Hidden Markov Models: learning and extensions

March 22, 2017

Topics

- Basics:
  - Concepts from information theory
  - Relative frequency as maximum likelihood estimates
- Hidden Markov Models
  - Basic inference methods
  - Learning and inference
- Parameter learning in HMMs
  - Approaches for incomplete data
    - Data imputation (completion) by most probable values (Viterbi)
    - Data imputation (completion) by random values (Gibbs)
    - Exact calculations and analytic usage of expectations (E-M)
  - The Expectation-Maximization method
  - The Baum-Welch method

Entropy and mutual information

If p_i is a discrete probability distribution, its entropy is

$$\mathrm{H}(\underline{\boldsymbol{p}}) = -\sum_i p_i \log(p_i), \tag{1}$$

Conditional entropy $H(Y|X)$ is defined as $\sum_x p(x) \sum_y p(y|x) \log(p(y|x))$. Mutual information is defined as $I(Y;X) = H(Y) - H(Y|X)$. The (conditional) *mutual information can be written as*

$$MI_p(X;Y|Z) = \mathrm{KL}(p(X,Y|Z)|p(X|Z)p(Y|Z)). \tag{2}$$

The chain rule for (joint distributions) and entropies:
$p(X_1,\ldots,X_n) = \prod_i p(X_i|X_1,\ldots,X_{i-1})$
$H(X_1,\ldots,X_n) = \sum_i H(X_i|X_1,\ldots,X_{i-1})$
And also

$$= \quad H(X_1,\ldots,X_n) \tag{3}$$

$$= \quad \sum_{i=1}^{n} H(X_i) - \sum_{i=1}^{n} I(X_i;X_1,\ldots,X_{i-1}). \tag{4}$$

Optimality of relative frequencies

Relative frequency is a maximum likelihood estimator in multinomial sampling: Assume $i = 1, \ldots K$ outcomes assuming multinomial sampling with parameters $\theta = \{\theta_i\}$ and observed occurrencies $n = \{n_i\}$ $(N = \sum_i n_i)$. Then

$$\log \frac{p(n|\theta^{ML})}{p(n|\theta)} = \log \frac{\prod_i (\theta_i^{ML})^{n_i}}{\prod_i (\theta_i)^{n_i}} = \sum_i n_i \log \frac{\theta_i^{ML}}{\theta_i} = N \sum_i \theta_i^{ML} \log \frac{\theta_i^{ML}}{\theta_i} > 0 \quad (5)$$

We are ready, because the last quantity is the “KL-divergence”, which is always positive. Proof: if $\hat{p}_i, p_i$ are discrete probability distributions, the *cross-entropy H and the Kullback-Leibler (semi)distance KL are as follows*

$\mathrm{H}(\underline{\boldsymbol{p}}||\underline{\hat{\boldsymbol{p}}}) = -\sum_i p_i \log(\hat{p}_i)$

$\mathrm{KL}(\underline{\boldsymbol{p}}||\underline{\hat{\boldsymbol{p}}}) = \sum_i p_i \log(p_i/\hat{p}_i)$

$0 < KL(\theta^{ML}||\theta)$*:*

$$-KL(p||q) = \sum_i p_i \log(q_i/p_i) \leq \sum_i p_i((q_i/p_i) - 1) = 0 \quad (6)$$

using $\log(x) \leq x - 1$.

Frequently pseudocounts are used to avoid imprecise estimates (e.g. divison by 0) and prior counts to incorporate bias/expertise.

HMM: definition

Hidden Markov Models (definitions/notations following DEKM)

1. π denotes a state sequence (of a Markov chain), π_i is the ith state
2. a_{kl} the transition probabilities $p(\pi_i = l | \pi_{i-1} = k)$ in the MC (extra state 0 for start/end)
3. $e_k(b)$ are the emission probabilities $p(x_i = b | \pi_i = k)$

Inferences in HMMs

Note $|\pi| = \mathcal{O}(|S|^L)$

- -,L $p(x,\pi) = a_{0\pi_l} \prod_{i=1}^{L} e_{\pi_i}(x_i) a_{\pi_i \pi_{i+1}}$
- ?,L "decoding": $\pi^* = \arg\max_\pi p(x,\pi)$
- ?,L sequence probability:$p(x) = \sum_\pi p(x,\pi)$ (or $p(x|M)$ "model likelihood" or filtering)
- ?,L smoothing/posterior decoding:$p(\pi_i = k|x)$
- ?,OK? parametric inference (training/parameteresation)
- ?,OK? structural inference (model selection)

HMM: Viterbi algorithm

Goal: "decoding": $\pi^* = \arg\max_\pi p(x, \pi)$
Note: "best joint-state-sequence explanation"$\neq$ "joint sequence of best-state-explanations"
Inductive idea: extend most probable paths with length i to i+1
$v_k(i)$ denotes the probability of the most probable path ending in state k with observation i
Then

$$v_l(i+1) = e_l(x_{i+1}) \max_k (v_k(i) a_{kl}) \tag{7}$$

Algorithm 1 Algorithm: Viterbi

Require: HMM,x
Ensure: $\pi^* = \arg\max_\pi p(x, \pi)$

```
Ini: (i=0): v_0(0) = 1, v_k(0) = 0 for 0<k
for i = 1 to L do
    v_l(i) = e_l(x_i) max_k (v_k(i-1) a_kl)
    ptr_i(l) = arg max_k (v_k(i-1) a_kl)
End: p(x, π*) = max_k (v_k(L) a_k0), π*_L = arg max_k (v_k(L) a_k0)
for i = L to 1 do {Traceback}
    π*_{i-1} = ptr_i(π*_i)
```

Note, small probabilities may cause positive underflow (length can be up to 10^3 <)=> log.
Note, $\pi^* = \arg\max_\pi p(x, \pi) = \arg\max_\pi p(\pi | x)$

HMM: forward algorithm

Goal: sequence probability:$p(x) = \sum_\pi p(x,\pi)$ (or $p(x|M)$ "model likelihood" or filtering)

Approximation: $p(x) = \sum_\pi p(x,\pi) \approx p(x,\pi^*) = a_{0\pi_l^*} \prod_{i=1}^{L} e_{\pi_i^*}(x_i) a_{\pi_i^* \pi_{i+1}^*}$ (π^* by Viterbi)

Inductive idea(dynamic programming): extend the probability of generating observations $x_{1:i}$ being in state k at i to i+1

By introducing $f_k(i) = p(x_{1:i}, \pi_i = k)$, we can proceed

$$f_l(i+1) = e_l(x_{i+1}) \sum_k (f_k(i) a_{kl}) \tag{8}$$

Algorithm 2 Algorithm: forward

Require: HMM M,x

Ensure: $p(x|M)$

1: Ini: (i=0): $f_0(0) = 1$,$f_k(0) = 0$ for 0<k

2: **for** $i = 1$ to L **do**

3: $\quad f_l(i) = e_l(x_i) \sum_k (f_k(i-1) a_{kl})$

4: End: $p(x|M) = \sum_k (f_k(L) a_{k0})$

Note, we have to sum small probabilities! => log transformation is not enough, scaling methods..

HMM: backward algorithm

Goal: smoothing/posterior decoding $p(\pi_i = k|x)$

Idea: $p(\pi_i = k|x) = \frac{p(\pi_i=k,x)}{p(x)}$ ($p(x)$ can be computed by the forward algorithm)

$$p(\pi_i = k, x) = p(\pi_i = k, x_{1:i}) p(x_{i+1:L}|\pi_i = k, x_{1:i}) = f_k(i) \underbrace{p(x_{i+1:L}|\pi_i = k)}_{b_k(i)}$$

Ensure: $b_k(i) = p(x_{i+1:L}|\pi_i = k)$

1: Ini: (i=L): $b_k(L) = a_{k0}$ for all k

2: **for** $i = L - 1$ to 1 **do**

3: $\quad b_k(i) = \sum_l a_{kl} e_l(x_{i+1}) b_l(i+1)$

4: End: $p(x|M) = \sum_l a_{0l} e_l(x_1) b_l(1)$

Note, conditionally most probable state at i $\neq$ state in most probable explanation at i.

HMM parameter learning

Assume n independent/exhangeable sequences $x^{(1)}, \dots, x^{(n)}$

$$p(x^{(1)}, \dots, x^{(n)} | \theta) = \prod_{i=1}^{n} p(x^{(i)} | \theta) \tag{9}$$

1. structure known, state sequences are known: ML parameter computation from counts
2. structure known, state sequences are unknown
   2.1 manual/heuristic matching: ML parameter computation from counts
   2.2 : Viterbi training: iterative ”multiple alignment-ML parameter computation from counts"
   2.3 : Baum-Welch training: iterative computation of mean counts and improved parameters from mean counts (EM-based)
3. structure unknown, state is unknown

Estimation using known state sequences

Recall relative frequency is a maximum likelihood estimator in multinomial sampling.

Assume $i = 1, \dots K$ outcomes assuming multinomial sampling with parameters $\theta = \{\theta_i\}$ and observed occurrencies $n = \{n_i\}$ $(N = \sum_i n_i)$. Then

$$\log \frac{p(n|\theta^{ML})}{p(n|\theta)} = \log \frac{\prod_i (\theta_i^{ML})^{n_i}}{\prod_i (\theta_i)^{n_i}} = \sum_i n_i \log \frac{\theta_i^{ML}}{\theta_i} = N \sum_i \theta_i^{ML} \log \frac{\theta_i^{ML}}{\theta_i} > 0 \quad (10)$$

because $0 < KL(\theta^{ML}||\theta)$

$$-KL(p||q) = \sum_i p_i \log(q_i/p_i) \leq \sum_i p_i((q_i/p_i) - 1) = 0 \quad (11)$$

using $\log(x) \leq x - 1$.

Thus using the counts of state transitions A_{kl} and emissions $E_k(b)$

$$a_{kl} = \frac{A_{kl}}{\sum_{l'} A_{kl'}} \text{ and } \mathrm{e_k(b)} = \frac{\mathrm{E_k(b)}}{\sum_{\mathrm{b'}} \mathrm{E_k(b')}} \quad (12)$$

So called *pseudocounts* to avoid imprecise estimates (e.g. divison by 0) and *prior counts* to incorporate bias/expertise.

$\Rightarrow A'_{kl} = A'_{kl} + r_{kl}\ E'_k(b) = E_k(b) + r_k(b)$

HMM parameter learning: Viterbi

Idea: using the actual parameters compute the most probable paths $\pi^*(x^{(1)}), \ldots, \pi^*(x^{(n)})$ for the sequences and select ML parameters based on these.

Require: HMM structure, $x^{(1)}, \ldots, x^{(n)}$
Ensure: $\approx \arg\max_\theta p(x^{(1)}, \ldots, x^{(n)} | \theta, \pi^*(x^{(1)}, \theta), \ldots, \pi^*(x^{(n)}, \theta))$

```
Ini: draw random model parameters θ_0 (e.g. from Dirichlet)
repeat
    set A and E values to their pseudocount
    for i = 1 to n do
        Compute π*(x^(i)) using θ_t with the Viterbi algorithm
    Set new ML parameters θ_{t+1} based on current counts A and E from
    x^(1),...,x^(n), π*(x^(1)),...,π*(x^(n))
    Compute model likelihood L_{t+1} = p(x^(1),...,x^(n) | θ_{t+1})
until NoImprovement(L_{t+1},L_t,t)
```

Note, that this finds a θ maximizing $p(x^{(1)}, \ldots, x^{(n)} | \theta, \pi^*(x^{(1)}, \theta), \ldots, \pi^*(x^{(n)}, \theta))$ and not the original goal $p(x^{(1)}, \ldots, x^{(n)} | \theta)$.

HMM parameter learning: Baum-Welch

Idea: compute the expected number of transitions/emissions A_t,E_t based on θ_t, then update to θ_{t+1} based on A_t,E_t...
The probability of $k \to l$ transition at position i in sequence x is

$$p(\pi_i = k, \pi_{i+1} = l|x) \tag{13}$$

$$= \frac{p(\overbrace{x_1, \ldots, x_i, \pi_i = k}^{f_k(i)}, x_{i+1}, \overbrace{\pi_{i+1} = l, x_{i+2}, \ldots, x_L}^{b_l(i+1)})}{p(x)} = \frac{f_k(i) a_{kl} e_l(x_{i+1}) b_l(i+1)}{p(x)} \tag{14}$$

The mean of the number of this transition and the mean of the number of emission b from state k is

$$A_{kl} = \sum_j \frac{1}{p(x^{(j)})} \sum_i f_k^{(j)}(i) a_{kl} e_l(x_{i+1}^{(j)}) b_l^{(j)}(i+1) \tag{15}$$

$$E_k(b) = \sum_j \frac{1}{p(x^{(j)})} \sum_{i|x_i^{(j)}=b} f_k^{(j)}(i) b_k^{(j)}(i), \tag{16}$$

Apply the same iteration as in Viterbi training ($\theta_t \to A_t, E_t \to \theta_{t+1} \to \ldots$)
Why does it converge? Baum-Welch is an Expectation-Maximization algorithm

Derivation of Baum-Welch I: Expectation-Maximization (E-M)

Goal: from observed x, missing π: $\theta^* = \arg\max_\theta \log(p(x|\theta))$
Idea: improve "expected data log-likelihood" $Q(\theta|\theta_t) = \sum_\pi p(\pi|x,\theta_t)\log(p(x,\pi|\theta))$
Using $p(x,\pi|\theta) = p(\pi|x,\theta)p(x|\theta)$ we can write that

$$\log(p(x|\theta)) = \log(p(x,\pi|\theta)) - \log(p(\pi|x,\theta)) \tag{17}$$

Multiplying with $p(\pi|x,\theta_t)$ and summing over π gives

$$\log(p(x|\theta)) = \underbrace{\sum_\pi p(\pi|x,\theta_t)\log(p(x,\pi|\theta))}_{Q(\theta|\theta_t)} - \sum_\pi p(\pi|x,\theta_t)\log(p(\pi|x,\theta)) \tag{18}$$

We want to increase the likelihood, i.e. want this difference to be positive

$$\log(p(x|\theta)) - \log(p(x|\theta_t)) = Q(\theta|\theta_t) - Q(\theta_t|\theta_t) + \underbrace{\sum_\pi p(\pi|x,\theta_t)\log(\frac{p(\pi|x,\theta_t)}{p(\pi|x,\theta)})}_{KL(p(\pi|x,\theta_t)||p(\pi|x,\theta))} \tag{19}$$

Because $0 \geq KL(p||q)$, so

$$\log(p(x|\theta)) - \log(p(x|\theta_t)) \geq Q(\theta|\theta_t) - Q(\theta_t|\theta_t). \tag{20}$$

E-M, Expectation-Maximization: using expectations, select the best:

$$\theta_{t+1} = \arg\max_\theta Q(\theta|\theta_t) \tag{21}$$

Generalised E-M: if we can select a better θ w.r.t. $Q(\theta|\theta_t)$ then asymptotically it converges to a local or global maximum (note that the target θ has to be continuous).

Derivation of Baum-Welch II: E-M

The probability of a given path π and observation x is

$$p(x,\pi|\theta) = \prod_{k=1}^{M}\prod_{b}[e_k(b)]^{E_k(b,\pi)}\prod_{k=0}^{M}\prod_{l=1}^{M}a_{kl}^{A_{kl}(\pi)} \tag{22}$$

using this we can rewrite $Q(\theta|\theta_t) = \sum_\pi p(\pi|x,\theta_t)\log(p(x,\pi|\theta))$ as

$$Q(\theta|\theta_t) = \sum_{\pi} p(\pi|x,\theta_t)\sum_{k=1}^{M}\sum_{b}E_k(b,\pi)\log(e_k(b)) + \sum_{k=0}^{M}\sum_{l=1}^{M}A_{kl}(\pi)\log(a_{kl}) \tag{23}$$

Note that the expected value of A_{kl} and $E_k(b)$over πs for a given x is

$$E_k(b) = \sum_{\pi} p(\pi|x,\theta_t)E_k(b,\pi) \quad \mathrm{A_{kl}} = \sum_{\pi}\mathrm{p}(\pi|\mathrm{x},\theta_\mathrm{t})\mathrm{A_{kl}}(\pi), \tag{24}$$

Doing the sum first over πs gives (also over multiple sequences in general)

$$Q(\theta|\theta_t) = \sum_{k=1}^{M}\sum_{b}E_k(b)\log(e_k(b)) + \sum_{k=0}^{M}\sum_{l=1}^{M}A_{kl}\log(a_{kl}) \tag{25}$$

Derivation of Baum-Welch III: E-M

Recall that A_{kl} and $E_k(b)$ are computable with forward/backward algorithms using current θ_t, whereas the a_{kl} and $b_k(l)$ parameters form the new candidate θ .
The $Q(\theta|\theta_t)$ is maximized by $a_{kl}^0 = \frac{A_{ij}}{\sum_k A_{ik}}$, because the difference for example for the A term is

$$\sum_{k=0}^{M}\sum_{l=1}^{M} A_{kl}\log(\frac{a_{kl}^0}{a_{kl}}) = \sum_{k=0}^{M}(\sum_{l'} A_{kl'})\sum_{l=1}^{M} a_{kl}^0 \log(\frac{a_{kl}^0}{a_{kl}}) \tag{26}$$

which is a KL distance, so not negative.

Summary

- Expectations by inference methods
- Maximization by maximum likelihood optimization