Switching linear dynamical models for automatic speech recognition

Fiona Couper BSc

Supervisor Simon King PhD

MSc Speech and Language Processing
University of Edinburgh
September 2002

Abstract

The field of speech recognition research has been dominated by the Hidden Markov Model (HMM) in recent years. The HMM has known weaknesses, such as the strong "independence assumption" which presumes observations to be uncorrelated. New types of statistical modelling are now being investigated to overcome the weaknesses of HMMs. One such model is the Linear Dynamical Model (LDM), whose properties are more appropriate to speech. Modelling phone segments with LDMs gives fairly good classification and recognition scores, and this report explores possible extensions to a system using such models. Training only one model per phone cannot fully model variation that exists in speech, and perhaps training more than one model for some segments will improve accuracy scores. This is investigated here, and four methods for building two models instead of one for any phone are presented. Three of the methods produce significantly increased classification accuracy scores, compared to a set of single models.

Declaration

Acknowledgements

I would like to thank my supervisor, Simon King, for his constant enthusiasm, guidance and genius that continually inspire me; for introducing me to speech recognition so clearly and positively; and for always being available.

I am also grateful to those at CSTR, where the research was carried out, for looking after me; particularly Joe Frankel for his immense patience in explaining so much to me; and Korin Richmond and Rob Clark for their emacs, latex and matlab expertise.

As always, my parents have supported me in every way this year - thank you. I couldn't ask for more. Finally, I would like to thank my fiance, Laurence Kenney, for his support and consistency, and for choosing us.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

The task in speech recognition is to form relationships between the continuous acoustic speech signal and the discrete symbolic word sequence such that translation between the two is possible. Achieving this mapping is complex, requiring knowledge from many different fields of research, from the phonetic level of speech to higher level language and linguistics study. The work contained in this thesis is concerned only with phonetic modelling.

The Hidden Markov Model (HMM) was applied to speech recognition in the early 1970s, and was the model that made automatic speech recognition (ASR) possible. By the end of the twentieth century HMMs dominated research into ASR. They are considered to be robust and effective for modelling. However, for reasons discussed later, HMMs may not be the best model to use for speech.

Recently, researchers have investigated alternative models which are more appropriate to speech. Segment modelling is a statistical modelling technique that is being applied to speech. Results from systems built using such models are promising.

This dissertation is concerned with Linear Dynamical Models, a type of segment model. It is believed that Linear Dynamical Models (LDMs) will exceed the capability of the familiar HMM.

While having some conceptually similar characteristics to HMMs, LDMs differ in a number of key ways, which will be described in this thesis.

The system this work is based on is a working recognition system consisting of one LDM per phone segment. The purpose of this research is to determine whether or not an extended model set improves the classification accuracy of the system.

The structure of the dissertation is as follows. The remainder of this chapter briefly summarises some of the main considerations in modelling speech, including an outline of HMMs, before introducing the subject of my investigation, Linear Dynamical Models. The weaknesses of LDMs and some proposed solutions are discussed in Chapter 2. The concept of switching models in a recognition system is described, with details of pilot experiments that explore the validity of switching as a possible solution. Chapters 3, 4 and 5 consist of the main experiments. Chapter 3 details the experimental procedure, while chapters 4 and 5 detail results of individual experiments. Finally, conclusions and suggestions for further research are presented in chapter 6.

1.2 Modelling speech

1.2.1 Speech parameterisation

There are a number of ways that frames of speech can be represented as a series of coefficients for processing. In this work, I have chosen to use data parameterised with Mel Frequency Cepstral Coefficients (MFCCs). The process for computing MFCCs is described briefly here. The waveform is first divided into overlapping frames, typically of length 25ms, spaced every 10ms. Fourier analysis is then used to convert from a time domain to a frequency domain. Human frequency perception is not linear, and there are frequency scales to reflect this. MFCCs use the Mel scale, a non-linear frequency scale based on perceptual data, and filter the frequency coefficients from the Fourier transform accordingly, with wider and more widely spaced filters for higher frequencies. Since the output of adjacent filters are highly correlated, a cosine transform of the outputs is the final step in the process. The cosine coefficients are less correlated with one another, and are often assumed to be independent. The first 12 contain enough discriminative information for speech, and usually energy is appended for each frame giving 13 coefficients in total.

1.2.2 Parameter Estimation

In training a recognition system using a maximum likelihood criterion, the goal is to maximise the likelihood of the observed (training) data given the model. An iterative process is used to achieve this, using the ubiquitous **Expectation Maximisation** (EM) algorithm. EM has two steps: the E-step and the M-step. In the E-step the expectation of the data given the current model parameters is found. In the M-step this value is maximised and the model parameters updated. The algorithm is guaranteed to converge to a local maximum. Variations of EM are used for training parsers and context free grammars. For the purposes of this dissertation, training is done using a version of EM developed specifically for linear dynamic models. There is no derivation here, since training is not altered in any way in my experiments. For further details about EM for segment models, the reader is referred to [13, p.17-8] and [2, ch. 5].

1.3 Classification and Recognition

There are a number of tasks that must be carried out in speech recognition. A trained system has models for segments of speech - maybe phones, biphones, triphones. When the system is presented with new data, its task is to assign labels to each part of the data. The recogniser needs, then, to segment the data in order to find which segment model is best for each part. Since segment duration affects the decision about which is the most likely model, the two tasks of segmentation and the assignment of segment labels must be done simultaneously. In a complete search this means that the data is segmented in every possible way, and each of these tested by every segment model. Obviously this is computationally infeasible, so in practice the search space is pruned so that only a proportion of the more likely segmentations are considered.

In classification tasks, the system is given segmented but unlabelled data, so the task is to identify the most likely segment model to have generated each segment. The investigation in this dissertation uses only classification, not full recognition: segment boundaries are always known. **Maximum a Posteriori** (MAP) estimation is used for classification. The function to maximise in classifying each segment is the sequence of labels x (words or phones, for example) for a given sequence of observations Z. **Bayes' formula** (equation 1.1) is used to maximise the posterior to get the MAP label sequence (equation 1.2).

$$p(x|Z) = \frac{p(Z|x)p(x)}{p(Z)} \tag{1.1}$$

$$\hat{x}(Z) = \arg\max_x p(x|Z) = \arg\max_x [p(Z|x)p(x)] \tag{1.2}$$

So finding the MAP estimate of the best series of labels $\hat{x}$ for an observation sequence Z requires the knowledge of $p(Z|x)$, the acoustic model, and $p(x)$, the language model (LM).

Computing $p(x)$ makes use of information above the segment level, to build words from the sub-word units (segments) and then join the words to make sentences. These decisions require prior probabilities of the concatenation of segments and words, and these probabilities are learnt from data. Language models give the probability of a word sequence, estimated from the training data. A full language model, given infinite amounts of data, would have, for each word sequence W,

$$P(W) = \prod_{i=1}^{N} P(W_i|W_1, W_2, \dots W_{i-1}, W_{i+1}, \dots W_N) \tag{1.3}$$

In practical systems, there is never enough data, since there are infinite ways of arranging words and only finite amounts of data, and so $P(W)$ has to be approximated. Also, in order that the language model can be used during recognition, where partially complete sentences exist, only the left context of a word is used. So the estimate is

$$P(W) \approx \prod_{i=1}^{n} P(W_i|W_{i-(n-1)}, \dots W_{i-2}, W_{i-1}) \tag{1.4}$$

This is an *n-gram* language model.

The ranked MAP likelihoods for each segment and the language model form a search space over which a Viterbi search is carried out. In such a search, the most likely sequence of segments is found, given the probabilities of the individual tokens(the MAP likelihoods), and the probability of the sequence of labels, x.

1.4 Hidden Markov Models

Hidden Markov Models (HMMs) are generative models used for many tasks in Artificial Intelligence and pattern recognition. In speech, the speech process is represented by a series of hidden

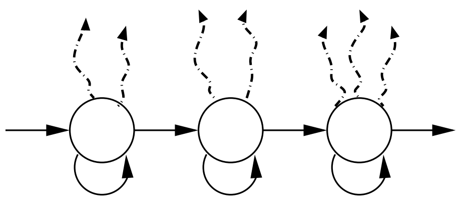

Figure 1.1: Hidden Markov Model

discrete states. The speech observations are outputs of these states, according to the output probability distributions of the states. A three state HMM is illustrated in figure 1.1. The states exhibit the Markov property, which states that probability of the current state depends only on the previous state, or “the past is independent of the future given the present”. The components of an HMM are

- States
- Transition probabilities, expressing the likelihood of transitioning between every pair of (connected) states
- Observation likelihoods, the probability of an observation being generated by a state

The parameters for the observation likelihoods are Gaussian, or Gaussian mixtures, in each state.

A general introduction to HMMs is found in [8, chapter 7]. A more in depth look at their properties is found in [14].

1.5 Linear Dynamical Models

1.5.1 Motivation

Modelling speech using HMMs involves the strong assumption that one frame of speech is independent of all others given the model state. Co-articulation and other well studied phenomena show that this view of speech is clearly not true, and modelling in this way ignores some potentially useful information. HMMs are generative models, generating conditionally independent frames of speech. Since adjacent frames of speech are not independent, a model generating *sequences* of frames would reflect speech more closely. Linear dynamic models (LDMs) are generative models, generating entire segments. The use of the word segment in this dissertation refers to a generic unit of speech rather than a linguistic segment.

Two key results reported by Digalakis [2] also motivate linear dynamic models. Firstly, he states that there is a strong correlation between the frames within a segment of speech. This result motivates entire segment modelling, which is achieved by LDMs. Secondly, *within* a segment there is no evidence that anything but linear dependencies exist. This motivates models which contain relationships which are no more complex than linear. Therefore it seems plausible that modelling speech using linear entire segment models has a lot of potential.

HMMs have a limited capability for modelling duration: the difference between a short and a long token of the same speech unit is the number of times the 'handle is turned' on the particular HMM. Duration is a useful source of information in the perceptual classification of speech units, it is a cue in human perception, and so a model capturing this information would be expected to outperform one that cannot model such phenomena.

Another limitation of HMMs, cited in [12] is that "restrictions on feature extraction imposed by frame-based observations", which again motivates entire segment modelling.

1.5.2 Introducing the models

I begin by giving the formal definition of the models, then describe some of their characteristics. The models exist on two planes, one describing the hidden activity, and one describing the observations. The hidden activity is in the state space, with state vector x_t, the position in the state space at time t. The vector y_t is the observation at time t. The full definition of the model is as follows:

$$y_t = Hx_t + v_t \text{ with } v_t \sim N(\mu_v, C) \tag{1.5}$$

$$x_t = Fx_{t-1} + w_t \text{ with } w_t \sim N(\mu_w, D) \tag{1.6}$$

Vectors v_t and w_t are random variables which model error. Two types of noise are modelled in this way. It is important to model error in some way since it will always exist in practical

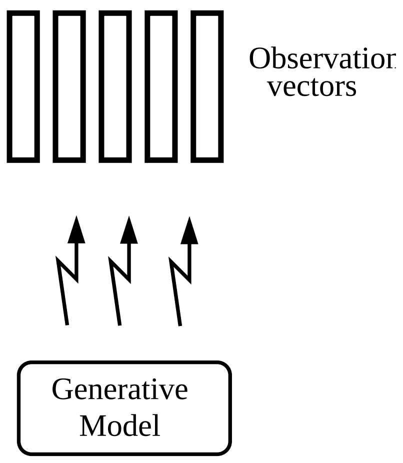

Figure 1.2: An LDM is a generative model

systems. w_t is the system noise, while v_t is the noise inherent in the observations. Both terms are normally distributed. For one segment model, the model parameters H, F, μ_v, C, μ_w, D are constant.

The models are **linear** since x_t is linearly related to x_{t-1}, and observation vector y_t is linearly related to x_t. The evolution of the state space in equation 1.6 is such that the position of a vector (or point in the space) is dependent only on the position of the vector at the previous time step. So the process is again Markov (see section 1.4). The models are **dynamic** since they create moving trajectories in both the state and observation spaces.

The simple diagram in figure 1.2 illustrates a generative model. The box contains the model parameters, and the processing of the equations can be thought to occur within it. The vectors generated are the observations. An HMM has this form, but generates all the observation vectors independently. An LDM also can be thought of as a model in a box, but the generated observation vectors are not independent. The state space evolves by equation 1.6, which produces a *continuous* and smooth trajectory. This is then mapped onto the observation space by equation 1.5 at each time step, also producing smooth and continuous trajectories. Figure 1.3 illustrates this concept.

The two-dimensional toy examples that follow illustrate the way the model makes these trajectories. Certain characteristics of the model for different values of F are also introduced by

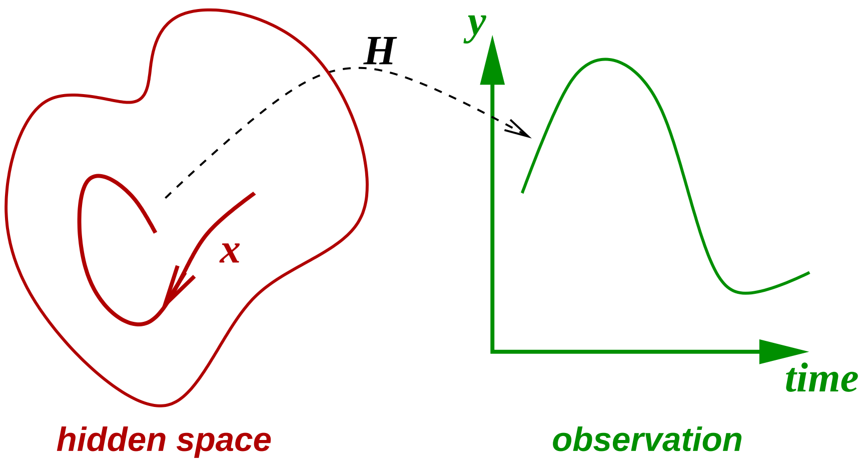

Figure 1.3: LDMs generate smooth trajectories (picture reproduced with permission, from [4])

these examples. We will see that the model is stable and tends to a target value is $|F| < 1$, whereas if $|F| > 1$, there is exponential growth, which is unstable. This is proved below (see equation 1.9 and text)

Letting $F = \begin{pmatrix} 0.1 & 0.1 \\ 0.5 & 0.3 \end{pmatrix}$ with $H = \begin{pmatrix} 1.1 & 2 \\ 0.3 & 0.1 \end{pmatrix}$, $\mu_v = \begin{pmatrix} 0.5 \\ 0.1 \end{pmatrix}$, $\mu_w = \begin{pmatrix} 0.2 \\ 1.0 \end{pmatrix}$, and $x_0 = \begin{pmatrix} 0.2 \\ 0.1 \end{pmatrix}$, the trajectories the model makes in the state and observation space are plotted for 30 frames in figure 1.4

Using the same μ and H parameters but changing one of the values in F such that $F = \begin{pmatrix} -1.5 & 0.1 \\ 0.5 & 0.3 \end{pmatrix}$, $|F|$ is now greater than 1, so there is exponential growth, as seen in figure 1.5

So, we can now appreciate the relationship between x_t and y_t, and see some of the differences between HMMs and LDMs. Relating the model more closely to speech, the trajectories described in the state space could be thought of as the movement of the articulators: a continuous series of events, the position at each time being dependent on only the position at the previous time. This continuous movement has as its outcome speech, dependent only on the position of the articulators. As in any system, there are also noise terms to model any drift, necessary since there will always be error, so the model will never *exactly* fit the data.

Since the state vector x is in a different space to the observation y, it can be of different dimensionality. This means that the model can be flexible, since the dimension of the parameters are not dependent on the format of the data. The dimensionality becomes another parameter to be optimised in a full system. However, this factor is not examined in any way in this work. In the experiments detailed in chapters 4 and 5, the state is 6-dimensional while the observations have 13 dimensions.

If we ensure that $|F| < 1$, it can be shown that the state trajectory is heading for a target, as illustrated above. Beginning with the state equation (equation 1.6):

$$x_t = Fx_{t-1} + w_t$$

It is easy to see that if $|F| < 1$, $\lim_{t\to\infty} x_t = x_{t-1}$ and $\lim_{t\to\infty} w_t = \mu_w$

So

$$\begin{aligned} x_t &= Fx_{t-1} + w_t \\ \lim_{t\to\infty} x &= Fx + \mu_w \qquad (1.7) \\ x(I - F) &= \mu_w \qquad (1.8) \\ x_\infty &= (I - F)^{-1}\mu_w \qquad (1.9) \end{aligned}$$

The target, then is dependent on the matrix F and the vector μ_w. This will be useful for experimentation detailed in chapter 5.

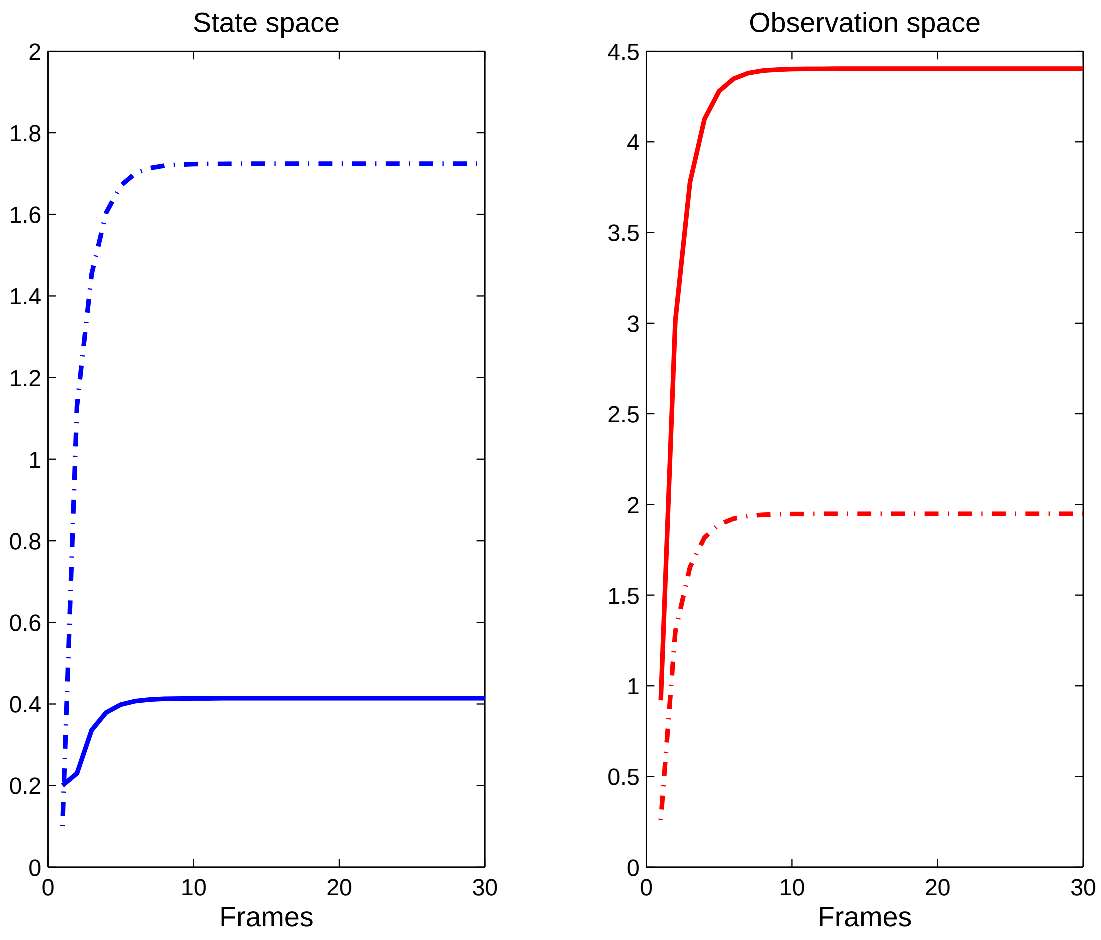

Figure 1.4: Trajectories for 2-dimensional LDM with $|F| < 1$ (see text). Solid lines are 1st dimension, dashed 2nd.

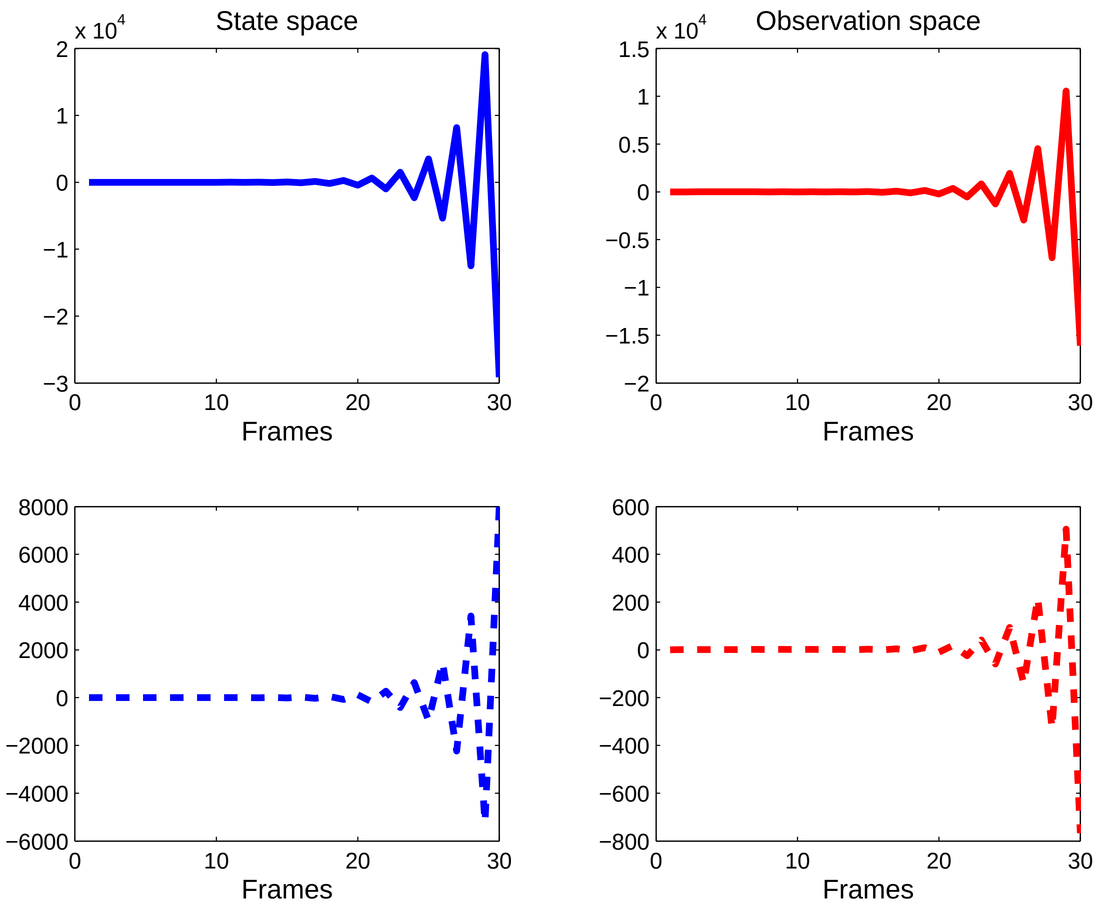

Figure 1.5: Trajectories for 2-dimensional LDM with $|F| > 1$ (see text). Solid lines are 1st dimension, dashed 2nd.

Chapter 2

Switching LDMs

Linear dynamic models were introduced in section 1.5 as statistical models with properties that make them appropriate for modelling speech. The current system at CSTR and its known weaknesses will now be described, introducing the extensions my work provides.

LDMs are versatile and can model units of any length, without any limitations on duration distributions [2]. However, a simple approach has necessarily been adopted in the current system: all models model phoneme segments and there is only one LDM per phoneme. This is illustrated in figure 2.1.

2.1 Weaknesses

Since statistical learning-from-data methods outperform hand crafted rules in general, it is believed that a system where the unit inventory is automatically determined from data would be a more effective recogniser than one with predetermined units and unit lengths. The current system has both these characteristics predetermined.

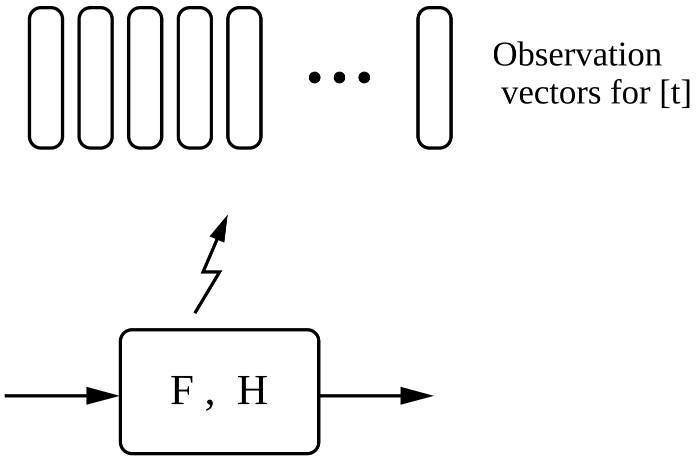

Figure 2.1: One segment model generating the phone [t]

2.1.1 Unit length

Predefining the unit length means that the system cannot model variation in smaller sized segments, in this case sub-phonetic units.

2.1.2 Multi-modality

There are multiple realisations of any one phoneme in speech. One segment of speech can be articulated in different ways, and thus have different properties, but the outcome is the same phone. These different ways of producing segments can be described as modes or regimes, and the task in speech recognition is to identify the mode and map it onto the correct output segment. Ostendorf [10] challenges the modelling assumption that a word is a sequence of phoneme segments, “beads-on-a-string”, concatenated. Adjacent segments in fact influence each other, with effects such as coarticulation. Since the current system, as illustrated in figure 2.1, has only one model for each segment, the multi-modality observed in speech cannot be modelled.

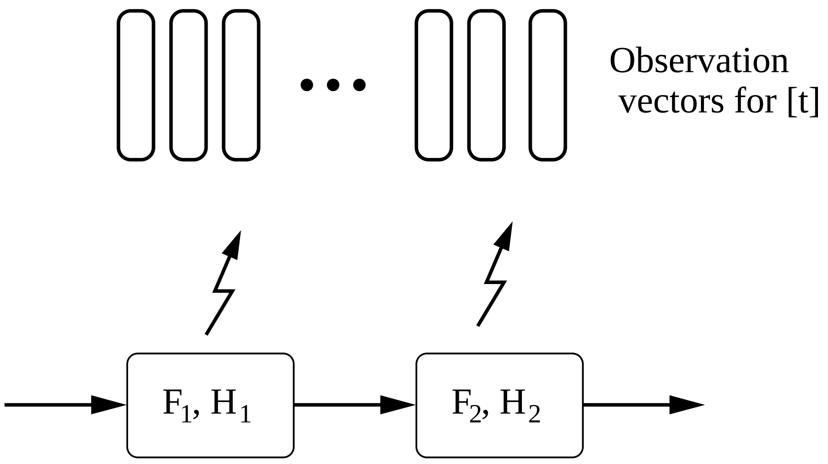

Figure 2.2: One segment model generating the phone [t]

2.2 Switching as a possible solution

Solving the problem of determining the optimal unit inventory is a large task, too large for a work of this size. Approaching it, though, is possible. One way of moving from the unimodal models for fixed units to a multi-modal variable unit length system is to research the multi-modality, keeping the unit length and designation fixed.

If the system were to be viewed as a finite state network, the states being individual LDMs, the topology of the current system would be a linear, left-to-right sequence of phone models.

There are two obvious ways of extending the current system to include models of different behaviours. Both involve the concept of switching. Switching effectively means choosing, so instead of having one possible set of parameters for any one segment, there is a choice, reflecting our understanding of the speech process. The choice of when to switch would be automatic.

2.2.1 Temporal switching

The first use of switching is to switch the parameters of a segment at some point within it: a temporal switch, as depicted in figure 2.2. In this structure, the form of the models remain the same, and the internal parameters are allowed to change or switch. This clearly gives more

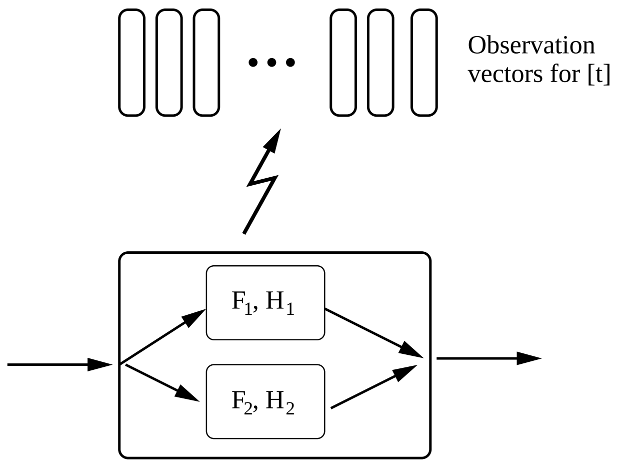

Figure 2.3: One segment model generating the phone [t]

permutations of sequences available for recognition. A pilot experiment was carried out by Joe Frankel (private communication), in which a different segment types have different numbers of regimes, and the time spent in each regime of the model is deterministic. Fricatives, stop closures and silence were constructed with one regime; affricates, nasals, semi-vowels and glides were constructed with two; and vowels and stops with three. The time spent in each regime is equal for all classes except vowels and stops. For vowels, the ratio of time spent in each regime was 2:3:2, while for stops more time was spent in the last regime. The outcome of this simple, deterministic experiment was an improvement in phone classification accuracy of 2.4%.

2.2.2 Multi-modal switching

Similarly, for a particular segment, more than one model could be trained according to a chosen property of the data. In general terms, creating these multiple models per segment causes a choice to exist between different targets. This is described further in chapter 5, and can be seen in equation 1.9. A simple switch is illustrated in figure 2.3.

The multiple parameter sets for a unit need to be allocated some subset of the tokens of the unit they are modelling, for training. This would also be an automatic, learnt from data, decision. But clearly there must be some criteria for splitting the tokens between different models, since to test every possible permutation of designations would be computationally infeasible.

2.3 Pilot Experiments

Pilot experiments were initially carried out on a small data set with small, incremental steps to establish whether or not the theoretical ideas proposed seem to translate into positive results. The purpose of the pilot experiments here is to begin to validate the hypothesis that switching LDMs will improve classification scores.

2.3.1 Mocha Corpus

The dataset chosen for the pilot experiments is a small articulatory / acoustic corpus collected at Queen Margaret University College, Edinburgh for the purpose of training automatic speech recognisers. Each speaker says 460 sentences, which include the main speech processes found in English. From this corpus, I used a single female speaker (fsew0). The data is labelled using a phone set of 39 phones.

2.3.2 Experiment

In order to begin a switching process in the recognition system, more than one model must exist to choose between for a segment. These models need data to be trained on, so the full set of training data needs to be divided up in some way. This way of making two models will be referred to as “splitting the data”.

Since classification and recognition are based on log likelihoods of observations, splitting the training tokens based on their log likelihood values should create two models that better represent the data.

Initially, a set of 39 single models, one model per phone, were trained. Each of these models was then presented with all the tokens of their phone from the training data, and the log likelihoods of the model generating each token was collected.

These figures were used to plot a histogram of log likelihoods. These were examined to determine how successful a model is at representing the dataset. A good model will have a large peak at the high log likelihood end of the graph and tail off quickly, as this would mean that the majority of the tokens in the data set are modelled with a high likelihood. The histogram

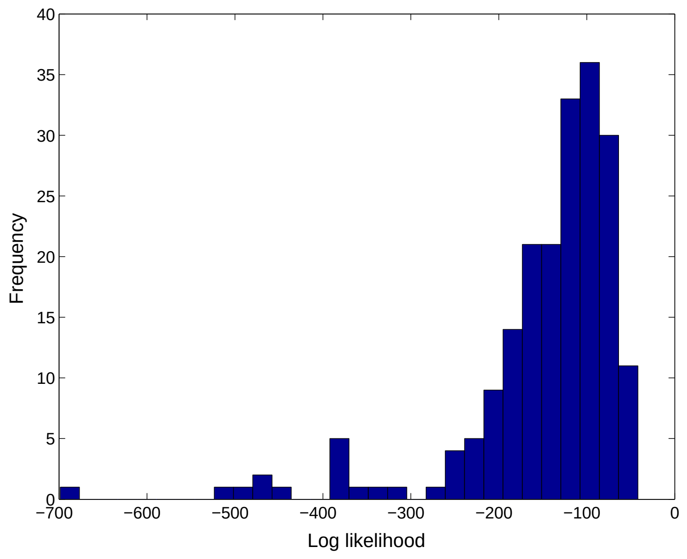

Figure 2.4: Histogram of likelihoods of training data tokens of ai

in figure 2.4 of the log likelihoods of the phone *ai* illustrates such a distribution. The height vertical bars represent the number of tokens in each log likelihood range. For graphs with more than one peak or a more uniform distribution, it is believed that the model is sub-optimal since it models only some of the tokens well.

Histograms of the stops [k, p, t] and the fricative [sh] were not of the desired shape, and the models for these phones were split. The median log likelihood was chosen to be the threshold for splitting: for each phone, tokens were allocated to one of two new models according to which side of this threshold their likelihood (when generated by the single model) was. The splitting criterion is reached for each phone *ph* as:

- Take the trained single model of *ph*.
- Find the log likelihood of this model generating each token of *ph* in data set 1.
- Find the median log likelihood.
- Label tokens whose log likelihood is greater than the median as $ph0$ and the remaining half $ph1$.

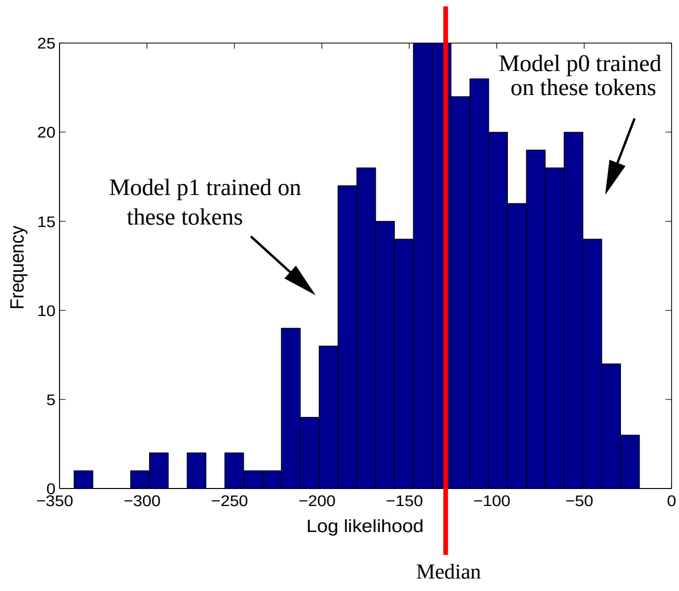

Figure 2.5: Illustration of splitting model for phone p by the median log likelihood of -122. Histogram is of the likelihoods of the training data tokens of p

This is simply illustrated for the data for model p in figure 2.5.

These models were then built using 2 iterations of training, each having the same number of training tokens for training. The split models were again presented with all the tokens of that phone in the training data, the model generating each token with the higher likelihood recorded with the likelihood score. The means of this data are seen in table 2.1.

model	no. of tokens	median	mean before	mean after
k	439	-149	-149	-137
p	307	-122	-124	-111
t	710	-135	-139	-127
sh	120	-154	-162	-151

Table 2.1: Mean log likelihoods of tokens in training data before and after pilot split.

These results indicate a clear improvement on log likelihood scores of the model against the tokens it is expected to classify.

The same log likelihood data were collected using the tokens in the test set, yielding better results for each of the split models, as reported in table 2.2.

model	single model mean	mean after split
k	-160	-146
p	-137	-125
t	-141	-129
sh	-147	-144

Table 2.2: Mean log likelihoods of test data tokens before and after split.

It is interesting to note at this point the distribution of tokens between the two split models. This method of splitting the models using likelihood scores means that one model is trained on the data it was very good at modelling before, and the other on the rest. Does this mean that the first model generalises well or not? The results in table 2.3 indicate that on average a third of the tokens achieve a higher log likelihood using the first model, and the second model better models the rest.

model	training data	test data
k	24%	15%
p	31%	26%
t	34%	28%
sh	39%	38%

Table 2.3: Percent of tokens in training and test data better whose log likelihood scores are higher with model 0

These results motivated investigation into token redistribution between the two models according to which was modelling each token better. The criterion for 'modelling better' was as before, the higher log likelihood score.

Two iterations of training were carried out to form a set of single models (one per phone), and two further iterations with the phones [k, p, t, sh] split into two models. The models were then used to 'test' all the tokens of their phone in the training data, and reallocate tokens to whichever model better represented them. A further iteration of training was then carried out, using the already trained set as initial models, before reallocating again.

Table 2.4 shows the proportion of tokens used to train the first of the two models for each phone at each iteration.

These figures indicate a consistent trend exists between models split this way: there is a con-

model	4	5	6	7	8
k	50	24	21	19	18
p	49	31	29	30	33
t	50	34	29	27	24
sh	53	39	35	33	33

Table 2.4: Percent token allocation to model 0 in training

vergence at around one third of the tokens. However, as can be seen in tables 2.5 and 2.6 , the models may be over-fitting the data, as the likelihood scores do not converge in the same way.

model	4	5	6	7
k	-140	-139	-143	-147
p	-113	-112	-114	-117
t	-128	-128	-130	-133
sh	-154	-154	-161	-171

Table 2.5: Mean log likelihoods for training data tokens at different iterations of training

model	4	5	6	7	8
k	-146	-149	-153	-159	-164
p	-125	-127	-131	-135	-139
t	-129	-130	-133	-135	-139
sh	-144	-148	-156	-166	-177

Table 2.6: Mean log likelihood scores for test data tokens at different iterations of training

Raw classification scores (not using a language model) are reported in table 2.7. The models at iteration 2 are all single, and have been trained for two iterations. The five further columns of data are the individual percent correct scores for each of the split phones at the different iterations of training. The overall percent correct scores for each iteration are also reported. All classification is on (unseen) test data.

The graphs in figure 2.6 and the data in the corresponding tables show a downward trend in mean log likelihood figures for all four phones. The percent correct results indicate three possible hypotheses:

1. Redistribution of the tokens between the models at every iteration of training does not improve classification.

model	it2	it4	it5	it6	it7	it8
k	56.4	72.9	68.8	70.8	68.8	62.5
p	59.5	47.6	45.2	38.1	35.7	38.1
t	41.8	48.0	45.9	48.0	49.0	49.0
sh	76.9	61.5	61.5	69.2	69.2	69.2
overall	56.4	56.4	54.4	52.0	50.7	49.0

Table 2.7: Percent correct classification results for models at different iterations of training (see text)

2. Different models require different numbers of iterations of training.
3. Mean log likelihood scores do not correlate with percent correct results for individual models: there is a more complex relationship.

Testing these three hypotheses satisfactorily would require more time than a project of this size allows. The focus of this work is more general, exploring whether or not systems with switching LDMs can be more effective than systems with one model per phone. As such, investigation into the finer parameters such as the number of iterations of training to use for each model cannot be included. However, we will see some confirmation of the first hypothesis through the experiment in section 5.1.

These pilot tests indicated improvement is possible in classification when more than one model exists for some tokens. The chapters 4 and 5 extend these experiments to a larger dataset and more extensive investigation.

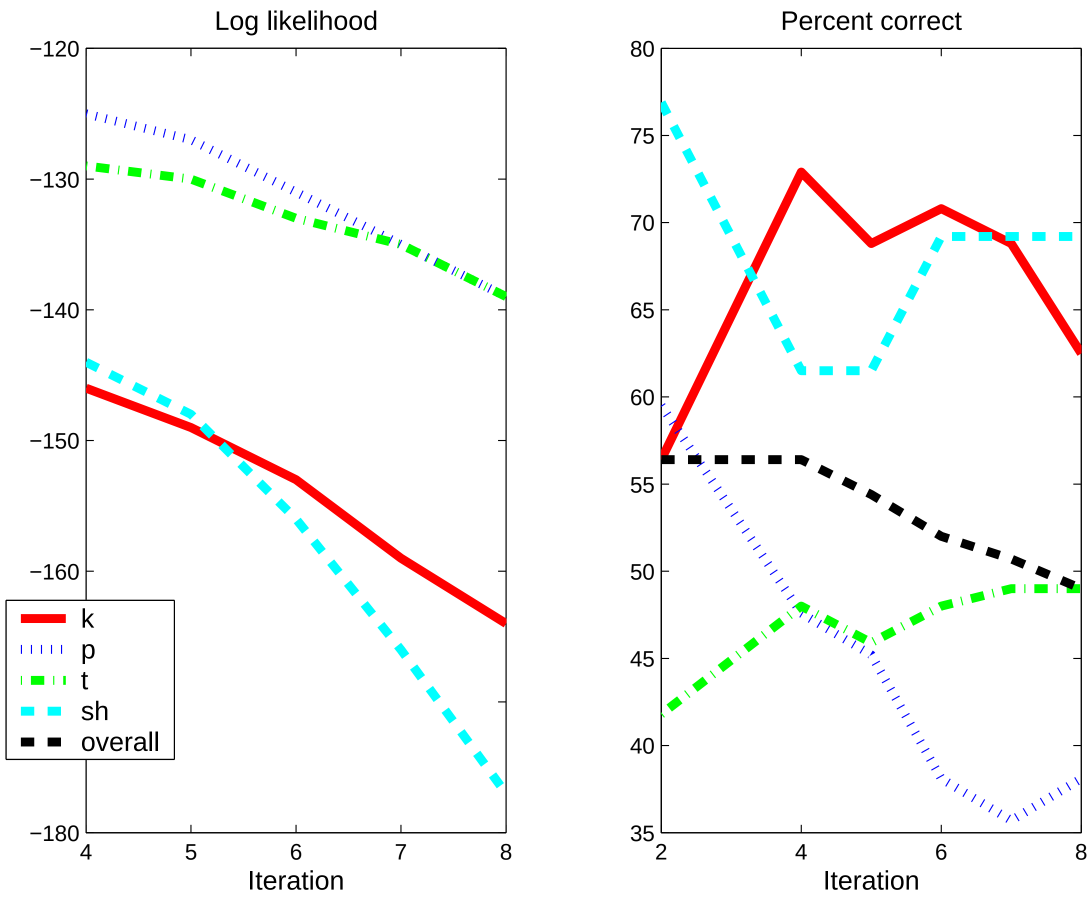

Figure 2.6: Graph of the log likelihood and percent correct scores as reported in tables 2.6 and 2.7

Chapter 3

Experimental Method

The pilot experiments in section 2.3 indicate the potential for improving speech recognition by switching phone models. Further investigation was carried out in the same direction, splitting the data to make more than one model, and automatically switching between the models. Instead of the small, single speaker data from the Mocha corpus, the entire TIMIT dataset was used.

3.1 TIMIT Corpus

TIMIT [7] is a multi speaker corpus, containing 6300 sentences: 10 sentences spoken by each of 630 speakers. The corpus is divided into 8 dialect regions, according to the geographical area in which the speakers grew up. The data is divided into training and test sentences; the division is standard so that test results are comparable between different systems.

Every speaker says the same two "shibboleth" sentences, then 8 more chosen from a wide selection. These shibboleth sentences skew the data, since their sequences of segments and words are repeated in the corpus 630 times. To avoid the effect of this skewing, the standard practice of ignoring these shibboleth sentences was adopted.

In order to set parameters in the experiments, detailed below, we need to hold out some data for validation. The TIMIT designated training data was split into two sets, for simplicity called set 1 and set 2. The test data remains as designated and will be referred to as set 3. These nominations are for clarity, since the words training, validation and test are processes as well as data sets.

Data set	No. of sentences	TIMIT designation
1	3168	Training
2	480	
3	1344	Test
TOTAL	4992	

Table 3.1: Division of the TIMIT corpus as used in this research

The test data, set 3 is completely unseen during training, so that final results are a true test of the system's capabilities. The validation sentences were chosen in such a way that the distribution of dialects reflects that of the test data. Table 3.1 shows the number of sentences in each data set.

3.2 General method

The various experiments will be discussed in detail in chapters 4 and 5. The procedure for all experiments is laid out in this section.

Since our goal is to improve classification scores from the current system where all segments have a single model, this should be our baseline. Initially we have a set of single models trained for 4 iterations[1] on the data set 1. These trained models provide the baseline. The scores from this baseline system are used for comparison later in the process.

A method by which to split the models is then chosen, either by a division of the data or by a permutation of some of the models' parameters. In training, the system builds models based on the names of the tokens. Dividing the data and distinguishing the tokens of a particular phone by different names, allows more than one model to be trained for a segment. The tokens are grouped by some characteristic, such as their durations; this is explained further in chapter 4. Permuting the model parameters is slightly different. Using some matrix techniques, some of

[1]Earlier work at CSTR showed that 3-4 iterations of training was sufficient. See [6]

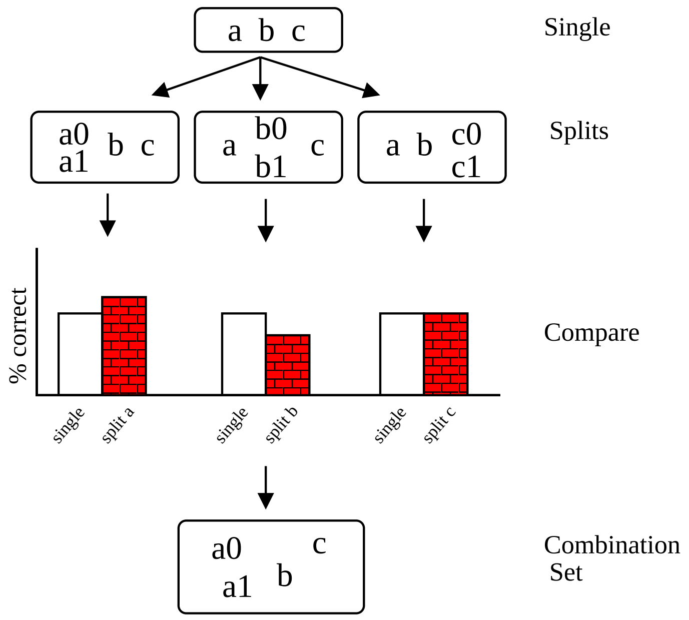

Figure 3.1: Simple example of the method: Choosing a full set of single or double models by independent decisions for each segment.

the parameters of the LDM can be changed so that there is more than one model. Then training can be carried out by finding out the likelihood of each the new models generating each token in the data, and assigning the most likely model that token. In this way the data is divided by the models. The new models can then be trained on these subsets of the data.

The single models are split one at a time by the chosen criterion. The model of one phone is split, and the other 60 remain single. This new model set of the split model and remaining single models is then used to classify the data in data set 2, and resulting scores are compared to those for a set of single models. Any improvement is attributed to the split model, since it is the only factor that is different to the control. Those models which improve classification after being split are put in a new "combination" set together along with single models for the other segments for further training. This is illustrated simply in figure 3.1.

The combination set of all the 'winning' models is then trained on data sets 1 and 2, and final classification scores are calculated using data set 3, the test data.

Pseudo code to explain the procedure is presented in figure 3.2 below.

```
• Train single models (data set 1)
• For each phone in phone set,
    – Split model by some criterion
    – Train split model (data set 1)
    – Classify using split model plus remaining single models (data set 2)
    – If (classification score for new model set) > (classification score for
      set of single models)
        ∗ Then set current model = split
        ∗ Else set current model = single
• Make combination model set depending on split/single choices
• Train combination model set (data sets 1 and 2)
• Classify using combination model set (data set 3)
```

Figure 3.2: Pseudo code for the general method

3.3 Reported results

For all experiments, there are three results reported for the final classification on data set 3 using the combination model set.

Firstly, a **raw classification** score for the set is reported. This is the percent correct value, calculated by finding the number of correctly classified tokens and dividing this value by the total number of tokens. A correctly classified token is a token whose actual label is the same as the label of the most likely model.

Then two scores using a **language model** are reported. Firstly, a score calculated using a bigram language model and viterbi search over all 61 models is stated. Then the output of this search is collapsed, and certain phone-confusions are allowed, reducing the phone set to a size of 39. This is a standard procedure. The confusions allowed are listed in table 3.2.

Phone confusions in collapsed phone set			
1.	aa ao	20.	g
2.	ae	21.	hh hv
3.	ah ax ax-h	22.	ih ix
4.	aw	23.	iy
5.	axr er	24.	jh
6.	ay	25.	k
7.	b	26.	ow
8.	bcl dcl epi gcl h# kcl pau pcl q tcl	27.	oy
		28.	p
9.	ch	29.	r
10.	d	30.	s
11.	dh	31.	sh zh
12.	dx	32.	t
13.	eh	33.	th
14.	el l	34.	uh
15.	em m	35.	uw ux
16.	en n nx	36.	v
17.	eng ng	37.	w
18.	ey	38.	y
19.	f	39.	z

Table 3.2: Allowable confusions in TIMIT phone set, from 61 to 39 distinctions

Chapter 4

Creating multiple models by splitting the data

The first approach to creating two models is to split the data labelled as a particular phone into two subsets. Two models can then be trained on this data instead of one for this phone. In this chapter we look at two ways of splitting the data for this purpose. Firstly, the pilot experiment is extended, splitting the data based on the log likelihood of the tokens as generated by the original single models. The second method uses an intrinsic characteristic of the data, the duration, and the tokens are divided by their duration.

4.1 Log likelihood criterion

The first way of splitting the data in order to make two models was used in the pilot experiments, section 2.3. The splitting criterion using the median log likelihood is detailed there.

As described in the general method, section 3.2 the two new models are trained, and then make a set with single models for all the other phones for testing. The combination set is chosen based on comparing raw classification scores between the split sets and the single sets, on data set 2. Table 4.1 shows that classification of phones [b d dh h# ix iy k kcl l m n p r s sh t tcl] improved by splitting the models.

Percent correct scores on validation data					
Phone	Score	Phone	Score	Phone	Score
aa	44.22	epi	44.19	ow	44.21
ae	44.21	er	44.12	oy	44.19
ah	44.26	ey	44.15	**p**	**44.31**
ao	44.26	f	44.26	pau	44.01
aw	44.12	g	44.16	pcl	44.16
ax	44.22	gcl	44.11	q	44.18
ax-h	44.20	**h#**	**44.83**	**r**	**44.31**
axr	44.14	hh	44.15	**s**	**44.41**
ay	44.20	hv	44.18	**sh**	**44.28**
b	**44.35**	ih	44.24	**t**	**44.44**
bcl	44.17	**ix**	**44.51**	**tcl**	**44.50**
ch	44.22	**iy**	**44.46**	th	44.12
d	**44.31**	jh	44.22	uh	44.03
dcl	44.20	**k**	**44.38**	uw	44.16
dh	**44.31**	**kcl**	**44.33**	ux	44.19
dx	44.18	**l**	**44.38**	v	44.28
eh	44.25	**m**	**44.33**	w	44.23
el	44.12	**n**	**44.59**	y	44.07
em	44.15	ng	44.19	z	44.24
en	44.12	nx	44.13	zh	44.21
eng	44.27				

Table 4.1: Percent correct scores for each phone split by likelihood (see text), compared to 44.28% for all models single. Entries in bold are models improved by splitting.

These successful split models form the mixture set to be trained, along with single models for the other phones. Training is now performed using all the data in both the training and validation sets.

The results after classification on the test data, data set 3, are shown in table 4.2.

4.2 Token duration criteria

A more phonetically motivated way of splitting the data that we investigated was to split by the duration of the tokens in the data. Since duration is a cue in speech perception, it is expected that training different models for different duration ranges will improve recognition.

	Single	Mixture
raw classification	43.0	45.6
with LM: 61 phones	58.0	58.4
with LM: 39 phones	67.1	67.3

Table 4.2: Classification scores for mixture set of likelihood split models. Both raw classification and classification with Viterbi are reported.

Still only modelling bimodality in the data, we create two models for each segment by assigning each token of that segment in the training data to one of the models according to which side of a threshold its duration lies. The threshold used for each phone was the median duration of all tokens of that phone in the training data, so the two models are trained on equal number of tokens.

Table 4.3 shows the raw classification scores for the individually split models, and as is seen there, the models for phones [b d f h ix iy k kcl l m n p r s sh t tcl v] improved by splitting.

Using these to make the mixture set, and training on training and validation data, and classifying the test data with these trained models, we give the results in table 4.4

Percent correct scores on validation data					
Phone	Score	Phone	Score	Phone	Score
aa	44.25	epi	44.12	ow	44.23
ae	44.15	er	44.12	oy	44.17
ah	44.23	ey	44.12	**p**	**44.34**
ao	44.21	**f**	**44.31**	pau	44.13
aw	44.12	g	44.11	pcl	44.18
ax	44.23	gcl	44.12	q	44.21
ax-h	44.08	**h#**	**45.02**	**r**	**44.34**
axr	44.12	hh	44.20	**s**	**44.43**
ay	44.21	hv	44.19	**sh**	**44.29**
b	**44.29**	ih	44.27	**t**	**44.42**
bcl	44.15	**ix**	**44.51**	**tcl**	**44.44**
ch	44.21	**iy**	**44.46**	th	44.13
d	**44.34**	jh	44.26	uh	44.04
dcl	44.23	**k**	**44.42**	uw	44.10
dh	44.21	**kcl**	**44.39**	ux	44.20
dx	44.08	**l**	**44.40**	**v**	**44.28**
eh	44.21	**m**	**44.34**	w	44.20
el	44.12	**n**	**44.66**	y	44.20
em	44.13	ng	44.19	z	44.21
en	44.12	nx	44.04	zh	44.23
eng	44.26				

Table 4.3: Percent correct scores for each phone split by duration (see text), compared to 44.28% for all models single. Entries in bold are models improved by splitting.

	Single	Mixture
raw classification	43.0	45.9
with LM: 61 phones	58.0	58.6
with LM: 39 phones	67.1	67.4

Table 4.4: Results using test data for the mixture set of models split by duration.

4.3 A Baseline: splitting the data randomly

We have been using the set of single models as a baseline up to now, but we have not asked what might happen to the scores if the data is split randomly. Here, we follow the method laid out in section 3.2, with the split defined simply by allocating alternate tokens of each segment in the training data to each of the two models.

The raw classification scores of these models on the validation data are reported in table 4.5

Percent correct scores on validation data					
Phone	Score	Phone	Score	Phone	Score
aa	44.28	epi	44.20	ow	44.26
ae	44.25	er	44.22	oy	44.23
ah	**44.33**	**ey**	**44.31**	**p**	**44.29**
ao	44.28	**f**	**44.29**	pau	44.17
aw	44.25	g	44.24	pcl	44.24
ax	44.23	gcl	44.21	q	44.26
ax-h	44.18	**h#**	**44.99**	**r**	**44.31**
axr	44.26	hh	44.23	**s**	**44.33**
ay	**44.28**	hv	44.23	**sh**	**44.31**
b	44.25	ih	44.26	**t**	**44.31**
bcl	44.27	**ix**	**44.36**	**tcl**	**44.31**
ch	44.25	**iy**	**44.39**	th	44.23
d	44.20	jh	44.25	uh	44.23
dcl	**44.29**	**k**	**44.31**	uw	44.25
dh	**44.28**	**kcl**	**44.32**	**ux**	**44.29**
dx	44.17	**l**	**44.34**	**v**	**44.31**
eh	44.26	**m**	**44.33**	w	44.26
el	44.22	**n**	**44.36**	y	44.17
em	44.22	ng	44.25	z	44.25
en	44.21	nx	44.09	zh	44.26
eng	**44.31**				

Table 4.5: Percent correct scores for each phone split randomly (see text), compared to 44.28% for all models single. Entries in bold are models improved by splitting.

For data set 1		For data set 2	
single models	double models	single models	double models
44.24	44.09	44.28	44.07

Table 4.6: Comparison of percent correct scores between single models and randomly split models

Table 4.6 gives the classification scores for the set of single models and compares this to the set of all split models, where the split is random. As expected, the random split gives a worse percent correct score for both data sets 1 and 2.

In accordance with the general methods applied in this chapter and the next, a combination set of models was created, of two models for segments where a split improved the raw classification score, and single models otherwise. The combination set in this case is made up of phones [ah ay dcl dh eng ey f h# ix iy k kcl l m n p r s sh t tcl ux v] split and the remainder single. Training uses sentences from both data set 1 and data set 2, and results for this split and for the set of single models are in table 4.7.

	Single	Mixture
raw classification	43.0	44.2
with LM: 61 phones	58.0	58.3
with LM: 39 phones	67.1	67.1

Table 4.7: Results using test data for models split randomly, with mixtures chosen in two ways.

Chapter 5

Creating multiple models by splitting the model parameters

When there exists only one model per segment, there can only be one target (for a stable system, with $|F| < 1$), as we saw in the introduction to linear dynamical models (section 1.5.2). Since there are different speech styles and accents, there could be multiple targets for different speakers saying the same segment. Our aim is to more effectively model this diversity, and it seems plausible that altering these targets will enable better modelling of the data.

Equation 1.9 relates this target, F, and μ_w as follows:

$$x_\infty = (I - F)^{-1}\mu_w$$

From this relationship, we see that there are two ways to change the target of a model: we can alter F or μ_w. This chapter explores one way of doing each of these, firstly creating two different matrices F using singular value decomposition, then creating two vectors μ_w by moving a fraction of a standard deviation away from the original mean.

5.1 Singular Value Decomposition

There are many ways of decomposing a matrix in order to assess its properties and behaviour. A way of analysing the 'energy' or size in certain directions is to do singular value decomposition (SVD). SVD makes it possible to express any matrix as diagonal, simply by changing the basis.

The basis of a vector space is a set of vectors which fully span the space, and are linearly independent (are not parallel), such that any vector in the space can be expressed as a linear combination. If a basis is

$$x_1, x_2, \dots x_n$$

for an n-dimensional vector space, then every element in the space can be represented *uniquely* by a sum

$$a_1x_1 + \cdots + a_jx_j + \cdots + a_nx_n$$

Now to introduce SVD itself. SVD allows the $m \times n$ matrix M to be expressed

$$M = USV^T \tag{5.1}$$

where U and V are orthogonal $m \times m$ and $n \times n$ matrices respectively[1] and V^T is the transpose of V. S is an $m \times n$ diagonal matrix whose entries are the singular values of M, in size order. If

$$S = diag(\sigma_1, \sigma_2 \dots \sigma_m)$$

then

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_m.$$

So M is reduced to the diagonal matrix S when the basis of the columns of U is the range and the basis of the columns of V is the domain. Bases are in general not unique, hence the SVD solution is not unique. However, the singular values are unique, so S is a stable solution to manipulate.

The singular values of M are in fact the square roots of the eigenvalues of M^TM. Eigenvalue decomposition of M itself would also make sense for this purpose, but since it is guaranteed to have a solution, we shall use SVD.

[1]A matrix A is orthogonal if $A^TA = I$ where I is the identity matrix

5.1.1 Toy example illustrating SVD

The toy examples below illustrate the way changing S affects the trajectory of the state. Let

$$F = \begin{pmatrix} 4 & 5 \\ 1.5 & 3 \end{pmatrix}, \mu_w = \begin{pmatrix} 0.1 \\ 1 \end{pmatrix}, \textit{ and } \mu_{x0} = \begin{pmatrix} 0.5 \\ 1 \end{pmatrix}$$

Singular value decomposition of F gives the following matrices

$$U = \begin{pmatrix} -0.89 & -0.46 \\ -0.46 & 0.89 \end{pmatrix}, S = \begin{pmatrix} 7.20 & 0 \\ 0 & 0.62 \end{pmatrix}, V = \begin{pmatrix} -0.59 & -0.81 \\ -0.81 & 0.59 \end{pmatrix}$$

For the model to tend to a target, $|F| < 1$. This can be ensured by replacing any element of S that is greater than 1 with 0.99. So now, $S = \begin{pmatrix} 0.99 & 0 \\ 0 & 0.62 \end{pmatrix}$ and F becomes $\begin{pmatrix} 0.75 & 0.54 \\ -0.18 & 0.69 \end{pmatrix}$.

The solid line in figure 5.1 is the trajectory of the state where the model uses the original F (corrected so that $|F| < 1$), the diamond is the target of the model. The dashed line is the trajectory of the state where the state matrix is permuted, $F_{new} = US_{new}V^T$ where $S_{new} = 0.9 * S$, the square is the target of the new trajectory.

5.1.2 Results

The goal is to permute the original F matrix into two matrices so that two distinct models can be initialised. One approach is to use the original matrix and one permuted matrix, but this is not in keeping with the experimental process of the previous experiments. Instead, two new matrices are formed using SVD and a 'splitting factor' k, such that the two new S matrices are calculated as $S0 = (1 - k) * S$ and $S1 = (1 + k) * S$.

Initially, the splitting factor k must be found. Choosing $k = 0.1, 0.5, 0.01$, the procedure for deciding which k to use is:

- For each phone *ph*,
  - Do SVD on F of the single model for this phone: $F = USV^T$.

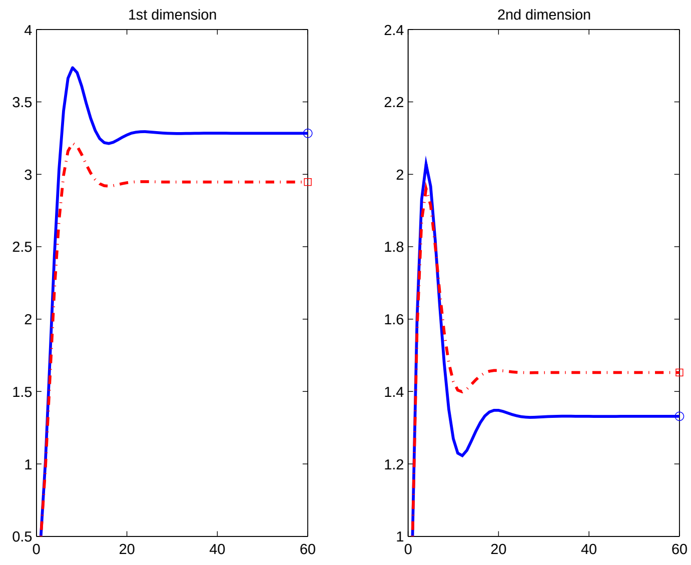

Figure 5.1: A toy example illustrating the effect of SVD on the state trajectory (see text)

- For k in 0.1, 0.5, 0.01,
  - Compute two new S matrices from this S, $S0 = (1-k)*S$ and $S1 = (1+k)*S$.
  - Construct two new F matrices from $S0$ and $S1$, $F0 = U(S0)V^T$ and $F1 = U(S1)V^T$.
  - Using the remaining parameters H, μ_v, C, μ_w, D from the single model for this phone, construct entire LDMs for $ph0$ and $ph1$.
  - Test models $ph0$ and $ph1$ against data in data set 1, finding the log likelihood of each model on each token and recording the highest. Find the mean log likelihood for ph.
- Compare the mean log likelihood figures for each split to the mean log likelihoods of the single model.

• Sum the improvements for each splitting factor. The one with the largest number of improvements will be used to construct initial models in the experiment.

Table 5.1 shows the mean likelihoods for these initial splits. The outcome of this initial experiment was that a splitting factor of $k = 0.05$ caused the largest number of improvements and so the initial models for each phone in subsequent experiments will use $S = (1 \pm 0.05)S$.

	Mean likelihood scores			
phone	single	split $k = 0.1$	split $k = 0.05$	split $k = 0.01$
aa	-430.55	**-430.53**	**-429.79**	**-429.96**
ae	-469.05	**-470.04**	**-468.48**	**-468.37**
ah	-318.32	**-318.01**	**-317.76**	**-317.98**
ao	-436.79	**-436.78**	**-436.17**	**-436.29**
aw	-552.37	**-551.05**	**-549.29**	**-549.14**
ax	-178.28	**-178.11**	**-178.15**	**-178.26**
ax-h	-120.82	**-120.74**	**-120.77**	-120.83
axr	-278.03	**-278.01**	**-277.88**	-278.07
ay	-531.51	-534.26	-531.86	**-531.21**
b	-64.32	-64.35	-64.35	-64.36
bcl	-216.76	-216.94	-216.95	**-217.06**
ch	-287.01	**-286.59**	**-286.73**	-287.06
d	-88.59	**-88.54**	**-88.57**	-88.60
dcl	-183.46	**-183.28**	**-183.36**	-183.47
dh	-130.13	**-129.95**	**-130.02**	**-130.10**
dx	-107.47	**-107.42**	**-107.45**	-107.48
eh	-332.93	-333.55	**-332.98**	-333.00
el	-319.83	-320.17	**-319.74**	-319.87
em	-287.49	-288.01	**-287.45**	-287.49
en	-281.41	**-281.04**	**-281.00**	**-281.27**
eng	-250.52	-253.20	-252.38	-252.17
epi	-156.08	**-155.84**	**-155.94**	**-156.05**
er	-420.44	-420.88	**-420.14**	**-420.21**
ey	-431.96	-438.18	-436.99	-436.79
f	-333.42	**-332.75**	**-332.99**	**-333.32**
g	-101.98	**-101.94**	**-101.97**	-101.99
gcl	-201.75	**-201.69**	-201.76	-201.86
h#	-604.70	**-514.40**	**-514.45**	**-514.93**
hh	-234.55	**-234.49**	**-234.47**	-234.62
hv	-257.07	-257.12	**-257.03**	-257.13

continued on next page

Table 5.1:

ih	-280.44	**-280.34**	**-280.17**	**-280.30**
ix	-188.06	**-187.94**	**-187.96**	-188.09
iy	-333.45	**-333.05**	**-332.94**	**-333.16**
jh	-209.29	**-209.04**	**-209.15**	-209.33
k	-192.81	**-192.59**	**-192.66**	**-192.78**
kcl	-217.57	**-217.32**	**-217.42**	**-217.54**
l	-227.46	**-227.26**	**-227.24**	**-227.42**
m	-236.22	**-235.95**	**-235.91**	**-236.11**
n	-202.57	**-202.33**	**-202.35**	**-202.52**
ng	-223.62	**-223.53**	**-223.49**	-223.65
nx	-107.24	-107.26	-107.27	-107.28
ow	-450.57	-451.02	**-450.07**	**-450.04**
oy	-598.89	-607.82	-604.15	-602.86
p	-156.01	-156.17	**-156.20**	-156.27
pau	-658.26	**-594.93**	**-594.91**	**-595.35**
pcl	-230.42	**-230.16**	**-230.25**	**-230.38**
q	-243.62	**-243.29**	**-243.35**	**-243.55**
r	-205.82	-205.96	-205.91	-205.98
s	-374.61	**-374.09**	**-374.20**	**-374.57**
sh	-385.24	**-384.69**	**-384.68**	**-385.08**
t	-172.75	**-172.51**	**-172.61**	**-172.75**
tcl	-190.58	**-190.34**	**-190.44**	**-190.55**
th	-307.38	**-306.74**	**-306.97**	**-307.26**
uh	-276.12	**-275.96**	**-275.88**	**-276.08**
uw	-370.77	**-370.41**	**-370.07**	**-370.34**
ux	-342.30	**-341.90**	**-341.74**	**-341.95**
v	-209.44	**-208.99**	**-209.13**	-209.36
w	-219.07	-219.14	-219.10	-219.24
y	-193.84	-194.42	-194.35	-194.38
z	-290.59	-290.85	-290.88	-291.12
zh	-276.26	**-276.21**	**-276.09**	-276.30

Total improvements:	43	49	33

Table 5.1: Mean likelihoods of each model generating each token, for initial svd split (see text). Improvements marked in bold.

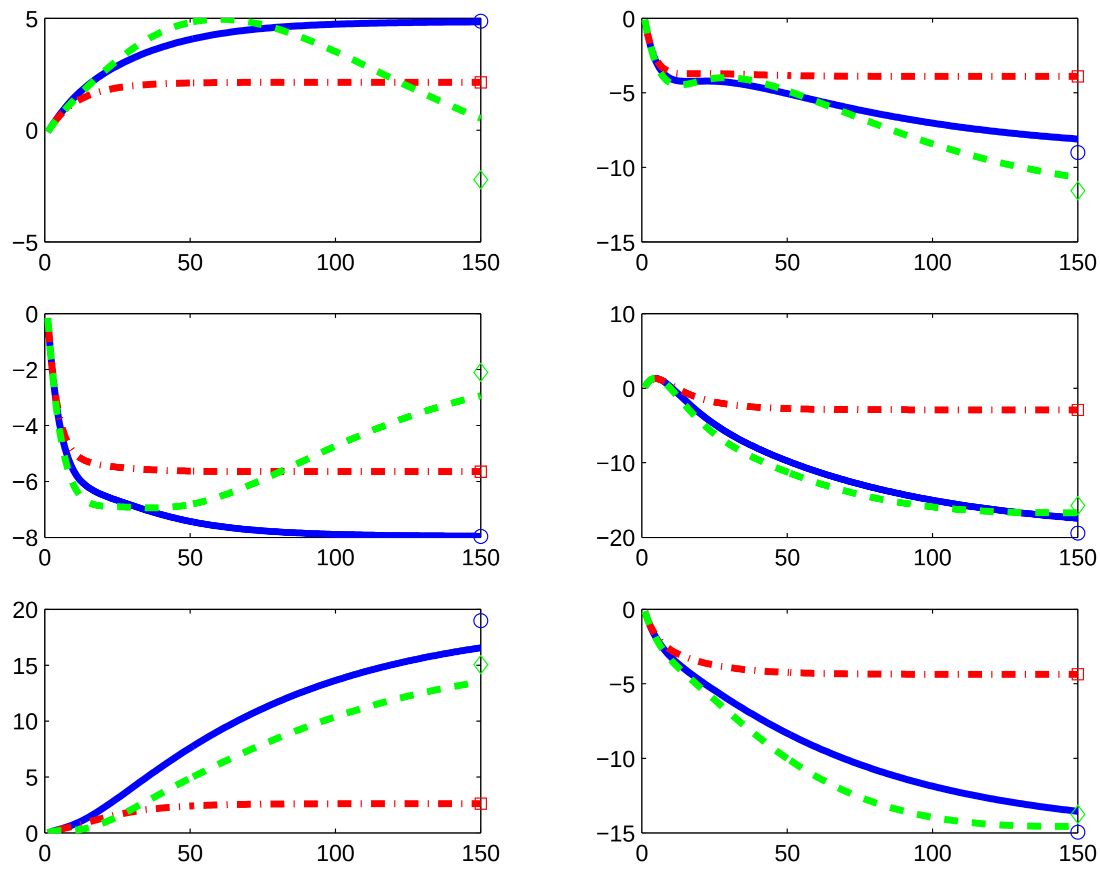

Figure 5.2: State trajectories for *el* after SVD split with $S = (1 \pm 0.05)S$. Solid line is original state, dot-dash line is $S = 0.95S$, dashed line is $S = 1.05S$

The state trajectories for model *el* can be seen in figure 5.2 for the case when $S = (1 \pm 0.05)S$. The targets of most of the dimensions are spread out, indicating that the models are distinct. Comparing these trajectories to those of *eng* for the same split, we see in figure 5.3 that for all dimensions the trajectories converge to the same target. It is perhaps not surprising, then that, as seen in table 5.1, the mean log likelihood of the training data tokens for *el* improved (compared to the single model set) for $S0 = (1-0.05)S$ and $S1 = (1+0.05)S$, while it worsened for *eng*.

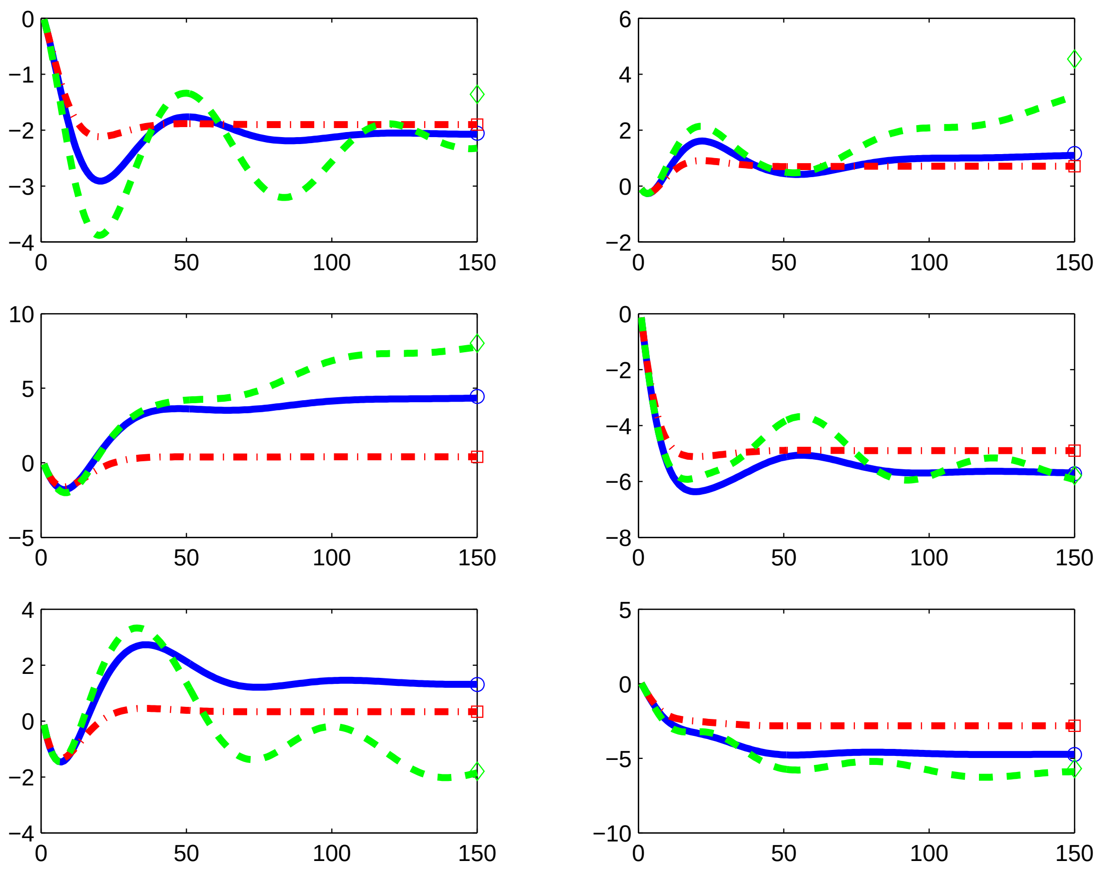

Figure 5.3: State trajectories for *eng* after SVD split with $k = 0.05$. Solid line is original state,dot-dash line is $S0 = 0.95S$, dashed line is $S1 = 1.05S$

Since a split with $S0 = (1-0.05)S$ and $S1 = (1+0.05)S$ gave the most individual improvements, these models were trained further. Continuing to train one model split at a time, with the remainder single, as described in the general method (section 3.2), the following procedure was carried out:

- split model by permuting S as $S0 = (1 - 0.05)S$ and $S1 = (1 + 0.05)S$
- allocate tokens to the model that has the higher likelihood of generating it
- train on data set 1 for two iterations

- reallocate each training token to the model that has a higher log likelihood of generating it
- train for a further two iterations

The combination set of models was chosen based on improvement of the raw classification scores on data set 2, as explained in the method, section 3.2. However, after the models had been trained for the full four iterations, none of the splits gave rise to any improvement over the single models. It is interesting that this should be the case, since in the process of reallocation each token is allocated to the model which is more likely to have generated it. Perhaps the models are over-fitting the training data.

On the intermediate (second) iteration of training, however, the classification scores show improvement between the single and split models for the phones [b h# ix iy k n s tcl v]. These improved models remained split, and the remaining 52 phones were modelled by single LDMs in the combination set. In order to retrain on both data sets 1 and 2 we must decide how to distribute the data initially. The fact that reallocation seemed to worsen the results, comparing the outcome from iteration 2 and 4 of training, led me to assume that the models are more successful when trained on the initial split. This initial split involved singular value decomposition on the matrix F of the trained single model directly, and this was again used for the initial models for the training of the combination set on data sets 1 and 2. The combination set was trained for 4 iterations.

	Single	Mixture
raw classification	43.0	44.0
with LM: 61 phone	58.0	58.2
with LM: 39 phone	67.1	67.2

Table 5.2: Results using test data (data set 3) for the mixture set of models split by SVD.

5.2 Moving the noise means

Another way of adjusting the model parameters to create two models is to shift the mean μ_w of the noise term w_t of the state evolution equation (equation 1.6). As seen in equation 1.9, moving μ_w affects the state target. Using an approach similar to that used for finding mixture

Gaussians, new mean vectors, $\mu_w 0$ and $\mu_w 1$, were found by adding or subtracting a proportion of the standard deviation. For component i of new mean $\mu_w 0$, with chosen proportion z as the 'splitting factor', the following operation was carried out:

$$\mu_w 0(i) = \mu_w(i) + z * \sqrt{D(i,i)}$$

where D is the noise covariance matrix. Similarly, for $\mu_w 1$,

$$\mu_w 1(i) = \mu_w(i) - z * \sqrt{D(i,i)}$$

The effect on the state space trajectories of such a split on the model of phone *hv* is seen for all six dimensions in figure 5.4.

5.2.1 Results

Simple experimentation was necessary initially to see the effect of moving the mean different standard deviations away from the initial mean. Choosing the splitting factor z to be 0.1, 0.2 and 0.3, we use a procedure parallelling that of choosing splitting factor k in the SVD split above. The results collected for these initial splits are seen in table 5.3.

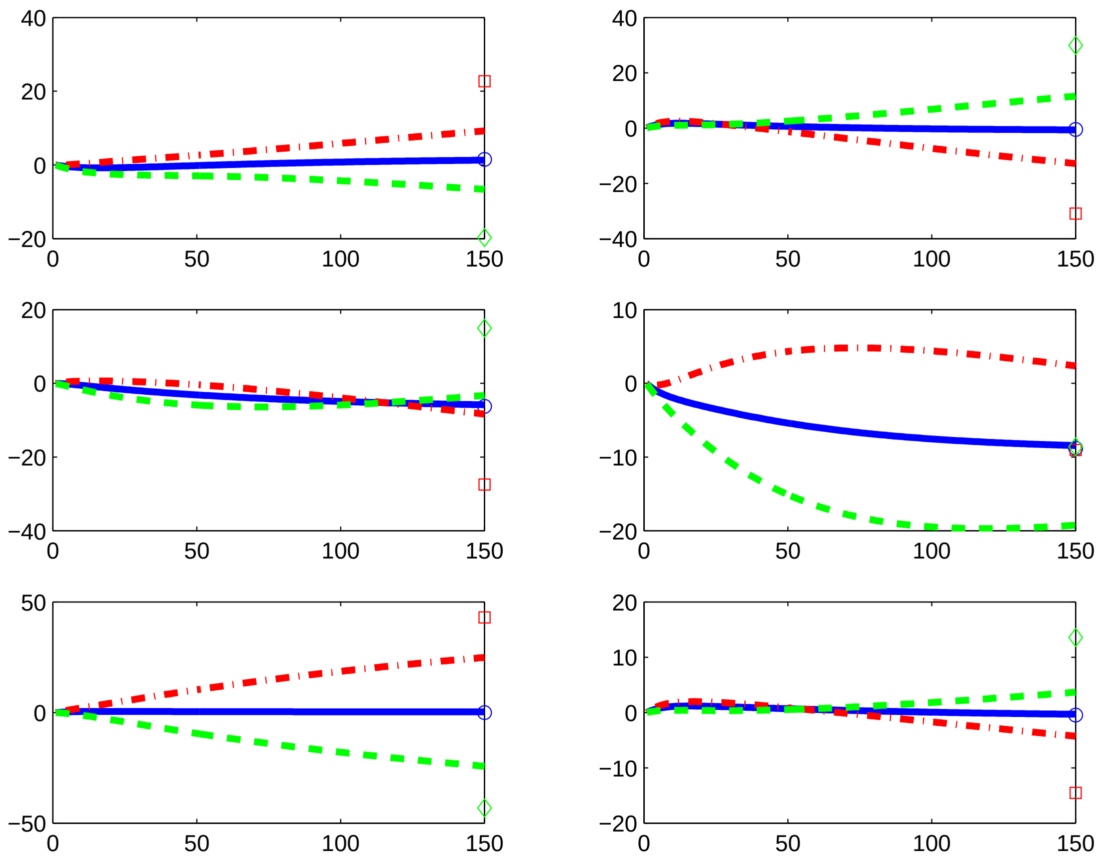

Figure 5.4: State trajectories in each of the 6 dimensions for hv after noise mean split with splitting factor $z = 0.1$. Solid line is original, dot-dash has $\mu_w 0$, dashed has $\mu_w 1$.

	Mean likelihood scores			
phone	single	split $z = 0.1$	split $z = 0.2$	split $z = 0.3$
aa	-430.55	**-429.73**	-432.65	-438.86
ae	-469.05	**-468.42**	-472.05	-479.41
ah	-318.32	**-317.60**	-319.31	-323.23
ao	-436.79	**-436.16**	-439.01	-444.92
aw	-552.37	**-548.80**	-552.79	-561.16
ax	-178.28	**-177.79**	**-178.01**	-178.95
ax-h	-120.82	**-120.50**	**-120.39**	**-120.51**
axr	-278.03	**-277.46**	-278.59	-281.41
ay	-531.51	**-529.14**	-533.07	-541.28
b	-64.32	**-64.15**	**-64.05**	**-64.03**
bcl	-216.76	**-216.20**	**-216.25**	-216.92
ch	-287.01	**-286.20**	**-286.49**	-287.83
d	-88.59	**-88.34**	**-88.22**	**-88.23**
dcl	-183.46	**-182.98**	**-182.90**	**-183.23**
dh	-130.13	**-129.85**	**-129.77**	**-129.87**
dx	-107.47	**-107.20**	**-107.05**	**-107.01**
eh	-332.93	**-332.41**	-334.26	-338.48
el	-319.83	**-319.33**	-320.72	-323.99
em	-287.49	**-286.62**	**-287.13**	-289.03
en	-281.41	**-280.67**	**-280.64**	**-281.32**
eng	-250.52	**-249.61**	**-249.60**	**-250.49**
epi	-156.08	**-155.67**	**-155.61**	**-155.89**
er	-420.44	**-419.83**	-422.84	-429.21
ey	-431.96	**-431.34**	-434.30	-440.67
f	-333.42	**-332.73**	**-333.04**	-334.34
g	-101.98	**-101.71**	**-101.57**	**-101.56**
gcl	-201.75	**-201.28**	**-201.25**	**-201.65**
h#	-604.70	**-514.66**	**-516.24**	**-519.86**
hh	-234.55	**-233.94**	**-234.18**	-235.26
hv	-257.07	**-256.46**	**-257.10**	-258.99

continued on next page

Table 5.3:

ih	-280.44	**-279.72**	-280.62	-283.01
ix	-188.06	**-187.55**	**-187.70**	-188.50
iy	-333.45	**-332.64**	**-333.37**	-335.47
jh	-209.29	**-208.66**	**-208.74**	-209.52
k	-192.81	**-192.41**	**-192.29**	**-192.46**
kcl	-217.57	**-217.15**	**-217.19**	-217.68
l	-227.46	**-226.92**	**-227.29**	-228.59
m	-236.22	**-235.61**	**-235.78**	-236.71
n	-202.57	**-202.02**	**-201.82**	**-201.98**
ng	-223.62	**-222.96**	**-222.79**	**-223.11**
nx	-107.24	**-106.91**	**-106.71**	**-106.65**
ow	-450.57	**-449.77**	-453.80	-461.91
oy	-598.89	**-597.82**	-602.17	-611.46
p	-156.01	**-155.43**	**-155.43**	**-156.01**
pau	-658.26	**-595.09**	**-596.71**	**-600.37**
pcl	-230.42	**-230.01**	**-230.04**	-230.52
q	-243.62	**-243.07**	**-243.45**	-244.76
r	-205.82	**-205.30**	**-205.75**	-207.17
s	-374.61	**-373.83**	-374.79	-377.37
sh	-385.24	**-384.42**	-385.30	-387.88
t	-172.75	**-172.27**	**-172.18**	**-172.48**
tcl	-190.58	**-190.18**	**-190.16**	**-190.52**
th	-307.38	**-306.76**	**-307.23**	-308.78
uh	-276.12	**-275.62**	-276.53	-278.86
uw	-370.77	**-369.98**	-371.55	-375.25
ux	-342.30	**-341.54**	-342.67	-345.46
v	-209.44	**-208.80**	**-208.94**	-209.88
w	-219.07	**-218.50**	**-218.54**	-219.20
y	-193.84	**-193.31**	**-193.41**	-194.12
z	-290.59	**-289.81**	-290.59	-292.93
zh	-276.26	**-275.36**	-276.37	-279.28
Total improvements:		61	39	20

Table 5.3: Mean likelihoods of each model generating each token, for initial noise mean split (see text). Improvements marked in bold.

Since the most individual improvements were caused by using the splitting factor $z = 0.1$, further training was carried out using these models as the initial set, and following the procedure laid out in section 3.2. The initial allocation of data between the two models for splitting factor $z = 0.1$ had a average of 52% of data tokens labelled model 0.

After four iterations of training on data set 1, and classification on data set 2 for all splits, none of the models were improved by being split. Thus, following the laid out method, no split models are collected to form the combination set for this split type.

This is a surprising result, since the mean log likelihood score was improved for *every* phone model at the initial split, as seen in table 5.3. One possible hypothesis to explain the lack of improvement in classification after training is that the models are over-fitting the training data. Perhaps the models perform better after fewer iterations of training. Due to the constraint of time, it was not possible to test these models further.

Chapter 6

Conclusions

6.1 Remarks on results

This thesis began with the hypothesis that creating more than one model per segment and implementing automatic switching would improve classification accuracy. In this work we have approached some solutions to finding the automatic switching criteria, and have experimented with five ways of dividing models into two, four 'intelligent' methods and a random split for comparison.

As noted at the end of chapter 5, the final way of splitting the models, by the noise mean, did not yield a combination set, and therefore no data to compare with the other splitting criterion. However, this should not be immediately considered an unsuccessful method, since there were initially promising results in the mean log likelihood scores for the models, calculating the log likelihood for each token in the training data.

Table 6.1 summarises the results for the combination sets of the other four ways of making multiple models for switching. No scores decrease from the single model scores. T-tests with 23 degrees of freedom show that the improved scores for classification with the language model and phone set of size 61 are all statistically significant improvements, when the single models are compared to the each split set ($p < 0.005$).

	Single	Random	Likelihood	Duration	SVD
raw classification	43.0	44.2	45.6	45.9	44.0
with LM: 61 phones	58.0	58.3	58.4	58.6	58.2
with LM: 39 phones	67.1	67.1	67.3	67.4	67.2

Table 6.1: Overall results for all four methods of splitting for switching. Combination sets chosen by raw classification scores of split sets

It is an interesting result that the improvement as a result of the random split is as statistically significant as that of the more intelligent splits, although the size of the improvement is not as large as the best of the other methods. Numerically, the two data splitting methods (log likelihood, and duration) give greater improvements in all three tests.

6.1.1 Alternative combination set criterion

All the combination sets reported so far in this thesis were determined based on the raw classification scores of the individual splits, as described in the method in section 3.2. This is not the only possible criterion that could be used for such a task. An alternative criterion is piloted here with the first set of experiments (split by data). In this new test, the choice of which phones to model with two LDMs and which to leave with one was made based on the classification score on data set 2 when a bigram language model is added. The method remains as described in chapter 3, except that this criterion is slightly altered.

For the models that are split by log likelihood, with the combination set chosen by the new criteria, scores improved when phones [dx eh el ey h# hh ih iy n p s] were split and these phones were thus modelled by two LDMs in the combination set. For the duration split, [d el en f h# iy jh l m s sh] remained split, and for the random split [aw ay d dcl dx eh en eng er ey f h# hh ih iy k n ng ow p pcl r sh v w] caused improvement when split.

Table 6.2 shows the classification results for these combination sets on the test data, data set 3. The results show that this criterion for choosing combination sets yields improved scores for both the log likelihood and the duration splits. In all cases, though, the final scores are better using the original criterion of raw classification scores.

	Single	Random	Likelihood	Duration
raw classification	43.0	43.7	44.0	44.2
with LM: 61 phones	58.0	58.3	58.3	58.4
with LM: 39 phones	67.1	67.0	67.1	67.3

Table 6.2: Overall results for all three methods of data-splitting. Combination sets chosen by classification scores with LM, 61 phones

6.1.2 Bi-modality

As an aside, it is interesting that certain phone models always perform better when split, and thus are always chosen to be part of the combination set as pairs. Table 6.3 is a chart of all the combination sets reported above, using both the original and alternative criterion for choosing the make up of the combination sets. The models of certain phones, such as *aa* and *q* are never split, while *h#*, *iy* and *n* are split in almost every combination set. These results indicate a possible bi-modality in the data which is independent of the split type chosen.

Comb. set Criterion	Original (raw classification)				Alternative (with LM)		
phone	rand	l'hood	dur	SVD	rand	l'hood	dur
aa	-	-	-	-	-	-	-
ae	-	-	-	-	-	-	-
ah	✓	-	-	-	-	-	-
ao	-	-	-	-	-	-	-
aw	-	-	-	-	✓	-	-
ax	-	-	-	-	-	-	-
ax-h	-	-	-	-	-	-	-
axr	-	-	-	-	-	-	-
ay	✓	-	-	-	✓	-	-
b	-	✓	✓	✓	-	-	-
bcl	-	-	-	-	-	-	-
ch	-	-	-	-	-	-	-
d	-	✓	✓	-	✓	-	✓
dcl	✓	-	-	-	✓	-	-

continued on next page

Table 6.3:

dh	✔	✔	-	-	-	-	-
dx	-	-	-	-	✔	✔	-
eh	-	-	-	-	✔	✔	-
el	-	-	-	-	-	✔	✔
em	-	-	-	-	-	-	-
en	-	-	-	-	✔	-	✔
eng	✔	-	-	-	✔	-	-
epi	-	-	-	-	-	-	-
er	-	-	-	-	✔	-	-
ey	✔	-	-	-	✔	✔	-
f	✔	-	✔	-	✔	-	✔
g	-	-	-	-	-	-	-
gcl	-	-	-	-	-	-	-
h#	✔	✔	✔	✔	✔	✔	✔
hh	-	-	-	-	✔	✔	-
hv	-	-	-	-	-	-	-
ih	-	-	-	-	✔	✔	-
ix	✔	✔	✔	✔	-	-	-
iy	✔	✔	✔	✔	✔	✔	✔
jh	-	-	-	-	-	-	✔
k	✔	✔	✔	✔	✔	-	-
kcl	✔	✔	✔	-	-	-	-
l	✔	✔	✔	-	-	-	✔
m	✔	✔	✔	-	-	-	✔
n	✔	✔	✔	✔	✔	✔	-
ng	-	-	-	-	✔	-	-
nx	-	-	-	-	-	-	-
ow	-	-	-	-	✔	-	-
oy	-	-	-	-	-	-	-
p	✔	✔	✔	-	✔	✔	-
pau	-	-	-	-	-	-	-
pcl	-	-	-	-	✔	-	-

continued on next page

Table 6.3:

q	-	-	-	-	-	-	-
r	✔	✔	✔	-	✔	-	-
s	✔	✔	✔	✔	-	✔	✔
sh	✔	✔	✔	-	✔	-	✔
t	✔	✔	✔	-	-	-	-
tcl	✔	✔	✔	✔	-	-	-
th	-	-	-	-	-	-	-
uh	-	-	-	-	-	-	-
uw	-	-	-	-	-	-	-
ux	✔	-	-	-	-	-	-
v	✔	-	✔	✔	✔	-	-
w	-	-	-	-	✔	-	-
y	-	-	-	-	-	-	-
z	-	-	-	-	-	-	-
zh	-	-	-	-	-	-	-

Table 6.3: Chart showing which models were split in the combination sets.

6.2 Suggestions for further research

With any piece of research there is room for development. Immediate further work in this area could include

- further analysis and investigation into the method of splitting the models by shifting the noise mean (section 5.2).
- splitting models into more than two distinct models and examining the effects of such a system.
- experiment with different numbers of training iterations for different phones. This is the second hypothesis indicated by the pilot experiment, chapter 2.

- combining temporal switching with the multi-modal switching described here.
- further investigation into the hypothesis that the data contains bimodality, as highlighted in section 6.1.2

There are many such directions that could be taken from the results and conclusions reported here. It is believed however that an optimal system will not be constrained to phone sized units, but will instead learn the unit length from data. Research into this area will be much more valuable. Since the relatively simple techniques used to extend the recognition system in this dissertation produced improvements, it would be surprising if employing more sophisticated learnt-from-data methods did not also enhance the system to a greater degree.

Bibliography

[1] Oxford english dictionary, 2nd edition. online, 1989.

[2] V Digalakis. *Segment-based stochastic models of spectral dynamics for continuous speech recognition.* PhD thesis, Boston University Graduate School, Boston, 1992.

[3] V. Digalakis, J.R. Rohlicek, and M. Ostendorf. A dynamical system approach to continous speech recognition. *IEEE*, 1991.

[4] J. Frankel. *Forthcoming.* PhD thesis, University of Edinburgh, 2003.

[5] J. Frankel and S. King. Articulatory speech recognition. In *Proc. Eurospeech-01, Scandinavia*, 2001.

[6] J. Frankel and S. King. Speech recognition in the articulatory domain: investigating an alternative to acoustic hmms. In *Workshop for Innovations in Speech Processing (WISP)*, pages 37–46, 2001.

[7] J.S. Garofolo. *Getting started with the DARPA TIMIT CD-ROM: An acoustic phonetic continuous speech database.* National Institute of Standards and Technology (NIST), Gaithersburgh, MD, 1988.

[8] D. Jurafsky and J.H. Martin. *Speech and language processing.* Prentice Hall, 2000.

[9] O.A. Kimball. *Segment modeling alternatives for continuous speech recognition.* PhD thesis, Boston University College of Engineering, 1995.

[10] M. Ostendorf. Moving beyond the 'beads-on-a-string' model of speech. In *IEEE ASRU Workshop*, 1999.

[11] M. Ostendorf and V. Digalakis. The stochastic segment model for continuous speech recognition. *IEEE*, 1991.

[12] M. Ostendorf, V. Digalakis, and O.A. Kimball. From HMMs to segment models: A unified view of stochastic modeling for speech recognition. *IEEE Trans. on Speech and Audio Processing*, 1996.

[13] A-V.I. Rosti and M.J.F. Gales. Generalised linear gaussian models. Technical Report CUED/F-INFENG/TR.420, Cambridge University Engineering, 2001.

[14] S. Roweis and Z. Ghahramani. A unifying review of linear gaussian models. *Neural Computation*, 11(2):305–345, 1999.

[15] R. Singh, B. Raj, and R.M. Stern. Automatic generation of subword units for speech recognition systems. *IEEE*, 2002.