Hidden Markov Model

Zvengin
zvengin@nii.ac.jp

January 6, 2018

1 Markov Chain

Markov Chain and Hidden Markov Model are extention of finite automation. Markov Chain is a special case of weighted finite automation, which is defined by a set of states and a set of weighted arcs. In Markov Chain, weighted arcs amongs states are replaced by a set of transition probabilities among states.
Definition of Markov Chain

$$Q = \{q_1, q_2, ..., q_N\} \text{ a set of states} \tag{1}$$
$$A = \{a_{01}, a_{02}, ..., a_{NN}\} \text{ a transition probability matrix with size of N} \times \text{N} \tag{2}$$
$$q_0, q_F \text{ initial state and final state} \tag{3}$$

Markov Chain has a very useful property. With respect to Nth order Markov Chain, current state only depends on last Nth states.

$$P(q_t|q_0, q_1, ..., q_t - 1) = p(q_t|q_{t-N}, ..., q_{t-1}) \tag{4}$$

However, Markov Chain is only suitable to represent a sequence of unambiguious labels. If internal state doesn't represent observed sequence explicitly, then Hidden Markov Model may be a good alternative.

2 Hidden Markov Model

Given a observed sequence, the events in which we are interested may be unable to be observed directly. Therefore, Hidden Markov Model is used to model assumed causal relationship between events observed and hidden events in which we are interested. That is, a hidden state doesn't corresponding to a specific event, but a probability distribution. Observed events are sampled from these distribution. Definition of Hidden Markov Model

$$Q = \{q_1, q_2, ..., q_N\} \text{ a set of hidden states} \tag{5}$$
$$A = \{a_{01}, a_{02}, ..., a_{NN}\} \text{ a transition probability matrix with size of N} \times \text{N} \tag{6}$$
$$O = o_1, ..., o_t \text{ a sequence of observed events} \tag{7}$$
$$B = b_i(o_t) \text{ emission probability expressing the probability of an observation } o_t \text{ being generated from state } i \tag{8}$$
$$q_0, q_F \text{ initial state and final state} \tag{9}$$

Two Properties of Hidden Markov Model
1. Markov Assumption With respect to Nth order Markov Chain, current state only depends on last Nth states.

$$P(q_t|q_0, q_1, ..., q_t - 1) = p(q_t|q_{t-N}, ..., q_{t-1}) \tag{10}$$

2. Output Independence The probability of an output observation o_i depends only on the state that produced the observation q_i and not on any other states or any other observations.

$$P(o_t|q_0, ..., q_t, o_1, ..., o_{t-1}) = P(o_t|q_t) \tag{11}$$

Three Fundamental Problems of Hidden Markov Model
1. Given Hidden Markov Model $\lambda = \{A, B\}$ and an observation sequence O, we need to decide the probability of this observation sequence, $P(O|\lambda)$.
2.Gievn Hidden Markov Model $\lambda = \{A, B\}$ and an observation sequence O, we need to decide the hidden state sequence $Q = q_0, ..., q_t$.
3. Given many observations and the sets of hidden states, we need to estimate HMM's parameters $\{A, B\}$.

3 Compute Forward Liklihood

Given an observation O and Hidden Markov Model $\lambda = \{A, B\}$, we need to estimate the probability of this observation sequence, $P(O|\lambda)$. However, as a specific observation event doesn' t correspond to a specific hidden state, we have no idea what hidden state sequence looks like. Therefore, an intuitive idea is that we can sum over all possbile hidden state sequence which can generate observation O. But exponentionally computation makes this idea impossible. So we turn to dynammic programming for help. The main idea behind dynammic programming is storing intermediate values as it builds up the probability of the observation sequence. an algorithm named forward algorithm, a kind of dynammic programming, perfectly solved this problem.
Forward Algorithm defines a table to store intermediate values and each cell, $\alpha_t(j)$, of the table represents the probability of $q_t = j$ and up to time step t, observation sequence being $o_1, o_2, ..., o_t$.

$$\alpha_t(j) = P(o_1, o_2, ..., o_t, q_t = j|\lambda) \tag{12}$$

$$\begin{aligned}
\alpha_t(j) &= P(o_1, o_2, ..., o_t, q_t = j|\lambda) && (13)\\
&= \sum_{q_{t-1}=k, k\in Q} P(o_1, o_2, ..., o_t, q_{t-1} = k, q_t = j|\lambda) && (14)\\
&= \sum_{q_{t-1}=k, k\in Q} P(o_t|q_t)P(q_t|q_{t-1})P(o_1, o_2, ..., o_{t-1}, q_{t-1} = k|\lambda) && (15)\\
&= \sum_{q_{t-1}=k, k\in Q} P(o_t|q_t = j)P(q_t = j|q_{t-1} = k)\alpha_t(j) && (16)\\
&= \sum_{q_{t-1}=k, k\in Q} b_j(o_t)a_{kj}\alpha_{t-1}(k) && (17)\\
& && (18)
\end{aligned}$$

Therefore, we can calculate $\alpha_t(j)$ iteratively and obtain a table of intermediate values. When time step $t = N$ and hMM jump to end state, we will get liklihood of an observation of sequence $o_1, ..., o_N$.

$$P(O|\lambda) = \alpha_N(F) = \sum_{q_N=i, i\in Q} a_{iF}\alpha_N(i) \tag{19}$$

Formal Definition of Foward Algorithm
1. Initialization

$$\alpha_1(i) = a_{0i}b_i(o_1) \tag{20}$$

2. Recursion

$$\alpha_t(j) = \sum_{q_{t-1}=i, i\in Q} b_j(o_t)a_{ij}\alpha_{t-1}(i) \tag{21}$$

3. Termination

$$P(O|\lambda) = \alpha_N(F) = \sum_{q_N=i, i\in Q} a_{iF}\alpha_N(i) \tag{22}$$

I will give some opinions on how forward algorithm comes up. When calculating $P(O|\lambda)$, we will intuitively choose to take hidden states into consideration since hidden states decide what kind of observation sequence we will observe. However, if we take all hidden states into consideration at once, it will lead to expontional computation problem. Therefore, at a specific tiem step t, we introduce a hidden state like $\alpha_t(j) = P(o_1, o_2, ..., o_t, q_t = j|\lambda)$. In the next step, we introduce hidden state of last time step $t-1$ in order to calcualte current probability in an iterative way like eq(14).

4 Estimate Hidden State Sequence

The basic idea behind the problem of finding hidden state sequence of given observation sequence is selecting a hidden state sequence which maximizes probability $P(Q|O, \lambda)$. Dynammic programming also is applied here in order to avoid exponential computation problem. Similar to Forward Algorithm, An algorithm named The Viterbi Algorithm is proposed to solve this problem. The Viterbi also defines a table to store intermediate values and each cell,$v_t(j)$, of the table reprsents the maximum probability.

$$v_t(j) = max_{q0,q_1,...,q_{t-1}} P(q_0, ..., q_{t-1}, o_1, .., o_t, q_t = j|\lambda) \tag{23}$$

$$\begin{aligned}
v_t(j) &= max_{q0,q_1,...,q_{t-1}} P(q_0, ..., q_{t-1}, o_1, .., o_t, q_t = j|\lambda) && (24)\\
&= max_{q0,q_1,...,q_{t-1}} P(o_t|q_t = j)P(q_t = j|q_{t-1})P(q_0, ..., q_{t-1}, o_1, .., o_{t-1}|\lambda) && (25)\\
&= max_{q_{t-1}} P(o_t|q_t = j)P(q_t = j|q_{t-1})max_{q0,q_1,...,q_{t-2}} P(q_0, ..., q_{t-1}, o_1, .., o_{t-1}|\lambda) && (26)\\
&= max_{q_{t-1}} P(o_t|q_t = j)P(q_t = j|q_{t-1})v_{t-1}(q_{t-1}) && (27)
\end{aligned}$$

Formal Definition of Foward Algorithm
1. Initialization

$$\begin{aligned}
v_1(i) &= a_{0i}b_i(o_1) && (28)\\
bt_1(j) &= a_{0i} && (29)
\end{aligned}$$

2. Recursion

$$\begin{aligned} v_t(j) &= max_{\{}q_{t-1} = i, i \in Q\}b_j(o_t)a_{ij}v_{t-1}(i) \quad (30) \\ bt_t(j) &= max_{\{}q_{t-1} = i, i \in Q\}a_{ij}v_{t-1}(i) \quad (31) \end{aligned}$$

3. Termination

$$\begin{aligned} P(O, Q|\lambda) &= v_N(F) = max_{\{}q_N = i, i \in Q\}a_{iF}v_N(i) \quad (32) \\ P(Q|\lambda) &= v_N(F) = max_{\{}q_N = i, i \in Q\}a_{iF}v_N(i) \quad (33) \end{aligned}$$

5 Train HMM

The essence of training Hidden Markov Model is estimating parameters, matrix A and B, of Hidden Markov Model. Forward-Backward algorithm which is a kind of EM algorithm, is proposed for this problem. One obvious characteristic of this algorithm is that an initial estimate for the paramters is computed and then we use these estiamtes to compute a better estimate, and so on, iteratively improving the probabilities that it learns. With respect to Matrix A, each cell $a_{i,j}$ represents probability of transiting from state i to state j. Therefore, we have estimating formula as following.

$$\hat{a}_{ij} = \frac{\text{expectation of transition from state i to state j}}{\text{expectation of transition from state i}} \quad (34)$$

We first need to calculate transition probability from state i to state j for a given time step t, that is, $\xi(i, j) = P(q_t = i, q_{t+1} = j|O, \lambda)$.

$$\xi(i, j) = P(q_t = i, q_{t+1} = j|O, \lambda) = \frac{P(q_t = i, q_{t+1} = j, O|\lambda)}{P(O|\lambda)} \quad (35)$$

$$\begin{aligned} & P(q_t = i, q_{t+1} = j, O|\lambda) \\ = \; & P(q_{t+1} = j, o_{t+1}, ..., o_T|q_t = i, o_1, ..., o_t, \lambda)P(q_t = i, o_1, ..., o_t, |\lambda) \\ = \; & P(o_{t+2}, ..., o_T|q_{t+1} = j)P(o_{t+1}|q_{t+1} = j)P(q_{t+1} = j|q_t = i) \\ & P(o_1, ..., o_t, q_t = i|\lambda) \end{aligned}$$

Backward probability, $\beta_t(j) = P(o_{t+1}, ..., o_T|q_t = j)$, is introduced here in order to compute above probability in an iterative way. As you can see, backward probability allows us to compute probability from backward in an iterative way.

$$\begin{aligned} \beta_t(j) &= P(o_{t+1}, ..., o_T|q_t = j) \\ &= \sum_{q_{t+1}=k, k \in Q} P(o_{t+1}, ..., o_T, q_{t+1}|q_t = j) \\ &= \sum_{q_{t+1}=k, k \in Q} P(o_{t+2}, ..., o_T|q_{t+1} = k)P(o_{t+1}|q_{t+1} = k)P(q_{t+1} = k|q_t = j) \\ &= \sum_{q_{t+1}=k, k \in Q} \beta_{t+1}(k)a_{jk}b_k(o_{t+1}) \end{aligned}$$

Therefore, $P(q_t = i, q_{t+1} = j|O, \lambda) = \beta_{t+1}(j)b_j(o_{t+1})a_{i,j}\alpha_t(i)$. we can estimate $\hat{a}_{i,j}$ as following.

$$\hat{a}_{i,j} = \frac{\sum_t \xi_t(i,j)}{\sum_t \sum_k \xi(i,k)} \tag{36}$$

With respect to Matrix B, the formula used to estimate $b_j(o_t)$ look like this.

$$b_j(o_t = v_k) = \frac{\text{expectation of emitting } o_t = v_k \text{ when in state j}}{\text{expectation of being in state j}} \tag{37}$$

Here the prbability in state j need first to be computed. Let us denote the probability of being in state j in time step t as $\gamma_t(j)$.

$$\gamma_t(j) = \frac{P(q_t = j, O|\lambda)}{P(O|\lambda)} \tag{38}$$

$$= \frac{P(q_t = j, o_1, ..., o_t|\lambda)P(o_{t+1}, ..., o_T|q_t = j, \lambda)}{P(O|\lambda)} \tag{39}$$

$$= \frac{\alpha_t(j)\beta_t(j)}{\alpha_T(F)} \tag{40}$$

Then we have estimate $\hat{b}_j(o_t) = \frac{\sum_{t,s.t.o_t=v_k} \gamma_t(j)}{\sum_t \gamma_t(j)}$