SPIRAL: Efficient and Exact Model Identification for Hidden Markov Models

Yasuhiro Fujiwara (NTT Cyber Space Labs)
Yasushi Sakurai (NTT Communication Science Labs)
Masashi Yamamuro (NTT Cyber Space Labs)

Speaker: Yasushi Sakurai
Motivation

- **HMM (Hidden Markov Model)**
  - Mental task classification
    - Understand human brain functions with EEG signals
  - Biological analysis
    - Predict organisms functions with DNA sequences
  - Many other applications
    - Speech recognition, image processing, etc

- **Goal**
  - Fast and exact identification of the highest-likelihood model for large datasets
Mini-introduction to HMM

• Observation sequence $X = (x_1, x_2, \cdots, x_n)$ is a probabilistic function of states

• Consists of the three sets of parameters:
  - Initial state probability: $\pi = \{\pi_i\} \ (1 \leq i \leq m)$
    - State $u_i$ at time $t = 1$
  - State transition probability: $a = \{a_{ij}\} \ (1 \leq i, j \leq m)$
    - Transition from state $u_i$ to $u_j$
  - Symbol probability: $b(v) = \{b_i(v)\} \ (1 \leq i \leq m)$
    - Output symbol $v$ in state $u_i$
Mini-introduction to HMM

• HMM types
  – Ergodic HMM
    • Every state can be reached from every other state
  – Left-right HMM
    • Transitions to lower number states are prohibited
    • Always begin with the first state
    • Transition are limited to a small number of states

Ergodic HMM

Left-right HMM
• Viterbi path in the trellis structure
  – Trellis structure: states lie on the vertical axis, the sequence is aligned along the horizontal axis
  – Viterbi path: state sequence which gives the likelihood
Mini-introduction to HMM

- Viterbi algorithm
  - Dynamic programming approach
  - Maximize the probabilities from the previous states

\[
P = \max_{1 \leq i \leq m} (p_{in})
\]

\[
p_{it} = \begin{cases} 
\max_{1 \leq j \leq m} (p_{j(t-1)} \cdot a_{ji}) \cdot b_{i}(x_t) & (2 \leq t \leq n) \\
\pi_i \cdot b_{i}(x_1) & (t = 1)
\end{cases}
\]

\(p_{it}\) : the maximum probability of state \(u_i\) at time \(t\)
Problem Definition

• Given
  – HMM dataset
  – Sequence $X = (x_1, x_2, \ldots, x_n)$ of arbitrary length

• Find
  – Highest-likelihood model, estimated with respect to $X$, from the dataset
Why not ‘Naive’

• Naïve solution
  1. Compute the likelihood for every model using the Viterbi algorithm
  2. Then choose the highest-likelihood model

But..

– High search cost: $O(nm^2)$ time for every model
  • Prohibitive for large HMM datasets

$m$: # of states
$n$: sequence length of $X$
Our Solution, SPIRAL

• Requirements:
  – High-speed search
    • Identify the model efficiently
  – Exactness
    • Accuracy is not sacrificed
  – No restriction on model type
    • Achieve high search performance for any type of models
Likelihood Approximation

Reminder: Naive
Likelihood Approximation

• Create compact models (reduce the model size)
  – For given $m$ states and granularity $g$,
  – Create $m/g$ states by merging ‘similar’ states
Likelihood Approximation

- Use the vector $F_i$ of state $u_i$ for clustering:
  \[ F_i = (\pi_i; a_{i1}, \ldots, a_{im}, a_{1i}, \ldots, a_{mi}; b_i(v_1), \ldots, b_i(v_s)) \]
  \( s: \) number of symbols

- Merge all the states $u_i$ in a cluster $C$ and create a new state $u_C$

- Choose the highest probability among the probabilities of $u_i$
  \[ \pi'_C = \max_{u_i \in C} (\pi_i) \quad a'_{Cj} = \max_{u_i \in C, u_j \not\in C} (a_{ij}) \]
  \[ a'_{CC} = \max_{u_i, u_k \in C} (a_{ik}) \quad a'_C = \max_{u_i \in C, u_j \not\in C} (a_{ji}) \quad b'_C(v) = \max_{u_i \in C} (b_i(v)) \]
  Obtain the upper bounding likelihood
Likelihood Approximation

• Compute approximate likelihood $P'$ from the compact model

$$P' = \max_{1 \leq i \leq m'}(p'_{in})$$

$$p'_{it} = \begin{cases} 
\max_{1 \leq j \leq m'}(p'_{j(t-1)} \cdot a'_{ji}) \cdot b'_i(x_t) & (2 \leq t \leq n) \\
p'_i \cdot b'_i(x_1) & (t = 1)
\end{cases}$$

$p'_{it}$: maximum probability of states

• Upper bounding likelihood
  
  – For approximate likelihood $P'$, $P' \geq P$ holds
  
  – Exploit this property to guarantee exactness in search processing
Likelihood Approximation

Advantages

• The best model can not be pruned
  – The approximation gives the upper bounding likelihood of the original model

• Support any model type
  – Any probabilistic constraint is not applied to the approximation
Multi-granularities

• The likelihood approximation has the trade-off between accuracy and computation time
  – As the model size increases, accuracy improves
  – But the likelihood computation cost increases

• Q: How to choose granularity $g$?
Multi-granularities

• The likelihood approximation has the trade-off between accuracy and computation time
  – As the model size increases, accuracy improves
  – But the likelihood computation cost increases

• Q: How to choose granularity $g$ ?
• A: Use multiple granularities
  – $h+1 (h = \lfloor \log_2 m \rfloor)$ distinct granularities that form a geometric progression $g_i = 2^i$ ($i=0,1,2,\ldots,h$)
  – Geometrically increase the model size
Multi-granularities

- Compute the approximate likelihood $P'$ from the coarsest model as the first step
  - Coarsest model has $\left\lfloor \frac{m}{2^n} \right\rfloor (=1)$ states
- Prune the model if $P' < \theta$, otherwise

$\theta$ : threshold
Multi-granularities

• Compute the approximate likelihood $P'$ from the second coarsest model
  - Second coarsest model has $\left\lfloor m/2^{h-1} \right\rfloor$ states
• Prune the model if $P' < \theta$
Multi-granularities

- Threshold $\theta$
  - Exploit the fact that we have found a good model of high likelihood
    - $\theta$: exact likelihood of the best-so-far candidate during search processing
    - $\theta$ is updated and increases when promising model is found
    - Use $\theta$ for model pruning
Multi-granularities

• Compute the approximate likelihood $P'$ from the second coarsest model
  – Second coarsest model has $\left\lfloor m/2^{h-1} \right\rfloor$ states
• Prune the model if $P' < \theta$, otherwise
  – $\theta$: exact likelihood of the best-so-far candidate

\[
\sum_{i=1}^{m} \frac{P(q^i)}{}\]
Multi-granularities

• Compute the likelihood $P'$ from more accurate model
• Prune the model if $P' < \theta$
Multi-granularities

- Repeat until the finest granularity (the original model)
- Update the answer candidate and best-so-far likelihood if $P \geq \theta$
Multi-granularities

- Optimize the trade-off between accuracy and computation time
  - Low-likelihood models are pruned by coarse-grained models
  - Fine-grained approximation is applied only to high-likelihood models
- Efficiently find the best model for a large dataset
  - The exact likelihood computations are limited to the minimum number of necessary
Transition Pruning

- Trellis structure has too many transitions
- Q: How to exclude unlikely paths
Transition Pruning

- Trellis structure has too many transitions
- Q: How to exclude unlikely paths
- A: Use the two properties
  - Likelihood is monotone non-increasing (likelihood computation)
  - Threshold is monotone non-decreasing (search processing)
Transition Pruning

- In likelihood computation, compute the estimate \( e_{it} \):

\[
e_{it} = \begin{cases} 
p_{it} \cdot (a_{\text{max}})^{n-t} \cdot \prod_{j=t+1}^{n} b_{\text{max}}(x_j) & (1 \leq t \leq n-1) 
p_{in} & (t = n) \end{cases}
\]

where \( a_{\text{max}} = \max_{1 \leq i, j \leq m}(a_{ij}) \), \( b_{\text{max}}(v) = \max_{1 \leq i \leq m} b_i(v) \)

- \( e_{it} \) : conservative estimate of the likelihood \( p_{it} \) of state \( u_i \) at time \( t \)

- If \( e_{it} < \theta \), prune all paths that pass through \( u_i \) at \( t \)
  - \( \theta \) : exact likelihood of the best-so-far candidate
Transition Pruning

- Terminate the likelihood computation if all the paths are excluded
- Efficient especially for long sequences
- Applicable to approximate likelihood computation
SPIRAL needs the same order of memory space, while can be up to $m^2$ times faster

<table>
<thead>
<tr>
<th>Accuracy</th>
<th>Complexity</th>
<th>Memory Space</th>
<th>Computation time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viterbi</td>
<td>Guarantee exactness</td>
<td>$O(m^2 + ms)$</td>
<td>$O(nm^2)$</td>
</tr>
<tr>
<td>SPIRAL</td>
<td></td>
<td></td>
<td>At least $O(n)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>At most $O(nm^2)$</td>
</tr>
</tbody>
</table>
Experimental Evaluation

• Setup
  – Intel Core 2 1.66GHz, 2GB memory

• Datasets
  – EEG, Chromosome, Traffic

• Evaluation
  – Mainly computation time
  – Ergodic HMM
  – Compared the Viterbi algorithm and Beam search
    • Beam search: popular technique, but does not guarantee exactness
Experimental Evaluation

- Evaluation
  - Wall clock time versus number of states
  - Wall clock time versus number of models
  - Effect of likelihood approximation
  - Effect of transition pruning
  - SPIRAL vs Beam search
Experimental Evaluation

- Wall clock time versus number of states
  - *EEG*: up to 200 times faster
Experimental Evaluation

- Wall clock time versus number of states
  - *Chromosome*: up to 150 times faster
Experimental Evaluation

- Wall clock time versus number of states
  - Traffic: up to 500 times faster
Experimental Evaluation

• Evaluation
  – Wall clock time versus number of states
  – Wall clock time versus number of models
  – Effect of likelihood approximation
  – Effect of transition pruning
  – SPIRAL vs Beam search
Experimental Evaluation

- Wall clock time versus number of models
  - EEG: up to 200 times faster
Experimental Evaluation

• Evaluation
  – Wall clock time versus number of states
  – Wall clock time versus number of models
  – Effect of likelihood approximation
  – Effect of transition pruning
  – SPIRAL vs Beam search
Experimental Evaluation

- Effect of likelihood approximation
  - Most of models are pruned by coarser approximations
Experimental Evaluation

- Evaluation
  - Wall clock time versus number of states
  - Wall clock time versus number of models
  - Effect of likelihood approximation
  - Effect of transition pruning
  - SPIRAL vs Beam search
Experimental Evaluation

- Effect of transition pruning
  - SPIRAL find the highest-likelihood model more efficiently by transition pruning

![Bar chart showing wall clock time for different categories: EEG, Chromosome, Traffic. SPIRAL and SPIRAL without transition pruning are compared.}

39
Experimental Evaluation

• Evaluation
  – Wall clock time versus number of states
  – Wall clock time versus number of models
  – Effect of likelihood approximation
  – Effect of transition pruning
  – SPIRAL vs Beam search
Experimental Evaluation

• SPIRAL vs Beam search
  – SPIRAL is significantly faster while it guarantees exactness

![Likelihood error ratio](image1)

![Wall clock time](image2)

Likelihood error ratio
Note: SPIRAL gives no error

Wall clock time
SPIRAL is up to 27 times faster
Conclusion

• Design goals:
  – High-speed search
    • SPIRAL is significantly (up to 500 times) faster
  – Exactness
    • We prove that it guarantees exactness
  – No restriction on model type
    • It can handle any HMM model type

• SPIRAL achieves all the goals