Automatic continuous speech recognition with rapid speaker adaptation for human/machine interaction

Nikko Ström

Department of Speech, Music, and Hearing
Kungliga Tekniska Högskolan
Stockholm, 1997

Akademisk avhandling som med tillstånd av Kungliga Tekniska Högskolan i Stockholm framlägges till offentlig granskning för avläggande av teknologie doktorsexamen fredagen den 5 december 1997 kl 14.00 i Kollegiesalen Valhallavägen 79, KTH, Stockholm. Avhandlingen försvaras på engelska.

ISRN KTH/TMH/FR--97/6--SE
TRITA-TMH 1997:6
ISSN 1104-5787

Abstract

This thesis presents work in three main directions of the automatic speech recognition field. The work within two of these – *dynamic decoding* and *hybrid HMM/ANN speech recognition* – has resulted in a real-time speech recognition system, currently in use in the human/machine dialogue demonstration system WAXHOLM, developed at the department. The third direction is *fast unsupervised speaker adaptation*, where "fast" refers to adaptation with a small amount of adaptation speech.

The work in dynamic decoding has involved the development of a continuous speech decoding engine based on the A* search paradigm. An efficient implementation of the algorithms has made real-time continuous speech recognition possible in the WAXHOLM dialogue system with a lexicon of about 1000 words. Features of the search algorithms that are important for the real-time performance are proposed. These include efficient use of beam-pruning, and graph reduction methods that greatly reduce the effective search space.

The hybrid HMM/ANN recognition is an area of work in its own right, but is also important in the speaker adaptation experiments. A very flexible ANN architecture has been developed and refined during the course of the thesis work. The architecture is a generalization of the TDNN and the RNN architecture, and allows both delayed and look-ahead connections. In the latest experiments, sparsely connected networks were investigated. Sparsely connected networks were shown to perform significantly better than their fully connected counterparts with an equal number of connections. In an experiment with phoneme recognition of the TIMIT database, the recognition rate of the hybrid HMM/ANN system is in the range of the highest reported, and only outperformed by another hybrid system.

The fast speaker adaptation work is based on the notion that an explicit *a priori* model of the speaker variability helps to rapidly adapt to a new speaker. In the experiments, a parametric speaker characterization is introduced in the ANN by adding special-purpose speaker-space input units whose activity values are determined by the speaker adaptation. Experiments have been made both with the American English TIMIT database and the Swedish WAXHOLM database, and a positive adaptation effect is detected after only a few syllables.

Keywords: Automatic speech recognition (ASR), hybrid HMM/ANN, lexical search, speaker adaptation, speaker characterization, human/machine dialogue system.

Contents

Included papers

The dissertation consists of this summary and the following papers, listed in chronological order of publication, except for Paper 4 – its position is moved back to indicate the time of creation because of the delay to the publication. Section 4 of the summary is an extended version of a presentation (Ström, 1995) at the IEEE Workshop on Automatic Speech Recognition, 1995. This extended version is not previously published.

Paper 1 Nikko Ström (1994): "**Optimising the Lexical Representation to Speed Up A* Lexical Search**," *STL QPSR 2-3/1994*, pp. 113-124.

Paper 2 Nikko Ström (1995): "**A Speaker Sensitive Artificial Neural Network Architecture for Speaker Adaptation**," *ATR Technical Report, TR-IT-0116,* ATR, Japan.

Paper 3 Nikko Ström (1996): "**Continuous Speech Recognition in the WAXHOLM Dialogue System**," *STL QPSR 4/1996*, pp. 67-96.

Paper 4 Nikko Ström (1997): "**Speaker Modeling for Speaker Adaptation in Automatic Speech Recognition**," in: *Talker Variability in Speech Processing,* Chapter 9, pp. 167-190, Eds.: Keith Johnson and John Mullennix, Academic Press, ISBN 0-12-386560-3.

Paper 5 Nikko Ström (1996): "**Speaker Adaptation by Modeling the Speaker Variation in a Continuous Speech Recognition System**," *Proc. ICSLP '96*, Philadelphia, pp. 989-992.

Paper 6 Nikko Ström (1997): "**Phoneme Probability Estimation with Dynamic Sparsely Connected Artificial Neural Networks**," *The Free Speech Journal (http://www.cse.ogi.edu/CSLU/fsj/),* Vol. **1**(5).

Abbreviations

ANN	Artificial Neural Network. The framework that is used in the hybrid HMM/ANN recognizer to compute local phoneme probabilities.
ASR	Automatic Speech Recognition.
CDHMM	Continuous Density Hidden Markov Model. Currently the most popular flavor of the HMM for ASR.
CSR	Continuous (automatic) Speech Recognition
CTT	Centrum för talteknologi (Centre for Speech Technology), based at KTH, Stockholm.
DAG	Directed acyclic graph
DFA	Deterministic Finite State Automaton.
DP	Dynamic Programming. An computationally efficient scheme for solving certain optimization problems.
EM	Expectation Maximization. Statistical method for parameter estimation.
EUROSPEECH	European Conference on Speech Communication and Technology.
FSA	Finite State Automaton.
HMM	Hidden Markov Model. A statistical time-series model that forms the basis of most contemporary state-of-the-art ASR systems.
ICASSP	International Conference on Acoustics, Speech, and Signal Processing.
ICSLP	International Conference on Spoken Language Processing.
IEEE	The Institute of Electrical and Electronic Engineers, Inc.
JASA	Journal of the Acoustic Society of America.
KTH	Kungliga Tekniska Högskolan (Royal Institute of Technology), Stockholm, Sweden.
MAP	Maximum *a posteriori*. Optimization criterion for the estimation of statistical parameters.
MCE	Minimum Classification Error. General optimization criterion for statistical classifiers.
ML	Maximum Likelihood. Optimization criterion for the estimation of statistical parameters.
NFA	Non-deterministic Finite State Automaton.
SD	Speaker Dependent.
SI	Speaker Independent.
STL-QPSR	Speech Transmission Laboratory – Quarterly Progress and Status Report, THM, KTH.
TMH	Institutionen för tal, musik och hörsel (Department of Speech, Music and Hearing), KTH, Stockholm.
TIMIT	Speech database recorded and processed by Texas Instruments and Massachusetts Institute of Technology.

1. Introduction

Continuous speech recognition (CSR), not long ago imaginable only in science-fiction stories, is a reality today. The first commercial, large vocabulary, continuous speech dictation system for use on standard PCs is already on the market (Naturally Speaking™ by Dragon Systems), and others will follow. IBM has announced a similar product to be released at the end of 1997. There is however a long way to go from contemporary state-of-the-art recognition systems to a system that is comparable with human speech perception. The research on the SWITCHBOARD corpus of spontaneous speech over the telephone is an illustration of this discrepancy (Cohen, 1996). For this task, today's technology is clearly not sufficient.

Today's best large vocabulary systems are used for one speaker only, because it is desirable to fine-tune to the speaker's voice, and the systems are still vulnerable to noisy conditions. Also, humans have an unlimited lexicon as new words can always be formed. This is true in particular for many Germanic languages where there is virtually no limit on the compounding of words. CSR systems recognize at best a few ten thousand words.

The recent commercial break-through is due to a gradual improvement in small steps of the prevalent HMM method. The improvements have walked hand-in-hand with the development of the computer technology that has allowed increasingly computation demanding models and the storage of larger probabilistic grammars. Although this development has been very fruitful, concerns have been raised about the long term development of the field. In order to bridge the remaining gap between human perception and automatic speech recognition, new innovative solutions may be required. It has been argued that the presence of one dominant technique, that has been tuned for many years, suppresses work along such innovative lines (Bourlard, 1995). CSR is a complex process that naturally breaks down in different sub-tasks (see Figure 1). Because of the years of tuning of the standard system, almost any significantly new approach in one of the modules of the system will most probably lead to an increase in error-rate when first investigated. This thesis reports in part on work that can be characterized as tuning the standard model, but the focus is on work on alternative methods.

In the first two blocks of Figure 1, signal processing and feature extraction, the speech signal is transformed to a time-series of feature vectors that is a

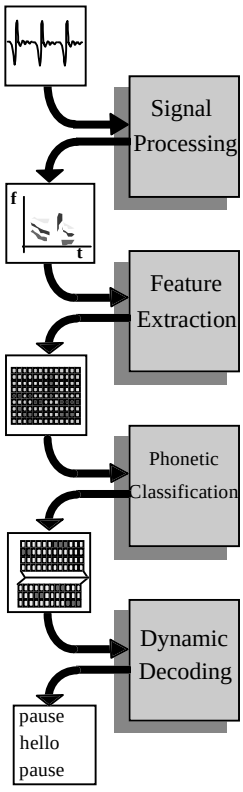


Figure 1. Overview of the ASR process. Most contemporary speech recognition systems conform to this chain of processing. In the signal processing module the raw speech waveform is transformed to the frequency domain. In the feature extraction module a data compressing transform is applied and the resulting output stream is a time-series of feature vectors. The phonetic classification module evaluates the feature vectors phonetically and in the dynamic decoding module the lexical and grammatical constraints are enforced.

suitable representation for the subsequent statistical phonetic evaluation in the next module. The sample rate of the feature vectors is usually about 100 Hz. These first two blocks are often called the front-end of the ASR system. Apart from the implementation of a few of the most popular features, e.g., mel-frequency cepstrum coefficients, the work reported on in this thesis is not much concerned with the front-end.

The standard method for implementing the phonetic classification module is to evaluate the feature vectors phonetically using multivariate Gaussian probability density functions. The density functions are conditioned by the phoneme in context, i.e., they estimate the likelihood of the feature vector given the hypothesized phoneme and its neighboring phonemes. However, several alternative methods have been proposed. For example, Digalakis, Ostendorf and Rohlicek (1992) use a stochastic model of phone-length segments instead of evaluation of the feature vectors independently, and Glass, Chang, and McCandless (1996) use a method based on phone-length segment feature-vectors and discrete landmarks in the speech signal.

Another alternative approach is the hybrid HMM/ANN method (Bourlard and Wellekens, 1988), where an ANN is used for the phonetic evaluation of the feature vectors. The hybrid HMM/ANN method is used in Paper 3, Paper 5 and Paper 6 of this thesis and is discussed in more detail in the next section of this summary.

The dynamic decoding module is where the recognition system searches for sequences of words that match phoneme sequences with high likelihood in the phonetic evaluation. This is also the module where the probabilistic grammar is included. Thus, word sequences are evaluated on the merits of their phonetic match and their grammatical likelihood. The output from the module is the most likely sequence of words, or a set of likely word sequence hypotheses that are then further processed by other modules of the system.

The dynamic decoding search of a large vocabulary CSR system can be computationally very costly. Therefore, current research and development in this field are dominated by computational issues. Popular themes for reducing computation are *pruning* and *fast search*. In pruning methods, partial hypotheses that are relatively unlikely are not pursued in the continued search. A particularly efficient pruning is proposed in Paper 3. In fast search methods, an initial fast, but less

accurate search is used to guide a subsequent, more accurate search, in order to pursue the most promising hypotheses.

Multi-pass methods are generalizations of fast search. In a multi pass method, the output of the first search pass is a set of the most likely hypotheses given the knowledge sources available in this first pass. Subsequent passes *re-score* the set of hypotheses based on additional knowledge sources. The additional knowledge sources require more computation, so it is beneficial to apply them only on the selected set of hypotheses instead of the whole search space. Examples of knowledge sources that can be added in later passes are of the type that span word boundaries, e.g., higher order N-grams in the probabilistic grammar, and tri-phones that condition on phones across word boundaries.

A concept that can be applied to all the modules of Figure 1 is speaker adaptation. Although the role of speaker adaptation in the human perception process is not completely understood, there is convincing evidence that such a process takes place. In particular, the presence of a rapid adaptation was proposed by Ladefoged and Broadbent (1957). This process operates on a time scale of a few syllables.

Speaker adaptation methods have been successfully applied also for ASR. The success should be no surprise, because of the discrepancy in the performance between speaker-dependent (SD) and speaker-independent (SI) systems (e.g., Huang and Lee, 1991). Different adaptation methods can be classified by the amount of supervision required from the user and on what time-scale they operate. At one end of the spectrum are methods that require the user to read a sample text that is then used to re-train the system. This method can asymptotically reach the performance of a corresponding SD system if the size of the sample text is increased (e.g., Brown, Lee, and Spohrer, 1983; Gauvain and Lee, 1994). More advanced methods collect the speaker-dependent sample as the system is running, and do re-training based on the data collected, without the need of a special training session by the user. This latter scheme is called unsupervised adaptation. However, both supervised and unsupervised re-training methods are typically relatively slow – the adaptation effect is significant only after a minute or more.

At the other end of the spectrum of speaker adaptation, are methods that have access to a model of

speaker variability. This additional knowledge lets the system concentrate on adapting to voices that are likely to exist in the population of speakers, instead of blindly trying to adapt based on the (initially) very small sample from the new speaker. These methods can reach a positive adaptation effect after only a few syllables of speech. The speaker modeling approach to speaker adaptation is developed and investigated in the context of hybrid HMM/ANN recognition in Paper 2, Paper 4 and Paper 5. However, the approach does not require an ANN model – a related adaptation scheme in the HMM domain, is found in a paper by Leggeter and Woodland (1994), and Hazen and Glass (1997) use a related method in a segment-based system.

The remainder of this summary is organized in sections corresponding to the three main directions of work: hybrid ANN/HMM recognition, dynamic decoding, and fast speaker adaptation. The exception is section 4, that contains previously unpublished material on the representation of large sets of alternative hypotheses – the output of the dynamic decoding search. Section 6 contains a brief presentation of the applications of the developed CSR system. Finally, section 7 contains brief summaries and comments on the included papers. In particular, the original contribution and innovative elements of each paper are discussed in this section.

2. Hybrid HMM/ANN speech recognition

2.1 Introduction

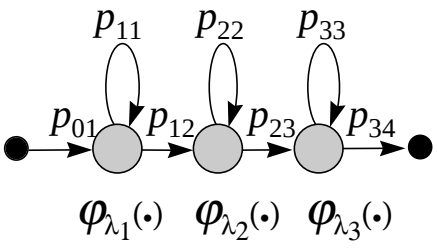
The continuous density hidden Markov model (CDHMM) is the dominant technology for automatic speech recognition today. However, a few significant limitations of the standard CDHMM suggest that other methods, or combinations of CDHMM with other methods, may improve the performance of the recognition. One class of such methods is based on artificial neural networks (ANNs).

Speech recognition was one of the first problems that ANN models were applied to during the rapid spread of the methods in the 1980's. The complex mapping between the acoustic domain, and the set of phonemes, has properties that are regarded to be modeled well with ANN methods. One reason for this is the good discriminative power of ANN models.

ANN models have been used for word recognition directly (e.g., Ström, 1992; English and Boggess, 1992; Li, Naylor, and Rossen, 1992), but with limited success. It has been proven difficult to model the temporal aspects in an efficient manner within the ANN formalism. Combinations of an ANN with HMM that models most of the temporal constraints have been more successful. Several different methods have been proposed for combining the two. A few of the more well-known are: "Hybrid HMM/ANN Architecture" (Bourlard and Wellekens, 1988), "Linked Predictive Neural Networks" (Tebelskis and Waibel, 1990), "Hidden Control Neural Architecture" (Levin, 1990), and "Stochastic Observation HMM" (Mitchel, Harper and Jamieson, 1996).

The most successful architecture for combining CDHMM and ANN technology is currently the hybrid HMM/ANN model, where the ANN is utilized to estimate the observation probabilities of a CDHMM. This combination makes use of the discriminative power of the ANN approach, and relies on the HMM to model temporal aspects and the invocation of a probabilistic grammar.

In this chapter, the hybrid system used in several of the included papers is reviewed. First the CDHMM that models the dynamic constrains imposed by the lexicon and grammar is discussed. Then I describe the ANN that models the mapping between acoustic observations



$$\varphi_{\lambda=\{w_i,\mu_{ij},\sigma_{ij}\}}(o) = \sum_{i=1}^{M} w_i \cdot \frac{1}{\sqrt{2\pi D}} e^{\sum_{j=1}^{D} \frac{(o_{ij}-\mu_{ij})^2}{2\sigma_{ij}^2}}$$

$$\sum_{i=1}^{M} w_i = 1$$

Figure 2. Graphical representation of an HMM. The HMM has one "hidden" stochastic process and one observable stochastic process. At each point in time, the model "is" in one of its states. The hidden process is the sequence of states visited. This process is governed by the transition probabilities, p_{ij} in the figure, for moving from one state to another. The observable process is the output acoustic feature vector at each state. These vectors are mixtures of multivariate Gaussian stochastic variables (with density function φ in the figure). This particular structure with three states and transitions from left to right, self-loops but no skips, is typically used for phoneme models.

and phonemes, and in section 2.5, sparse connection and pruning in the ANN are covered.

2.2 The standard CDHMM

Although used for speech recognition, the HMM is most easily described as a speech production model. An HMM based speech recognizer has a set of different HMMs representing different units of speech, e.g., phonemes or words. The recognition process is a search to find the sequence of models that are the most likely to have produced the utterance.

In the standard CDHMM formalism, the speech signal is assumed to be produced by a probabilistic finite state automaton (FSA). A graphic representation of an HMM, modeling a phoneme is shown in Figure 2. In the following I assume that all HMMs model phonemes or special segments such as pauses etc. The model produces the signal by emitting, at each state of the FSA, an output observation and then moving to a new state. The emission of observations occurs at equidistant time-points, typically every 10 ms, and the output observation is a random variable with a different probability density function for each state. Also the motion between the states is governed by statistical laws. The allowed transitions have transition probabilities associated with them such that some paths through the FSA are more likely than others. This is the only inherent duration modeling in the standard CDHMM.

The output probability densities are typically modeled by mixtures of multivariate Gaussian probability distributions with diagonal covariance matrices. This functional form is selected mostly for its mathematical properties, and in the case of the choice of diagonal covariance matrices, to reduce computations.

The observations are representations of the speech signals in the short time frame of circa 10 ms covered by each emission. A set of features, the *feature vector*, is computed for each frame. A popular basic feature vector is the so called cepstrum coefficients vector. The cepstrum coefficients are the cosine coefficients of the spectrum of the signal in the frame. Typically, the cepstrum features, the total energy in the frame, and the first and second time-derivatives of these features, are the components of the observation vectors modeled by the multivariate mixture of Gaussian probability distributions. The size of these observation vectors are

typically around 40. The feature vector extraction is discussed in detail in Paper 3.

The framework of the CDHMM makes it straightforward to optimize the model parameters using well-known parameter estimation paradigms from statistic theory. In particular, the maximum likelihood (ML) method is used. In short, it means that the parameters of the model – the means, variances and mixture weights of the distributions, and the transition probabilities – should be chosen in such a way that the probability that the correct sequence of HMMs produced the utterances of a training database is maximized. Or put in another way, the probability that the training utterances were produced by the models is maximized. There exists a computationally efficient algorithm, the Baum-Welch algorithm, that performs this maximization in an iterative expectation-maximization (EM) procedure. Each iteration of the algorithm is guaranteed to increase the probability of the training utterances, and thus convergence is established.

The ML training with Baum-Welch's algorithm is popular mostly because of its computational attractiveness, but it is actually optimizing the model's ability to produce speech, instead of recognizing it. Parameters computed by MAP (maximum *a posteriori*) estimation give the recognizer, at least in theory, better discrimination ability, but are harder to compute. In this paradigm, the probability of the correct string of symbols (phonemes or words) is maximized, given the observations. This is a more intuitive optimization criterion for a model to be used for recognition (see Figure 3).

A third alternative is to search for the parameters that minimize the actual error rate of the recognizer on the training data (minimum classification error, MCE). This is clearly the optimal training criterion, but this optimization is a much harder task than finding the ML parameters by Baum-Welch's algorithm or even MAP estimation.

Speech recognition in the CDHMM framework, is typically a procedure of finding the sequence of HMMs that are the most likely to have produced the utterance to recognize. Because the number of models can be rather large, and even more importantly, the boundaries between the models are unknown, this search is a very large problem that is in general not possible to solve without approximations. In practice, some variation of the Viterbi approximation is virtually always employed for continuous speech recognition (see section 3.2). In

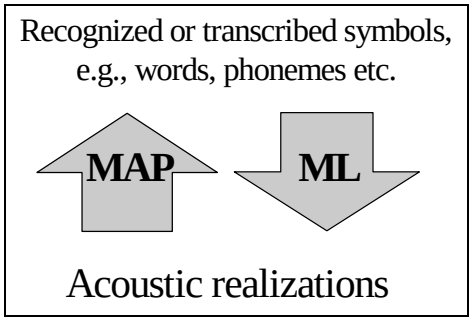


Figure 3. The difference between the maximum likelihood (ML) optimization criterion and the maximum a posteriori (MAP) criterion can be illustrated by this figure. In the ML framework, the probability of the produced acoustic realizations, given the symbols to recognize is optimized. In the MAP framework, the probability of the symbols is instead optimized given the acoustic realizations.

this approximation it is assumed that the probability of one path through the HMMs has much higher probability than all other paths. It is very clear that this underlying assumption is often not justified, but once again it is the attractive computational properties of the so called Viterbi algorithm that has made it the standard decoding method for CDHMM.

2.3 Problems with the standard model

The weaknesses of the standard CDHMM model, touched upon in the previous section, have been pointed out by several authors (e.g., Bourlard and Morgan 1993; Robinson, 1994), and can be summarized in the following points:

1. Poor discrimination due to the fact that model parameters are estimated by maximum likelihood (ML) estimation instead of an estimation method that attempts to explicitly minimize the classification error. Examples of such estimation schemes are: minimum classification error (MCE), and maximum a posteriori (MAP) estimation.
2. The *a priori* choice of model topology, and in particular the choice of functional form of the statistical distributions, e.g., assuming that the emission probabilities for the acoustic observations are mixtures of multivariate Gaussian densities with diagonal covariance matrices. This choice is based on the mathematical properties of the family of distribution functions and is not necessarily optimal.
3. The so called Markov assumption that state sequences are first-order Markov chains, i.e., the probability density distributions depend only on the current state.
4. The correlation between successive acoustic observations is not acknowledged. Note that this is different from the previous point where the influence of the state sequence was considered.
5. There is a mismatch between the Baum-Welch training and evaluation of the HMMs because the Viterbi approximation is active only in the evaluation phase.

Thus, it seems that we have a rather strong case against the HMM technology. However, during the history of the model, designers of state-of-the-art CDHMM recognition systems have addressed all the above points and found ways to reduce the negative effects of them. The effects of point (3) have been reduced by introducing context dependent models, e.g., tri-phones.

Point (4) has been addressed by adding delta and delta-delta coefficients to the observation vector. Parameter-tying schemes have made it possible to train very general probability density functions that can solve problems due to point (2), and recently, the ML training scheme has been complemented with MAP training in response to point (1).

To summarize the analysis of the weaknesses of the CDHMM: on the foundation of an initial model that seems rather inappropriate for the task, an increasingly complex and more accurate model has emerged through incremental refinement. It is noteworthy that the original motivation for the choice of model that leads to the discussed weaknesses was to keep computation down, but the subsequent improvements that partially solved the initial problems have increased the computational demands dramatically. This is true in particular for the many models needed for triphone modeling and the computationally demanding probability distribution functions used. Thus, the improvements go hand in hand with the rapid development of computers.

Given the above, it is not self-evident that the CDHMM would be the choice of model if automatic speech recognition were to be reinvented today. But because of the complexity of the problem and the large research and development investments in the current technology, it is very difficult to make a competitive system based on a completely new framework. When Bourlard (1995) discusses the present situation in the ASR field, he characterizes the prevailing CDHMM architecture as a local optimum in the space of recognition systems. He argues that it is necessary to change the architecture in a manner that increases recognition error rates in the short run, but has potential of long term improvement, thus escaping from the local optimum.

In the hybrid HMM/ANN architecture, the standard framework is kept intact, but the observation probabilities are computed by an ANN. This addresses point (1) and (2) above, by the selection of model and training paradigm chosen – ANNs put very weak *a priori* constraints on the distributions and are trained in the MAP paradigm. Point (5) is also neutralized because the Baum-Welch algorithm is not used, but many of the imperfections of the standard model remain, e.g., the Viterbi approximation and points (3) and (4).

2.4 The artificial neural network

In the late 1980's it was pointed out by several authors that the output activities of ANNs trained with the back-propagation algorithm approximate the *a posteriori* class probabilities (Baum and Wilczek 1988; Bourlard and Wellekens, 1988; Gish, 1990; Richard and Lippman, 1991). In the case of an ANN trained to recognize phonemes, this means that the ANN estimates the probability of each phoneme given the acoustic observation vector. This observation is of fundamental importance for the theoretical justification of the hybrid HMM/ANN model. By application of Bayes' rule, it is easy to convert the *a posteriori* probabilities to observation likelihoods, i.e,

$$p(o|c_i) = \frac{p(c_i|o)}{p(c_i)} p(o) \quad \textbf{(1)}$$

where c_i is the event that phoneme i is the correct phoneme, o is the acoustic observation, and $p(c_i)$ is the *a priori* likelihood of phoneme i. The unconditioned observation probability, $p(o)$, is a constant for all phonemes and can consequently be dropped without changing the relative relation between phonemes, and the *a priori* phoneme probabilities are easily computed from relative frequencies in training speech data. Thus, equation (1) can be used to define output probability density functions of a CDHMM.

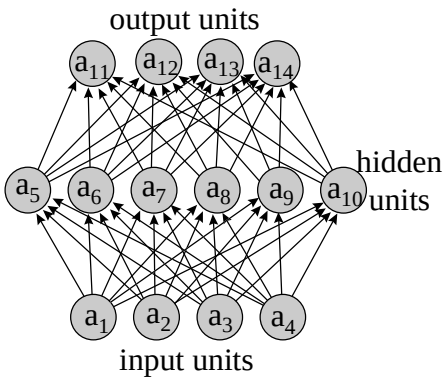
Back-propagation ANNs have intrinsically many of the features that have been added to the standard CDHMM in the development process discussed in the previous section. The normal back-propagation estimates MAP phoneme probabilities, not ML estimates that is the normal estimation method for CDHMM. As mentioned, MAP has better discrimination ability than ML, and is a more intuitive method to train a model for recognition.

Also the parameter sharing/tying is available in the ANN at no extra cost. This was introduced in the CDHMM with added complexity as a consequence, to be able to use complex probability density functions without introducing too many free parameters. In Figure 4 it is seen that all output-units in the ANN share the intermediate results computed in the hidden units. However, this total sharing scheme can sometimes hurt performance and it is therefore beneficial to limit the sharing of hidden units. This is discussed more in section 2.5.

The important short time dynamic features – formant transitions etc. – have been captured in ANNs by time-delayed connections between units (Waibel *et al.*, 1987). This is a more general mechanism than the simple dynamic features (1st and 2nd time-derivatives) used in the standard CDHMM. One use of time-delayed connections is to let hidden units merge information from a window of acoustic observations, e.g., a number of frames centered at the frame to evaluate. The same mechanism can be used to feed the activity of the hidden units at past times, e.g., the previous time point, back to the hidden units. This yields recurrent networks that utilize contextual information by their internal memory in the hidden units (e.g., Robinson and Fallside, 1991). A general ANN architecture that encompasses both time delay windows and recurrency, is presented in Paper 6.

Problems that are related to the HMM-part of the hybrid, are of course not solved by the introduction of the ANN. The Markov assumption, the Viterbi approximation etc. still remain. In many cases, the *ad hoc* solutions developed to reduce the effects of these problems for CDHMM can easily be translated to the hybrid environment, but in the case of context dependent models, e.g., tri-phones, there is an extra complication. It is not as straight-forward to apply Bayes' rule to the output activities when they are conditioned by the surrounding phoneme identities. It turns out that, to compute the observation likelihood in this case, the probability of the context given the observation is needed, i.e.,

$$p(o|l,c,r) = \frac{p(c|l,r,o)p(o|l,r)}{p(c|l,r)} = \frac{p(c|l,r,o)p(l,r|o)p(o)}{p(c|l,r)p(l,r)} \qquad \textbf{(2)}$$



$$a_i = \sigma\left(\sum_{j=1}^{n_i} w_{ji} a_j\right), \sigma(x) = \frac{1}{1+e^{-x}}$$

$$\text{e.g., } a_5 = \sigma\left(w_{15}a_1 + w_{25}a_2 + w_{35}a3 + w_{45}a_4\right)$$

Figure 4. Graphical representation of a feed-forward ANN. Associated with each node at each time is an activity. This is a real bounded number, e.g., [0; 1] or [-1; 1]. The activities of the input units constitutes the input pattern to classify. The activities of all other units are computed by taking a weighted sum of the activities of the units in lower layers, and then applying a compressing function σ to get a bounded value. The activities of the output units are the network response to the input pattern. To train an ANN for a particular task, a training database is prepared with input patterns and corresponding target vectors for the output units. The weights, w_{ij}, of the ANN are adjusted to make the output units' activities as close as possible to the target values. This is done iteratively in the so called back-propagation training.

where *c* is the phoneme, and *l* and *r* are the right and left context phonemes. This problem has been solved by (Bourlard and Morgan, 1993; Kershaw, Hochberg and Robinson, 1996) by introducing a separate set of output units for the context probabilities, $p(l,r|o)$, but their results indicate that the gain with tri-phones is smaller for the hybrid model than for the standard CDHMM.

2.5 Sparse connectivity and pruning in the ANN

The number of hidden units determines to a large extent a network's ability to estimate the *a posteriori* probabilities accurately. One could argue that it is the number of free trainable parameters in the network that is the important factor. In this view, it is not the number of units, but the number of connections that is important. However, from experience we know that not only the number of parameters is important, but also how they are put to use. In large, fully-connected networks, the number of connections is several orders of magnitude higher than the number of units. This means that each unit has several hundred, sometimes thousands of in-flowing connections. It is unlikely that all these connections can provide useful information to the one-dimensional output of the unit. Also, by studying the distribution of the weights in trained ANNs it can be noted that there is a high concentration of weights close to zero. Therefore it can be advantageous to work with sparsely connected networks where units are connected only to a fraction of all units in higher layers.

In Paper 6 I show that the ANN performance can be greatly increased by shifting the balance between the number of units and connections by introducing sparse connectivity. Two different approaches for achieving sparse connectivity are explored: connection pruning and sparse connection.

Connection pruning is a method that is applied after the network is trained. The network is analyzed and each connection is given a measure of *salience*. A fraction of the connections with the smallest salience is then removed and the resulting, smaller ANN is retrained. The most well-known pruning criterion is due to Le Cun, Denker, and Solla (1990), and was given the imaginative name "Optimal Brain Damage". In this method the salience depends on the second derivative of the objective function of the back-propagation. In

Paper 6, a more simplistic measure is used, the salience is simply the magnitude of the connection weight.

In Paper 6, a reduction to about half the number of connections was found to be possible without significant performance degradation. The computational complexity for running the ANN is proportional to the number of connections. Thus, this may be of great importance when computation is critical. Since pruning reduces the number of free parameters, it can also improve the network's generalization ability (e.g. Le Cun, Denker and Solla, 1990; Sietsma and Dow, 1991), but since we used a truncated training that handles problems with over-adaptation to the training data, this potential benefit is less important in our case.

Pruning the connections of an already trained network has no impact on the computational effort for training. To be able to apply the pruning criterion to the connections, the weights of the fully connected network must first be trained. Therefore, a second method – to start the training with already sparsely connected networks – was also explored in Paper 6. Before training, there is no available information about which connections are salient, so a random set of connections must be selected. Of course, this is in general not an optimal set, but the results show that sparsely connected ANNs perform much better than their fully connected counterparts with equal number of connections.

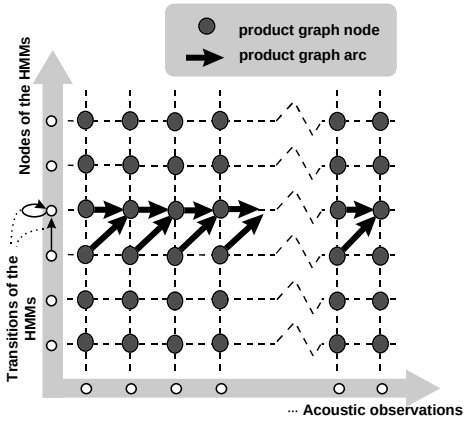


Figure 5. The product graph. This graphical construction visualizes the Viterbi search. The nodes of the product graph are the Cartesian product of the HMM-states and the sequence of acoustic observations. The Viterbi search is a DP search to find the most likely path through the graph. In the figure, only two of the many transition arcs of the HMMs are shown. The corresponding arcs of the product graph are also drawn. Details of the construction of the product graph are found in Paper 1 and Paper 3.

3. Dynamic Decoding

3.1 Introduction

The processing step after the acoustic classification is the dynamic decoding. These are the methods which are used to incorporate the dynamic constraints imposed by the HMM part of the hybrid HMM/ANN. Paper 3 covers in detail the dynamic decoding methods used in the WAXHOLM system and several of the included papers (see also section 6.1). The most basic and widespread method is the Viterbi algorithm. It finds the most likely path of HMM-states for an utterance. A more accurate method is to compute the most likely sequence of phones or words given the utterance, summed over all state sequences. However, this method is too computationally demanding for all but the simplest applications.

The Viterbi algorithm is discussed in the next section, 3.2. Section 3.3 deals with the method used in several of the included papers for finding multiple promising hypotheses instead of only the most likely one (the A* algorithm). Multiple hypotheses constitutes a better interface for information exchange between the recognizer and higher levels of the application. This is because the top-down constraints accessible in the higher levels can then be utilized (e.g., Hetherington, Phillips, Glass and Zue, 1993; Murveit, Butzberger, Digalakis and Weintraub, 1993; Ney and Aubert, 1994). In section 3.4, an efficient storage format for the multiple hypotheses is discussed, and in section 4 the methods presented in (Ström, 1995) for minimization of the storage are covered.

3.2 Viterbi decoding

The Viterbi algorithm (Viterbi, 1967) solves the problem of finding the most likely sequence of HMM-states given a sequence of acoustic input observations, i.e., an utterance to recognize. It is the standard decoding scheme used in different flavors in virtually all state-of-the-art recognition systems. To understand the Viterbi algorithm it is useful to think of the decoding problem as an optimal path-search in two dimensions – time-points and HMM-states. Figure 5 shows the states of the HMMs of a recognizer duplicated for each time-point, and lined up in columns. This graphical construct was introduced in (Ström, 1994a) and was named the “product graph” because the

nodes are the Cartesian product of the nodes of the HMMs and the time-points.

In Paper 3, I account in detail for how the best path is found in the 2-dimensional grid of Figure 5. The general principle is *dynamic programming* (DP), which yields a time-synchronous search, i.e., time-points are processed one at a time. Therefore the search can be performed in real time as the speaker utters a sentence. This is an important feature in human/machine dialogue where short response times often are critical for users' acceptance of the systems.

The Viterbi search is the most computationally demanding part of the dynamic decoding of the system described in Paper 3, but several approximations are used to keep computation down. The most important one is beam pruning. A combination of two different pruning criteria – both a threshold on the path probabilities and a maximum number of alive paths – are enforced. A section of Paper 3 is devoted to this topic that is of great importance for fast computation. It is shown that the combination of the pruning criteria is more efficient than each criterion by itself.

3.3 The N-best paradigm and A* search

A human/machine dialogue system is a large and complex software program that is impossible to construct and maintain without a great deal of modularity. For example, the ASR module takes audio speech input and delivers a symbolic representation of the utterance as its output. Interpretation of the meaning of the utterance is left to higher level modules. The symbolic representation can simply consist of the most likely string of words computed by the Viterbi algorithm. However, because the ASR module does not necessarily have access to information about the current dialogue state, or a deep syntactic, semantic and pragmatic analysis of the utterance, this is probably not the optimal interface to other modules.

An output representation that takes better advantage of the acoustic evidence is one where the ASR module delivers a set of acoustically scored hypotheses to higher modules. These modules can then select the most likely word-string hypothesis on the basis of both the acoustical evidence computed by the ASR module, and the syntactic, semantic and pragmatic analysis. A popular representation of this set of hypotheses is an *N-best list*. This is simply a list of the *N* most likely word strings given the sequence of acoustic observation vectors. Thus, an N-best list is a

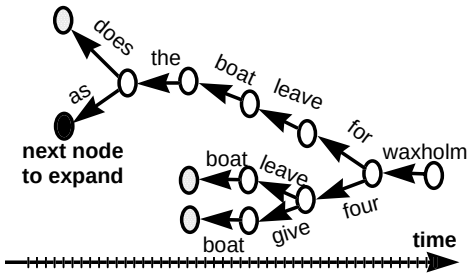


Figure 6. The search tree of a stack-search backwards from the end of an utterance. During the search, the tree is expanded by selecting the most "promising" leave and growing new word-branches from it. The key to a successful algorithm is how to determine which leave is the most promising. See the main text for details.

generalization of the output of the Viterbi algorithm (in the case of the Viterbi algorithm, $N = 1$).

Finding the N-best word-strings is a significantly harder problem than just finding the most likely word-string. It can be solved by extending the DP search and keeping a stack of hypotheses at each search point in Figure 5 instead of just the most likely as in the standard Viterbi algorithm. This has the advantage of preserving the time-synchronous feature of the Viterbi algorithm, but the complexity is proportional to N, i.e., the already computationally demanding algorithm becomes slower in proportion to the desired number of hypotheses.

A more computationally efficient method is the so called A* (a-star) search. This method is based on the stack decoding paradigm where a search tree is built with partial hypotheses represented by paths from the root to the leaves. The word-symbols are associated with the branches of the tree (see Figure 6). During the search, most paths are *partial hypotheses*, i.e., they do not cover the whole utterance. The partial path that is most promising according to a *search heuristic* is expanded by growing new branches from its leave. When there are enough complete paths in the tree, the search is terminated.

The key to an efficient A* search is the search heuristic. When evaluating partial paths, the likelihood of the words in the path so far, given the acoustic observation, is easily accessible and should of course be utilized. But it turns out that this is not enough – it is also necessary to estimate the influence of the remainder of the utterance. It is possible that a partial path with a high likelihood turns out to be in a “dead-end”, and can not be completed with a high likelihood.

It was an observation of Soong and Huang (1991) that made A* practically applicable to the N-best problem in ASR. They realized that the likelihoods computed in the Viterbi algorithm constitute a particularly well behaved A* heuristic. In the Viterbi search, the highest likelihoods of the observations from the beginning of the utterance to all points in the product graph (see Figure 5) are computed. If the A* search is performed backwards from the end of the utterance, the partial paths are from some interior point to the end of the utterance, and the remainder of the utterance is from the beginning to the interior point. Thus, the best possible likelihood for the remainder of the utterance is the likelihood computed in the Viterbi search. This particular heuristic has many attractive

features, one is that complete paths are expanded in order of likelihood, i.e., when *N* complete paths with different word strings have been expanded, the search can be terminated.

The development of an A* search algorithm, currently used in the WAXHOLM dialogue system has been a significant part of my work in the ASR field, and is reported on in Paper 1 and Paper 3. The most original aspect of the work in this area is the optimization of the "lexical network", introduced in Paper 1, that greatly reduces computation in the search. The optimization method is developed further in Paper 3, where the concept of *word pivot arcs* is introduced. Although in this case it is used only in the A* framework, word pivot arcs are relevant also in standard Viterbi decoding, because they can be used to significantly reduce the size of the product graph.

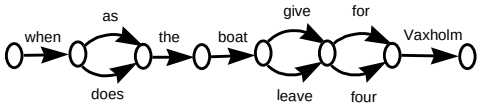
3.4 Word lattice representation of multiple hypotheses

Passing N-best lists from the ASR module as the recognition result to higher level modules is a step that increases the coupling between the modules without decreasing the modularity. It is a clean interface between modules, but in some cases the entry in the N-best list that is optimal after considering knowledge sources provided by higher level modules, is very far down the list. Thus, it is necessary to pass long lists between the modules. In this case, a so called word-lattice or word-graph is a better representation of the hypotheses because it is more compact (see Figure 7 and Figure 8).

In Paper 3, the A* search is modified to produce a word-lattice instead of an N-best list. A word lattice is a graph that generates all hypotheses, including all time-alignments of the words, above a likelihood threshold. This is the method now used in the WAXHOLM demonstration system (see section 6.1). The dialogue module of the WAXHOLM system requires an N-best list as input, but this is computed in a subsequent search in the word-lattice.

The direct construction of a word-lattice in the A* search gives a cleaner implementation, but also improves the performance. It is computationally advantageous to make a separate search for the N-best hypotheses in the produced word-lattice instead of producing the list while performing the first A* search. The reason is that finding the N-best list is inherently an

Word graph



N-best list

when as the boat give for Vaxholm
when does the boat give for Vaxholm
when as the boat leave for Vaxholm
when does the boat leave for Vaxholm
when as the boat give four Vaxholm
when does the boat give four Vaxholm
when as the boat leave four Vaxholm
when does the boat leave four Vaxholm

Figure 7. Two different representations of a set of hypotheses. From this simple example, the difference between the two formats is clearly seen. The entries in the N-best list are typically similar to each other, therefore it is sometimes practical to work with the more compact word graph representation.

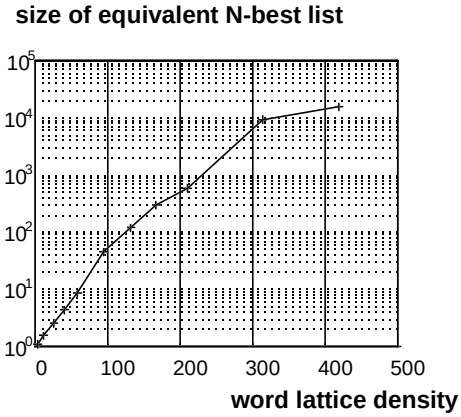


Figure 8. This graph shows the size of the word lattice (number of arcs per word) versus the number of entries in an hypothetical N-best list covering the same hypotheses. It is easy to see the benefit of the word lattice for large sets of hypotheses. The example is taken from Paper 3.

exponential complexity problem while constructing the word-lattice is of polynomial complexity (this is discussed in Paper 3).

4. Word graph minimization

This section is an extended version of the summary (Ström, 1995) of a presentation at the IEEE Workshop on Automatic Speech Recognition, 1995. This extended material has not been published before.

4.1 Introduction

With the development of complex speech technology applications, the ability of an automatic speech recognition (ASR) system to output multiple recognition hypotheses is becoming increasingly important. Examples of such applications are human/dialogue applications and machine translation where the ASR must interact with modules that handle discourse, semantics and pragmatics. Passing multiple recognition hypotheses to the higher levels of the system is a way of increasing the coupling without decreasing the modularity, which is essential for the development of large complex systems.

Another use is hypothesis re-scoring, where the multiple hypotheses are an internal, intermediate representation generated by a fast initial search. These are then used to limit the search-space in a second, more accurate search.

Word-lattices and word-graphs provide a compact representation of sets of hypotheses as described in section 3.4. I will mean by a word-lattice, a word graph that contains all time alignments that have a likelihood above the pruning threshold.

For re-scoring using new acoustic models, the word-lattice constructed in the A* search is a good representation because in this case, it is necessary to re-score the different alignments. However, if the objective is re-scoring using only a new grammar, the multiple time-alignments are excessive, and a more compact representation is better.

4.2 Problem formulation

It is not trivial to define a measure of size for word-graphs. From a practical point of view, the size of the graph should reflect the amount of computation required to perform critical operations on the graph. Such an operation may for example be searching for the most likely path through the graph. The number of arcs is in many cases a good indication on the amount of computation required, and a wide-spread measure of word-graph size is *lattice density*. The lattice density of a word-graph is the number of arcs divided by the

number of words actually uttered (Ney and Aubert, 1994).

Another measure of size is the number of nodes in the graph. In contrast to the number of arcs, the number of nodes has typically only a small impact on the computation required for search operations. Instead the memory requirement of the algorithms is affected because search algorithms often store state-information for the nodes at each time-frame.

If the objective is re-scoring word-graphs with a new grammar, the time-alignments of words is not important and it is therefore desirable to reduce the size as much as possible without altering the generated set of word-strings. Thus, the problem can be defined as finding the word-graph with the minimum number of arcs or nodes that generates exactly the same word-strings as a given word-lattice.

4.3 Approximative methods

A word lattice is a directed acyclic graph (DAG) – a subclass of non-deterministic finite state automatons (NFA) (see for example Hopcroft and Ullman, 1979). Minimizing an NFA is a hard problem that can not in general be solved in polynomial time. Therefore, the problem has been attacked using heuristic methods that reduce the graph but not to its minimal size.

In the word-pair approximation (Ney and Aubert, 1994), it is assumed that the positions of word boundaries are independent. In most cases this is a sound assumption, but it may lead to search errors for short words and when a minimum duration constraint is enforced for phones.

In the A* search by Hetherington, Phillips, Glass and Zue (1993), arcs are not added to a node of the word-graph if they do not introduce a new partial word-sequence. This reduces the constructed graph significantly.

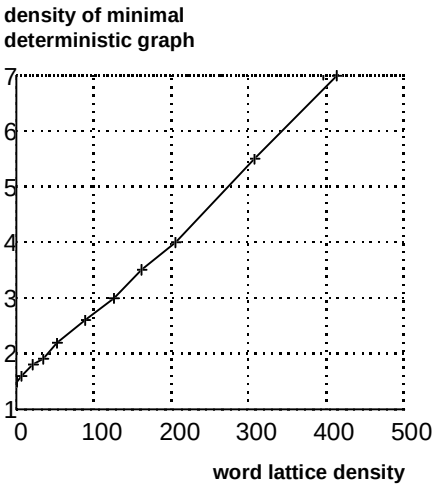


Figure 9. Size of the minimal deterministic graph as a function of the size of the original lattice. Sizes are measured in arcs per word – lattice density. This example was produced by running the ASR of the WAXHOLM dialogue system with varying pruning threshold. This figure is taken from Paper 3.

4.4 The minimal deterministic graph

Although the approximate methods can be quite effective, it is not satisfactory to settle with an approximation without having established a baseline with an exact method.

A well-known minimization algorithm exists for minimizing NFAs. The procedure results in a minimal (with respect to the number of nodes) equivalent deterministic finite automaton (DFA). This is a well defined objective, and in addition the deterministic

property of the resulting graphs has some other nice effects.

The algorithm (Hopcroft and Ullman, 1979) can be broken down into two steps. The first step is to construct an equivalent deterministic finite state automaton (DFA) – *the "determinization"*. A deterministic FSA has the property that, given a sequence of words, there is at most one path through the graph that generates it. This is accomplished for example by allowing only one start state and no more than one out-flowing arc for each word from any state. The second step is minimization of the DFA to obtain the minimal deterministic graph – *the minimization*.

The resulting reduction of the number of connections in the graph can be seen in Figure 9. In this example the reduction is about 50 times, but this is likely to be a problem dependent number.

Of the two steps, the determinization is the computationally hardest. In general the determinization can not be done in polynomial time or space. But as we will see, the particular structure of word-graphs can be utilized to attain an acceptable computational cost. The minimization algorithm has *N log(N)* complexity in general, but in the case of word-graphs this can be reduced to approximately linear complexity. In the next two sections the two steps are reviewed briefly. A more thorough account of the classic algorithm can be found in (Hopcroft and Ullman, 1979). Here we focus on the aspects that are specific to optimizing word-graphs – the short-cuts due to the special structure of word-graphs, and computational considerations given this particular type of NFA.

4.5 Determinization

The key to the determinization algorithm is the identification of sets of nodes in the original graph with single nodes in the resulting deterministic graph. This is also what causes the exponential computational complexity as there are 2^N sets of nodes in an original graph of N nodes. Fortunately, all sets will not be considered during the construction of the deterministic graph. Still, the main effort in the implementation of the algorithm involves the storage of sets of nodes in an efficient manner.

To simplify the algorithm slightly, we assume that there is exactly one start node and one end node in the graph. This can easily be enforced in the case of word-lattices, in particular if all utterances must begin and end with the special "silence" symbol.

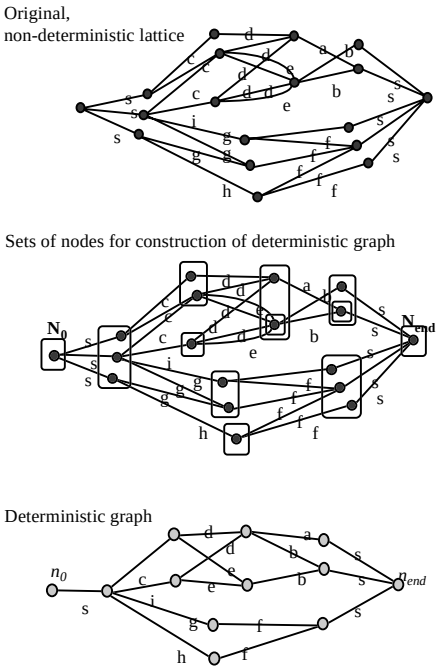


Figure 10. Illustration of the determinization algorithm described in the main text. The set N_0 is the first to be taken care of in step 4, and there are only words labeled "s" flowing out from it so *w*=s. The set of nodes reached by the s-arcs is the next node of the deterministic graph. This set in turn has four different words flowing out from it (c, i, g, h), and four corresponding new sets are formed. The process continues until no new sets are formed.

The algorithm can now be described as follows:

1) Identify nodes n_i of the deterministic graph with sets of nodes $\boldsymbol{N_i}$ in the original latice.
2) Define $\boldsymbol{N_0}$ to be the set containing only the start node of the lattice, and $\boldsymbol{N_{end}}$ to be the set containing only the end node of the lattice.
3) Initialize the deterministic graph with the node n_0.
4) For each node, n_x in the deterministic graph and each word w

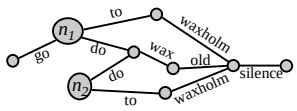
   Let $\boldsymbol{N_y}$ be the set of nodes in the lattice that can be reached from a node in $\boldsymbol{N_x}$ with the word w.

   If there is no node n_y in the deterministic graph - add it.

   Add an arc with word w from n_x to n_y.

The procedure is illustrated in Figure 10.

Computationally, the greatest problem is how to store the sets of nodes in an efficient manner. It is important to be able to quickly determine whether a set already exists, or if it represents a new node in the deterministic graph. In the implementation, this was solved by a combination of a hash-table and comparing sorted sets of nodes. The hash table is based on a hash-key that is independent of the order of the nodes, and the sets are not sorted unless necessary. This means that whenever an already existing set is looked up, it will be necessary to sort it and the lists with the same hash-key among the already existing. But no sorting is needed to insert a new set unless a hash-table clash occurs. Because large sets are likely to be looked up only once, the effect is that larger sets are seldom sorted.



n_1 and n_2 both generate exactly: {<to waxholm silence>, <do wax old silence>} and are therefore merged.

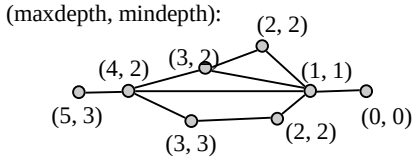


Figure 11. Illustration of the minimization. Each pair of nodes are checked to see if they generate exactly the same set of partial word-sequences forward. Because word-graphs are acyclic, it is possible to efficiently compute the longest and shortest generated word-sequences generated forward from each node (max and min depth). This reduces the number of node-pairs to compare significantly because only those with the same maximum and minimum depth need to be considered.

4.6 Minimization

The deterministic graph constructed in the previous section can now be minimized by considering the partial word sequences that can be generated forward from each node. We define for any path of arcs from a particular node to the end-node, the sequence of words on the arcs as *generated* by the node forward.

Any pair of nodes that generates exactly the same word-sequences can clearly be merged without changing the set of complete word-sequences generated by the graph. The special structure of word-graphs makes this merging particularly simple. Because all arcs flow from left to right, the nodes are processed from right to left. It is then guaranteed that when processing a particular node, all nodes to the right of it that can be merged, are in fact already merged.

Therefore, all nodes to the right of the current node generate different sets of sequences forward.

The notion of "left" and "right" can be formalized by introducing maximum and minimum depth. This is the length of the shortest and the longest word sequence generated from the node (see Figure 11). Only nodes that have exactly the same maximum and minimum depth need to be considered for merging. Further, if nodes are processed in order of increasing minimum depth, all that needs to be compared is the words on the out-flowing arcs and the node that they flow to. There is no need to consider more than one arc forward because all nodes forward generate different word-sequences. The procedure of merging nodes is illustrated in Figure 12 where the deterministic graph constructed in Figure 10 is continued with the minimization step.

In general, the minimization step has *N log(N)* computational complexity. However, because of the acyclic structure of word-graphs and the distribution of word duration, the number of nodes with identical maximum and minimum depth is approximately a linear function of utterance length.

4.7 Computational considerations

As noted earlier, the determinization procedure has, in the worst case, exponential computational complexity with respect to the number of nodes. However, the input lattice-size has at least two distinguishable dimensions: the utterance-length and the graph-density (number of arcs per utterance-length, Ney and Aubert, 1994). Graph-density corresponds to the amount of pruning and utterance-length corresponds to the length of the input. Of the two, the length of the input is varying, but the amount of pruning can be controlled in the A* search.

We studied empirically the time-complexity of the construction of the DFA in the 1000-word spontaneous speech task of the WAXHOLM-project (Blomberg *et al.*, 1993; Paper 3). In the top graph Figure 13 the exponential behavior can be seen, but the bottom graph shows approximately linear time-dependence with respect to utterance-length when the dependency of the graph-density has been eliminated. This difference in behavior for the two dimensions is very important because it gives us the possibility to control the "exponential explosion" of the algorithm by tuning the pruning.

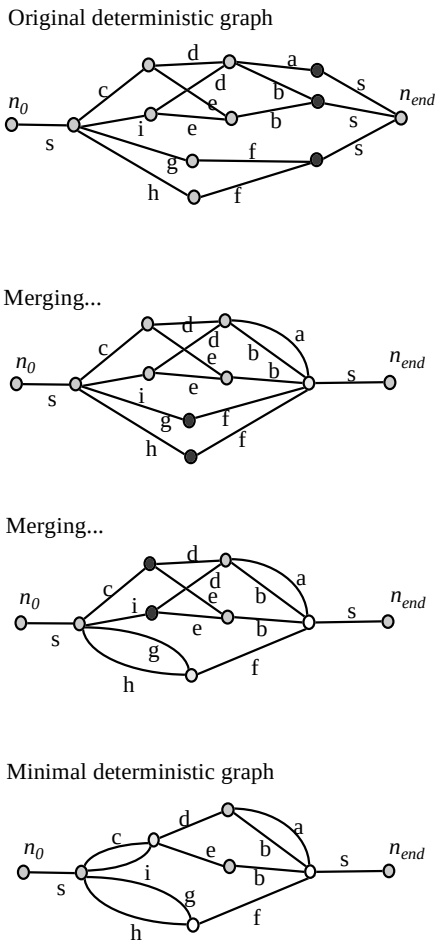


Figure 12. Illustration of the merging of nodes in the minimization step of the algorithm. Merging proceeds from right to left and because all nodes to the right are merged if possible, it is easy to check if two nodes generate the same word sequences forward. See the main text for details.

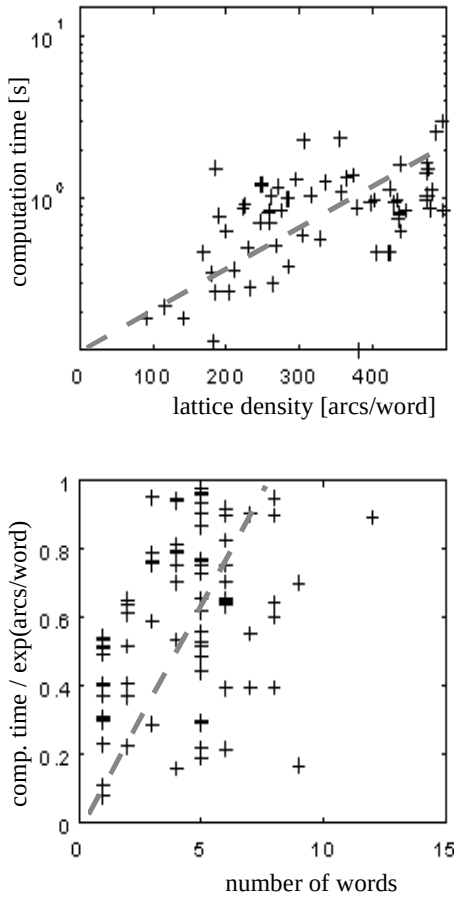


Figure 13. Top: Time for construction of the DFA as a function of the lattice-density.Bottom: Construction time divided by the exponential of the lattice density. The simulation was made on a SPARC 10 workstation.

For comparison, the word-pair approximation was also investigated. Applying the word-pair approximation to the word-lattice resulted on average in a reduction in the number of arcs to 8.1% of the original lattice. The minimal deterministic graph for the same word-lattice was on average reduced to 1.4% of the lattice.

4.8 Discussion

The exact minimization algorithm investigated in this study was clearly shown to produce smaller graphs than the approximate word-pair approximation method. This is true in spite of the fact that the exact algorithm minimizes the number of nodes, but the two methods are compared on the basis of the number of arcs. In the example investigated, the exact algorithm produced almost six times smaller graphs than the approximate method, but the exact quotient is likely to be dependent on the lexicon, pruning aggressiveness etc.

In contrast to the word-pair approximation, the graph of the proposed algorithm generates exactly the same word-strings as the original lattice (no hypotheses are lost by approximation). However, since the proposed method is exponential with respect to lattice-density, a hybrid-approach where the lattice is first reduced by the word-pair approximation and then minimized, seems to be an attractive alternative.

The computational demands of the exact minimization are probably too high for use in the on-line recognition mode of a real-time ASR system (with contemporary computer technology). However, for off-line tasks, where the same minimized word-graphs are used repeatedly, the effort may be worthwhile. Iterative estimation of grammar parameters during training of the system is one example.

5. Speaker adaptation

5.1 Introduction

Speaker adaptation is one of the main areas of work of this thesis. In particular, rapid adaptation to a new voice has been investigated. Although there is evidence for such an adaptation process in speech perception, the role of speaker adaptation in the speech perception process is still not completely understood. In a famous experiment with synthetic speech stimuli, Ladefoged and Broadbent (1957) found that altering the formant frequencies in a precursor utterance resulted in differently perceived identity of a target word. To account for this effect they postulated a psychological adaptation process and concluded that:

> *"... unknown vowels are identified in terms of the way in which their acoustic structure fits into the pattern of sounds that the listener has been able to observe."*

This view has been very influential, but it has been challenged by others arguing that the effect is of minor importance in the perception of natural speech. In particular, the dynamic patterns of the vowels in their consonantal context have been ascribed more importance (Verbrugge and Strange 1976, Strange 1989). Strange (1989) wrote:

> *"If, as perceptual results suggest, there is sufficient information within single syllables to allow the listener to identify intended vowels, even when those vowels are coarticulated by different speakers in different consonantal contexts, then the need to postulate psychological processes by which the perceiver enriches otherwise ambiguous sensory input is eliminated."*

In ASR, the performance gap between speaker-independent (SI) and speaker dependent (SD) systems (e.g., Huang and Lee 1991) indicates that there is a lot to be gained from adapting SI-models to the speaker. Certainly, the variation due to consonantal context is recognized as an important property of the speech-signal also for ASR. This is usually modeled by context dependent models, e.g., triphones. Nevertheless, there is general agreement that speaker adaptation schemes can further improve recognition performance

significantly. An overview of existing methods for speaker adaptation in ASR is given in Paper 4.

Paper 2, Paper 4 and Paper 5 represent a line of work that aims at performing rapid speaker adaptation by accessing knowledge from an explicit model of the speaker variability. The idea is that *a priori* knowledge of the speaker variability will reduce the amount of adaptation data necessary from the new speaker by constraining the parameter space. For example, the variability due to varying vocal tract length affects the formant patterns of all voiced phones in a coherent and systematic fashion. Thus, if the rules governing this variability are known, it is not necessary to collect adaptation data for all phonemes.

Key concepts of the framework of Paper 2, Paper 4 and Paper 5 are the *speaker model* that models the speaker variability, and the *speaker space* that is the domain of a set of parameters describing different voice characteristics. A speaker model is basically a probability distribution function over a speaker space.

5.2 Speaker modeling

A speaker model is a statistical model of the speaker variability. It includes a parametric description of the variability and an *a priori* probability distribution for the different characteristics. For example, vocal-tract length is a characteristic that can be of importance in a speaker model, independent of the particular ASR method used.

Vocal-tract length is an example a of *speaker parameter* – a parameter that describes the speaker. The speaker-space is the domain of the speaker parameters. In this framework, individual speakers have positions in the speaker-space and adapting to a speaker involves estimating this position. Note however that speaker parameters are not necessarily physiologically related as in the previous example. In Paper 2, I experiment with a data driven method to extract a speaker space that does not explicitly correspond to any knowledge-based parameters.

An explicit model of speaker variation may offer other advantages than increased speech recognition performance. The speaker-model can provide an interface for coupling with other modules of the human-machine interface. For example, consider the two possible speaker parameters dialect and age, both known to affect the acoustic realization of the speech. But speakers of differing dialects also have different

lexical preferences (Shiel 1993), and speakers of different age are interested in different subjects.

5.3 Speaker-sensitive phonetic evaluation

In the speaker modeling approach to speaker adaptation, discussed in the previous section, the phonetic evaluation is conditioned by the speaker parameters, i.e., an utterance is recognized differently depending on the hypothesized voice characteristics of the speaker. A speaker-sensitive phonetic classifier is a phonetic classifier that is dependent on the speaker parameters of the hypothesized speaker. In the case of an ANN classifier, the speaker parameters can be introduced in the network as extra input units (see Figure 14).

The speaker parameters can be thought of as the "source" of the variation. Thus, changing a speaker parameter related to the vocal tract length should alter the ANN's probabilities for all voiced phonemes. The ease of modeling such complex systematic variation shared by different phonemes is an important strength of explicitly modeling the speaker variability.

An example of a speaker-sensitive ANN can be found in (Carlson and Glass, 1992a,b), where the acoustic observations as well as the speaker parameters are input to the network. This study inspired the work reported in Paper 2, Paper 4 and Paper 5, where the same basic ANN structure is used. In this continued work, the domain has been extended from only the vowels to the whole phoneme inventory, and from classifying segments to the complete continuous speech ASR problem.

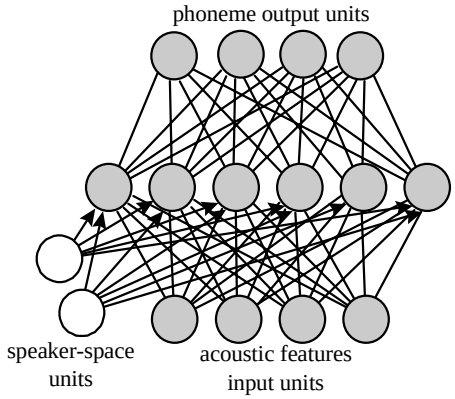


Figure 14. Structure of a speaker-sensitive ANN for phoneme probability estimation. Speaker parameters are introduced to the ANN by means of the special-purpose speaker-space units. The speaker space units are connected to other units like other input units, but they get their activation values from the speaker adaptation.

5.4 Speaker adaptation in the speaker modeling framework

In the previous sections it was discussed how to model the speaker variability, but the particular adaptation procedure is not specified. This is not a coincidence – in fact, one of the advantages of the speaker modeling framework is that the adaptation procedure is not determined by the model of the variability. Different adaptation schemes may be chosen for different tasks. For example the amount of adaptation speech available, text dependent/independent adaptation, real-time requirements etc. may be of relevance for the particular adaptation scheme chosen.

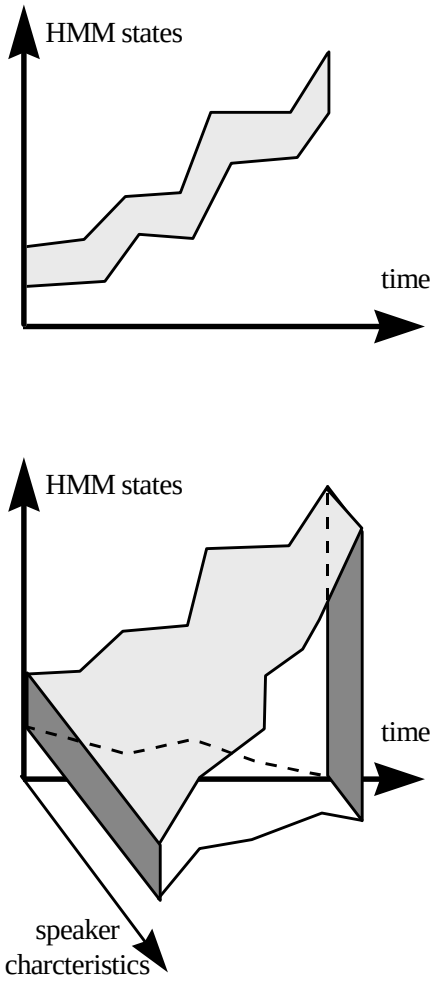


Figure 15. Search-space of the dynamic decoding. Top: the standard search space with the two dimensions time and HMM states. The objective of the dynamic decoding is to find the most likely path through the search-space. Because of beam-pruning, many paths in the search space are never investigated. This is indicated by the shadowed "beam" in the figure. Bottom: the search space with an additional dimension of speaker characteristics. At the beginning of the search, all possible speaker characteristics are inside the beam, but as the search progresses, unlikely speaker characteristics are successively pruned, and the speaker's position in the speaker space is gradually more specified.

Clearly, speaker adaptation in the speaker modeling framework includes, in one form or the other, estimation of the current speaker's position in the speaker-space. If the speaker's position is known, this information can be used to condition the phonetic evaluation, and give a more accurate recognition. If the exact position in the speaker-space is unknown, it must be estimated from the knowledge sources at hand. This includes speech recorded previously from the speaker (possibly only the one utterance to recognize), but it could also include other types of information that is available to the system. For the analysis of the recorded speech, features that are often discarded in ASR can potentially be of use for the speaker characteristics estimation, e.g., fundamental frequency, that is strongly correlated with gender.

The estimation of speaker-space position needs not be explicit. An useful concept is the so called *speaker consistency principle*. This is a formulation of the observation that an utterance is spoken by one and the same speaker, from the beginning to the end. This constrains the observation space and can therefore be used to reduce the variation in the ASR model. In the speaker modeling framework, the speaker consistency principle can be introduced by enforcing constant speaker parameters throughout the utterance. This can be implemented by adding a new dimension to the search space of the dynamic decoding. The original two dimensions: time and HMM state, are then complemented with the third dimension of the speaker parameters. This is the method used in Paper 5. The extended search space is illustrated in Figure 15.

The dynamic decoding search in the extended search space of Figure 15 is pruned with beam-pruning just like in the case of the standard search in two dimensions. The effect is that partial hypotheses with low probability will not be further investigated in the search, leaving more computational resources for the more promising hypotheses. In the extended search space, a part of a hypothesis is the speaker characteristics, so the effect of beam pruning is that hypotheses with unlikely speaker parameters will be pruned. In effect this is speaker adaptation – as the Viterbi search progresses, unlikely speaker characteristics are successively pruned, and the speaker's position in the speaker space will gradually be more specified. Consequently, more resources can be allocated for the other dimension. Thus, in this framework, adaptation in the sense that something in

the system is progressively changed, is the changed balance of attention in the search from speaker characteristics to HMM states.

As a final note, we point out that the speaker modeling approach and speaker consistency modeling do not require an ANN model – a related adaptation scheme in the HMM domain is found in a paper by Leggeter and Woodland (1994), and a slightly different approach to implement the speaker consistency principle is taken by Hazen and Glass (1997). In their consistency model, the key concept is long range correlation between speech sounds. No explicit speaker model is used, but the method is successfully combined with speaker clustering and a technique called “reference speaker weighting”. Both of these implicitly define speaker models by the space spanned by the clusters and reference speakers respectively.

6. Applications

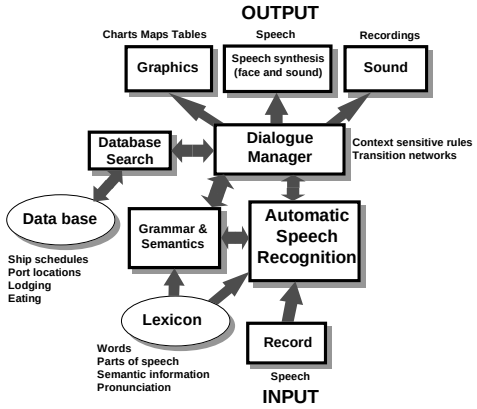


Figure 16. Overview of the WAXHOLM demonstrator system. See the main text for details.

6.1 The WAXHOLM dialogue demonstrator

The WAXHOLM human-machine dialogue demonstrator is built on a generic framework for human/machine spoken dialogue under continuous development at the speech group at the department for Speech, Music and Hearing of KTH. The domain of the WAXHOLM application is boat traffic and tourist information about hotels, camping grounds, and restaurants in the Stockholm archipelago. The application database includes timetables for a fleet of some twenty boats from the Waxholm company, which connects about two hundred ports. The user input to the system is spoken language exclusively, but the responses from the system include synthetic speech as well as pictures, maps, charts and timetables (see Figure 16).

The ASR module of the system, described in detail in Paper 4, has a domain-dependent vocabulary of about 1000 words. The application has similarities to the ATIS domain within the ARPA community, the Voyager system from MIT (Glass *et al.*, 1995) and European systems such as SUNDIAL (Peckham, 1993), Philips's train timetable information system (Aust *et al.*, 1994) and the Danish dialogue project (Dalsgaard and Baekgaard, 1994). Summaries of the WAXHOLM dialogue system and the WAXHOLM project database can be found in (Bertenstam *et al.* 1995a,b) and an early reference is Blomberg *et al.* (1993).

The demonstration system is currently mature enough to be displayed and tested outside the laboratory by completely novice users. A successful such attempt was made at "Tekniska Mässan" (the technology fair) in Älvsjö in October '96. Visitors with no prior experience with the system were invited to try the demonstrator in a rather noisy environment.

6.2 An instructional system for teaching spoken dialogue systems technology

Human/machine dialogue systems are large complex software projects, and have traditionally required high expertise to design and develop. However, recently efforts have been made to make this type of user interface available for developers that are not experts in the area. Central to this development are the modular toolkits that are being developed, e.g., OGI's CSLUsh (Sutton *et al.*, 1996), and MIT's Sapphire (Hetherington and McCandless, 1996) toolkits. A toolkit in the same spirit is also being developed at the department for Speech, Music and Hearing (Sjölander and Gustafson, 1997). Existing components for speech recognition, speech synthesis, visual speech synthesis and NLP tools have been extracted and re-designed to fit in a common framework under the Tcl language. The Tcl language has many shortcomings, but is convenient for rapid prototyping and development work on the system integration level, and offers good graphical support through the accompanying Tk-widget set. The ASR system described in the previous sections developed for the WAXHOLM system is the underlying speech recognition module of the toolkit.

The increased availability of the technology makes student courses in the subject matter possible. A simple, but fully functioning dialogue instructional system has been developed using the toolkit for educational purposes (Sjölander and Gustafson, 1997). The system has been used for courses at the MSc level at KTH, and at Linköping University in Sweden. In this environment, students are presented with a simple spoken dialogue application for yellow pages search on a few selected topics using voice input. The application is accompanied by a development environment that allows the students to interactively follow the processing in the system and modify the different modules even while it runs. A screen-shot of the system in use is shown in Figure 17

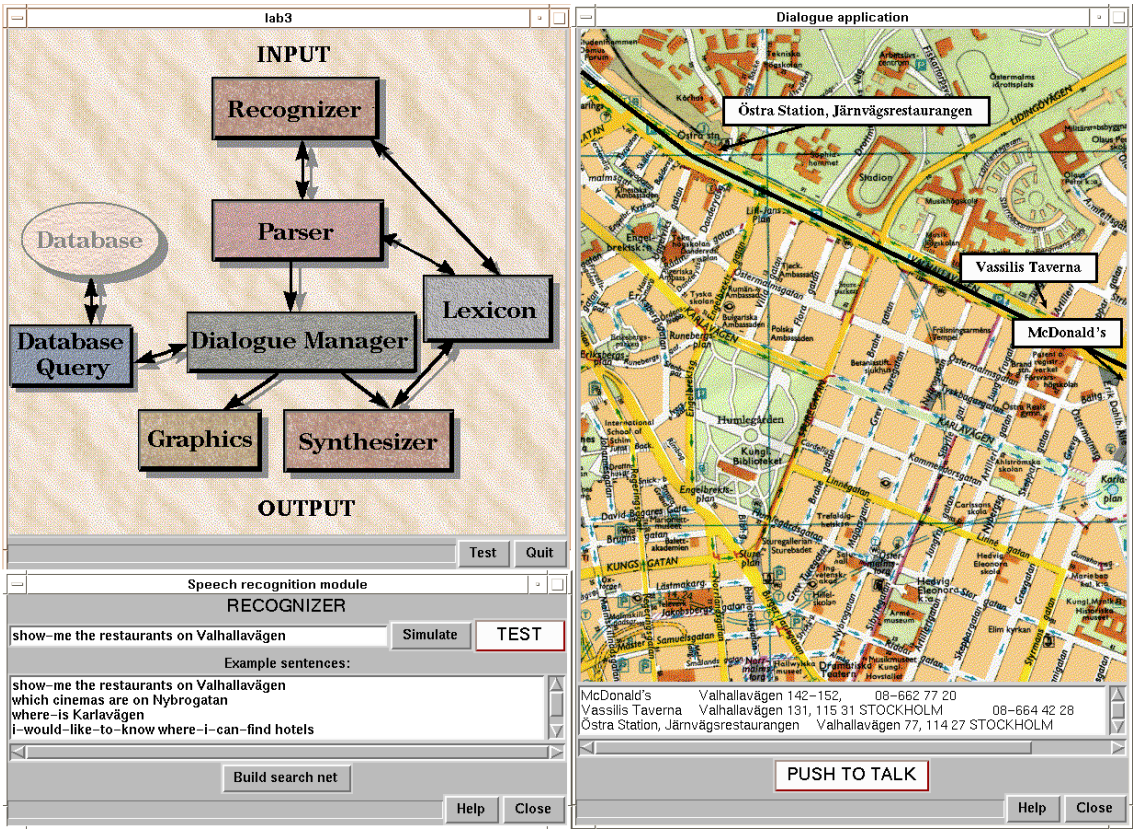


Figure 17. screen-shot of the instructional dialogue system. Upper left: the control window. Right: the dialogue application. Bottom left: the speech recognition module.

7. Summaries and comments on individual papers

7.1 Paper 1.

"Optimising the Lexical Representation to Speed Up A* Lexical Search,"
STL QPSR 2-3/1994, pp. 113-124.

This paper was written after the first implementation of the ASR module of the WAXHOLM demonstration system. Since then, the system has continuously been improved, and has been one of the main tools for the experiments of this thesis.

The A* algorithm is defined in a graph formalism, where both the input and output of the algorithm, and the lexicon and grammar model are represented by directed graphs. The acoustic input observation of an utterance, and the word sequence hypotheses that are the output, are represented by graphs called observation graphs. The nodes of observation graphs have a time-tag, and arcs between the nodes represent some aspect of the observation between the two times. The graph representing the lexicon and grammar model is a probabilistic finite state automaton, e.g., an HMM. The graph formulation of the A* algorithm is also used in Paper 3 and has been a useful conceptual tool in the development of the search.

A large part of the paper is concerned with computational aspects of the search. CPU and memory requirements are discussed for the implementation of the first pass of the A* algorithm, but it is the graph reduction methods in the second, stack-decoding, pass of the implantation that is the most important contribution of this paper. The idea is to utilize the regularities of the words in the lexicon to reduce the effective search space. For example, many words begin with the same phoneme, and it is not necessary to evaluate them more than once at a particular time. This notion is generalized with the help of quotient graphs. The quotient graph is typically a significantly smaller graph than the original model graph, but all nodes of the original graph have a corresponding node in the quotient graph. The search can be made in the quotient graph, and when a search result is needed for a node in the original graph, there is always a reference to a quotient graph node where the result is available.

7.2 Paper 2

"A Speaker Sensitive Artificial Neural Network Architecture for Speaker Adaptation,"
ATR Technical Report, TR-IT-0116, 1995,ATR, Japan.

In this report the speaker sensitive ANN, introduced in Ström (1994b), is further investigated. The general framework, an ANN for phoneme evaluation with a set of extra input units that characterizes the speaker is inspired by the work of Carlson and Glass (1992a,b), and is also used in Paper 5 and discussed in Paper 4.

In this report, the speaker parameters supplied by the extra input units are automatically extracted, i.e., they have no explicit relation to any knowledge-based parameters. However, using a novel analysis-by-synthesis method, the influence of the automatically extracted parameters was visualized in the formant space. An ensemble of synthetic vowels were generated by a formant synthesizer driven by an LF voice source (Fant, Liljencrants and Lin, 1985). The individual vowels of the ensemble were varied only in F1 and F2, (the first and second formant frequencies). The synthetic vowels were then fed to the ANN and the vowel classification was recorded. This makes it possible to draw a map with the phoneme boundaries in the F1/F2 space for the ANN. By repeating the procedure with different speaker parameters, the effect of the speaker parameters on the phoneme boundaries can be studied. It was found that, in agreement with theory, the phoneme boundaries were lower in frequency for speaker parameters corresponding to male voices than female. In another analysis, a correlation between the knowledge-based parameter fundamental frequency, and one of the two automatically generated parameters was also found.

This report was written during my stay as a guest researcher at ATR, Kyoto, Japan. As a curiosity I can mention that, as part of my attempt to learn the Japanese language, it was written on a Macintosh with only Japanese labels on all buttons and menu items. However, I never quite succeeded in learning the kanji characters, so sometimes some pretty spectacular things happened on the screen. This was how I learned to recognize the “undo” character.

7.3 Paper 3

"Continuous Speech Recognition in the WAXHOLM Dialogue System,"
STL QPSR 4/1996, pp. 67-96.

This paper presents the status of the continuous speech recognition engine of the WAXHOLM project at the end of 1996. At this point the demonstration system was mature enough to be displayed and tested outside the laboratory by completely novice users. A successful such attempt was made at "Tekniska Mässan" (the technology fair) in Älvsjö in October '96 where visitors with no prior experience with the system were invited to try the demonstrator in a rather noisy environment.

All parts of the ASR module are described in some detail in the paper, including the different modes of operation: standard CDHMM, hybrid HMM/ANN, and a general phone-graph input mode. However, the focus is on the aspects of the system that are original in some sense, and the ANN part of the system is covered more thoroughly in Paper 6.

Most of the report is devoted to the dynamic decoding block of the system. The graph representation of the search space of the dynamic decoding is described in detail, and algorithms for graph reduction, Viterbi search, and A* stack decoding are given.

The optimization of the lexical graph is perhaps the most original aspect of the decoding block of the system. This is a continuation of the work reported in Paper 1. The resulting graph has only one word-end and one word-start node for each word class. This results in a very small number of word connecting arcs. Without the graph reduction, the algorithm would spend a large part of its CPU time processing the word connecting arcs. The key concept that was utilized to achieve this high degree of reduction is *word pivot arcs*. This is an innovation that was not used in Paper 1.

For the estimation of bigram grammar probabilities, a novel estimate based of Zipf's law for word frequency is proposed. However, the WAXHOLM corpus is not large enough for a conclusive judgment of the method.

7.4 Paper 4

"Speaker Modeling for Speaker Adaptation in Automatic Speech Recognition,"
in: Talker Variability in Speech Processing, Chapter 9, pp. 167-190, Eds: Keith Johnson and John Mullennix, 1997, Academic Press.

This book illuminates the subject of talker variability from the different viewpoints of research in auditory word recognition, speech perception, voice perception and ASR. A common theme of all chapters is to approach speaker variability as a source of information rather than unwanted noise. In particular in the ASR field, the latter have often been the case in the past. The approach is very apparent in my chapter, that describes the ideas behind the experiments with fast speaker adaptation, reported on in Paper 2 and Paper 5. The concepts of a speaker space, and an explicit model for the speaker variability are laid out.

The chapter also contains a survey of existing speaker adaptation methods, and comparison with adaptation effects in the human speech perception process is made.

7.5 Paper 5

"Speaker Adaptation by Modeling the Speaker Variation in a Continuous Speech Recognition System,"
Proc. ICSLP '96, Philadelphia, pp. 989-992.

This is the latest included paper on fast speaker adaptation. In this paper, the speaker modeling approach introduced in (Ström, 1994b), and developed further in Paper 2, is used in a full continuous speech recognition experiment.

The adaptation performance was evaluated on the WAXHOLM database. Overall, the adaptation effect was not very high, but the database contains a large portion of very short utterances (only a few words), and this was not enough for the adaptation to give a positive effect. On average the adaptation had positive effect on the word-error rate for utterances longer than three words, but increased the word-error rate for shorter utterances. The utterance level result, i.e., the fraction completely correctly recognized utterances, was in contrast, slightly improved for all utterance lengths.

Although it was shown that the proposed adaptation method can improve recognition both at the word and utterance level, the merit of this paper is not the results achieved. The ANN used in the study is rather small, the speaker space used in the experiments is simplistic (only two speaker parameters), and the sampling of the speaker space is coarse. The main contribution of this paper is that it showed that the method is feasible in a full ASR system.

7.6 Paper 6

"Phoneme Probability Estimation with Dynamic Sparsely Connected Artificial Neural Networks," *The Free Speech Journal*, Vol. **1**(5), 1997.

This paper focuses on the latest development of, and provides detailed information on a complete ANN toolkit, developed during the course of my thesis studies. The department of Speech, Music and Hearing at KTH has a history of research in the area (Elenius and Takacs, 1990), and the first step of the development of the toolkit is reported on in Ström, 1992. Formulae are given for back-propagation training of the generalized dynamic ANN architecture used. The dynamic networks require that units' activities are computed in a particular, time-asynchronous order, and an algorithm for computing this order is described in detail. All aspects of network training and evaluation are discussed; network topology, weight initialization, input feature representation, the "softmax" output activation function, and the theory that establishes the link between multi-layer perceptrons and *a posteriori* phoneme probabilities.

This paper also announces the toolkit as a publicly available software resource, available for free by the research community. In an appendix, detailed instructions are given for training and evaluating ANNs for phoneme probability estimation using the toolkit. The source code and documentation are available on the Internet. To this date, the toolkit has been downloaded from more than 75 different sites worldwide.

The innovative part of the paper is the sparse connection and connection pruning of the networks, that to my knowledge have not been used for phone probability estimation ANNs before. The sparse connectivity of the networks allows considerably enlarged hidden layers without increasing the computational demands, and this is shown to improve recognition accuracy significantly. The phoneme recognition results for the TIMIT database are in the range of the lowest error-rate reported, and outperforms all HMM based systems for the core test set of the database.

Since this paper was published, the development of the toolkit, and a new ANN architecture have resulted in further improvement of the phoneme recognition. Currently our lowest error-rate for the core test set of the TIMIT database is 26.7% (Ström, 1997). The development of other systems have also resulted in reduced error-rates, Chang and Glass (1997) report an error-rate of 26.6% using their segment based approach.

Acknowledgments

First of all, I would like to thank Evolution for coming up with the brilliant idea of using the same device for chewing, breathing, and emitting sounds for communication, making automatic speech recognition one of the most challenging engineering tasks one can think of.

I am grateful to my colleagues for making my work environment so stimulating, both at KTH, and during my stay as a guest researcher at ATR. In particular Rolf Carlson has given important guidance over the whole period. It was also studies of Rolf Carlson and James Glass that inspired to my work in the area of speaker adaptation. Michael Phillips deserves acknowledgment for introducing me to the A* algorithm and lexical search algorithms, and Kjell Elenius has always been a good conversation partner in ANN related issues.

Further, I would like to thank Björn Granström for allowing me to work in this area, and for guidance and inspiration. I would also like to thank Yoshinori Sagisaka for inviting me to work in the stimulating research environment at ATR in Kyoto, Japan.

I also take the opportunity to thank all my teachers, all the way from kindergarten to graduate course teachers. Of course, this thesis would have been impossible without them.

I thank my mother and father for all their support, and for providing a home where education was a natural part of life. All my friends deserve acknowledgment for being so patient with me when I occasionally completely forget about everything but my work. Finally, thank you Linda for love, support, and understanding.

My first two years of doctoral studies were sponsored by a donation by VOLVO AB, and the remaining studies were financially supported by NUTEK and HSFR, and during the last year also by CTT, a center for speech technology, jointly sponsored by, NUTEK, Swedish industry and KTH.

References

Ahadi S. M. and Woodland P. C. (1995) : “Rapid speaker adaptation using model prediction,” *Proc. ICASSP* 1995, pp. 684-687.

Aust H., Oerder M., Seide F., and Steinbiss V. (1994) : “Experience with the Philips automatic train timetable information system,” *Proc. of IEEE Workshop on Interactive Voice Technology for Telecommunications Applications* (IVTTA94), pp. 67-72.

Bertenstam J., Blomberg M., Carlson R., Elenius K., Granström B., Gustafson J., Hunnicutt S., Högberg J., Lindell R., Neovius L., de Serpa-Leitao A., and Ström N. (1995a) : “The Waxholm application database,” *Proc. EUROSPEECH '95*, Madrid, pp. 833-836.

Bertenstam J., Blomberg M., Carlson R., Elenius K., Granström B,. Gustafson J., Hunnicutt S., Högberg J., Lindell R., Neovius L., de Serpa-Leitao A., Nord L., and Ström N. (1995b) : “Spoken dialogue data collection in the Waxholm project,” *STL-QPSR, KTH, 1/1995*, pp. 50-73.

Blomberg M., Carlson R., Elenius K., Granström B., Gustafson K., Hunnicutt S., Lindell R., and Neovius L. (1993) : “An experimental dialogue system: Waxholm,” *EUROSPEECH '93*, pp. 1867-1870.

Bourlard H. (1995) : “Towards increasing speech recognition error rates,” *Proc. EUROSPEECH ’95*, pp. 883-894.

Bourlard H. and Morgan N. (1993) : “Continuous speech recognition by connectionist statistical methods,” *IEEE trans. on Neural Networks*, Vol. **4**(6), pp. 893-909.

Bourlard H. and Wellekens C. J. (1988) : “Links between Markov Models and Multilayer Perceptrons,” *IEEE Trans on PAMI*, **12**(12), pp. 1167-1178.

Brown P. F., Lee C-H, and Spohrer J. C. (1983) : “Bayesian adaptation in speech recognition,” *Proc. ICASSP 1983*, pp. 761-764.

Carlson, R. and Glass J. (1992a) : “Vowel classification based on analysis-by-synthesis,” *STL-QPSR* 4/1992, pp. 17-27, Dept. of Speech Communication and Music Acoustics, KTH, Sweden.

Carlson R. and Glass J. (1992b) : “Vowel classification based on analysis-by-synthesis,” *Proc. ICSLP 1992*, pp. 575-578.

Carlson R. and Hunnicutt S. (1996) : “Generic and domain-specific aspects of the Waxholm NLP and dialog modules” *Proc. ICSLP 1996*, pp. 677-680.

Chang J. and Glass J. (1997) : “Segmentation and modeling in segment-based recognition,” *Proc. EUROSPEECH 1997*, pp. 1199-1202.

Cohen J. (1996): “The summers of our discontent,” *Proc. ICSLP 1996, distributed on CDROM version*.

Dalsgaard P. and Baekgaard A. (1994) : “Spoken language dialogue systems,” *Proc. of Artificial Intelligence, Infix.* Presented at the CRIM/FORWISS workshop on Progress and Prospects of Speech Research and Technology, Munich.

Digalakis V. V., Ostendorf M., and Rohlicek J. R. (1992) : “Fast algorithms for phone classification and recognition using segment-based models,” *IEEE Trans. on Signal Processing*, Vol. **40**, pp. 2885-2896.

Elenius K. and Takacs G. (1990) : “Acoustic-phonetic recognition of continuos speech by artificial neural networks,” *STL-QPSR* 2-3/1990, pp. 1-44, KTH, Dept. of Speech Communication and Music Acoustics, Sweden.

English T. M. and Boggess L. C. (1992) : “Back-propagation training of a neural network for word spotting”, *Proc. ICASSP ´92*, Vol. **2**, pp. 357-360.

Fant G., Liljenkrans J., and Lin Q. (1985) : “A four-parameter model of glottal flow,” *STL-QPSR 4/85*, pp. 1-13, KTH, Dept. of Speech, Music and Hearing, Sweden.

Gauvain J. L. and Lee C. H. (1994) : “Maximum a posteriori estimation for multivariate Gaussian observations of Markov chains,” *IEEE Trans. Speech and Audio Processing*, Vol. 2(2), pp. 806-814.

Gish H. (1990) : A probabilistic approach to the understanding and training of neural network classifiers,” *Proc. ICASSP '90*, pp.1361-1364.

Glass J., Chang J., and McCandless M. (1996) : “A probabilistic framework for feature-based speech recognition,” *Proc. ICSLP ’96*, pp. 2277-2280.

Glass J., Flammia G., Goodine D., Phillips M., Polifroni J., Sakai S., Seneff S., and Zue V. (1995) : “Multilingual spoken language understanding in the MIT voyager system,” *Speech Communication* 17/1-2, pp. 1-18.

Hazen T. J. and Glass J. R. (1997) : “A comparison of novel techniques for instantaneous speaker adaptation,” *Proc. EUROSPEECH 1997*, pp. 2047-2050.

Hetherington L. and McCandless M. (1996) : “SAPPHIRE: An extensible speech analysis and recognition tool based on Tcl/Tk,” *Proc ICSLP ’96*, pp. 1942-1945.

Hetherington L., Phillips M., Glass J., and Zue V. (1993) : “A* word network search for continuous speech recognition,” *ICASSP '93*, pp. 1533-1536.

Hopcroft J. and Ullman J. (1979) : Introduction to automata theory, languages and computation, Addison and Wesley, ISBN 0-201-02988X.

Huang X. D. and Lee K. F. (1991) : "On speaker-independent, speaker-dependent and speaker-adaptive speech recognition," *Proc. ICASSP 1991*, pp. 877-880.

Kershaw D. J., Hochberg M. M., and Robinson A. J. (1996) : "Context-dependent classes in a hybrid recurrent network-HMM speech recognition system," *In Advances in Neural Information Processing Systems*, Vol. **8**, eds: Touretsky D. S., Mozer M. C, and Hasselmo M. E., Morgan Kaufmann.

Ladefoged P., and Broadbent D. E. (1957) : "Information conveyed by vowels," *JASA* **29**(1), pp. 99-104.

Le Cun Y., Denker J. S., and Solla S. A. (1990) : "Optimal brain damage," *In Advances in Neural Information Processing Systems* Vol. **II**, ed: Touretsky D. S., pp. 589-605, San Mateo, California IEEE, Morgan Kaufmann.

Leggetter C. J. and Woodland P. C. (1994) : "Speaker adaptation of continuous density HMMs using multivariate linear regression," *Proc. ICSLP 1994*, pp. 451-454.

Levin E. (1990): "Word recognition using hidden control neural architecture", *Proc. ICASSP '90*, Vol. **1**, pp. 433-436.

Li K. P., Naylor J. A., and Rossen M. L. (1992) : "A whole word recurrent neural network for keyword spotting," *Proc. ICASSP ´92*, Vol. **2**, pp. 81-84.

Mitchel C. D., Harper M. P., and Jamieson L. H. (1996) : "Stochastic observation hidden Markov models," *Proc. ICASSP '96*, pp. 617-620.

Necioglu B. F., Ostendorf M., and Rohlicek J. R. (1992) : "A Bayesian approach to speaker adaptation for the stochastic segment model," *Proc. ICASSP 1992*, pp. I-437 - I-440.

Ney H. and Aubert X. (1994) : "A word graph algorithm for large vocabulary, continuous speech recognition," *Proc. ICSLP '94*, pp. 1355-1358.

Peckham J. (1993) : "A new generation of spoken dialog systems: results and lessons from the SUNDIAL Project," *Proc. Eurospeech '93*; pp. 33-40.

Richard M. D. and Lippman R. P. (1991) : "Neural network classifiers estimate Bayesian *a posteriori* probabilities," *Neural Computation*, Vol. **3**, pp. 461-483.

Robinson A. J. (1994) : "An application of recurrent nets to phone probability estimation," *IEEE trans. on Neural Networks* Vol. **5**(2), pp. 298-305.

Robinson T. and Fallside F. (1991) : "A recurrent error propagation network speech recognition system," *Computer Speech & Language* 5:3, pp. 259-274.

Shiel F. (1993) : “A new approach to speaker adaptation by modelling pronounciation in automatic speech recognition,” *Speech Communication* Vol. **13**, pp. 281-286.

Sietsma J. and Dow R. J. F. (1991) : “Creating artificial neural networks that generalize,” *Neural Networks*, **4**(1) pp. 67-69.

Sjölander K. and Gustafson J. (1997) : “An integrated system for teaching spoken dialogue systems technology,” *Proc. EUROSPEECH ‘97*, pp. 1927 - 1930.

Soong and Huang (1991) : “A tree-trellis based fast search for finding the N best sentence hypotheses in continuous speech recognition,” *Proc. ICASSP ‘91*, pp. 713-716.

Strange W. (1989): “Evolving theories of vowel perception,” *JASA* **85**(5), pp. 2081-2087.

Ström N. (1992): “Development of a recurrent time-delay neural net speech recognition system,” *STL-QPSR* 2-3/1992, pp. 1-44, KTH, Dept. of Speech Communication and Music Acoustics, Sweden.

Ström N. (1994a) : “Optimising the lexical representation to improve A* lexical search”, *STL-QPSR 2-3/1994*, pp. 113-124.

Ström N. (1994b): “Experiments with a new algorithm for fast speaker adaptation,” *Proc. ICSLP 1994*, pp. 459-462.

Ström N. (1995): Generation and minimisation of word graphs in continuous speech recognition,” *Proc. Workshop on Automatic Speech Recognition*, pp. 125-126, Snowbird, Utah.

Ström N. (1997): Nikko Ström (1997): “A tonotopic artificial neural network architechture for phoneme probability estimatio,” *To appear in Proc. of the 1997 IEEE Workshop on Speech Recognition and Understanding*, Santa Barbara, CA.

Sutton S., de Veilliers J., Schalkwyk J., Fanty M., Novick D. and Cole R. (1996): “Technical specification of the CSLU toolkit;” *Tech. Report No. CSLU-013096*, CSLU, Dept. of Computer Science and Engineering, Oregon Graduate Institute of Science and Technology, Portland. OR.

Tebelskis J. and Waibel A. (1990): “Large vocabulary recognition using linked predictive neural networks,” *Proc. ICASSP '90*, Vol. **1**, pp. 437-440.

Verbrugge R. R. and Strange W. (1976): “What information enables a listener to map a talker’s vowel space,” *JASA* **60**(1), pp. 198-212.

Viterbi A.J. (1967): “Error bounds for convolutional codes and an asymptotically optimum decoding algorithm,” *IEEE Trans. Information Theory*, Vol. IT-13, pp. 260-269.

Waibel A., Hanazawa T., Hinton G., Shikano K. and Lang K. (1987) : “Phoneme recognition using time-delay neural networks,” *ATR Technical Report TR-006*, ATR, Japan.