents generic data with known nonlinearities. The second task is a simple classification task with data for the two classes synthesized from models trained on speech-like data.

D.6.1 Two-Way Classification with Synthetic Data

A simple 2-way classification experiment was designed to study the performance of MixAR and GMM. Two-dimensional data for the first class was generated using a linear dynamic system:

$$x(n-1) = A \times x(n-1) + B \times E(n)$$
$$A = \begin{bmatrix} 0.5 & 0.0 \\ 0.0 & 0.5 \end{bmatrix}; B = \begin{bmatrix} 1.0 & 0.0 \\ 0.0 & 1.0 \end{bmatrix} \quad . \tag{32}$$

Data for the second class was generated using the simple nonlinear equation:

$$x(n-1) = A \times \text{sign}(x(n-1)) + B \times E(n)$$
$$A = \begin{bmatrix} 0.5 & 0.0 \\ 0.0 & 0.5 \end{bmatrix}; B = \begin{bmatrix} 1.0 & 0.0 \\ 0.0 & 1.0 \end{bmatrix} \quad . \tag{33}$$

In both cases, ***E*** denotes an uncorrelated 2-D Gaussian (normal) random variable with a zero mean and unit variance.

For each class, the training data consisted of a sequence of 10,000 vectors, and evaluation data consisted of 100 segments of 200 feature vectors each (the log-likelihood of the entire segment was used to assign a segment to a class). The classification error results are stated in Table 6. Clearly, when using only static features, MixAR does much better than GMM if nonlinearities are present. The use of dynamic features enhances GMM performance considerably but still falls far short of MixAR's performance.

Table 6. MixAR classification (% error) results for synthetic data (the numbers of parameters are shown in parentheses).

# mix.	GMM Static	MixAR Static	GMM Static+Δ	MixAR Static+Δ
0	36.0(12)	6.5(20)	10.0(24)	5.5(40)
4	35.5(24)	6.0(40)	11.5(48)	4.5(80)

D.6.2 Two-way Classification with Speech-like Data

In order to evaluate how well MixAR does as compared to GMM for speech-like signals, two speakers from the 2001 NIST

SRE Corpus [29] were selected. A 3-state HMM with 4 Gaussian mixtures per state and a MixAR model with 4 mixtures were trained over 12 static MFCC coefficients for each speaker. For each class (speaker), two speech-like signals of 40,000 vectors were generated – a linear speech-like signal ($\boldsymbol{X_1}$) was synthesized from the HMM model, and a nonlinear speech-like signal ($\boldsymbol{X_2}$) was generated from the MixAR model. To simulate a range of signals with varying degrees of nonlinearity, the two signals were mixed with a mixing coefficient alpha:

$$X_\alpha = (1-\alpha)X_1 + \alpha X_2 . \tag{34}$$

The first 20,000 vectors from each $\boldsymbol{X_\alpha}$ were used as a training set while the remaining vectors were split into 200 segments of 100 vectors each for evaluation. The results are shown in Table 7.

From the table we can see that when the amount of nonlinearity is insignificant, GMM performs as well as MixAR. However, as the amount of nonlinearity in the signal increases, MixAR performs significantly better with just static features as compared to GMM with static+Δ features. This clearly demonstrates the superiority of MixAR when dynamics in the data are nonlinear.

D.7 Speaker Recognition Experiments with Mixture Autoregressive Models

The goal in speaker recognition is to validate the identity of a speaker given speech data from that speaker. The two types of speaker recongition tasks are speaker identification and speaker verification. In speaker identification, the identity of a speaker is found from a set of apriori known database of speakers. Speaker verification, on the other hand, involves a particular claim for a speaker and the claim is accepted or rejected. In the suite of experiments presented here, we are concerned with speaker identification. An overview of this task is presented in Figure 10.

Table 7. MixAR classification error rate (%) with 12 speech MFCC-like synthetic features for GMM and MixAR Number of parameters in each case is in paranthesis. (*: For this case, GMM performed better with only static features, and this value is stated).

α	GMM-8mix. Static+Δ	MixAR-4-mix. Static
0.0*	1.5 (288)	1.5 (240)
0.25	3.25 (576)	3.5 (240)
0.50	10.25 (576)	6.25 (240)
0.75	24.75 (576)	9.75 (240)
1.0	26.75 (576)	13.75 (240)

In spite of advancements like HMMs and the use of dynamic features (deltas) in speech recognition, straight-forward GMM modeling of speakers has proven to be the most effective in speaker recognition. While these advancements over GMMs have improved performance significantly for speech recognition, it is clear that they are inept at capturing dynamical information for speakers. Our work attempts to remedy this situation by using MixAR models for quantitatively capturing the nonlinear dynamics of speakers. In the following we describe the data we used for speaker recognition and the corresponding results.

We use TIMIT database for speaker recongition. All of the 168 speakers in the test part of the database were utilized. For each speaker, the four SX and four SI sentences were used for training. The remaining two SA sentences were combined to form the test data for that speaker. We first performed tuning experiments with only 26 speakers from the DR2 dialect

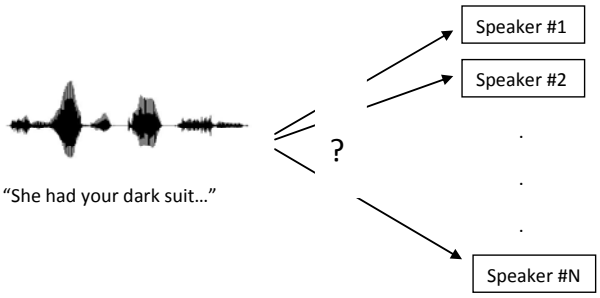

Figure 10. An overview of the speaker recognition problem.

region, while the final experiments were comprised of tests with all 168 test speakers.

We extracted only 12 static MFCC features since, as mentioned earlier, dynamic features are ineffective in speaker recognition. In addition, we did not use the energy coefficient, since this lowered the recognition performance slightly. Cepstral mean subtraction (CMS) was used on the MFCCs to compensate for mismatches in channel conditions.

Since our model formulation was for the scalar case only, we applied all four types of model for each MFCC coefficient individually. Thus, we have forced an assumption of independence between the features, but this is not too restrictive for MFCCs since they are known to be uncorrelated. Note that this is a common assumption even with GMM-based approaches where the covariance matrices are assumed to be diagonal. However, there is a weak tying between the scalar features within each Gaussian in the form of a commonly tied weight. We modeled each MFCC scalar completely independent of the other, even with distinct weights for each model component for each scalar. It was done this way because Type 2 and Type 4 MixARs have frame-dependent weights and these weights are dependent on the previous frame MFCC values. Hence, it is not possible to tie the weights across scalar components for these two models. For uniformity, we chose to use untied weights with all four types of models.

As a first experiment to test for the efficacy of each of the four types of models and to fix the number of mixtures, we used a smaller database of 26 speakers from the dialect region DR2 and compared the speaker recognition error rate. First we ran an experiment to tune the number of mixtures using GMM model. The results are shown in Table 8. From these experiments, four mixtures per feature was found to be about optimal for each speaker and we used this setting in the following experiments. In this case, the decrease in performance for 16 mixtures is contrary to expectations, and could be due to the limited amount of training data.

Next, the speaker recognition performance for the four types of MixAR models were found and results are in Table 9. From this table we see that while Types 2 and 3 do not perform well compared to GMM and Type 4 (full MixAR) does better than GMM. However, this study suffers from the fact that there are only 26 speakers and so this may not be statistically significant.

Table 8. Speaker recognition tuning experiments for the number of mixtures (26 speakers from DR2 dialect region of TIMIT test data).

GMM # mixtures	1	2	4	8	16
% Speaker Recog. Error	46.15	19.23	15.38	15.38	19.23

To obtain results with higher statistical significance and to test for robustness in noisy conditions, we applied the MixAR model to the 1-speaker detection task in the 2001 NIST SRE Corpus [29]. Only the development database was used. All 60 speakers were used for training and all 78 utterances were used for evaluation. Each training utterance was about 2 minutes long, while the test utterances were of varying length not exceeding 60 seconds. Static (13 MFCCs), delta (26 MFCCs) and delta-delta (39 MFCCs) features were extracted.

Table 9. Speaker recognition performance with 26 speakers from DR2 dialect region of TIMIT test data.

MixAR Type	**% Speaker Recog. Error**
GMM	15.38
Mix. Experts	30.77
MAR	38.46
MixAR	11.54

First we evaluated performance with and without delta features and energy for a fixed number of mixtures. The results are tabulated in Table 10. For GMM, substantial improvement is obtained using the delta features and marginal improvements were obtained using delta-delta features. For MixAR, the use of any delta features provides

no measurable improvements. This clearly indicates that MixAR can extract all necessary information from only the static features.

MixAR and GMM performance was then evaluated as a function of the number of mixtures. The EER results are shown in Table 11. Also indicated in parenthesis is the number of parameters for each case. From this table it is clear that MixAR can achieve about the same performance using almost 4x fewer parameters than GMM. This reduction in the number of parameters points to the efficiency of MixAR in capturing the dynamic information. Moreover, even when considering the best case scenario for GMM with a large number of parameters (8 mixtures with static as well as velocity and acceleration coefficients), there is a 10.6% relative reduction in EER with MixAR. This is a strong indication that there is some amount of nonlinear evolution information in speech features that GMM model cannot capture using linear derivatives alone and MixAR can effectively employ this information for achieving better speaker recognition. The detection error trade-off (DET) curves are shown in Figure 11.

Table 10. Speaker recognition EER for the NIST corpus for different feature combinations.

Features	GMM (16-mix.)	MixAR (8-mix.)
Static(12)	22.1	19.1
Static+E(13)	33.1	41.1
Static+Δ(24)	20.6	20.4
Static+Δ+ΔΔ(36)	20.5	20.5

Table 11. Speaker recognition EER with NIST for MixAR and GMM as a function of #mix. (the numbers of parameters are shown in parentheses).

# mix.	MixAR Static+Δ+ΔΔ	MixAR Static
2	23.1 (216)	24.1 (120)
4	21.7 (432)	19.2 (240)
8	20.5 (864)	19.1 (480)
16	20.5 (1728)	19.2 (960)

D.8 Speech Recognition Experiments with Mixture Autoregressive Models

For speech recognition, we applied the MAR model (MixAR Type 3 model) in the framework of HMMs to phone classification and recognition tasks [29]. The MAR-HMM model we have developed is a generalized version of [22] that has been extended to handle vector observations, so that we can operate on the speech feature vector stream rather than speech samples. One property of MAR that is of particular relevance is the ability of MAR to model nonlinearity in time series. Though the individual component AR processes are linear, the probabilistic mixing of these AR processes constitutes a nonlinear model. In a GMM, the distribution remains invariant to the past samples due to the static nature of the model. For MAR, the conditional distribution given past data varies with time. This model is capable of modeling both the conditional means and variances. Thus, MAR can model time series that evolve nonlinearly.

We first wanted to test the efficacy of MAR-HMM in a simple speech recognition setting, so we performed a sustained phone classification experiment. We made 16 kHz recordings of three distinct phones – "aa" (vowel), "m" (nasal), and "sh" (sibilant). For each phone and for each speaker,

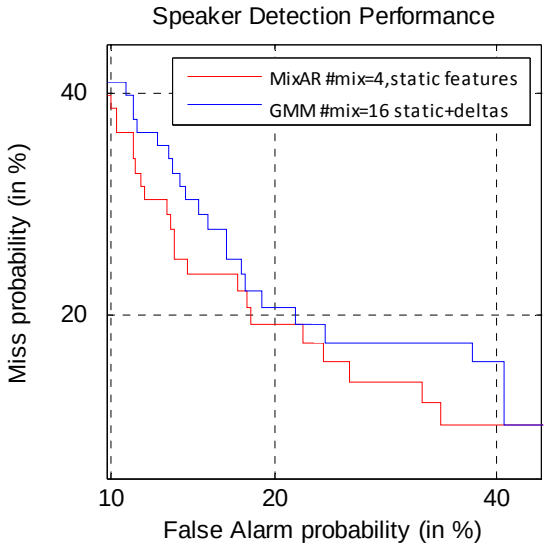

Figure 11. Speaker detection performance (DET curves) for the experiments on the NIST data.

35 recordings were made to serve as training database, while another 15 were reserved for testing. Silence was removed so that we could focus on the ability of the approach to model speech. We evaluated the performances of 2-, 4-, 8-, and 16 mixture GMM-HMM and MAR-HMM with the 13 dimensional static MFCC features. The results are shown in Table 12.

Table 12. Sustained phone classification (% accuracy) results with MAR and GMM using static MFCC features (the numbers of parameters are shown in parentheses).

# mixtures	GMM	MAR
2	77.8 (54)	83.3 (80)
4	86.7 (108)	90.0 (160)
8	91.1 (216)	94.4 (320)
16	93.3 (432)	95.6 (640)

For an equal number of parameters, MAR outperformed GMM significantly. For instance, MAR-HMM achieved a phone classification accuracy of 94.4% with only 320 parameters while a GMM system using 432 parameters could only achieve a 93.3%.

To determine whether MAR is more effective at exploiting dynamics than what GMM can achieve using dynamic features, we also performed another experiment with 39-dimensional features containing both static as well as velocity and acceleration coefficients. The results are in Table 13.

Table 13. Sustained phone classification (% accuracy) results with MAR and GMM using static+Δ+ΔΔ MFCC features (the numbers of parameters are shown in parentheses).

# mixtures	GMM	MAR
2	92.2 (158)	94.4 (236)
4	94.4 (316)	97.8 (472)
8	96.7 (632)	97.8 (944)
16	100.0 (1264)	98.9 (1888)

In this case, the results were not conclusive. While MAR-HMM showed an accuracy rate of 97.8% with 472 parameters, GMM-HMM attained only 96.7% accuracy with 632 parameters. Unfortunately, the performance of MAR-HMM saturated with an increase in the number of parameters. For example, MAR-HMM at 1888 parameters achieved only 98.9% accuracy while GMM-HMM achieved 100% with 1264 parameters. We suspect that this could be due to the fact that our parameter estimation and likelihood computation procedures assume that the features are independent. It is well-known that the static MFCC features are uncorrelated (at least, theoretically), but obviously the delta features are correlated with the static ones. While this should also cause problems for GMM, the problem is more acute for MAR because in this case, unlike GMMs, we employ the past history explicitly [22].

Since the positive results with static features on the simple phone classification experiment above was encouraging, we next applied MAR-HMM to a larger scale phone recognition experiment with TIMIT database. We used the full training part of TIMIT for training and the core test part for testing. Since for all speakers the SA sentences had the same transcriptions, these were avoided both during training and testing to avoid biasing the results. A bigram language model was used with 16 mixture MAR-HMM and a 16-mixture GMM-HMM served as the baseline. The results are shown in Table 15 and Table 14.

In Table 15, it is seen that for recognition with static as well as dynamic features, GMM performs significantly better. We expected this to be the case from our experience with sustained phone experiments, where MAR proved superior to GMM with static only features, but inferior when derivative features are included. Unfortunately, contrary to our expectations, we found that MAR performance was worse than for GMM even for static features only case, as shown in Table 14. These provide indications that MAR is perhaps not suited for speech recognition.

Table 14. Performance for 39 (static+dynamic) MFCCs and with 16 mixtures.

Acoustic Model for HMM	Phone Recognition Accuracy (%)
GMM	69.5
MAR	59.2

Table 15. Performance for 13 static MFCCs and with 16 mixtures.

Acoustic Model for HMM	Phone Recognition Accuracy (%)
GMM	51.5
MAR	42.3

However, this approach is limited by the assumption that the dynamics of speech features are linear and can be modeled with static features and their derivatives. In this work, a nonlinear mixture autoregressive model was used to model state output distributions (MAR-HMM). This model can handle both static and dynamic features.

E. Linear Dynamic Models for Speech Recognition

Hidden Markov models (HMMs) have been the most popular approach for acoustic modeling in speech recognition along with the diagonal covariance matrix assumption in which correlations between feature vectors for adjacent frames are ignored. Linear Dynamic Models (LDMs) [34] use a state space-like formulation that explicitly models the evolution of hidden states using an autoregressive process. This smoothed trajectory model allows the system to better track speech dynamics especially for noisy speech signals. In this work, we proposed LDMs as an alternative to hidden Markov models (HMMs) for robust speech recognition in noisy environments. Our evaluation results showed that, for complex recognition task Aurora 4 [36], LDM classifiers achieved a 4.9% relative accuracy increase for the clean evaluation data and a 6.5% relative accuracy increase for the noisy data over a comparable HMM system with 3-state models. Based on these preliminary results, we are in the process of developing a HMM/LDM hybrid decoder architecture to model the frame correlation using LDMs as well as utilizing HMMs techniques for phone segment alignment.

E.1 State Space Models

Over the last few decades, a variety of linear Gaussian models have been applied in a wide variety of domains including control, machine learning and financial analysis. This forumlation draws heavily on a state-space model, in which data is described as a realisation of some unseen process. The following two equations describe a general state space model:

$$\begin{aligned} y_t &= h(x_t, \varepsilon_t) \\ x_t &= f(x_{t-1}, \ldots, x_1, \eta_t) \end{aligned} \quad (35)$$

A p-dimensional observation y_t is linked to a q-dimensional state vector x_t by the first equation, and the state's evolution is governed by the second equation.

In state space models, the observations are seen as realisations of some unseen, usually lower-dimensional, process. This provides a means of distinguishing the underlying system from the observations which represent it. The state and observation spaces are linked by the transformation h. The observation noise ε_t characterises the variation due to a range of external sources, for example measurement error or noise. Furthermore it offers a degree of smoothing which is useful when there is a mismatch between training and testing data. Uncertainty in the modelling of the state process is described by the state noise η_t. An important feature of these models is that the observation at time t is conditional only on the state at that time. However, the state can take a variety of forms, such as static

distributions, long-span autoregressive processes or sets of discrete modes. Figure 12 represents such a model, where motions in the state space give rise to the observed data.

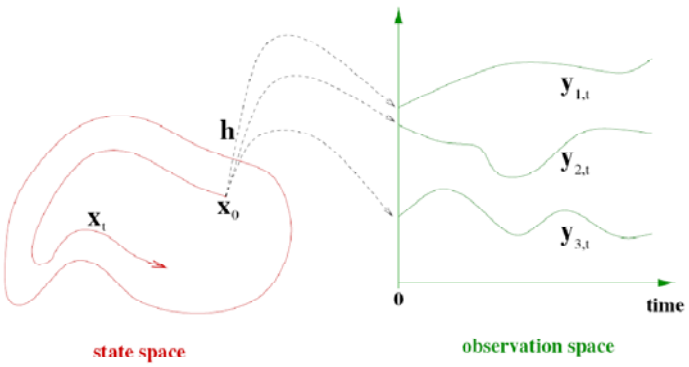

Figure 12. An overview of a state space model.

State-space models are useful in many real-life situations where systems contain a different number of degrees of freedom, usually fewer, than the data used to represent them. In these cases, a distinction can be made between the production mechanism at work and the parameterization chosen to represent it. The hidden state variable can have just as many degrees of freedom as are required to model any underlying processes, and then a state-observation mapping shows how these are realised in observation space. This offers a means of making a compact representation of the data. In fact, dimensionality reduction is a common application of this class of models.

There are two problems which must usually be solved when applying a state-space model. First, it should be possible to infer information about the internal states of the model for a given set of parameters and sequence of observations. Second, the parameters which identify the model must be capable of being estimated from suitable training data.

E.2 Linear Dynamic Models

Linear Dynamic Models (LDMs) [34] are an example of a Markovian state space model, and in some sense can be regarded as analogous to an HMM since LDMs do use hidden state modeling. With LDMs, systems are described as underlying states and observables combined together by a measurement equation. Every observable will have a corresponding hidden internal state, as shown in Figure 13.

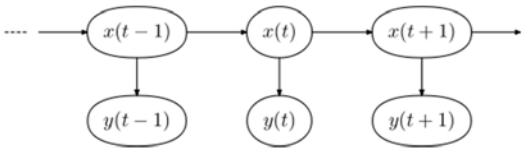

Figure 13. Internal states and observations in a LDM.

The general LDM process is defined by:

$$\begin{aligned} y_t &= Hx_t + \varepsilon_t \\ x_{t+1} &= Fx_t + \eta_t, \quad x_1 \sim N(\pi, \Lambda) \end{aligned} \tag{36}$$

where,

y_t : p-dimensional observation feature vectors

x_t : q-dimensional internal state vectors

x_1 : initial state with mean π and covariance matrices Λ

H: state evolution matrix

F: observation transformation matrix

ε_t : uncorrelated white Gaussian noise with mean ν and covariance matrices C

η_t : uncorrelated white Gaussian noise with mean w and covariance matrices D

LDM assumes that the dynamics underlying the data can be accounted for by the autoregressive state process. This describes how the Gaussian-shaped cloud of probability density representing the state evolves from one time frame to the next. A linear transformation via the matrix F and the addition of some Gaussian noise, η_t, represents the dynamic portion of the model. The complexity of the motion that second equation can model is determined by the dimensionality of the state variable, and will be considered below. The observation process shows how a linear transformation with the matrix H and the addition of measurement noise ε_t relate the state and output distributions.

The system's hidden states are the deterministic portion of an LDM which are also affected by random Gaussian noise [37]. The state and noise variables can be combined into one single Gaussian random variable. Based on Figure 13, the conditional density functions for the states and output can be written as follows:

$$\begin{aligned} P(y_t|x_t) &= (2\pi)^{-P/2}|C|^{-1/2}\exp\left\{-\frac{1}{2}[y_t - Hx_t]^T C^{-1}[y_t - Hx_t]\right\} \\ P(x_t|x_{t-1}) &= (2\pi)^{-k/2}|D|^{-1/2}\exp\left\{-\frac{1}{2}[x_t - Fx_{t-1}]^T D^{-1}[x_t - Fx_{t-1}]\right\} \end{aligned}. \tag{37}$$

According to the Markovian assumption, the joint probability density function of the states and observations becomes:

$$P(\{x\},\{y\}) = P(x_1)\prod_{t=2}^{T} P(x_t|x_{t-1})\prod_{t=1}^{T} P(y_t|x_t). \tag{38}$$

The system's states are hidden. We need to estimate the hidden state evolution given an N-length observation sequence y_t and the model parameters. This can be accomplished using a Kalman filter combined with a Rauch-Tung-Striebel (RTS) smoother [32]. The Kalman filter provides an estimate of the state distribution at time t given all the observations up to and including that time. The RTS smoother gives a corresponding estimate of the underlying state conditions over the entire observation sequence. For the smoothing part, a fixed interval RTS smoother is used to compute the required statistics once all data has been observed.

The RTS smoother adds a backward pass that follows the standard Kalman filter forward recursion [32]. In addition, in both the forward and the backward pass, we need some additional recursions for the computation of the cross-covariance. The RTS equations are:

$$\hat{x}_{t-1|N} = \hat{x}_{t-1|t-1} + A_t\left(\hat{x}_{t|N} - \hat{x}_{t|t-1}\right). \tag{39}$$

$$\Sigma_{t-1|N} = \Sigma_{t-1|t-1} + A_t\left(\Sigma_{t|N} - \Sigma_{t|t-1}\right)A_t^T. \tag{40}$$

$$A_t = \Sigma_{t-1|t-1}F^T\Sigma_{t|t-1}^{-1}. \tag{41}$$

$$\Sigma_{t,t-1|N} = \Sigma_{t,t-1|t} + \left(\Sigma_{t|N} - \Sigma_{t|t}\right)\Sigma_{t|t}^{-1}\Sigma_{t,t-1|t} \tag{42}$$

A synthetic LDM model with two-dimensional states and one-dimensional observations was created to demonstrate the contribution of RTS smoothing. In Figure 14 we show the state predictions of this LDM model using traditional Kalman filter. In Figure 15, the performance of the Kalman filter with RTS smoothing is shown. In both figures, the green lines represent the trajectories of the two-dimensional true state evolution for our synthetic LDM model. The blue points are the scatter plot of the noisy observations of the LDM model. We can see the predicted results roughly simulate the true state evolution. After adding RTS smoothing into the Kalman filtering process, we observed significantly better prediction for the system internal states.

E.3 EM Training and Implementation Issues

The Expectation-maximization (EM) algorithm [37] is used to find the maximum likelihood estimates of parameters for a specific word or phone, where the model depends on unobserved latent variables. The relevant equations are:

$$E\left[x_t \middle| y, \theta^{(i)}\right] = \hat{x}_{t|N}. \tag{43}$$

$$E\left[x_t x_t^T \middle| y, \theta^{(i)}\right] = \Sigma_{t|N} + \hat{x}_{t|N}\hat{x}_{t|N}^T \tag{44}$$

$$E\left[x_t x_{t-1}^T \middle| y, \theta^{(i)}\right] = \Sigma_{t,t-1|N} + \hat{x}_{t|N}\hat{x}_{t-1|N}^T \tag{45}$$

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 observations and model parameters. The RTS smoother described previously can be used to compute the complete-data estimates of the state statistics. EM for LDM then consists of evaluating the ML parameter estimates by replacing x_t and $x_t x_t^T$ with their expectations.

The EM algorithm converges quickly and is stable for our synthetic LDM model of two-dimensional states and one-dimensional observations. After initilizing this LDM model with an identity state transition matrix and random observation matrix, the first iteration of ML parameter estimation was applied to update the model parameters. Log-likelihood scores of observation vectors were calculated and saved in order to perform further analysis.

EM training was applied for 30 iterations. After the training recursion, intermediate log-likelihood scores of observation vectors for each iteration of LDM were plotted as a funtion of the number of iterations. This plot is refered as the EM evolution curve. We explored 1-, 4-, 6-, and 10-dimensions for a state in the LDM approach, and applied EM training for each specified dimension. In Figure 16, the EM

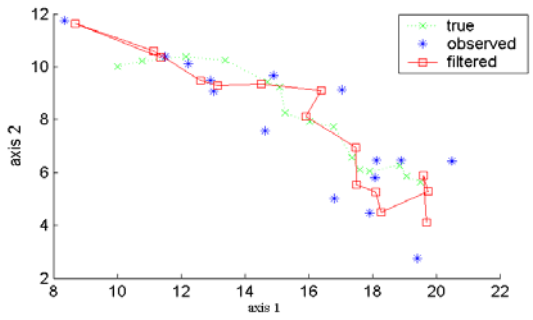

Figure 14. State predictions of an LDM using a traditional Kalman filter.

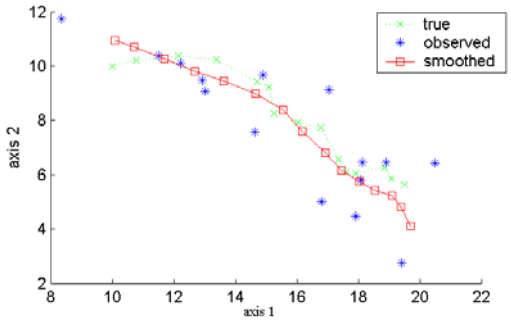

Figure 15. State predictions of an LDM with RTS smoothing.

evolution curve is shown as a function of the state dimension.

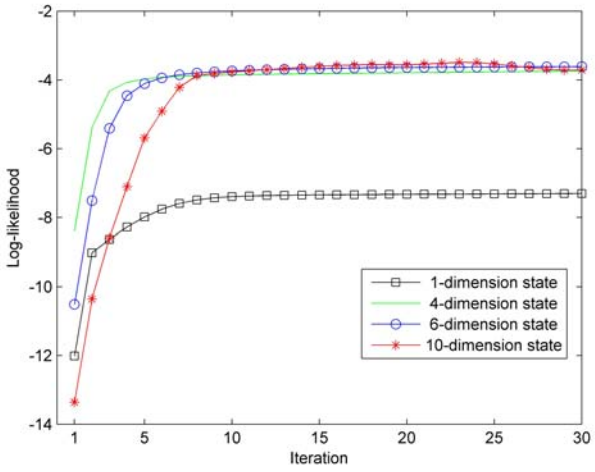

Figure 16. EM evolution vs. state dimension.

One important practical issue about our EM implementation is that the linear transformation matrix F might lead the ML parameter estimation to produce erroneous parameters when $|F| > 1$. The reason for this is that the LDM state evolution would grow exponentially if the matrix F is not a decaying transformation [34]. Such behavior may not be apparent over a small numbers of frames, but it appears quite often when the training dataset gets large, especially in the situation where the state is not reset between models.

In this case, the most common solution is to use Singular Value Decomposition (SVD) to force $|F| < 1$ after each iteration of EM training. SVD provides a pair of orthonormal bases U and V, and a diagonal matrix of singular values S such that:

$$F = USV^T. \tag{46}$$

Every element of S greater than $1-\varepsilon$ will be replaced by $1-\varepsilon$ for a small number of ε (usually $\varepsilon = 0.005$). By adding the SVD component, we attain good model stability for LDM training, as was described in [32].

For a given speech segment, the likelihood that this segment was generated from a specific LDM can be calculated from Kalman filter equations. For a standard Kalman Filter, the state estimation error at time t can be represented as:

$$e_t = y_t - \hat{y}_t = y_t - H\hat{y}_{t|t-1}. \tag{47}$$

After replacing y_t with the observation equation, the error term becomes:

$$\begin{aligned} y_t - H\hat{y}_{t|t-1} &= H\left(x_t - \hat{x}_{t|t-1}\right) + \varepsilon_t \\ e_t &= H\left(x_t - \hat{x}_{t|t-1}\right) + \varepsilon_t . \end{aligned} \tag{48}$$

The associated covariance is:

$$\Sigma e_t = E\left[e_t e_t^T\right] = H\Sigma_{t|t-1} + C. \tag{49}$$

Since errors are assumed uncorrelated and Gaussian, the log-likelihood of an N-length observation sequence y_t given the model parameters can be calculated as

$$\log\left(p_1^N|\theta\right) = -\frac{1}{2}\sum_{t=1}^{N}\left\{\log\left|\Sigma e_t\right| + e_t^T\Sigma^{-1}e_t\right\} - \frac{Np}{2}\log(2\pi). \tag{50}$$

where e_t and Σe_t are computed as part of the standard Kalman filter recursions. In classification applications, the latter normalization term can be omitted because it is constant [32].

Some researchers report that the state's contribution to the error covariance Σe_t is detrimental to classification performance [32]. During EM training, the resulting fluctuations in the likelihoods computed during the segment-initial frames have the most effect on the overall likelihood of shorter phone segments. For shorter speech segments, it is recommended to replace the error covariance calculation:

$$\Sigma e_t = H\Sigma_{t|t-1}H^T + C \quad where \quad \Sigma e_t^{'} = R\,. \tag{51}$$

However, our experimental results did not show a performance improvement for shorter speech segments by using this approach. Hence, in the following experiments, the LDM implementations used the traditional error covariance form.

E.4 Pilot Classification Experiments

Since LDM has proven to be effective on simulated data, a logical next step was to apply it to the classification of phonetic segments in speech. Our first experiment involved evaluating LDM as a classifier on a simple database consisting of a few phones clearly articulated by a small group of speakers. This data was used to gain a better understanding of key algorithm parameters and their impact on convergence. We refer to this data as the sustained phones database.

The sustained phone database is composed of 2 speakers with 3 phones recorded for each speaker. Each speaker produced 0.5 second utterances of the following phonemes: one vowel 'aa', one nasal 'm' and one fricative 'sh' at a sampling rate of 16 kHz. Feature vectors were generated by computing 12 mel-scaled cepstral coefficients and absolute energy. A frame duration of 10 milliseconds and a window duration of 25 milliseconds was used for feature extraction. The training set consisted of 210 examples (70% of the sustained phone database) of 3 phones from two speakers and the test set consisted of 90 examples (30% of the sustained phone database).

After the data recording and feature extraction, we initialized 3 LDMs (phonemes 'aa', 'm', and "sh") using the following strategy: ; the state transition matrix as an identity matrix multiplied by a factor 0.1; the observation matrix as random entries; the observation noise covariance as an identity matrix. The EM algorithm was used for training. We observed that EM training converges after approximately 5 iterations. Different dimensionalities of the state space were examined and we found 13 dimensions were adequate. Increasing the dimensionality of the state space to 40 did not improve the classification accuracy in this case.

An HMM system with GMMs was built as the benchmark to evaluate LDM as a phoneme classifier. Table 16 summarizes the relative difference in classification accuracy between LDMs and HMMs. We see that the classification accuracy of the LDM system is 98.9%, which outperforms the best HMM baseline classification accuracy of 91.1% (8-mixture). For each phoneme, the LDM model has 858 parameters and the 8-mixture HMM model

Table 16. Classification (% accuracy) results for the sustained phone database.

Model	Vowel *aa*	Nasal *m*	Fricative *sh*	Total
HMM (2-mixt)	67	70	97	78
HMM (4-mixt)	90	70.	100	87
HMM (8-mixt)	100	73.	100	91
LDM	100	97	100	99

has 630 parameters. The total running time of LDM classification experiment was in the same range of HMM classification experiment. In the next section, we will further assess LDMs on a large vocabulary evaluation corpus Aurora-4.

E.5 Aurora Experiments

Motivated by the encouraging results on the sustained phone classification experiment, we continued to evaluate LDMs on the Aurora-4 large vocabulary evaluation corpus [36]. This corpus is a well established LVCSR benchmark that does not require extensive computational resources. The data was generated from a machine readable corpus of Wall Street Journal news text. The corpus is divided into a training set and an evaluation set. The training set consisted of 7,138 utterances from 83 speakers totaling in 14 hours of speech. The evaluation set consisted of 330 utterances from 8 speakers. All utterances were generated at 16 kHz.

The HMM system is used to generate alignments at the phone level. Each phone instance is treated as one segment. A total of 40 LDM phone models, one classifier per model, were used to cover the pronunciations. Each classifier was trained using the segmental features derived from 13-dimensional frame-level feature vectors comprised of 12 cepstral coefficients and absolute energy. The full training set has as many as 30k training examples per classifier. Each phone-level classifier is trained as a one-vs-all classifier. The classifiers are used to predict the probability of an acoustic segment.

Table 17 summarizes the results of the Aurora-4 phoneme classification experiments. The baseline system is composed of 3-state HMMs with varying numbers of mixtures. We show results only for 4-mixture GMMs since the performance increase for larger mixtures was only marginal. The HMM system achieves up to 46.9% and 36.8% accuracy for the clean evaluation data and noisy evaluation data respectively. For the noisy evaluation data, six different kinds of noise (Airport, Babble, Car, Restaurant, Street, and Train) were added randomly to better simulate the real world noisy environment.

Table 17. Classification (% accuracy) results for the Aurora-4 large vocabulary corpus (the relative improvements are shown in parentheses).

Model	Clean Data	Noisy Data
HMM (4-mixt)	46.9 (-)	36.8 (-)
LDM	49.2 (4.9%)	39.2 (6.5%)

From Table 17 we can see that the LDM classifiers achieve superior performance to the HMM classifiers with a classification accuracy of 49.2% for the clean evaluation data and 39.2% for the noisy evaluation data. This represents a 4.9% relative and a 6.5% relative increase in performance over a comparable HMM system with 3-state models. For each phoneme, a 4-mixture HMM model has 318 parameters while the LDM model has 858 parameters. LDM appears to offer improved generalization over the HMM baseline system across different channel conditions, which makes LDM a more robust speech recognition technique.

F. Summary and Future Work

This project represented a unique opportunity to look beyond the HMM paradigm and incorporate recent research in nonlinear system analysis. Throughout the course of the project, many techniques involving nonlinear processing have been evaluated. Though many of these techniques have been extensively published, and have shown improvements on small tasks or non-speech data, few delivered improvements on realistic speech processing tasks. In this final report, we have described three promising approaches that have produced measurable improvements on speech tasks: an MFCC feature vector

augmented with features respresenting estimates of the nonlinearity in the signal, a mixture autoregressive model and a linear dynamic model.

The first of these was the topic of an MS thesis by Daniel May. The second topic will be the dissertation topic for Sundar Srinivasan, who is expected to complete his Ph.D. in Spring'2010. The third topic will be the dissertation topic for Tao Ma, also expected to complete his Ph.D. in Spring'2010.

There are several aspects of the MixAR model which need further investigation. First, we would like to synthesize synthetic speech-like data to study MixAr performance for speaker as well as speech recognition. Next we want to evaluate the applicability of MixAR for speech recognition. Finally, it could also be worthwhile to investigate discriminative training approaches for MixAR modeling. Over the past few years, there has been rapidly growing interest in discriminative training for GMM and HMM, and it is highly likely the advantages of these methods for GMM also carry over to MixAR modeling.

With respect to LDM, we are currently developing an HMM/LDM hybrid decoder architecture to model the frame correlation using LDMs as well as utilizing HMMs techniques for phone segment alignment. Results will be presented on the Alphadigits (AD) and Resource Management (RM) speech corpora. This HMM/LDM hybrid decoder architecture will be a good evaluation of LDMs on continuous speech recognition tasks, and can be compared to other hybrid decoders we have developed that utilize other nonlinear statistical models (e.g., support vector machines).

G. References

[1] X. Huang, A. Acero, and H.W. Hon, *Spoken Language Processing - A Guide to Theory, Algorithm, and System Development*, Prentice-Hall, Upper Saddle River, New Jersey, USA, 2001.

[2] M. Saraclar and S. Khudanpur, "Pronunciation change in conversational speech and its implications for automatic speech recognition," *Computer Speech and Language*, vol. 18, no. 4, pp. 375-395, January 2004.

[3] G.E. Peterson and H.L. Barney, "Control methods used in a study of the vowels," *Journal of the Acoustical Society of America*, vol. 24, no. 2, pp. 175-184, March 1952.

[4] A. Andreou, T. Kamm, and J. Cohen, "Experiments in vocal tract normalization," presented at the CAIP Workshop: Frontiers in Speech Recognition II, Piscataway, NJ, USA, July-August 1994.

[5] M.J.F. Gales and P.C. Woodland, "Mean and Variance Adaptation Within the MLLR Framework," *Computer Speech and Language*, vol. 10, no. 4, pp. 249-264, October 1996.

[6] D. Jurafsky and J.H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, Prentice-Hall, Upper Saddle River, New Jersey, USA, 2009.

[7] R. E. Best, *Phase-locked Loops: Design, Simulation and Applications*, McGraw-Hill, Chicago, Illinois, USA, 2003.

[8] "LM565/LM565C Phase Locked Loop," *http://cache.national.com/ds/LM/LM565.pdf*, National Semiconductors, Santa Clara, California, USA, May 1999.

[9] E. Lorenz, *The Essence of Chaos*, University of Washington Press, Seattle, Washington, USA, 1996.

[10] D. May, Nonlinear Dynamic Invariants For Continuous Speech Recognition, M.S. Thesis, Department of Electrical and Computer Engineering, Mississippi State University, May 2008.

[11] P. Maragos, A.G. Dimakis and I. Kokkinos, "Some Advances in Nonlinear Speech Modeling Using Modulations, Fractals, and Chaos," presented at the International Conference on Digital Signal Processing (DSP-2002), Santorini, Greece, July 2002.

[12] A.C. Lindgren, M.T. Johnson and R.J. Povinelli, R. J., "Speech Recognition Using Reconstructed Phase Space Features," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing* (ICASSP), pp. 60-63, Hong Kong, April 2003.

[13] A. Kumar and S.K. Mullick, "Nonlinear Dynamical Analysis of Speech," *Journal of the Acoustical Society of America*, vol. 100, no. 1, pp. 615-629, July 1996.

[14] M. Banbrook, *Nonlinear Analysis of Speech From a Synthesis Perspective*, Ph.D. Thesis, The University of Edinburgh, Edinburgh, UK, 1996.

[15] I. Kokkinos and P. Maragos, "Nonlinear Speech Analysis using Models for Chaotic Systems," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 6, pp. 1098-1109, Nov. 2005.

[16] S. Prasad, S. Srinivasan, M. Pannuri, G. Lazarou and J. Picone, "Nonlinear Dynamical Invariants for Speech Recognition," *Proceedings of the International Conference on Spoken Language Processing*, pp. 2518-2521, Pittsburgh, Pennsylvania, USA, Sept. 2006.

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

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

[19] V. Pitsikalis and P. Maragos, "Filtered Dynamics and Fractal Dimensions for Noisy Speech Recognition," *IEEE Signal Processing Letters*, vol. 13, no. 11, pp. 711-714, Nov. 2006.

[20] A. Morris, D. Wu and J. Koreman, "GMM based clustering and speaker separability in the TIMIT speech database," *IEICE Transactions on Fundamentals of Communications, Electronics, Informatics and Systems*, vol. E85-A/B/C/D, no. 1, March 2005.

[21] M. Zeevi, R. Meir, and R. Adler, "Nonlinear models for time series using mixtures of autoregressive models", *Unpublished Technical Report*, 1999, http://ie.technion.ac.il/~radler/mixar.pdf.

[22] C. S. Wong, and W. K. Li, "On a Mixture Autoregressive Model," *Journal of the Royal Statistical Society*, vol. 62, no. 1, pp. 95-115, February 2000.

[23] B. H. Juang, and L. R. Rabiner, "Mixture Autoregressive Hidden Markov Models for Speech Signals," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 33, no. 6, pp. 1404-1413, December 1985.

[24] Y. Ephraim, and W. J. Roberts, "Revisiting Autoregressive Hidden Markov Modeling of Speech Signals," *IEEE Signal Processing Letters*, vol. 12, no. 2, pp. 166-169, February 2005.

[25] A.B. Poritz, "Linear Predictive Hidden Markov Models," *Proceedings of the Symposium on the Application of Hidden Markov Models to Text and Speech*, Princeton, New Jersey, USA, pp. 88-142, October 1980.

[26] M. Jordan and R. Jacobs, “Hierarchical Mixture of Experts and the EM algorithm,” *Neural Computation*, vol. 6, no. 2, pp 181-214, March 1994.

[27] A. Dempster, N. Laird, and D. Rubin, “Maximum Likelihood From Incomplete Data Via the EM Algorithm," *Journal of the Royal Statistical Society, Series B*, vol. 39, no. 1, pp. 1-38, February 1977.

[28] J.A. Bilmes, “A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models,” *Technical Report ICSI-TR-97-021*, University of Berkeley, Berkeley, California, USA, April 1998.

[29] National Institute of Standards and Technology, “The 2001 NIST Speaker Recognition Evaluation,” http://www.nist.gov/speech/tests/spk/2001, 2001.

[30] S. Srinivasan, T. Ma, D. May, G. Lazarou and J. Picone, "Nonlinear Mixture Autoregressive Hidden Markov Models For Speech Recognition," *Proceedings of the International Conference on Spoken Language Processing*, pp. 960-963, Brisbane, Australia, September 2008.

[31] V. Digalakis, *Segment-based Stochastic Models of Spectral Dynamics for Continuous Speech Recognition*, Ph.D. Thesis, Boston University, Boston, Massachusetts, USA, 1992.

[32] J. Frankel, *Linear Dynamic Models for Automatic Speech Recognition*, Ph.D. Thesis, The Centre for Speech Technology Research, University of Edinburgh, Edinburgh, UK, 2003.

[33] 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 Transactions on Speech and Audio Processing*, vol. 1, no. 4, pp. 431–442, October 1993.

[34] J. Frankel and S. King, “Speech Recognition Using Linear Dynamic Models,” *IEEE Transactions on Speech and Audio Processing*, vol. 15, no. 1, pp. 246–256, January 2007.

[35] G. Tsontzos, V. Diakoloukas, C. Koniaris, and V. Digalakis, “Estimation of General Identifiable Linear Dynamic Models with an Application in Speech Recognition,” *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. IV-453–IV-456, Honolulu, Hawaii, USA, April 2007.

[36] N. Parihar and J. Picone, “An Analysis of the Aurora Large Vocabulary Evaluation,” *Proceedings of the European Conference on Speech Communication and Technology (EUROSPEECH)*, pp. 337-340, Geneva, Switzerland, September 2003.

[37] S. Roweis and Z. Ghahramani, “A Unifying Review of Linear Gaussian Models,” *Neural Computation*, vol. 11, no. 2, February 1999.

[38] M. Ostendorf, V. Digalakis, and O. Kimball, “From HMMs to Segment Models: A Unified View of Stochastic Modeling for Speech Recognition,” *IEEE Transactions on Speech and Audio Processing*, vol. 4, no. 5, pp. 360–378, September 1996.

[39] 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 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 109-112, Phoenix, Arizona, USA, May 1999.

[40] A. Rosti and M. Gales, “Generalized Linear Gaussian Models,” *Technical Report CUED/F-INFENG/TR.420*, Cambridge University Engineering, 2001.

[41] Z. Ghahramani and G.E. Hinton, “Parameter Estimation for Linear Dynamical Systems," *Technical Report CRG-TR-96-2*, University of Toronto, Toronto, Canada, 1996.