Viterbi path and training with (Bayesian) HMM

Jüri Lember

University of Tartu, Estonia

17.09.2021

EM and MM algorithm

Let $p(x, y|\theta)$ be a latent variable model. Typically:

$x = x_{1:n} = (x_1, \ldots, x_n)$ – observations, $y = y_{1:n}$ – latent variables, θ – parameters.

EM-algorithm (parameter, classic) ($\sum_y$ when y discrete)

$$\theta^{(i+1)} = \arg\max_{\theta} \left[\sum_y \ln p(y, x|\theta) p(y|\theta^{(i)}, x) \right].$$

increases likelihood

$$p(x|\theta^{(i+1)}) \geq p(x|\theta^{(i)}).$$

EM and MM algorithm

EM-algorithm (parameter, Bayes) $\pi(\theta)$ prior (density), so

$$\frac{p(x,y|\theta)\pi(\theta)}{p(x)} = p(y,\theta|x),$$

$$\begin{aligned}\theta^{(i+1)} &= \arg\max_\theta \Big[\sum_y \ln p(y,\theta|x)p(y|\theta^{(i)},x)\Big] \\ &= \arg\max_\theta \Big[\sum_y \ln p(y,x|\theta)p(y|\theta^{(i)},x) + \ln \pi(\theta)\Big].\end{aligned}$$

increases posterior

$$p(\theta^{(i+1)}|x) \geq p(\theta^{(i)}|x).$$

MM-algorithm (classic)

Replace the weighted sum $\sum_y \ln p(y, x|\theta) p(y|\theta^{(i)}, x)$ by the element with maximal weight.

$$y^{(i)} = \arg\max_y p(y, x|\theta^{(i)}) = \arg\max_y p(y|\theta^{(i)}, x)$$
$$\theta^{(i+1)} = \arg\max_\theta p(y^{(i)}, x|\theta)$$

$$\theta^{(i)} \to y^{(i)} \to \theta^{(i+1)} \to y^{(i+1)} \to \cdots (\hat{y}, \hat{\theta})$$

increases joint likelihood

$$p(y^{(i+1)}, x|\theta^{(i+1)}) \geq p(y^{(i)}, x|\theta^{(i+1)}) \geq p(y^{(i)}, x|\theta^{(i)}).$$

MM-algorithm (Bayes)

$$y^{(i)} = \arg\max_y p(y, \theta^{(i)}|x) = \arg\max_y p(y, x|\theta^{(i)}) = \arg\max_y p(y|\theta^{(i)}, x)$$
$$\theta^{(i+1)} = \arg\max_\theta p(y^{(i)}, \theta|x) = \arg\max_\theta \left[\ln p(y^{(i)}, x|\theta) + \ln \pi(\theta) \right]$$

increases joint posterior

$$p(y^{(i+1)}, \theta^{(i+1)}|x) \geq p(y^{(i)}, \theta^{(i+1)}|x) \geq p(y^{(i)}, \theta^{(i)}|x).$$

When the goal is parameter estimation, then $\hat{\theta}$ is taken as output.

HMM-like models

Mixture model: $Y_1, \ldots, Y_n$ iid random variables with values $\mathcal{Y} = \{0, \ldots, K-1\}$, and given $Y_t = k$, the $\mathbb{R}^d$-valued observation X_t is emitted independently of everything else from distribution with density $f(\cdot|\theta_k)$, $k \in \mathcal{Y}$.

Hidden Markov model (HMM): the same, but $Y_1, \ldots, Y_n$ is a (homogenuous) Markov chain; mixture model is a special case.

Markov switching model: $Y_1, \ldots, Y_n$ is a (homogenuous) Markov chain, but the conditional distribution of X_t depends also on X_{t-1} : $X_t|Y_t = k, X_{t-1} = x_{t-1}$ has density $f(\cdot|\theta_k, x_{t-1})$. Given $Y_1, \ldots, Y_n$ the observations $X_1, \ldots, X_n$ are not (conditionally) independent any more; HMM is a special case.

In all those models the parameters of Y, typically the transition matrix $\mathbb{P}$ and/or initial probabilities – transition parameters; the parameters θ_i $(i \in \mathcal{Y})$ – emission parameters.

HMM-like models

Pairwise Markov model (PMM): $Z_1, \ldots, Z_n$, with $Z_t = (X_t, Y_t)$ is a (homogenuous) Markov chain with state space $\mathbb{R}^d \times \mathcal{Y}$. Now Y-process might or might not be a Markov chain. Markov switching models is a special case.

For PMM, all HMM tools: forward-backward algorithm, Viterbi algorithm, EM-algorithm apply. Thus

$$v = \arg\max_{y \in \mathcal{Y}^n} p(y|\theta, x)$$

can be found via Viterbi algorithm and so the MM-algorithm for estimating θ is known as Viterbi training. The output of Viterbi algorithm is called Viterbi path.

For mixture model, no Viterbi algorithm is needed, because given the probabilities $(p_0, \ldots, p_{K-1})$ and emission parameters θ_k, the Viterbi path is found pointwise

$$v_t = \arg\max_{k \in \mathcal{Y}} p_k f(x_t|\theta_k), \quad k \in \mathcal{Y}, \quad t = 1, \ldots, n.$$

MM-algorithm (Viterbi training) for HMMs

Input: Observations $x = x_{1:n}$, initial probabilities $\pi(y_1)$;

Initialization: Initial transition matrix $\mathbb{P}^{(0)}$ and emission parameters $\theta_k^{(0)}, k \in \mathcal{Y}$;

Iteration: Given $\mathbb{P}^{(i)}$ and $\theta_k^{(i)}$, $k \in \mathcal{Y}$ find Viterbi path (using Viterbi algorithm)

$$y^{(i)} = \arg\max_{y \in \mathcal{Y}^n} p(y|x; \mathbb{P}^{(i)}, \theta_0^{(i)}, \ldots, \theta_{K-1}^{(i)}).$$

Given Viterbi path $y^{(i)}$ re-estimate transition parameters simply by counts:

$$p_{kl}^{(i+1)} = \frac{(\sharp \text{ pairs } (k,l) \text{ in Viterbi path})}{(\sharp \text{ states } k \text{ in Viterbi path}))}.$$

MM-algorithm (Viterbi training) for HMMs

For emission parameters estimates observe that Viterbi path splits observations into K subsamples/empirical distributions

$$P_{k,n}^{(i)}(A) = \frac{1}{m_k} \sum_{t=1}^{n} I_A(x_t), \quad m_k = \sum_t I_k(y_t^{(i)}), \quad k \in \mathcal{Y}, \quad A \in \mathcal{B}(\mathbb{R}^d)$$

and then find MLE for every sub-sample separately:

$$\theta_k^{(i+1)} = \arg\max_{\theta_k} \int \ln f(x|\theta_k) P_{k,n}^{(i)}(dx), \quad k \in \mathcal{Y}.$$

Repeat until no change/stopping criterion

Output: Parameter estimates $\hat{\mathbb{P}}$, $\hat{\theta}_k, k \in \mathcal{Y}$ (and corresponding Viterbi path $\hat{y}$).

MM-algorithm (Viterbi training) for PMMs

No separation between emission and transition parameters: $p(x_2, y_2|x_1, y_1; \theta)$.

Input: Observations $x = x_{1:n}$, initial probabilities $\pi(x_1, y_1)$;

Initialization: initial parameter(s) $\theta^{(0)}$;

Iteration: Given $\theta^{(i)}$ find Viterbi path (using Viterbi algorithm)

$$y^{(i)} = \arg\max_{y \in \mathcal{Y}^n} p(y|x; \theta^{(i)}).$$

Given Viterbi path and observations find empirical distribution $P_n^{(i)}$, where

$$P_n^{(i)}\Big((A_1 \times k) \times (A_2 \times l)\Big) = \frac{1}{n-1} \sum_{t=1}^{n-1} I_{A_1}(x_t) I_k(y_t^{(i)}) I_{A_2}(x_{t+1}) I_l(y_{t+1}^{(i)}), \quad A_i \in \mathcal{B}(\mathbb{R}^d),\ k, l \in \mathcal{Y}.$$

MM-algorithm (Viterbi training) for PMMs

Find MLE estimate

$$\theta^{(i+1)} = \arg\max_{\theta} \left[\ln \pi(x_1, y_1^{(i)}) + \sum_{t=1}^{n-1} \ln p(x_{t+1}, y_{t+1}^{(i)} | x_t, y_t^{(i)}; \theta) \right]$$
$$= \arg\max_{\theta} \left[\ln \pi(x_1, y_1^{(i)}) + \int \ln p(x_2, y_2 | x_1, y_1; \theta) P_n(d(x_1, y_1, x_2, y_2)) \right].$$

Repeat until no change/stopping criterion

Output: Parameter estimates $\hat{\theta}$ (and corresponding Viterbi path $\hat{y}$).

NB! One can always start with a path instead of parameters.

A (toy) example of MM algorithm going very wrong

Hidden Bernoulli model: Mixture model, where $Y_1, Y_2, \ldots$ Bernoulli(p); p-unknown; emission distributions normal with common σ (known), means μ_0 and μ_1 (known), i.e. $X_t | Y_t = k \sim \mathcal{N}(\mu_k, \sigma^2)$.

Simulations: Take $\sigma = 0.8$, $\mu_0 = 0$ and $\mu_1 = 1$ (Gaussian noise is added to Bernoulli outputs). Generate samples (say $n \geq 100$) and taking $p^{(0)} = p$ (starting with true value), one sees that MM-estimate is typically either 0 or 1 – extremely wrong estimate!

A (toy) example of MM algorithm going very wrong

Theoretical explanation: In this model finding Viterbi path v means comparing x_t with threshold

$$s(p, \mu_1, \mu_2) = \frac{\mu_1 + \mu_0}{2} - \frac{\sigma^2}{\mu_1 - \mu_0} \ln \frac{p}{1-p}, \qquad v_t = 1 \Leftrightarrow x_t > s.$$

With true parameters $p, \mu_0 < \mu_1$ as initial values and calculate $p^{(1)}$, $\mu_0^{(1)}$, $\mu_1^{(1)}$. With $s = s(p, \mu_0, \mu_1)$, we get

$$p_n^{(1)} = \frac{m_1}{n}, \quad m_1 = \sum_{t=1}^{n} I_1(v_t) = \sum_{t=1}^{n} I_{[s,\infty)}(x_t) \quad \text{proportion of ones in Viterbi path} v$$

$$\mu_{1,n}^{(1)} = \frac{1}{m_1} \sum_{t=1}^{n} x_t I_1(v_t) = \int x P_{1,n}^{(1)}(dx), \quad \mu_{0,n}^{(1)} = \frac{1}{n - m_1} \sum_{t=1}^{n} x_t I_0(v_t) = \int x P_{0,n}^{(1)}(dx).$$

A (toy) example of MM algorithm going very wrong

When $n \to \infty$, then by SLLN (here X has true mixture distribution), a.s.

$$p_n^{(1)} \to p^{(1)} := P(X_1 > s), \quad \mu_{0,n}^{(1)} \to \mu_0^{(1)} := E[X|X \leq s], \quad \mu_{1,n}^{(1)} \to \mu_1^{(1)} := E[X|X > s]$$

Moreover the empirical measures converge

$$P_{0,n}^{(1)} \Rightarrow Q_0 = P(X_1 \in \cdot | X_1 < s), \quad P_{1,n}^{(1)} \Rightarrow Q_1 = P(X_1 \in \cdot | X_1 > s), \quad \text{a.s.}.$$

Obviously the measures Q_i are very far from normal as the following picture shows.

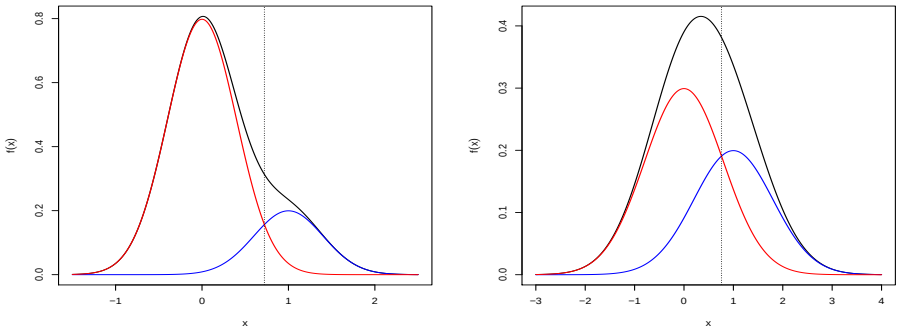


Figure: Mixture distributions (left $\sigma = 0.4, p = 0.2$, right $\sigma = 0.8, p = 0.4$). The dotted line is s. The area under black curve right of s is $p^{(1)}$.
The measures Q_0 and Q_1 correspond to the truncated black curve. Corresponding means: $\mu_0^{(1)} = -0.0017$, $\mu_1^{(1)} = 1.12$ (small σ); right: $\mu_0^{(1)} = -0.14$, $\mu_1^{(1)} = 1.4$.

A (toy) example of MM algorithm going very wrong

It turns out that when $p < 0.5$, then $p^{(1)} < p$. This means $p^{(1)} < p$ and then $s(p^{(1)}) > s(p)$ so that

$$p^{(2)} = P(X_t > s(p^{(1)})) < p^{(1)}$$

and so the "MM iterations" are monotonically decreasing

$$0.5 > p > p^{(1)} > p^{(2)} > p^{(3)} > \cdots$$

Whether the limit is 0 or not depends on σ as the following picture shows.

A (toy) example of MM algorithm going very wrong

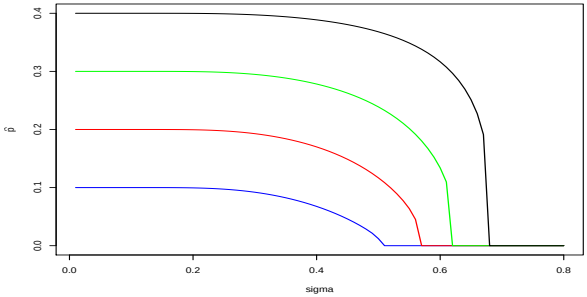


Figure: the final estimate $\hat{p}$ for different σ and $p = 0.1, 0.2, 0.3, 0.4$

For $\sigma = 0.8$ (used for simulations) it is 0. **So the MM-estimate $\hat{p}$ is very wrong even when one starts with true parameter p and sample size n is arbitrary large!**

Lack of asymptotic fixed point property

We saw $E(X|X > s) = p^{(1)} < p$ (when $p < 0.5$). So, one start with true parameter, n is infinitely big, yet the algorithm returns something else. The same holds, when one estimates μ_0 and μ_1. Indeed, (assuming p is known and $n = \infty$) taking the initial values as true parameters, the algorithm returns

$$\mu_0^{(1)} = E[X|X \leq s] = \int xQ_0(dx), \quad \mu_1^{(1)} = E[X|X > s] = \int xQ_1(dx)$$

and since Q_i is not equal to emission distributions (normal in our example), then also $\mu_i^{(1)} \neq \mu_i$. In our simple example, it is often so that

$$\mu_1^{(1)} > \mu_1, \quad \mu_0^{(1)} < \mu_0.$$

We see that the MM-algorithm **lacks asymptotic fixed point property** – starting with true parameters and having n arbitrarily large, the algorithm returns something else.

More general models

For mixture model, from SLLN it follows that the first iteration estimates based on Viterbi path v converge and the limits are (in general) different from true parameters, even when Viterbi path is found by using them. In particular $P_{k,n}^{(1)} \Rightarrow Q_k$, a.s.. Do the same convergence hold for general model?

For HMM's and more general models not trivial to show. Recall :

Observations $x_1, \ldots, x_{10}$. Subsamples based on Viterbi alignment ($K = 2$)

X :	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
Viterbi :	0	1	1	0	1	0	1	1	1	0

The subsamples (empirical measures) are

$x_1 \quad x_4 \quad x_6 \quad x_{10}$ $\quad P_{0,10}$ and $x_2 \quad x_3 \quad x_5 \quad x_7 \quad x_8 \quad x_9$ $\quad P_{1,10}$

More general models

(Except the mixture model) Viterbi path is global: adding one more observation x_{n+1} can change the whole path and influence heavily the measures:

X :	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
Viterbi :	0	1	1	0	1	0	1	1	1	0

X :	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}
Viterbi :	0	0	0	0	1	1	1	1	0	0	0

What about asymptotics in this case?

More general models: the Viterbi process

It can be show that under rather general conditions, the Viterbi path oh HMM stabilizes, and there exists so-called infinite Viterbi path that can be considered as a Viterbi decoding of $x_1, x_2, \ldots$. Since $X = X_1, X_2, \ldots$ is a random process, the the infinite Viterbi path/decoding forms a random process called Viterbi process $V = V_1, V_2 \ldots$.

Moreover, one can show that the 2-dim process (X, V) is regenerative. This implies the existence of limit proportions $p_{kl}^{(1)}$ so that

$$p_{kl,n}^{(1)} = \frac{\sum_{t=1}^{n-1} I_k(V_t) I_l(V_{t+1})}{\sum_{t=1}^{n-1} I_k(V_t)} \to p_{kl}^{(1)}, \quad \text{a.s}$$

and the existence of limit measures Q_k so that

$$P_{k,n}^{(1)} \Rightarrow Q_k, \quad \text{a.s}, \quad \forall k \in \mathcal{Y}.$$

More general models: the Viterbi process

Then also (under general conditions)

$$\theta^{(1)}_{k,n} = \arg\max_{\theta_k} \int \ln f(x|\theta_k) P^{(i)}_{k,n}(dx) \to \theta^{(1)}_k, \quad \text{a.s},$$

where

$$\theta^{(1)}_k := \arg\max_{\theta_k} \int \ln f(x|\theta_k) Q_k(dx), \quad \forall k \in \mathcal{Y}.$$

In general, the matrix $\mathbb{P}^{(1)} = (p^{(1)}_{kl}) \neq \mathbb{P}$ and $\theta^{(1)}_k \neq \theta^*_k$ – no asymptotic fixed point property!

For (a large class) of HMM's this was proven in (Kolydenko, L., 2008, 2010).

More general models: the Viterbi process

Recently the existence and (almost) regenerativity of Viterbi process were proven for (a large class of PMM's) in (Sova, L., 2020, 2021). For PMM's we speak about the measure $P_n^{(1)}$ (empirical distribution of pairs (Z_1, Z_2)); it holds $P_n^{(1)} \Rightarrow Q$, a.s., implying that

$$\theta_n^{(1)} = \arg\max_{\theta} \int \ln p(x_2, y_2 | x_1, y_1; \theta) P_n(d(x_1, y_1, x_2, y_2)) \to \theta^{(1)},$$

where

$$\theta^{(1)} = \arg\max_{\theta} \int \ln p(x_2, y_2 | x_1, y_1; \theta) Q(d(x_1, y_1, x_2, y_2))$$

and, again, $\theta^{(1)} \neq \theta^*$ – no asymptotic fixed point property!

Under general conditions EM-algorithm has asymptotic fixed point property!

Adjusted Viterbi training

Why MM-algorithm (Viterbi training)? Faster (bigger steps), cheaper (computationally), easier (to implement).

We saw that in general (θ^* true, $\theta^{(1)}$ output of 1-st iteration $\theta^{(0)} = \theta^*$ for $n = \infty$)

$$\Delta(\theta^*) := \theta^* - \theta^{(1)} \neq 0$$

Suppose $\theta \to \Delta(\theta)$ is known. Then there is an easy way to get fixed point property:

Adjusted Viterbi training (AVT): Given $\theta^{(i)}$ find Viterbi path and MM estimates as previously, let that be $\theta_{MM}^{(i+1)}$. Take

$$\theta^{(i+1)} = \theta_{MM}^{(i+1)} + \Delta(\theta_{MM}^{(i+1)}).$$

AVT has asymptotic fixed point property.

Simulations: more correct (comparable to EM), still fast and cheap.
Problem: $\Delta(\theta)$ is not known analytically (we only known it exists) except mixtures.

Segmentation with Bayes

HMM in Bayesian setup – transition matrix $\mathbb{P}$ has prior $\pi_{tr}(\mathbb{P})$ and emission parameters have priors $\pi_{em}(\theta) = \pi_{em}(\theta_0) \cdots \pi_{em}(\theta_{K-1})$, transition and emission priors independent. Transition priorπ_{tr} models every row independently with Dirichlet priors – typical choice. Thus

$$(p_{l,0}, \ldots, p_{l,K-1}) \sim \mathrm{Dir}(\alpha_{l,0}, \ldots, \alpha_{l,K-1}), \quad \forall l.$$

A special case $\alpha_{l,k} = 1$ – uniform: uninformative prior, every matrix is alike.

Hence – a new model (X, Y) with law $p(x, y)$, where for paths $x = x_{1:n}$ and $y = y_{1:n}$

$$p(x, y) = p(y)p(x|y), \quad p(y) = \int p(y|\mathbb{P})\pi_{tr}(d\mathbb{P}), \quad p(x|y) = \int p(x|y, \theta)\pi_{em}(d\theta).$$

Segmentation with Bayes

The new model is very far from HMM:

1) Y is not a Markov chain any more

2) Given Y, the observations $X_1, \ldots, X_n$ are not (conditionally) independent any more.

Most sadly, (X, Y) is not a PMM.

We aim to find MAP (maximum a posteriori or Viterbi) path

$$v = \arg\max_y p(y, x) = \arg\max_y p(y|x)$$

but Viterbi algorithm does not apply any more – no Markov property!

Segmentation with Bayes: how to find Viterbi/MAP path?

* Simulated annealing (expensive)

* Parameters first: find $\hat{\theta}$ and $\hat{\mathbb{P}}$ by Bayesian EM (posterior mode) and then apply Viterbi algorithm with these parameters.

* Variational Bayes algorithm (close to the previous one);

* Segmentation EM – just Bayesian EM algorithm with roles changed: (denoting all parameters with θ): given $y^{(i)}$ update

$$y^{(i+1)} = \arg\max_y \int \ln p(y, \theta|x) p(\theta|y^{(i)}, x) d\theta = \arg\max_y \int \ln p(y, x|\theta) p(\theta|y^{(i)}, x) d\theta.$$

It holds: $p(y^{(i+1)}|x) \geq p(y^{(i)}|x)$.

Segmentation with Bayes: how to find Viterbi/MAP path?

In our model (Dirichlet) – segmentation EM is implementable! (given emission priors are "nice").

* (Segmentation) MM – the same Bayesian MM, the output is path.

Comparison of the algorithms: (Gasbarra, Koloydenko, Kuljus, L. 2020, Koloydenko, Kuljus, L. 2021). No wonder that segmentation EM outperforms most of the other methods – it optimizes the right criterion. Syrprisingly MM performs equally well, sometimes even better!

Explanation why (segmentation) MM algorithm performs as well as (segmentation) EM algorithm: toy model

Mixture model, with $p \sim \mathrm{Beta}(\alpha, \alpha)$:

$$\begin{aligned} p &\sim \mathrm{Beta}(\alpha, \alpha) \\ Y_1, \ldots, Y_n | p &\overset{i.i.d.}{\sim} B(1, p) \\ X_i | Y_i, p &\overset{ind}{\sim} \mathcal{N}(Y_i, \sigma^2), \quad i = 1, \ldots, n. \end{aligned}$$

The hidden process $Y_1, Y_2, \ldots$ is now Beta-Bernoulli process with the following law:

Let $y = y_{1:n} \in \{0, 1\}$ such that $\sum_{i=1}^n y_i = m$ (m ones), the probability of that sequence is

$$p(y_{1:n}) := \frac{\alpha(\alpha+1)\cdots(\alpha+m-1)\cdot\alpha(\alpha+1)\cdots(\alpha+n-m-1)}{(2\alpha)(2\alpha+1)\cdots(2\alpha+n-1)}$$

Hidden Beta-Bernoulli model

When $\alpha \geq 1$ is an integer, then

$$p(y_{1:n}) = \frac{(\alpha+m-1)!}{(\alpha-1)!}\frac{(\alpha+n-m-1)!}{(\alpha-1)!}\frac{(2\alpha-1)!}{(2\alpha+n-1)!} = \frac{\binom{2\alpha-2}{\alpha-1}(2\alpha-1)}{\binom{2\alpha+n-2}{\alpha+m-1}(2\alpha+n-1)}$$

and when $\alpha = 1$ (uniform prior), it simplifies

$$p(y_{1:n}) = \frac{1}{\binom{n}{m}(n+1)}.$$

The model is now hidden Beta-Bernoulli model: with $x = x_{1:n}$ and $y = y_{1:n}$, it holds

$$p(x,y) = p(y)p(x|y) = p(y_{1:n})\prod_{t=1}^{n} f(x_t|y_t),$$

where $f(x_t|y_t)$ is Gaussian density with mean y_t evaluated at x_t.

Segmentation EM for hidden Beta-Bernoulli model

Input: Start with a sequence $y^{(0)}$ (or the number of ones $m^{(0)}$);

Iteration: Given sequence $y^{(i)}$ define

$$u_1^{(j)} := \exp[\psi(\alpha + m^{(j)}) - \psi(n + 2\alpha)], \quad u_0^{(i)} := \exp[\psi(\alpha + n - m^{(i)}) - \psi(n + 2\alpha)],$$

where $m^{(i)} = \sum_t y_t^{(i)}$ (number of ones in $y^{(i)}$) and ψ is digamma function.

Find the sequence $y^{(i+1)}$ as follows (can be done pointwise)

$$y^{(j+1)} = \arg\max_{y \in \mathcal{Y}^n} \left[\sum_{t=1}^{n} \ln f(x_t|y_t) + \ln q^{(i)}(y) \right],$$

where

$$q^{(i)}(y) = \left(u_1^{(i)}\right)^{\sum_t y_t} \left(u_0^{(i)}\right)^{n - \sum_t y_t}.$$

Output: Proceed until $y^{(i)} = y^{(i+1)}$, output is the sequence $y^{(i+1)}$.

About digamma

If $X \sim B(\alpha, \beta)$, then $E(\ln X) = \psi(\alpha) - \psi(\alpha + \beta)$, where

$$\psi(x) = \frac{\Gamma'(x)}{\Gamma(x)}.$$

For even moderate n, $\psi(n) \approx \ln(n - 0.5)$. Then, if $\alpha \ll n$

$$u_1^{(j)} \approx \frac{m^{(j)} + \alpha - 0.5}{n + 2\alpha - 0.5}, \quad u_0^{(j)} \approx \frac{n - m^{(j)} + \alpha - 0.5}{n + 2\alpha - 0.5}.$$

We see that (for big n)

$u_1^{(j)} \approx \frac{m^{(j)}}{n}$ proportion of ones in $y^{(j)}$, $u_0^{(j)} \approx \frac{n - m^{(j)}}{n}$ proportion of zeros in $y^{(j)}$.

So the algorithm is close to MM algorithm.

(Segmentation) MM for hidden Beta-Bernoulli model

Input: Start with a sequence $y^{(0)}$ (or the number of ones $m^{(0)}$);

Iteration: Given sequence $y^{(i)}$ define

$$p^{(i)} = \frac{m^{(i)}}{n} \text{ proportion of ones in } y^{(i)}$$

($m^{(i)} = \sum_t y_t^{(i)}$ is the number of ones in $y^{(i)}$).

Find the sequence $y^{(i+1)}$ as follows (can be done pointwise)

$$y^{(i+1)} = \arg\max_y \Big[\sum_{t=1}^{n} \ln f(x_t|y_t) + \ln q^{(i)}(y) \Big],$$

where $q^{(i)}$ is iid Bernoulli under parameter $p^{(i)}$:

$$q^{(i)}(y) = \big(p^{(i)}\big)^{\sum_t y_t} \big(1 - p^{(i)}\big)^{n - \sum_t y_t}.$$

Output: Proceed until $y^{(i)} = y^{(i+1)}$, output is the sequence $y^{(j+1)}$.

Posterior consistency

when $\alpha \ll n$, the (segmentation) MM algorithm preforms almost as (segmentation) EM algorithm. Another explanation of that similarity is posterior consistency: with $\theta \sim \mathrm{Beta}(\alpha, \alpha)$ and m being the number of ones in y, we have

$$p(\theta|y,x) = p(\theta|y) \sim \mathrm{Beta}(\alpha + m, \alpha + n - m).$$

The variance of $\mathrm{Beta}(\alpha + m, \alpha + (n-m))$ distribution is of order $O(\frac{1}{n})$ and so $p(\theta|y)$ is heavily peaked over its expectation

$$\frac{\alpha + m}{\alpha + n} \approx \frac{m}{n}.$$

This means that in the segmentation EM update the distribution $p(\theta|y^{(i)}, x)$ is peaked over $\frac{m^{(j)}}{n} = \theta^{(j)}$ and so the integral is close to maximum:

$$y^{(i+1)} = \arg\max_y \int \ln p(y,x|\theta) p(\theta|y^{(i)},x) d\theta \approx \arg\max_y \ln p(y,x|\theta^{(j)}) = \arg\max_y \ln p(y,\theta^{(j)}|x),$$

exactly what MM algorithm does.

MM: good in segmentation, bad in parameter estimation

MM-algorithm above (VT for mixtures) has formally nothing to do with Bayesian approach (no α in it!). Yet, it aims to find maximum likelihood path in hidden Beta-Bernoulli model with some "relatively small α" (like uniform prior). The same holds for Bayesian HMM's with Dirichlet priors on transition matrix.

Conclusion: MM-algorithm (Viterbi training) might be very poor in parameter estimation (it lacks asymptotic fixed point property, optimizes wrong criterion....), but works well in Bayesian segmentation (under Dirichlet priors).

Why it performs well in one task (segmentation) and fails in another (parameter estimation)?

Answer: It fails in parameter estimation just because it is good in segmentation!

Link to the statistical learning

For any model $p(x, y)$, the Viterbi (MAP) path

$$v = \arg\max_{y \in \mathcal{Y}^n} p(y|x) = \arg\max_{y \in \mathcal{Y}^n} \ln p(y, x) = \arg\max_{y \in \mathcal{Y}^n} [\ln p(x|y) + \ln p(y)].$$

For an hidden model (either mixture model or HMM or hidden Beta-Beroulli) with (conditionally) independent emission densities $f(\cdot|k)$, $k \in \mathcal{Y}$

$$v = \arg\max_{y \in \mathcal{Y}^n} [\sum_{t=1}^{n} f(x_t|y_t) + \ln p(y)].$$

If emission densities are normal with common σ^2 and means $\mu_k = k$ (as in toy model)

$$v = \arg\min_{y \in \mathcal{Y}^n} \left[\sum_{t=1}^{n} (x_t - y_t)^2 - 2\sigma^2 \ln p(y)\right].$$

Statistical learning objective, where $\sum_{t=1}^{n} (x_t - y_t)^2$ – loss function (depends on data); $\ln p(y)$ – penalty term (depends on model); σ^2 – regularization constant. The bigger is σ^2, the more model matters. So, if σ is big, the v aims to maximize $p(y)$.

Hidden Bernoulli model

The properties of MAP path depend on model $p(y)$.

$Y_1, Y_2, \ldots$ is Bernoulli process. The simplest possible model. Any 0-1 model be considered as a concatenation of blocks

$$\underbrace{000}_{0-\text{block}} \quad \underbrace{11}_{1-\text{block}} \quad \underbrace{00000}_{0-\text{block}} \quad \underbrace{111}_{1-\text{block}} \quad \underbrace{000}_{0-\text{block}} \quad \cdots$$

Block lengths are Geometrically distributed, the expected lengths of 0-block is $\frac{1}{p}$, expected length of 1-block is $\frac{1}{1-p}$.

The path $\arg\max_{y \in \mathcal{Y}^n} p(y)$ is constant (only 0's or 1's), if $p < 0.5$, it is constantly $0000 \cdots 0$ with probability $(1-p)^n$. The (any) second best path has one 1 and its probability is $(1-p)^{n-1}p$. The ratio between the best and second best path is constant: $(1-p)/p$.

Hidden Beta-Bernoulli model

$Y_1, Y_2, \ldots$ is Beta-Bernoulli process. Totally different properties: random variables $Y_1, Y_2, \ldots$ have the same B(1,0.5) distribution, but they are positively correlated:

$$\operatorname{cov}(Y_1, Y_2) = \frac{\alpha+1}{4\alpha+2} - \frac{1}{4} > 0.$$

The expected block length of both blocks is ∞, when $\alpha \leq 1$ (in particular, for uniform law and $\frac{2\alpha-1}{\alpha-1}$ otherwise). The max-likelihood path is constant (either $0000\cdots0$ or $1111\cdots1$), its probability is

$$\frac{\alpha \cdot (\alpha+1) \cdots (\alpha+n-1)}{2\alpha \cdot (2\alpha+1) \cdots 2\alpha+n-1}.$$

In particular, for $\alpha = 1$ (uniform prior),

$$p(0000\cdots0) = p(1111\cdots1) = \frac{1}{n+1}.$$

We see that under Beta-Bernoulli prior $p(0000\cdots0)$ is much higher than under Bernoulli model! For uniform prior: $\frac{1}{n+1} \gg (1-p)^n$.

Hidden Beta-Bernoulli model

The ratio between best and second best path:

$$\frac{p(000\cdots0)}{p(100\cdots0)} = \alpha + n - 1$$

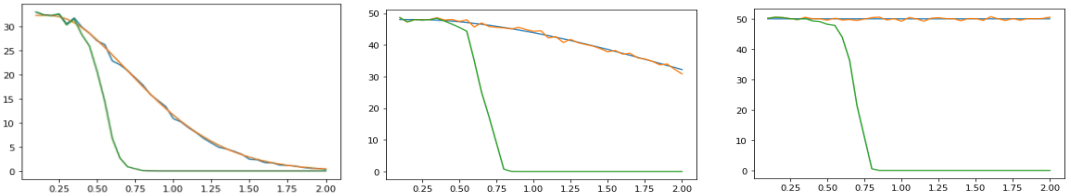
is much bigger than that of under Bernoulli model (constant).

For Beta-Bernoulli model the max probability path has relatively more weight than under Bernoulli model. Similarly the paths with less blocks have relatively more weights. This, in turn, implies that the solution to our statistical learning under Beta-Bernoulli model has less blocks / jumps less / is more conservative than under Bernoulli model.

MAP path jumps under Hidden Bernoulli and hidden Beta-Bernoulli model

In the picture below the number of jumps (number of blocks-1) of MAP path of hidden Bernoulli and hidden Beta-Bernoulli model are compared. For every σ (x-axes), 100 samples of size 100 are generated and the average number of jumps are found. Upper curve: Bernoulli model (blue line theoretical value), lower curve: Hidden Beta-Bernoulli model. Left $p = 0.2$, middle $p = 0.4$, right $p = 0.5$.

We see that MAP path of hidden Beta-Bernoulli model is much more conservative. If the number of jumps is 0, then $\hat{p}$ of VT is either 0 or 1. Form $\sigma \geq 0.8$ it is so.



Conclusion

In latent variable model: putting priors on parameter (to relaxe assumptions on parameters) induces assumptions on paths! For example, noninformative (uniform) priors on transition matrix in HMM induces long memory to hidden process. On the other hand hidden Bernoulli(0.5) process – all paths are equiprobable (no assumptions on paths), but strong assumption on parameter.

For many models (mixtures, HMM's) MM-algorithm (VT) resembles segmentation EM with Dirichlet priors, hence aims to find MAP of certain Bayesian model even when user is in fully frequentist setup.

When MM performs well is Bayesian segmentation, its must be bad in parameter estimation – the path properties are so different (MAP path is far of being typical)!

References:

About (adjusted) Viterbi training:

J. Lember, A. Koloydenko, Adjusted Viterbi training, *Probability in Engineering and Information Sciences* (2007)

J. Lember, A. Koloydenko, The adjusted Viterbi training for hidden Markov model, *Bernoulli* (2008)

About Viterbi process:

J. Lember, A. Koloydenko, A constructive proof of the existence of Viterbi processes, *IEEE Transactions on Information Theory* (2010)

J. Lember, J. Sova, Existence of infinite Viterbi path for pairwise Markov model, *Stochastic Processes and their Applications* (2020)

J. Lember, J. Sova, Regenerativity of Viterbi process, *Journal of Theoretical Probability* (2021)

About Bayesian segmentation:

J. Lember, D, Gasbarra, A. Koloydenko, K. Kuljus, Estimation of Viterbi path in Bayesian hidden Markov models, *METRON* (2019)

A. Koloydenko, K. Kuljus, J. Lember, MAP segmentation in Bayesian hidden Markov models: a case study, *Journal of Applied Statistics* (2020)