Proceedings of the Institute of Acoustics

OPTIMAL FEASIBLE HIDDEN MARKOV MODELS

Bill McKee and Ian Barlow

School of Electronic, Communication & Electrical Engineering
University of Plymouth, Drake Circus, Plymouth, Devon PL4 8AA

1. INTRODUCTION

This paper reviews the initial stages of research into a novel application of Hidden Markov Models (HMMs) for automatic speech recognition. The research aims to tackle multi-speaker recognition by tuning or 'personalizing' the models at recognition time into those of a speaker whose voice is similar to that which generated the utterance. Such tuning calls for an effective HMM adaptation algorithm, and efforts to date have been directed at developing a suitable algorithm. As a consequence of this work, some interesting features of the mathematics of Hidden Markov Models have come to light. Among these, there are indications that HMMs can be identified which are globally optimal with respect to the training data...so-called Optimal Feasible Hidden Markov Models.

After presenting the necessary background, this paper gives a descriptive account of the current progress.

2. BACKGROUND

Historically, there have been two main approaches to ASR (see Fig 1). In the cognitive or Knowledge-Based approach [1], one attempts to elicit and to automate the rules used by knowledgeable experts. Alternatively, in the information-theoretic or Template Matching approach [2], a collection of prototypical patterns are collected into a reference memory. Then a sample utterance is compared against these patterns, and a classification is made based upon the 'best' match. Recently, Neural Network recognizers have come along [3] which bridge the gap between the older approaches. The neural network performs a type of matching based upon features that it 'learns' by itself from the training data.

Hidden Markov Models (HMMs) belong to the template matching approach, but are generally superior to the other members of that family. Typical template matchers cluster the training data and retain the collection of centroids, which comprise the reference memory, but then discard other available information, such as the distribution of the clusters about their centroids. On the other hand, HMMs are able to incorporate this extra information and, consequently, tend to achieve better results.

A Hidden Markov Model can be viewed simply as a special type of finite state machine which does not depend upon external inputs. Instead, the transition between states is a random process that satisfies the Markov property, whereby the state at time t depends only upon the previous state. The sequence of states through which the model passes is not directly observable (hence the term 'hidden'). However, at each discrete time t, upon entering a state, the model emits a signal which is observable. The signal which is emitted is also random, with the choice of output controlled by a different random process per state.

If there are N states, and if the total collection of possible outputs is limited to a set of M discrete symbols, then an HMM can be specified in terms of three matrices:

an $N \times N$ matrix of transition probabilities, A , where element a_{ij} gives the probability of moving from state i to state j

an $N \times M$ matrix of output probabilities, B , where element b_{jk} gives the probability of generating output k from within state j

and an $N \times 1$ initial state distribution vector, p , where element p_i gives the probability of starting off in state i

In using an HMM to model data, one assumes that the process which generated the data was itself an HMM , and then attempts to guess the model parameters (i.e. matrix elements) of the underlying HMM . According to the usual criterion, the best model for a set of data is the triple (A, B, p) which exhibits the greatest probability that the observed data would be observed.

To carry out speech recognition a library of HMMs is constructed, with a model for each item (word, syllable, phoneme, etc.) in the vocabulary. A test pattern is compared against the reference HMMs by computing the probability that each HMM might have generated it. The HMM which exhibits the greatest probability is deemed the best match, and the test pattern is so classified.

The difficulties for automatic speech recognition can be summarized in terms of the two 'dimensions' shown below (Fig 2) . Where the characteristics of only a single speaker are involved, and the words are separated by clear gaps, systems can achieve high levels of success. But as the number of speakers or the fluency of the speech increases, recognition accuracy falls off dramatically.

The current research is focussed on the multi-speaker dimension of the problem, with the goal of developing recognizers which perform equally well for more than one speaker.

An obvious way to try to achieve speaker independence is to incorporate training data from multiple speakers into the models, with the intention that the resulting set of models will capture the characteristics of an 'average' speaker. Fortunately, existing HMM training algorithms can acccomodate a population of speakers. However, while it is clear that some sort of averaging takes place, it is also clear that the results are not those of an 'average' speaker, but something far more vague. Furthermore, any averaging inevitably leads to a 'blurring' or 'smearing' effect, and it is natural that this would impair the recognition accuracy.

In terms of accuracy, the best solution would be to score the unknown utterance against library models that were derived from a voice which is as similar as possible to that which generated the unknown utterance. Clearly, one approach would be to have a separate library for each possible speaker. This would reduce the multiple speaker problem to the single speaker case. However not only is this unmanageable in practice, but it is not a solution when the speaker is unknown.

The approach which this project is adopting is to train the set of models to a population of speakers, but then to transform the models at recognition time. This tuning would have the effect of quickly 'personalizing' the models into those of a speaker whose voice is similar to that which generated the utterance, rather than that of an average speaker.

3. WORK AND PROGRESS TO DATE

3.1 Adaptive HMMs

A critical part of model-tuning is the ability to adapt HMMs incrementally. For example, having established within a reasonable certainty that the last uttered word was "dog", one should then be able to use that utterance to update the "dog" model (and perhaps a few other models) accordingly. In this way, the library HMMs would converge to a personalized set of models.

Unfortunately, the existing algorithms for building HMMs don't lend themselves to incremental model-building/adaptation. For example, the classic Baum-Welch method performs a 'batch' calculation of the matrix elements using the assembled training data. If more data then comes along, the model has to be scrapped and the calculations performed again.

Eventually, the right sort of algorithm surfaced from the area of Visual Pattern Recognition [4,5,6]. Given an HMM derived from observations up through time $t - 1$, and assuming that at time t the m^{th} symbol is observed, then the elements of the A and B matrices are amended according to a given set of update formulas.

However, upon careful examination two important weaknesses became apparent :

(A) The procedure is somewhat 'ad hoc', lacking in theoretical foundations

(B) The formulas appear not to be mathematically sound. Essentially, their derivation misses the distinction between independent events and those which are conditionally independent [7]

3.2 HMMs as Neural Nets

While the above sources failed to provide an acceptable approach to adaptive HMMs, they did make an important contribution. The remaining HMM literature (see [8]) focusses exclusively on the three operations of

Learning	Given a body of training data derived from some real-world process, to compute the parameters of the HMM which best models that data
Classification	Given an observation sequence and an HMM, to calculate the probability that the sequence was generated by that HMM
Explanation	Given an observation sequence and an HMM, to determine the sequence of hidden states through which the HMM passed

while generating the sequence

The Gong paper [4] highlights a fourth operation, namely:

Generation Given an HMM, to predict the sequence of its states as well as the sequence of observations it is likely to generate

We know that π gives the N-element vector of probabilities that the model is in state i at time $t = 1$. In a similar way, the calculation $(A^{t-1})'\pi$ generates the N-element vector of probabilities that the model is in state j at time t. Furthermore, the calculation $B'(A^{t-1})'\pi$ generates the M- element vector of probabilities that symbol k appears at time t.

It was then recognized that this operation could be modelled as a neural network. If the state probability vectors at times $t - 1$ and t, as well as the output probability vector, are each represented as a layer of nodes, then the HMM is entirely equivalent to the special type of recurrent neural network shown in Fig 3 (see [9]). The connection weights between the input and hidden layers constitute the A matrix, and the weights between the hidden and output layers constitute the B matrix.

Finally, consider the M-element vector D_t consisting of all zeros except for a single 1 to indicate the codebook symbol actually observed at time t. This gives the desired output from the network at time t, as opposed to the predicted output given by the calculation $B'(A^{t-1})'\pi$. By adapting neural network training algorithms (e.g. back-propagation), using the collection of D_t vectors as the training data, a new method was suggested for building and adapting HMMs incrementally.

As a technique for training neural networks, back-propagation is very slow, requiring numerous passes through the entire body of training data ('epochs') in order to converge. Fortunately, unlike normal nodes which apply a non-linear transfer ('squashing') function to their inputs, the above nodes are strictly linear, which implies that a direct solution for the connection weights should be possible.

A preliminary analysis tended to confirm the existence of a direct solution based upon the full training data, and also suggested that a straight-forward method exists to build/adapt solutions based upon individual data. The fact that sums of the outer products of the input vectors are positive definite (almost surely) permits application of the Matrix Inversion Lemma [10] which yields a formula for incrementally updating the required matrix inverse. The entire procedure is similar to the Recursive Least Squares method for adaptive filters.

3.3 Matrix Derivatives

The previous analysis suggested how one might derive an algorithm to incrementally build/adapt HMMs .

Let
$$e = \sum_t \|D_t - B'(A^{t-1})'\pi\|^2$$

= total squared error between desired and predicted probability vectors

In outline:

(A) Find the (A, B, π) values which minimize e over the first T training data
(B) Find a similar solution for the first $T+1$ data
(C) Determine how to use the new data D_{T+1} to convert the first solution into the second

Note the similarity with Linear Prediction, where one tries to minimize

$$f = \sum_n [x_n - H'X_n]^2$$

$$\text{with} \quad H = [h_1 \, h_2 \ldots h_p]' \qquad X_n = [x_{n-1} \, x_{n-2} \ldots x_{n-p}]'$$

The procedure is:

(A) Calculate the partial derivative of f with respect to each of the unknown filter coefficients h_i and equate the individual derivatives to zero
(B) Bundle the resulting set of p equations into a single matrix equation of the form $\left(\frac{\partial f}{\partial h_i}\right) = 0$
(C) Solve the matrix equation for H

However, in contrast with the Linear Prediction case, HMM model-building involves 3 matrix unknowns and the differentiation of e with respect to a total of $N^2 + MN + N$ matrix elements, before they are eventually bundled into a system of 3 simultaneous matrix equations.

Clearly it would be an advantage to be able to derive the matrix equations $\left(\frac{\partial e}{\partial a_{ij}}\right) = 0$ directly, without having to calculate the individual partial derivatives. This bundle (array) of partial derivatives is one possible definition [11,12] of a 'matrix derivative' : $\frac{de}{dA} = \left(\frac{\partial e}{\partial a_{ij}}\right)$

The next step of the research was to assemble, and in some cases derive, a set of tools by which these matrix derivatives could be calculated.

3.4 Optimal Feasible HMMs

Given the ability to compute matrix derivatives, the way was clear to attempt to minimize

$$e = \sum_t \| D_t - B' (A')^t \pi \|^2$$

as per the previous discussion. Eventually a solution was found and, fueled by this success, attention was then directed at the customary (but much more challenging) HMM objective function [13]

$$\text{Maximize} \quad P(o_1 o_2 .. o_T) = 1' \, \mathrm{Diag}(BD_T) \, A' \ldots A' \, \mathrm{Diag}(BD_2) \, A' \, \mathrm{Diag}(BD_1) \, \pi$$

The analysis yielded these interesting results :

(A) Within the interior of the solution space, a simple test distinguishes locally optimal triples (A, B, π) --namely, the expression

$$L = \mathrm{Diag}(BD_1) \, A \, \mathrm{Diag}(BD_2) \, A \ldots A \, \mathrm{Diag}(BD_T) \, 1$$

must produce a column vector of identical values

(B) Among those feasible solutions, an upper bound exists on the value of the objective function $P(o_1 o_2 .. o_T)$

(C) Feasible triples can be identified which achieve this upper bound and are therefore 'globally' optimal within the interior of the solution space

3.5 Generalized Linear Programming

It must be emphasized that the solutions thus found are only 'globally' optimal within the interior of the solution space. In fact, superior triples are generally to be found on the boundary of the solution space, analogous to end-point maxima. The search for a true global optimum thus requires a methodical procedure for searching the boundary.

In the case of a linear objective function which is defined on a convex solution space, the theorems of Linear Programming show that the global optimum always occurs at a vertex of the solution space. Furthermore, to locate the optimum, one starts at any vertex and examines the edges emanating outward from it, to find the edge along which the objective function increases most rapidly. One then advances along that edge to the next neighboring vertex and repeats the procedure, until there is no increase along any edge, which signals that the current vertex is the optimal one.

In the present task, the solution space is convex but the objective function is far from linear. Nevertheless, some early work suggests that close parallels may exist with the linear case:

Firstly, it is possible to distill down the boundary planes into a collection of 'conceptual vertices', and to define a neighbor relationship among them (i.e. Vertex A is a neighbor of Vertex B). Secondly, one can specify a procedure to evaluate the objective improvement on the outgoing edges. Finally, advancing along the steepest edge appears always to produce a solution which is a true global optimum to the HMM training task.

4. FUTURE WORK

Much more work remains before the goal of tunable Hidden Markov Models has been achieved. The immediate objectives focus on the methodical search for optimal models around the boundary of the solution space:

(A) Derive a mathematical proof that the strategy described above will always lead to a global optimum

(B) Simplify the process of evaluating the conceptual edges, which currently involves optimizing fragments of the original objective function (Geometric Programming is being investigated for this purpose.)

(C) Discern how the global optimum solutions change from problem to problem as additi0nal data are included

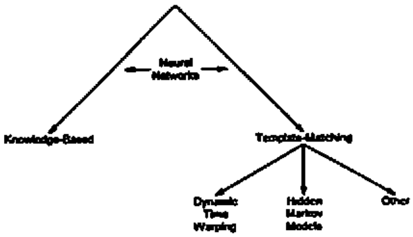

Fig 1 ASR Approaches

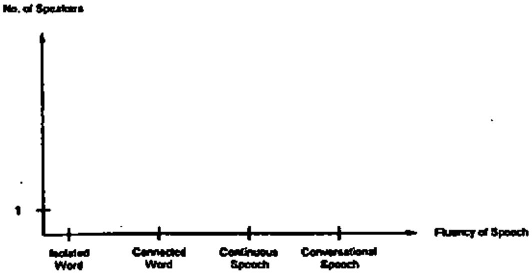

Fig 2 'Dimensions' of Difficulty

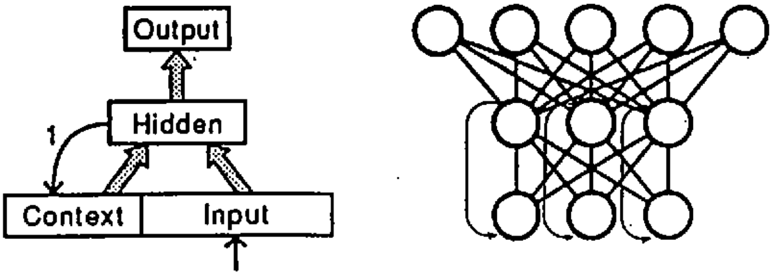

Fig 3 Two Views of an HMM-equivalent Neural Network

5. REFERENCES

[1] M ALLERHAND Knowledge-Based Speech Pattern Recognition. Kogan Page, 1987.

[2] D O'SHAUGHNESSY Speech Communication, Human and Machine. Addison-Wesley, 1987.

[3] D P MORGAN & C L SCOFIELD Neural Networks and Speech Processing. Kluwer Academic Publishers, 1991.

[4] S GONG Visual observation as reactive learning. IN : Adaptive and Learning Systems, 20-21 April 1992, Orlando,Florida. Proceedings of SPIE--The International Society for Optical Engineering. 1992. Vol. 1706, pp. 175-186.

[5] R D RIMEY & C M BROWN Selective Attention as Sequential Behavior : Modeling Eye Movements with an Augmented Hidden Markov Model. IN : DARPA Image Understanding Workshop, September 1990, Pittsburgh, Pennsylvania. Proceedings, pp. 840-849.

[6] R D RIMEY & C M BROWN Selective Attention as Sequential Behavior : Modeling Eye Movements with an Augmented Hidden Markov Model. Technical Report 327 (revised), Department of Computer Science, University of Rochester, April 1990.

[7] J PEARL Probabilistic Reasoning in Intelligent Systems : Networks of Plausible Inference. Morgan Kaufmann, 1988.

[8] L R RABINER A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proceedings of the IEEE. February 1989. Vol. 77, No. 2, pp. 257-286.

[9] J HERTZ, A KROGH & R G PALMER Introduction to the Theory of Neural Computation. Addison-Wesley, 1991.

[10] S HAYKIN Adaptive Filter Theory. Prentice-Hall, 1986.

[11] G S ROGERS Matrix Derivatives. Marcel Dekker, 1980.

[12] J R MAGNUS & H NEUDECKER Matrix Differential Calculus with Applications in Statistics and Econometrics. John Wiley & Sons, 1988.

[13] S E LEVINSON, L R RABINER & M M SONDHI An Introduction to the Application of the Theory of Probabilistic Functions of a Markov Process to Automatic Speech Recognition. The Bell System Technical Journal. April 1983. Vol. 62, No. 4, pp. 1035-1074.