PRODUCT-HMM - A NOVEL CLASS OF HMMS FOR SUB-SEQUENCE MODELLING

MUKUNDHN NAGARAJAN and T.V.SREENIVAS

Department of Electrical Communication Engineering, Indian Institute of Science, Bangalore 560 012, India
email: tvsree@ece.iisc.ernet.in

ABSTRACT
This paper presents a novel kind of HMM, called Product-HMM, that can be used for sub-sequence modelling. A sub-vector is formed by selecting particular components from the original vector. A sequence of such sub-vectors forms a sub-sequence. This paper considers the case of modelling a vector sequence in terms of its two sub-sequences. In the present framework of HMM, the architecture of the HMM is fixed for a class of vector sequences. The freedom of selecting a suitable architecture for each of the sub-sequences (of the class of vector sequences considered earlier) is not possible. Product-HMM offers this flexibility of selecting a separate architecture for each of the sub-sequences. Number of states and structure of the transition matrix (which decides whether the HMM is a left-to-right one or an ergodic one) constitute the architecture of a HMM. So, Product-HMM offers the freedom to choose different number of states for each of the sub-sequences and to choose independently whether each of them is to be modelled by a left-to-right or an ergodic HMM. It is shown that modelling using Product-HMM is better than modelling the two sub-sequences using two independent HMMs. This way of joint training of a HMM from two streams of sequential data has not been tried before. Product-HMM is an integrated statistical model which provides a way of integrating different HMMs that model the sub-sequences of a vector sequence. The possibility of having optimal HMM architectures for the sub-sequences results in utilising the training data better and hence in better estimates of model parameters.

1. INTRODUCTION
Hidden Markov Models are used to represent classes of vector sequences of varying lengths. If each vector in the sequence is split into two sub-vectors, we will have two sub-sequences. The temporal structures of the two sub-sequences might require different HMM architectures for modelling them optimally. One straightforward way of doing this would be to model the two sub-sequences using two independent HMMs. Normally, the emission densities of the HMMs are assumed to have diagonal covariance matrices implying that the components of the vectors are uncorrelated. But, the transition probabilities of the two HMMs may not be independent of each other. So, we need a framework in which the dependency of the transition probabilities of the two HMMs can be captured.

In this paper we propose a new framework of Product-HMM which allows the freedom of selecting a different architecture for modelling each of the sub-sequences and yet does not consider the sub-sequence models to be wholly independent of each other. Product-HMM provides a means of training a HMM from two groups of sequential data in a joint manner.

The organization of the paper is as follows: The Product-HMM architecture is first explained. Comparison of Product-HMM with other structured-HMMs is presented. Experiments that illustrate the advantages of Product-HMM are described followed by discussion of the results and inferences. Lastly, directions for future work and conclusions follow.

The following notation is consistently used throughout this paper.
- \( O_t \) - \( t \)’th vector of the unsplit vector sequence.
- \( O_{1t} \) - \( t \)’th first sub-vector of \( O_t \)
- \( O_{2t} \) - \( t \)’th second sub-vector of \( O_t \)
- \( O_t = [O_{1t}, O_{2t}] \)
- \( O \) - Unsplit vector sequence. \( O = \{O_t\}, t=1,2,...,T \), where \( T \) is the length of the vector sequence.
- \( O_1 \) - First sub-sequence of \( O \). \( O_1 = \{O_{1t}\}, t=1,2,...,T \)
- \( O_2 \) - Second sub-sequence of \( O \). \( O_2 = \{O_{2t}\}, t=1,2,...,T \)
- \( \lambda_1 \) - HMM Architecture chosen for \( O_1 \)
- \( \lambda_2 \) - HMM Architecture chosen for \( O_2 \)
- \( \lambda_p \) - Product-HMM
- \( N_1 \) - Number of states in \( \lambda_1 \)
Fig. 1. Product-HMM state configuration

- $N_2$ - Number of states in $\lambda_2$
- $N$ - Number of states in $\lambda_p$, $N=N_1\times N_2$
- $S_1=\{s_{11}, s_{12} \ldots s_{1N_1}\}$ - Set of states of $\lambda_1$
- $S_2=\{s_{21}, s_{22} \ldots s_{2N_2}\}$ - Set of states of $\lambda_2$
- $S=\{s_1, s_2 \ldots s_N\}$ - Set of states in $\lambda_p$, which is the cartesian product of $S_1$ and $S_2$

2. PRODUCT-HMM ARCHITECTURE

Fig. 1 illustrates the architecture of Product-HMM. $N_1$ number of states are chosen for modelling the class of sub-sequences $O_1$ and $N_2$ number of states are chosen for modelling the class of sub-sequences $O_2$. Each state in the Product-HMM is made up of 2 components. One of the component states is from $S_1$ and the other is from $S_2$. The set of states in Product-HMM is the cartesian product of the set of states in $\lambda_1$ and $\lambda_2$ and the number of states in Product-HMM is the product of number of states in $\lambda_1$ and $\lambda_2$ (i.e., $N=N_1 \times N_2$).

The probability of emission of the complete vector $O_1$ is given as $P(O_1|S_1) = P(O_{11}|S_{11})P(O_{12}|S_{12})$, where $O_{11}$ and $O_{12}$ are the sub-vectors of $O_1$ emitted by component states $S_{11}$ and $S_{12}$ respectively.

The transition matrix constraints necessary for $\lambda_1$ and $\lambda_2$ which decide whether the HMM is a left-to-right one or an ergodic one can be incorporated into $\lambda_p$. So, the $N \times N$ transition matrix of $\lambda_p$ takes care of the transition constraints of $\lambda_1$ and $\lambda_2$. For example, suppose that $\lambda_1$ has the constraint that the probability of transition from $s_{11}$ to $s_{12}$ is zero. This constraint will show up in $\lambda_p$ as the probability of transition from any state in which $s_{11}$ is the first component to any other state in which $s_{12}$ is the first component is zero. Thus, $\lambda_p$ preserves the architectures of both $\lambda_1$ and $\lambda_2$.

Refer Fig. 2. This illustrates the fact that $\lambda_p$ preserves the architectures of both $\lambda_1$ and $\lambda_2$. In this example, $\lambda_1$ is a 2 state ergodic HMM and $\lambda_2$ is a 3 state left-to-right HMM. The Product-HMM $\lambda_p$ made up of 6 combination states is ergodic. i.e., We can choose the transition matrix of $\lambda_p$ in such a way that the individual architectures of both $\lambda_1$ and $\lambda_2$ are preserved. The emission densities of each state in the Product-HMM consist of 2 densities one for each of the component states.

The essence of Product-HMM is to train $\lambda_p$ using $O$ ($O_1$ and $O_2$ jointly) rather than training $\lambda_1$ and $\lambda_2$ independently using $O_1$ and $O_2$. The advantage of Product-HMM is that the dependency between state transitions in $\lambda_1$ and $\lambda_2$ is captured (i.e., the incorrect assumption that the state transitions in $\lambda_1$ and $\lambda_2$ are independent of each other is not made).

For a given class of $O$, $\lambda_p$ has to be trained after splitting $O$ into $O_1$ and $O_2$. Re-estimation formulae for Product-HMM have been derived [7] for the case of mixture gaussian emission densities using Expectation-Maximization algorithm. The derivation is on the lines of the derivation for the regular HMMs [2].

We present next the comparison of Product-HMM with other important structured-HMMs currently available in the literature.
3. PRODUCT-HMM IN THE PERSPECTIVE OF OTHER STRUCTURED-HMMs

Several important structured hidden markov models are seen in the literature. As Product-HMM is a kind of structured-HMM, it is good to compare this with the other structured models in terms of modelling abilities and topologies. Four important structured-HMMs are taken for consideration: Factorial HMMs, HMM-squared, Hierarchical HMMs and Multi-stream HMMs.

3.1. Factorial-HMM

Factorial HMM introduced by Zoubin Ghahramani et al., is a generalization of HMMs in which the state is factored into multiple state variables and is therefore represented in a distributed manner. The paper [3] addresses the problem of constructing efficient learning algorithms for Hidden Markov Models with distributed state representations.

F-HMM enables us to have a HMM with large number of states in a structured way so that training with finite data is not a problem. This is a triply stochastic model (unlike regular HMM which is a doubly stochastic model) which uses 3 levels of stochastic processes (emission density, main-state, sub-state) to model the observation sequence.

The main difference between Product-HMM and Factorial-HMM is that in P-HMM (Product-HMM), the sub-states in each state model one half of the observation vector and in F-HMM, the sub-states in each state model the whole of the observation vector. This difference comes out from the motivation for introducing the P-HMM: to come up with a model that accommodates the different stochastic natures of the sub-sequences. One minor difference is that P-HMM is a doubly stochastic model (just like a regular HMM) whereas F-HMM is a triply stochastic model.

3.2. HMM-squared

HMM2 (HMM-squared) was introduced by Katrin Weber et al.[4]. In this, the external HMM emits a sequence of feature vectors, exactly as a conventional HMM does. Each of these feature vectors, however, is emitted by an internal HMM as a sequence of sub-vectors. The internal HMM thus replaces the Gaussian mixture distributions of conventional HMMs. An internal HMM is introduced in each state of the external HMM. The feature vector is cut into subvectors (called as ‘internal feature vectors’). The sequence of these internal feature vectors corresponding to a feature vector is modelled by the internal HMM. In the limiting case of HMM2, each coefficient of the feature vector can be an internal feature vector and each feature vector is modelled as a sequence of scalars emitted by the internal HMM. On the other hand, the whole feature vector could be taken as a single internal feature vector, in which case, the internal HMM reduces to the conventional GMM and HMM2 becomes a regular HMM.

The main motivation behind the HMM2 approach is better modelling of the correlation across feature vector components. i.e., contrarily to GMM-based systems where the usual assumption is that the feature vector components are uncorrelated, in HMM2, the only assumption is that a sequence of subvectors can be modelled by a first order HMM.

A comparison of the modelling properties of HMM2 and P-HMM will be interesting to make. HMM2 models a single feature vector as a sequence of subvectors. P-HMM models the feature vector sequence as two subvector sequences, which may have different stochastic properties. HMM2 models the temporal characteristics of the components in a single feature vector. P-HMM models the statistical correlation between the subsequences of the feature vector sequence. A feature vector sequence containing vectors whose components ‘evolve’ in time is better modelled by HMM2 and a feature vector sequence containing sub-sequences having different stochastic properties is better modelled by P-HMM.

3.3. Hierarchical HMM

Hierarchical HMM(HHMM) introduced by Shaifine et al.[5] generalizes the standard HMMs by making each of the hidden states an ‘autonomous’ probabilistic model on its own, that is, each state is an HHMM as well. Therefore, the states of an HHMM emit sequences rather than a single symbol. An HHMM generates sequences by a recursive activation of one of the substates of a state. This substate might also be composed of substates and would thus activate one of its substates, etc.

HHMM is a kind of structured segmental-HMM where each state of the ‘base’ or the ‘root’ HMM emits a sequence of vectors instead of a single observation vector. On the other hand, each sub-state in the Product-HMM emits part of a single observation vector. HHMM models the temporal characteristics of the vector sequence (leading to segmentation of the vector sequence) whereas Product-HMM models the statistical dependence between the sub-sequences of the vector sequence.

Product-HMM and HHMM can be combined together to model a vector sequence in terms of its sub-sequences and in terms of its segments respectively.

3.4. Multi-stream HMM

Multi-stream HMMs are used by J.Luettin et al.[6] for information fusion from two streams of data (Audio stream and Visual stream) for robust speech recognition. In this kind of HMMs, an independent HMM is available for each of the streams. The overall likelihood of the multi-stream sequence is the product of the likelihoods of the independent
streams, raised to appropriate stream exponents that capture the reliability of each individual model.

The difference between the Multi-stream HMM and the Product-HMM can be stated as follows: In the Multi-stream HMM, the stream likelihoods are taken as independent when calculating the overall likelihood and in the Product-HMM, it is not so. The statistical dependence between the streams is incorporated in the framework of the Product-HMM. It can be stated that the calculation of the overall likelihood emerges ‘naturally’ in Product-HMM.

One other difference is that the lengths of the streams need not be same in the Multi-stream HMM since the overall likelihood is just the product of the individual likelihoods. But, in Product-HMM, the lengths of the streams should be same and this restriction comes from the motivation of deriving a model that is suitable for sub-sequences rather than multi-stream sequences.

4. EXPERIMENTS

Experiments are conducted with simulated data to bring out the properties of Product-HMM. Experiment 1 is designed to illustrate the fact that Product-HMM can capture the statistical dependence between the two sub-sequences much better than the individual HMMs which model the two sub-sequences independently. Experiment 2 is designed to illustrate the fact that for the same amount of training data, a Product-HMM provides a better model of the data than two individual HMMs.

4.1. Experiment 1 - Capturing statistical dependence between 2 sequences

Fig. 3 shows the decomposition of a state transition in Product-HMM. $S_1$ consists of 2 component states $S_{11}$ and $S_{21}$. $S_2$ consists of two component states $S_{12}$ and $S_{22}$. So, a Product-HMM transition of $S_1$ to $S_2$ consists of 2 transitions $S_{11}$ to $S_{12}$ and $S_{21}$ to $S_{22}$.

Fig. 4 shows the overview of Experiment 1.

Two cases are considered in this experiment. In one case, the vector sequence $O$ is generated and then split into sub-sequences $O_1$ and $O_2$. We expect a statistical dependence between $O_1$ and $O_2$ in this case. In the other case, the sub-sequences $O_1$ and $O_2$ are generated separately. We do not expect any statistical dependence between $O_1$ and $O_2$ in this case. For each case, independent HMMs $\lambda_1$ and $\lambda_2$ are trained using $O_1$ and $O_2$ respectively and Product-HMM $\lambda_p$ is trained jointly using $O_1$ and $O_2$. Since the components of the vectors that make up $O, O_1$ and $O_2$ are independent of each other, a comparison of the transition matrix $A_p$ of the Product-HMM with the combination $(A_1, A_2)$, the transition matrices of $\lambda_1$ and $\lambda_2$, should reflect the statistical dependence of the sub-sequences. The method of combination of $A_1$ and $A_2$ to get a matrix $A_c$, and the method of comparison of $A_c$ with $A_p$ will be discussed.

4.1.1. Calculation of $A_c$ from $A_1$ and $A_2$

$\lambda_1$ and $\lambda_2$ are trained independently from $O_1$ and $O_2$. So, $A_1$ and $A_2$ are statistically independent. i.e, the transition of states in $O_1$ and $O_2$ are independent of each other. We want to calculate the transition matrix of the Product-HMM from $A_1$ and $A_2$. This calculated transition matrix is $A_c$. This is different from $A_p$ which is the transition matrix of the Product-HMM, obtained by joint training of $\lambda_p$. Refer Fig. 3. The elements of $A_1$ provide the transition probability information for the states of $\lambda_1$, i.e, $\{S_{11}, S_{12}...S_{1N_1}\}$. Similarly, the elements of $A_2$ provide the transition probability information for the states of $\lambda_2$, i.e, $\{S_{21}, S_{22}...S_{2N_2}\}$. To calculate the transition probability from $S_1$ to $S_2$ (which are states in the Product-HMM), we state that,

$$P(S_1, S_2) = P(S_{11}, S_{12}) * P(S_{21}, S_{22}) \quad (1)$$

eq. (1) shows the fact that we are considering the component transitions as statistically independent. The components of $A_c$ are calculated from the components of $A_1$ and $A_2$ using eq. (1).

4.1.2. Calculation of the distance between $A_c$ and $A_p$

$A_c$ is obtained by making the assumption of independence of transitions of $\lambda_1$ and $\lambda_2$. $A_p$ is obtained without any
such assumptions and through joint training. We propose the following measure \( D_{tr} \),

\[
D_{tr} = \sum_{i,j=1}^{N} (A_c(i,j) - A_p(i,j))^2
\]

(2)

\( D_{tr} \) is low if \( A_c \) is closer to \( A_p \), i.e., the independence assumption made for calculating \( A_c \) is valid. On the other hand, a high value of \( D_{tr} \) indicates the falseness of the independence assumption. So, \( D_{tr} \) provides a measure of statistical dependence of the transitions of \( O_1 \) and \( O_2 \). This can also be taken to mean that the Product-HMM is able to capture the statistical dependence of the state transitions in its transition matrix.

4.1.3. Results and Inference

For case 1, a vector sequence \( O \) is generated and split into two sub-sequences \( O_1 \) and \( O_2 \). \( \lambda_1, \lambda_2 \) and \( \lambda_p \) are trained with these sub-sequences. \( \lambda_1 \) and \( \lambda_2 \) are 2-state HMMs and \( \lambda_p \) is a 4-state HMM. \( A_1 \) and \( A_2 \) are used to calculate \( A_c \) using eq. (1). The dependence measure \( D_{tr} \) is calculated using eq. (2).

For case 2, two vector sequences \( O_1 \) and \( O_2 \) are generated separately. \( \lambda_1, \lambda_2 \) and \( \lambda_p \) are trained with these sub-sequences. \( \lambda_1 \) and \( \lambda_2 \) were 2-state HMMs and \( \lambda_p \) is a 4-state HMM. \( A_1 \) and \( A_2 \) are used to calculate \( A_c \) using eq. (1). The dependence measure \( D_{tr} \) is calculated using eq. (2).

From Table 1, we note that the dependence measure is higher in case 1 in which there is statistical dependence between transitions of \( \lambda_1 \) and \( \lambda_2 \), i.e., \( A_c \) is not close to \( A_p \) which indicates that we cannot calculate the Product-HMM transition matrix from the independent transition matrices and a joint training procedure is required to capture the statistical dependence between the two transition matrices. In case 2, where there is no statistical dependence between the 2 transition matrices, joint training of a Product-HMM is the same as independent training of the 2 separate HMMs. So, in case 1 where we split a vector sequence into its sub-sequences, possibility of transition dependence exists and it is better captured by jointly training a Product-HMM, rather than training two separate HMMs.

4.1.4. Mutual Information Interpretation of Experiment 1

\( A_1 \) and \( A_2 \) tell us the transition probabilities of the states of \( \lambda_1 \) and \( \lambda_2 \) respectively. \( A_p \) tells us the transition probabilities of the 2-component transitions. So, it is possible to calculate the mutual information that each transition from a state of \( \lambda_1 \) contains about another transition from a state of \( \lambda_2 \). We can calculate this mutual information for all the \( N_1N_2 \) combinations.

The mutual information \( I(X,Y) \) is a measure of dependence between the random variables \( X \) and \( Y \). The mutual information \( I(X,Y) \) (in bits) between two random variables \( X \) and \( Y \) with marginal distributions \( p(x) \) and \( p(y) \) respectively and joint distribution \( p(x,y) \) is given by,

\[
I(X,Y) = \sum_x \sum_y p(x,y) \log_2 \frac{p(x,y)}{p(x)p(y)}
\]

(3)

Each row in the transition matrix represents the distribution of the transitions to all the states from that state. Each row in the Product-HMM transition matrix represents the joint distribution of the 2-component transitions to all combinations of states from the individual HMMs, from that combination. So, we can calculate the mutual information between transitions in \( \lambda_1 \) and \( \lambda_2 \) using eq. (3).

For case 1 and case 2 of experiment 1, mutual information of the state transitions (one state from \( \lambda_1 \) and one from \( \lambda_2 \)) are calculated for all the 4 combinations of states. \( MI_1 \) is the mutual information vector for case 1 and \( MI_2 \) for case 2.

\[
MI_1 = [0.2451 1.0514 0.6861 0.0485] \quad \text{and} \quad MI_2 = [0.0001 0.0019 0.0062 0.0038].
\]

We see that the mutual information of state transitions of the 2 HMMs is more in case 1 and less in case 2. This shows that the Product-HMM is able to capture the dependence between the state transitions of the 2 HMMs. In the case of state transitions being independent, the mutual information will be zero and this is equivalent to saying \( A_c \) and \( A_p \) are same.

4.2. Experiment 2 - Product-HMM in finite training data cases

We focus on the estimation of the parameters of the emission densities in HMMs. We consider the number of samples available for estimation of the parameters of the emission densities. We know that the more the number of samples available for estimation the better will be the estimate. From a given number of training sequences \( M \), we have to estimate the parameters of \( N \) emission densities, where \( N \) is the number of stationary states in the HMM chosen to model the sequences. Suppose it is known a priori that the sub-sequences of the sequence have \( N_1 \) and \( N_2 \) number of stationary states. We will calculate the number of vectors per state available for training the regular HMM as well as the Product-HMM and show that it is more in the case of Product-HMM. This implies better estimates of the parameters of emission densities in Product-HMM. We will take an example and show that the estimates are better in the case of Product-HMM.

Refer Fig. 6. A given HMM \( \lambda_p \) generates 4 dimensional vector sequences of length 600. The 4 dimensional vector sequence \( O \) can be split into 2 sub-sequences \( O_1 \) and
Table 1. Details of Experiment 1

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Case 1</th>
<th>Case 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_1$</td>
<td>(0.7558 0.2442)</td>
<td>(0.8011 0.1989)</td>
</tr>
<tr>
<td></td>
<td>(0.3981 0.6019)</td>
<td>(0.2995 0.7005)</td>
</tr>
<tr>
<td>$A_2$</td>
<td>(0.7081 0.2919)</td>
<td>(0.1944 0.8056)</td>
</tr>
<tr>
<td></td>
<td>(0.5788 0.4212)</td>
<td>(0.6949 0.3051)</td>
</tr>
<tr>
<td>$A_c$</td>
<td>(0.5352 0.2206 0.1729 0.0713)</td>
<td>(0.1487 0.6530 0.0368 0.1615)</td>
</tr>
<tr>
<td></td>
<td>(0.4374 0.3184 0.1413 0.1029)</td>
<td>(0.5693 0.2324 0.1408 0.0575)</td>
</tr>
<tr>
<td></td>
<td>(0.2819 0.1162 0.4262 0.1757)</td>
<td>(0.0519 0.2278 0.1336 0.5867)</td>
</tr>
<tr>
<td></td>
<td>(0.2304 0.1677 0.3484 0.2535)</td>
<td>(0.1986 0.0811 0.5115 0.2088)</td>
</tr>
<tr>
<td>$A_p$</td>
<td>(0.8052 0.0978 0.0475 0.0495)</td>
<td>(0.1541 0.6490 0.0398 0.1571)</td>
</tr>
<tr>
<td></td>
<td>(0.0989 0.1096 0.6073 0.1842)</td>
<td>(0.5551 0.2454 0.1491 0.0504)</td>
</tr>
<tr>
<td></td>
<td>(0.1841 0.1000 0.1227 0.5932)</td>
<td>(0.0060 0.2220 0.1131 0.6043)</td>
</tr>
<tr>
<td></td>
<td>(0.2434 0.2532 0.2532 0.2502)</td>
<td>(0.1962 0.0808 0.5223 0.2007)</td>
</tr>
<tr>
<td>$D_{lr}$</td>
<td>0.7788</td>
<td>0.0016</td>
</tr>
</tbody>
</table>


$\mu_j$ of O2 State-j2: [22] 0  [44] 0  [66] 0  [88] 0  [1010] 0  [1212] 0

**Fig. 5.** Full vector and split vector stationary states for Experiment 2

**Fig. 6.** Overview of Experiment 2

$O_2$ consisting of 2 dimensional vectors. $O_1$, the first sub-sequence has 2 stationary states and $O_2$, the second sub-sequence has 6 stationary states. A regular HMM $\lambda_r$ would use 6 states to model the 4 dimensional vector sequence. Product-HMM $\lambda_p$ uses 2 states for $O_1$ and 6 states for $O_2$. The sequence $O$ is generated from a 6 state regular left-to-right HMM with single gaussian emission densities with the shown mean vectors and diagonal covariance matrices as shown in Figure 5. The sequence length is taken to be 600. For regular HMM, this means 100 vectors per state are available for initialization. For Product-HMM, 300 vectors per state are available for $O_1$ (2 states) and 100 vectors per state are available for $O_2$ (6 states). We see that Product-HMM, by offering optimal number of states to the sub-sequence models, enables better estimates of emission density parameters.
of the gaussian emission densities of segments.

erate 2 sub-sequences with different number of stationary mean vectors of emission densities are so chosen to generate sequences are generated from a 6 state left-to-right HMM whose covariance matrices of the gaussian emission densities by pairing properly. So, we have the following table of mean vectors of the states of regular HMM and regular HMM for comparison. From Table 2, we see that the first sub-sequence’s 2 mean vectors are better estimated because of the availability of more data. Refer Table. 2. 600 length, 4-dimensional vector sequences are generated from a 6 state left-to-right HMM whose mean vectors of emission densities are so chosen to generate 2 sub-sequences with different number of stationary segments. \( O_1 \), the first sub-sequence has 2 stationary states and \( O_2 \) has 6 stationary states. The covariance matrices of the gaussian emission densities of \( \lambda_g \) are chosen to be diagonal. A regular HMM \( \lambda_r \) having the same topology as \( \lambda_p \) is trained with the sequences generated from \( \lambda_g \). A Product-HMM \( \lambda_p \) which has only 2 states for the first subsequence \( O_1 \) and 6 states for the second sub-sequence \( O_2 \) is also trained with the same data. \( \lambda_p \) has totally 8 states whose 2-dimensional mean vectors are also shown in table. 2. We see that the 2 stationary states (with mean vectors [1 1] and [5 5]) are better estimated in \( \lambda_p \).

We saw in this section that Product-HMM can have better estimates of its emission density parameters than a regular HMM, because of its property of having optimal number of states for each sub-sequence. So, the sequence is better modelled by Product-HMM than a regular HMM. This will result in good recognition performance in speech recognition applications.

4.2.1. Results and Inferences

In the regular HMM, we needed to estimate six 4-dimensional emission densities. In the Product-HMM, eight 2-dimensional emission densities are estimated. From the eight 2-dimensional emission densities, we can 'form' a six 4-dimensional emission densities by pairing properly. So, we have the following table of mean vectors of the states of regular HMM and Product-HMM for comparison. From Table 2, we see that the first sub-sequence’s 2 mean vectors are better estimated because of the availability of more data.

Refer Table. 2. 600 length, 4-dimensional vector sequences are generated from a 6 state left-to-right HMM whose mean vectors of emission densities are so chosen to generate 2 sub-sequences with different number of stationary segments. \( O_1 \), the first sub-sequence has 2 stationary states and \( O_2 \) has 6 stationary states. The covariance matrices of the gaussian emission densities of \( \lambda_g \) are chosen to be diagonal. A regular HMM \( \lambda_r \) having the same topology as \( \lambda_p \) is trained with the sequences generated from \( \lambda_g \). A Product-HMM \( \lambda_p \) which has only 2 states for the first subsequence \( O_1 \) and 6 states for the second sub-sequence \( O_2 \) is also trained with the same data. \( \lambda_p \) has totally 8 states whose 2-dimensional mean vectors are also shown in table. 2. We see that the 2 stationary states (with mean vectors [1 1] and [5 5]) are better estimated in \( \lambda_p \).

We saw in this section that Product-HMM can have better estimates of its emission density parameters than a regular HMM, because of its property of having optimal number of states for each sub-sequence. So, the sequence is better modelled by Product-HMM than a regular HMM. This will result in good recognition performance in speech recognition applications.

5. CONCLUSIONS AND DIRECTIONS FOR FUTURE WORK

We conclude that Product-HMM offers a way of integrating 2 HMMs, which model a sub-sequence each, to provide a combined model for the complete vector sequence. The freedom of selecting an optimal HMM architecture for the sub-sequences and then the ability of combining them is the essence of Product-HMM. The generalization to a Product-HMM consisting of \( n \) sets of component states will be able to model \( n \) sequences jointly. Presently, the HMMs have emission densities which consider the components of the vector to be independent of each other. But, Product-HMM can capture the dependence between them in a natural way and this will make the model more accurate and in tune with the real nature of the sequences. Finally, the possibility of having one statistical model that can handle sequences of different statistical properties, can be used to study the statistical dependence between them.

6. REFERENCES


[7] Mukundh Nagarajan, Thesis for the M.Sc(Engg) degree in the department of ECE, IISc under the guidance of Dr. T. V. Sreenivas (Under preparation)