KL-Divergence Guided Two-Beam Viterbi Algorithm on Factorial HMMs

By

RAYMOND YEH

UNDERGRADUATE THESIS
Submitted in partial fulfillment of the requirements
for ECE496/499 in Electrical and Computer Engineering at the
University of Illinois at Urbana-Champaign, 2014

Urbana, Illinois

Adviser:

Professor Mark Hasegawa-Johnson
Abstract

This thesis addresses the problem of the high computation complexity issue that arises when decoding hidden Markov models (HMMs) with a large number of states. A novel approach, the two-beam Viterbi, with an extra forward beam, for decoding HMMs is implemented on a system that uses factorial HMM to simultaneously recognize a pair of isolated digits on one audio channel. The two-beam Viterbi algorithm uses KL-divergence and hierarchical clustering to reduce the overall decoding complexity. This novel approach achieves 60% less computation compared to the baseline algorithm, the Viterbi beam search, while maintaining 82.5% recognition accuracy.

Subject Keywords: factorial hidden Markov model, Viterbi beam, digit recognition.
# Table of Contents

1. **Introduction** .......................................................................................................................... 1  

2. **Background** ............................................................................................................................ 3  
   2.1 Factorial Hidden Markov Model & MIXMAX Approximation ........................................... 3  
      2.1.1 MIXMAX Approximation ............................................................................................... 3  
      2.1.2 Factorial Hidden Markov Model .................................................................................... 3  
      2.1.3 Output Probability Distribution ..................................................................................... 4  
   2.2 Viterbi Algorithm & Viterbi Beam ....................................................................................... 4  
      2.2.1 Viterbi Algorithm ........................................................................................................... 4  
      2.2.2 Viterbi Beam Search ...................................................................................................... 5  
   2.3 Hierarchical Clustering ......................................................................................................... 5  
   2.4 Kullback-Leibler Divergence (KL Divergence) .................................................................... 5  

3. **Two-beam Viterbi Algorithm** ............................................................................................... 7  
   3.1 Motivation and Main Ideas .................................................................................................. 7  
   3.2 Clustering HMM states ....................................................................................................... 7  
   3.3 Pre-compute during decoding ............................................................................................ 8  
   3.4 Variations of Viterbi Algorithm ......................................................................................... 8  
      3.4.1 Viterbi Algorithm ........................................................................................................... 8  
      3.4.2 Viterbi Beam Algorithm ................................................................................................ 8  
      3.4.3 Two-Beam Viterbi Algorithm ........................................................................................ 9  

4. **Experiments and Results** ...................................................................................................... 11  
   4.1 Experiment System ............................................................................................................... 11  
      4.1.1 Isolated Digit HMMs ..................................................................................................... 11  
      4.1.2 Simultaneous Digit Pair Recognition Baseline System ............................................. 12  
   4.2 Simultaneous Digit Pair Recognition with Viterbi Beam Result ....................................... 12  
   4.3 Simultaneous Digit Pair Recognition with Guided Viterbi Beam Search ......................... 13  
   4.4 Beam and Two-Beam Viterbi Results .................................................................................. 13  

5. **Discussion** ............................................................................................................................ 15  

6. **Future Work and Conclusion** .............................................................................................. 16  

References ....................................................................................................................................... 17
1. Introduction

Hidden Markov models, HMMs, have been used in a broad range of applications from automatic speech and handwriting recognition to financial economics. This model is particularly useful in modeling and capturing the temporal behavior of each application. This thesis will particularly focus on the Viterbi algorithm use to determine the most likely sequence of hidden states, the Viterbi path, of the HMMs in the context of automatic speech recognition.

The automatic speech recognition problem has been commonly solved with HMMs and the Viterbi algorithm [1]. By finding the most likely sequence of hidden states and mapping the sequence back into words, speech can be recognized. However, this method depends on the size and complexity of the HMMs. For example, in the context of large vocabulary speech recognition, traversing the entire HMM using the Viterbi algorithm may not be feasible due to long computation time [2]. A variety of decoding methods have been proposed to solve this running-time problem [2, 3, 4]. One common method used to solve this issue is the Viterbi beam search, or variations of Viterbi beam Search, which prunes paths that have low probability. This leads to more efficient running time but with a tradeoff between recognition accuracy and computing time/power [5, 6, 7].

This thesis explores a novel approach, which we call the “two-beam” Viterbi algorithm, to solve this running time issue in the context of a large-state trellis resulting from simultaneous recognition of a pair of isolated digits on one audio channel. The “two-beam” Viterbi utilizes pre-computed acoustic information, through clustering, to determine the more probable next states and prune the ones that are less probable.

The experiment of simultaneous digit pair recognition on one audio channel uses the factorial hidden Markov model (FHMM) and the MIXMAX approximation following the setup used in [8].

A FHMM mixes independent HMMs, causing the number of states to grow from O(n) to O(n^2). This fact is illustrated in Fig. 1.1 and Fig. 1.2, which show the resulting FHMM from mixing two left-to-right HMMs, each containing five states.
Fig. 1.1: Five states left-to-right HMM

Fig. 1.2: FHMM form by mixing two left-to-right five-state HMMs
2. Background

2.1 Factorial Hidden Markov Model & MIXMAX Approximation

The factorial hidden Markov model (FHMM) and MIXMAX approximation are used to model and recognition of simultaneous isolated digit utterance on one audio channel as done in [8].

2.1.1 MIXMAX Approximation

The MIXMAX approximation is formed on the observation that the Mel frequency spectral coefficient (MFSC) of a signal, consists of two additive signals in the time domain, can be approximate by the element-wise-maximum of the two signals’ log magnitude spectra [8]. Thus, consider the signal $Y(j\omega) = X(j\omega) + Z(j\omega)$, then the following approximation can be made

$$\log |Y(j\omega)| \approx \max(\log|X(j\omega)|, \log|Z(j\omega)|) \quad (2.1)$$

2.1.2 Factorial Hidden Markov Model

A FHMM can be interpreted as two separate HMMs evolving independently, each generating a cepstrum per frame, whose exponentiated transforms, are added together [8, 9, 10]. The observation pdf of the FHMM can be approximated using the MIXMAX approximation, [11] where the additive combination of two sound signals can be approximated with the element-wise-maximum of their log-magnitude spectra as explained in the previous section. The visualization of FHMM is shown in Fig. 2.1. Additionally, what is makes FHMM useful is that because the HMMs are independent, thus the training of a FHMM can be done independently, using the training method of a regular HMM. Furthermore, the FHMM can be converted into a single equivalent HMM using the following definition of the transition matrix.

![Fig. 2.1: FHMM illustration](image)
Given two HMM 1 with \( |P| \) states and HMM2 with \( |Q| \) states, the FHMM transition matrix of \( P \times Q \) states the FHMM transition matrix is defined as:

\[
a^{FHMM}_{(i,j)(k,l)} = a^{1}_{ik} \cdot a^{2}_{jl} \quad 1 \leq i, k \leq Q \\
1 \leq j, l \leq P
\]  

(2.2)

Where, \( a^m_{ij} \) corresponds to the \( i^{th}, j^{th} \) element in the transition matrix of the \( m^{th} \) HMM. As can be seen, each state in this equivalent HMM is indexed by a pair of state indices from the mixed HMMs.

2.1.3 Output Probability Distribution

Once we have this HMM topology, the output probability density, as derived in [8, 11], can be described as

\[
b_{lj}(y_t) = b^1_l(y_t) \int_{-\infty}^{y_t} b^2_j(z_t)dz_t + b^2_l(y_t) \int_{-\infty}^{y_t} b^1_j(x_t)dx_t
\]

(2.3)

where, \( b^k_l \) corresponds to the pdf of the \( k^{th} \) HMM at \( j^{th} \) state, which in our case is a single Gaussian distribution.

2.2 Viterbi Algorithm & Viterbi Beam

2.2.1 Viterbi Algorithm

The Viterbi algorithm is used to find the most likely sequence of hidden states sequences given a HMM and a sequence of observation vectors. Let, denote the sequence of observation as \( O \), and the sequence of states as \( Q \), and the model parameters as \( \lambda \). Then the most likely sequence can be denote as \( Q^* = \arg\max_{Q} P (O, Q|\lambda) \). The most naïve solution is simply iterated through all the possible \( Q \). However, this isn't a feasible method for any reasonable length of observation, and size of HMM. This is because; the number possible of \( Q \) is \((\text{Number of States})^{\text{Number of Time Steps}}\). Next, denote the state space as \( S \), and the number of time steps as \( T \), the running complexity of this approach is \( O(|S|^T) \). Thus, instead the Viterbi algorithm utilizes dynamic programming to reduce the running complexity to \( O(T \times |S|^2) \) [12]. However, as the number of states gets larger, the Viterbi algorithm again gets impractical to compute; For example in the context of large vocabulary speech recognition [2].
2.2.2 Viterbi Beam Search

The Viterbi beam search is a hill-climbing algorithm that attempts to find the most likely sequence of hidden states given a sequence of observations without traversing through all paths in the Viterbi lattice. This is done with pruning some of the paths during the search; by setting a minimum threshold on the likelihood at each time step during the Viterbi algorithm, or by keeping only certain percentage of nodes with the highest likelihood at each time step; this percentage is known as the "beam width". Note that this method doesn't guarantee finding the optimal path as the most promising sub-path isn't necessary contain in the final optimal path.

2.3 Hierarchical Clustering

Clustering is the task that partitions a set of object into groups such that the more similar objects are grouped together based on some distance metric. And hierarchical clustering merges of splits groups of objects using a greedy approach. The Agglomerative type of hierarchical clustering is the "bottom up" approach. The clustering starts with each object in an individual group, then the pair of groups that are most similar will now be union together to form a larger group then repeat the procedure [13]. The following is a pseudo code for the algorithm.

begin
  initialize c = n; n*; G_i = {o_i}; // i from 1 to n. o_i = objects, G_i = groups, n* = ideal cluster #
  while(c != n*)
    c = c-1;
    Find the most similar G_i and G_j pair;
    Union G_i and G_j;
  return G //G = the set of {G_i}
end

2.4 Kullback-Leibler Divergence (KL-Divergence)

As previously mentioned, when implementing a clustering algorithm the measure of "similarity" is necessary. This “similarity” has to be expressed as a distance metric for the objects. In this paper, we are clustering on distributions, and thus a distance metric that captures the similarity between distributions is the KL Divergence.

For distributions $P$ and $Q$ of continuous random variable, KL divergence is defined as the following
KL divergence measures the difference between two probability distributions. This measurement is non-symmetric, and captures the information lost when one distribution is used to approximate the other [14].

In our system clustering is done on multivariate Gaussians. The closed form of the KL divergence between multivariate Gaussian distributions of k dimension, with mean $\mu_p$, $\mu_q$, and their corresponding covariance matrix $\Sigma_p$, $\Sigma_q$ is as follows

$$D_{KL}(P||Q) = \int_{-\infty}^{\infty} \ln \left( \frac{p(x)}{q(x)} \right) p(x) \, dx$$

(2.4)

Additionally, the symmetric KL divergence can be express as

$$D_{KL \text{ symmetric}}(P||Q) = D_{KL}(P||Q) + D_{KL}(Q||P)$$

(2.6)

Symmetrized KL divergence is commonly used to cluster triphone states, e.g., [15].
3. Two-beam Viterbi Algorithm

The following sections introduce the two-beam Viterbi algorithm and its mathematical implementations.

3.1 Motivation and Main Ideas

The main motivation behind this two-beam Viterbi algorithm is to use a low complexity algorithm that prunes away, in advance, search paths expected to have a very low observation likelihood, which results in a lower overall time complexity during decoding. First, the HMM states are clustered, based on acoustic similarity. Each cluster is summarized by a representative pdf, which is computed in advance for every frame of the utterance. Finally, pruning at each time-step, leaving only the paths within the beam-width at each time step. The motivation behind this is the by introducing a time complexity of $O(CT)$, we can reduce the quadric complexity of $|S|^2T$, where $C$ stands for number clusters, $T$ for number of time steps and $S$ for the state space. For a fixed width of the forward beam, on average we retain clusters containing $n$ states retained per frame. The total complexity is reduced from $|S|^2T$ to $n^2T + CT$, which is an improvement in complexity provided that

$$|S|^2 - (n^2 + C) > 0$$

(3.1)

3.2 Clustering HMM states

Using the agglomerative hierarchical clustering and symmetric KL-divergence as the distance metric, each state is assigned to a cluster. We used was the mean linkage as the linkage criteria, where each cluster is averaged into a single normal distribution; Each cluster is averaged into one normal distribution with the mean $\bar{X}_{tC}$ and covariance $\Sigma_{tC}$ where

$$\bar{X}_{tC} = \frac{1}{|C|} \sum_{S_k \in C_t} X_{S_k}$$

(3.2)

$$\Sigma_{tC} = \frac{1}{|C|^2} \sum_{S_k \in C_t} Cov(X_{S_k})$$

(3.3)

where Eq. (3.3) generates a concentrated distribution centered on the cluster mean, for the purpose of improved trellis pruning.

All the clusters information was saved during the clustering process, in order to empirically determine the optimal number of cluster to be used during decoding.
3.3 Pre-compute during decoding

To use the cluster information during decoding, first we have to pre-compute the output probability for each cluster, using its cluster mean and variance, for each time step. Thus output will be a matrix $V$ of size $C \times T$, where $V_{i,j}$ corresponds to the $i^{th}$ cluster’s output probability given $j^{th}$ observation. Then sorting the cluster in order of descending output probability, the decoding will only keep certain percentage, beam width, of the cluster with the highest probability. The rest of the clusters are pruned prior to the decoding process.

3.4 Variations of Viterbi Algorithm

3.4.1 Viterbi Algorithm

Given a HMM with state space $S$, initial probabilities $\pi_i$, and the transition matrix $A$, where $A(i,j)$ is the transition probability from state $i$ to state $j$. Also given an observation sequence $O_T = o_1 ... o_T$. Then the Viterbi algorithm can be written recursively as follows

For each $k \in S$ compute

$$\delta_{1,k} = P(o_1|s_k) \cdot \pi_k$$

$$\delta_{t,k} = \max_{x \in S} (P(o_t|s_k) \cdot A(x,k) \cdot \delta_{t-1,x})$$

where $\delta_{t,k}$ is the probability given the most likely state sequence at observation $o_t$ and state $s_k$. Fig. 3.1 illustrates the computation of the algorithm.

3.4.2 Viterbi Beam Algorithm

Similarly, the Viterbi Beam algorithm can be written recursively as follows

For each $k \in S$ compute

$$\delta_{1,k} = P(o_1|s_k) \cdot \pi_k$$

$$\delta_{t,k} = \max_{x \in s_{beam,t}} (P(o_t|s_k) \cdot A(x,k) \cdot \delta_{t-1,x})$$

where $\delta_{t,k}$ is the probability given the most likely state sequence at observation $o_t$ and state $s_k$. And $s_{beam,t} = \{s_k | \delta_{t-1,k} \text{ is within the beam width}\}$ Fig. 3.2 illustrates the computation of the algorithm.
3.4.3 Two-Beam Viterbi Algorithm

Finally, the two-beam Viterbi algorithm can be written recursively as

For each \( k \in S_{\text{beam},t} \) compute

\[
\delta_{1,k} = P(o_1 | s_k) \cdot \pi_k \tag{3.8}
\]

\[
\delta_{t,k} = \max_x \in S_{\text{beam},t} (P(o_t | s_k) \cdot A(x, k) \cdot \delta_{t-1,x}) \tag{3.9}
\]

where,

\[
S_{\text{beam},t} = \{s_k | \delta_{t-1,k} \text{ is within beam width}\} \tag{3.10}
\]

\[
S_{\text{beam},t} = \{s_k | s_k \in C_i : V_{i,t} \text{ is within forward beam}\} \tag{3.11}
\]

where \( V_{i,j} \) is defined to be the \( i^{th} \) cluster’s output probability given the \( j^{th} \) observation.

Next, Fig 3.3 demonstrates the two-beam Viterbi algorithm. It is a standard Viterbi trellis, where each node corresponds to a state at a particular time, and each link represents the transition from a state to the next instant of time. For a Viterbi beam algorithm, Fig 3.2, at every time instant, each state should have a link to its previous more probable state. However, this is not the case with the two-beam Viterbi. This is because using the clustering information, at each time instant only the more probable next states within beamF are considered. From Fig. 3.1 - 3.3, it can observe that there is a reduction in the amount of computation between the full Viterbi, Viterbi Beam, and the two-beam Viterbi algorithm.

![Viterbi trellis illustration of full Viterbi algorithm](image-url)
Fig. 3.2: Viterbi trellis illustration of Viterbi beam algorithm

Fig. 3.3: Viterbi trellis illustration of two-beam Viterbi algorithm
4. Experiments and Results

4.1 Experiment System

Figure 4.1 is the overall experiment system block diagram from training to testing. Overall, the system is mainly written in Matlab. The HTK Toolbox was used to extract features and to train the isolated digit HMMs [16]. Then the trained HMMs were imported into Matlab and used as the bases of the FHMM.

4.1.1 Isolated Digit HMMs

The isolated digits system consists of 12 individual HMMs from zero to nine, including “oh” and a silence model. Each isolated digit HMM was trained on 50 boy and 50 girl speakers, each with two utterances, from the children’s speech portion of the TIDIGITS speech corpus.

The system was trained using HTK; the individual HMMs were initialized using a standard 12 states left-to-right model, each with a single Gaussian per state [16]. For the feature extraction, the observation sequence was a time series of 12 MFCCs, delta1 and delta2
concatenated vectors, with window size of 25 ms, frame period of 10 ms, and Hamming window.

The isolated digit HMMs were tested over 25 boy and 25 girl speakers different from the training speakers, using two utterances of each digit per test speaker. The isolated digit HMMs achieved 100% recognition accuracy when tested on this small isolated digit test corpus. And also achieved 93.59% on continuous digit recognition, again using the children test data set from TIDIGITS corpus.

4.1.2 Simultaneous Digit Pair Recognition Baseline System

As described in the background, FHMM for double digit recognition can be constructed by mixing two single digits HMMs. Using the isolated digit HMMs described above, the FHMM is mixed, and then converted to an equivalent HMM, where the traditional Viterbi algorithm on a regular HMM can be used, following the configuration of [8]. This is used as the baseline system when evaluating the performance of the Viterbi Beam algorithm, and our novel approach of the guided two-beam Viterbi algorithm, and the Viterbi beam algorithm.

Next, we follow the same evaluation method as in [8], where a “complete success” (CS) is successfully recognizing digits, and a ‘partial success, partial failure’ (PSPF) is recognition of one digit of the pair.

The recognition rate was computed by

\[
\text{Recognition Accuracy} \, (\%) = \frac{CS + 0.5PSPF}{N}
\]  

(4.1)

where \(N\) is the number of test cases.

The baseline system was decoded using a full Viterbi Search algorithm; the recognition accuracy was 83% and “Complete success” was achieved on 70% of 200 test cases mixed from the test speakers.

4.2 Simultaneous Digit Pair Recognition with Viterbi Beam Result

<table>
<thead>
<tr>
<th>% of computation</th>
<th>1%</th>
<th>1.5%</th>
<th>2%</th>
<th>3%</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS/N</td>
<td>35%</td>
<td>55%</td>
<td>70%</td>
<td>70%</td>
</tr>
<tr>
<td>Recognition</td>
<td>67.5%</td>
<td>75%</td>
<td>82.5%</td>
<td>82.5%</td>
</tr>
<tr>
<td>Accuracy</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Percent of computation is the percentage of the state transition computed of a full Viterbi search; in this case it is equal to the beam width.

### 4.3 Simultaneous Digit Pair Recognition with Guided Viterbi Beam Search

The percent of computation is the percentage of the state transition computed of a full Viterbi search; this includes the pre-compute transitions. In this case it is approximately equal to the product of beam width and forward beam width, as the pre-compute computations were always averaged into the percentage. The number of cluster was empirically determined to be 40 by sweeping different number of clusters for the experiment.

#### Table 2. Two-Beam Viterbi Recognition Result

<table>
<thead>
<tr>
<th>% of computation</th>
<th>0.4%</th>
<th>0.8%</th>
<th>1.2%</th>
<th>1.6%</th>
<th>2%</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS/N</td>
<td>55%</td>
<td>60%</td>
<td>60%</td>
<td>65%</td>
<td>70%</td>
</tr>
<tr>
<td>Recognition</td>
<td>75%</td>
<td>77.5%</td>
<td>77.5%</td>
<td>80%</td>
<td>82.5%</td>
</tr>
<tr>
<td>Accuracy</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The percent of computation is the percentage of the state transition computed of a full Viterbi search; this includes the pre-compute transitions. In this case it is approximately equal to the product of beam width and forward beam width, as the pre-compute computations were always averaged into the percentage. The number of cluster was empirically determined to be 40 by sweeping different number of clusters for the experiment.

**Figure 4.2: Recognition Accuracy for Two-Beam Viterbi vs. Viterbi Beam**

**Figure 4.3: Complete Success Rate for two-beam vs. Viterbi Beam**

#### 4.4 Beam and Two-Beam Viterbi Results

Figure 4.2, 4.3 and table 1, 2 contrasts the experiment results decoded with the Viterbi beam and the two-beam Viterbi algorithm; Fig. 4.2 shows recognition accuracy, while Fig. 4.3 shows complete success rate. For the Viterbi beam, percent of computation is the
computed percentage of the state transitions of a full Viterbi search; in this case it is equal to the beam width.

For the two-beam Viterbi, the percent of computation is again the computed percentage of the state transitions of a full Viterbi search, therefore the percentage is approximately equal to the product of beam width and forward beam width. The number of clusters was set to 40 based on the results of preliminary experiments.

As can be seen in Fig. 4.2 and 4.3, the accuracies of both algorithms converge to the accuracy of the full Viterbi search at 2% of computation. Also, as the percent of computation decreases, both \( \frac{CS}{N} \), and recognition accuracy fall for both algorithms, while the Viterbi beam has a steeper decay than the two-beam Viterbi.
5. Discussion

From the results, we can observe that the Viterbi beam and the two-beam Viterbi converge to the result of full Viterbi at 2% of computation. There is a trade-off between accuracy and the percentage of computation, as expected.

The recognition accuracy and $\frac{CS}{N}$ results from the Viterbi and two-beam Viterbi are plotted in Fig. 4.2 and 4.3, where we can observe that the drop in accuracy from the decrease in the percentage of computation is less for the two-beam Viterbi than that of the Viterbi beam. The two-beam Viterbi still obtained 75% accuracy with 0.4% of computation; the Viterbi beam’s recognition accuracy already drops to 67.5% at 1% of computation.

These results demonstrate the success and potential of the two-beam Viterbi algorithm for faster computation, compared to the Viterbi beam search, during the HMM decoding process. Furthermore, the two-beam Viterbi algorithm breaks the structure limitation of the Viterbi beam search on the lowest percentage of computation. For example, even if the beam width is chosen so that at each time step only the “most probable” state is extended and the rest are pruned, nevertheless the Viterbi beam Search will still have to perform $|S| \times T$ computations overall. However, with the two-beam Viterbi, it is possible to have the beam widths chosen so the minimum computation is as low as $(C + 1) \times T$, where $C$ is the number of clusters, and $1 \leq C + 1 \leq |S|$. Thus the two-beam Viterbi by the structure of the algorithm has the potential to prune more paths and lead to faster computation.
6. Future Work and Conclusion

In this thesis we present a novel approach, the two-beam Viterbi algorithm, to decode HMMs. The two-beam Viterbi was able to maintain the same recognition rate as the Viterbi beam-search baseline, with only 40% of the computation cost. The significance of this algorithm is that the structure of the two-beam Viterbi allows the possibility of lower computation complexity than with the Viterbi beam. We have demonstrated the potential of the two-beam Viterbi algorithm on the single channel simultaneous digit pair recognition task, and expect to expand this algorithm to other tasks, including large vocabulary automatic speech recognition.
References


