Making Gestural Input from Arm-Worn Inertial Sensors More Practical

Louis Kratz1,2, T. Scott Saponas1, Dan Morris1
lak24@drexel.edu ssaponas@microsoft.com dan@microsoft.com

1Microsoft Research, Redmond, WA
2Department of Computer Science, Drexel University, Philadelphia, PA

CONSTANT TIME COMPLEXITY

Eq. 10 in our main submission has complexity of $O(N^2D)$ due to the Forwards-Backwards procedure used to compute $p(O_{t_0+1:t+1})$ and $p(O_{t_m:t+1})$ at each time instance. We may efficiently compute $p(O_{t_0+1:t+1})$ and $p(O_{t_m:t+1})$ using values at the previous time instance. Specifically, we use the information at time $t$ to quickly compute the values at time $t+1$. Both values in the update step are formulated as the same problem: given a sequence $O_{t_w:t}$ of a fixed duration $(t-t_w)$ and the sequence’s likelihood $p(O_{t_w:t} | \lambda)$, compute the likelihood of the next subsequence $O_{t_w+1:t+1}$ without re-evaluating the entire sequence.

Review of the Forwards-Backwards Algorithm

The forwards backwards algorithm computes the likelihood of being in state $j$ and observing the sequence $O_{t_w:t}$ by

$$\alpha_t(j) = p(z_t = j, O_{t_w:t} | \lambda), \quad (0.1)$$

where $z_t$ is the value of the hidden state at time $t$, $\lambda$ is the HMM, and $\alpha$ is the forwards message from the forwards backwards algorithm.

The likelihood of the observation sequence given the model is computed by marginalization:

$$p(O_{t_w:t} | \lambda) = \sum_{j=1}^{N} p(z_t = j, O_{t_w:t} | \lambda) = \sum_{j=1}^{N} \alpha_t(j), \quad (0.2)$$

where $N$ is the number of states in the HMM.

The value of $\alpha_t(j)$ is computed sequentially by

$$\alpha_t(j) = \left[ \sum_{i=1}^{N} a_{ij} \alpha_{t-1}(i) \right] b_j(\alpha_t) , \quad (0.3)$$

where $a_{ij}$ is the HMM’s transition probability, and $b_j(\alpha_t)$ its emission density. The initialization is

$$\alpha_{t_0}(j) = \pi_j b_j(\alpha_{t_0}) , \quad (0.4)$$

where $\pi_j$ is the initial state likelihood.

Matrix Representation

First, we formulate the forwards step as a matrix multiplication. Let $\alpha_t$ be an $N \times 1$ vector. Also, let

$$B_t = \begin{bmatrix} b_1(\alpha_t) & 0 & \ldots & 0 \\ 0 & b_2(\alpha_t) & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & b_N(\alpha_t) \end{bmatrix} \quad (0.5)$$

We can re-write Eq. 0.3 as

$$\alpha_t = B_t \pi , \quad (0.7)$$

and the initialization in Eq. 0.4 as

$$\alpha_{t_0} = B_{t_0} \pi \quad (0.6)$$

Exploiting Redundant Computations

To store redundant computations in the forwards procedure, we define a matrix

$$G_{t_w:t} = A^T B_{t-1} \ldots A^T B_{t_w} . \quad (0.8)$$

Note that $\alpha_t$ (and subsequently $p(O_{t_w:t} | \lambda)$) may be computed from $G_{t_w:t}$ by

$$\alpha_t = B_t G_{t_w:t} \pi . \quad (0.9)$$

At time $t+1$, we compute the next matrix $G_{t_w+1:t+1}$ from $G_{t_w:t}$ by

$$G_{t_w+1:t+1} = A^T B_t G_{t_w:t} B_{t_w+1}^{-1} [A^T]^{-1} . \quad (10.10)$$

Note that $[A^T]^{-1}$ is fixed and may be pre-computed, and computing $B_{t_w+1}$ is trivial since $B_{t_{w+1}}$ is diagonal.

To summarize, we use Eq. 0.10 to update a matrix that we use to compute the likelihood via Eq. 0.9. The update step in Eq. 0.10 is only as computationally complex as matrix multiplication (naive algorithm $O(N^3)$). We use this algorithm to compute $p(O_{t_0+1:t+1})$ and $p(O_{t_m:t+1})$ in Eq. 10 of our main submission. As a result, at each time instance $t$ the complexity of the update step in Eq. 10 of our main submission is $O(N^3)$, constant with respect to the maximum duration $D$ of the gesture.

Numeric Stability

In practice, Eq. 0.10 may be numerically unstable if $B_{t_w}$ or $A^T$ are not invertible or poorly conditioned. Such conditions, however, are easy to identify. Our implementation identifies such numeric issues and uses the linear-time algorithm
on such occasions. Numerical issues may also be reduced by restricting all emission likelihoods to be nonzero (i.e., forcing $B_{tw}$ to be invertible) or scaling $B_{tw}$ with a procedure similar to the scaled Forwards-Backwards algorithm [1].

REFERENCES