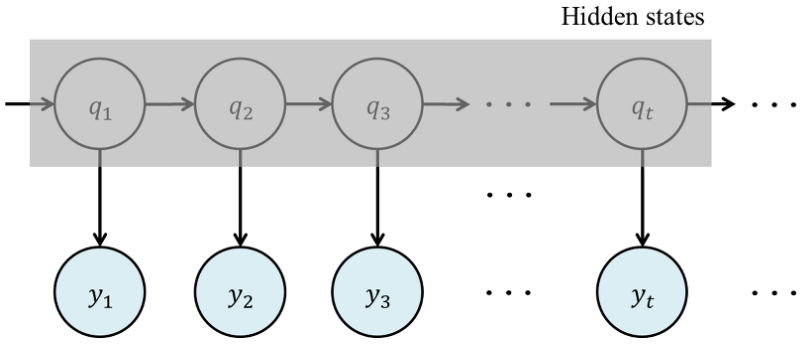
지능형시스템_

[머신 러닝] - 은닉 마르코프 모델 (Hidden Markov Model, HMM)

CHML 2016. 9. 2. 20:50

Markov model은 어떠한 날씨, 주식가격 등과 같은 어떠한 현상의 변화를 확률 모델로 표현한 것이다. Hidden Markov model (HMM)은 이러한 Markov model에 은닉된 state와 직접적으로 확인 가능한 observation을 추가하여 확장한 것이다. HMM은 observation을 이용하여 간접적으로 은닉된 state를 추론하기 위한 문제를 풀기 위해 사용된다.

아래의 [그림 1]은 은닉된 state와 그에 따른 observation의 개념을 나타낸다. HMM을 이용해 우리가 풀고자 하는 문제는 관측 가능한 것은 오직 y_t뿐이며, y_t는 q_t에 종속적으로 발생한다고 할 때, y_t의 sequence를 통해 q_t의 sequence를 추론하는 것이다.



[그림 1] 은닉된 state와 직접적으로 확인 가능한 observation

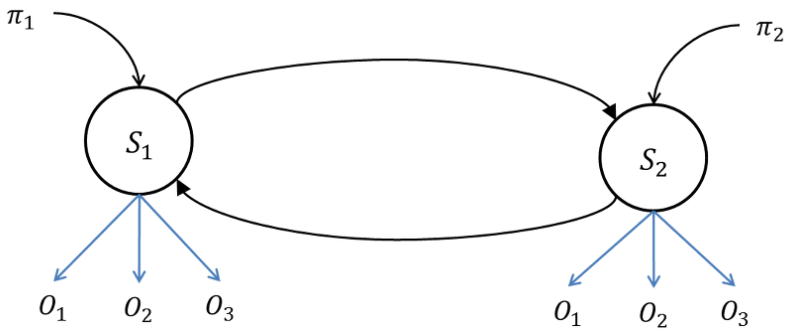
1950년대 이후 forward-backward recursion, Baum-Welch algorithm, Viterbi algorithm이 소개되면서 HMM에 대한 많은 연구가 진행되었고, 1970년대에는 음성 인식 분야에서 HMM이 활발히 응용되기 시작하였다. 현재는 필기체 인식, 동작 인식, 품사 태깅 등과 같이 이전 시간의 데이터에 영향을 받는 순차 데이터에서 패턴을 인식하기 위해 이용되고 있다.

1. HMM의 구조

HMM은 <Q, Y, π, T, E>라는 tuple로 정의되며, 각각의 요소가 의미하는 내용은 아래와 같다.

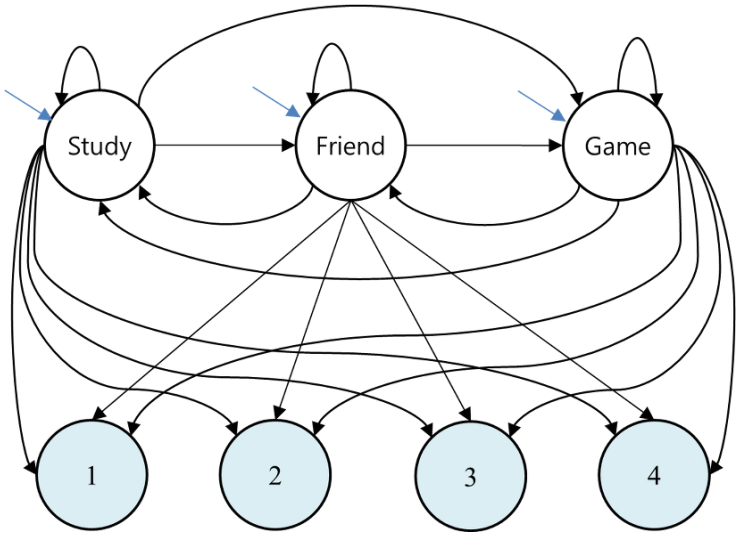
- $Q = \{q_1, q_2, \ldots, q_N\}$ - 은닉된 state들의 집합이다.
- $Y = \{y_1, y_2, \ldots, y_M\}$ - 은닉된 state에서 발생할 수 있는 observation들의 집합이다.
- $\pi : \mathbb{R}^N$ - 초기 state가 q_i일 확률을 나타내는 initial probability $p(q_i)$의 집합이다.
- $\boldsymbol{A} : \mathbb{R}^{N \times N}$ - q_i에서 q_j로 이동할 확률을 나타내는 transition probability $p(q_j|q_i)$의 집합이다.
- $\boldsymbol{B} : \mathbb{R}^{N \times M}$ - q_i에서 y_j가 발생할 확률을 나타내는 emission probability $p(y_j|q_i)$의 집합이다.

일반적으로 HMM은 아래의 [그림 1]과 같은 그래프의 형태로 표현된다. 그래프에서 s_i는 은닉된 state를 나타내며, Q에 있는 원소 중 한 가지 값을 갖는다. o_j는 s_i에서 발생한 observation을 의미하며, Y의 원소 중 한 가지 값을 갖는다.



[그림 2] 2개의 state와 3개의 observation을 갖는 HMM의 구조

HMM의 동작을 설명하기 위한 한 가지 예시로써 지출내역을 통해 과거의 행동을 유추하는 문제를 [그림 3]과 같은 HMM으로 정의할 수 있다. 오늘 하려는 행동은 무엇을 했는지에 영향을 받으며, 과거에 했던 행동은 과거로 돌아가지 않는 한 다시는 관측할 수 없기 때문에 과거의 행동은 은닉된 state에 해당한다. 이 예제에서 $Q = \{study, friend, game\}$으로, 각각 공부, 친구 만나기, 게임을 의미한다. 지출내역은 카드명세서를 통해 과거의 내역이더라도 직접적으로 관측할 수 있기 때문에 observation에 해당한다. 이 예제에서 $Y = \{1, 2, 3, 4\}$이며, 값이 클 수록 돈을 많이 사용했다는 것을 의미한다.



[그림 3] 지출내역을 통해 과거의 행동을 추론하기 위한 HMM

HMM을 구성하기 위해서는 위에서 정의한 Q와 Y 이외에도 HMM의 parameter에 해당하는 initial probabilities π, transition probabilities $\boldsymbol{A}$, emission probabilities $\boldsymbol{B}$를 추정해야 한다. 일반적으로 HMM의 parameter를 추정하기 위해서는 Baum-Welch algorithm이 사용되며, 이 예제에서는 $\pi, \boldsymbol{A}, \boldsymbol{B}$가 각각 아래의 [표 1, 2]와 같이 추정되었다고 가정한다. [표 1]에서 Initial 행은 initial probabilties를 나타낸다.

[표 1] Initial probabilities와 transition probabilities

	Study	Friend	Game
Initial	0.4	0.2	0.4
Study	0.4	0.3	0.3
Friend	0.7	0.1	0.2
Game	0.5	0.2	0.3

[표 2] Emission probabilities

State \ Observation	1	2	3	4
Study	0.7	0.15	0.1	0.05
Friend	0.1	0.2	0.3	0.4
Game	0.5	0.2	0.1	0.2

[표 1]을 통해서는 사람의 행동 양식을 알 수 있다. 예를 들어, 전날 친구를 만나고 오늘도 친구를 만날 확률은 0.1로 매우 작다는 것이나 전날 게임을 했다면 오늘은 0.5의 확률로 공부를 한다는 것이다. 또한, [표 2]를 통해서는 소비 습관을 알 수 있다. 예를 들어, 공부를 하는 날에는 돈을 거의 쓰지 않는다는 것이나 친구를 만나면 0.4의 확률로 많은 돈을 쓴다는 것이다. HMM은 이러한 확률적 사실들을 바탕으로 observation을 통해 은닉된 state를 추론하는 것이다.

2. HMM의 동작

[그림 3]과 [표 1, 2]로 정의된 HMM이 주어졌을 때, 우리는 HMM을 이용하여 아래의 두 가지 사항을 추론할 수 있다.

- 지출 내역이 1 1 4 2일 때, 이러한 지출 내역이 나타날 확률 계산
- 지출 내역이 1 1 4 2일 때, 이러한 지출 내역이 나타날 확률이 가장 큰 과거의 행동 추론

위와 같은 두 가지 동작은 결국 HMM을 이용하여 conditional probability를 계산하는 것과 같다. HMM을 구현할 때는 이러한 두 가지 추론 과정을 어떻게 구현할 것인지에 대해 고려해야 한다. 또한, [그림 3]의 예제에서는 HMM의 인자인 $\pi, \boldsymbol{A}, \boldsymbol{B}$가 주어졌다고 가정하였지만, 원래는 데이터로부터 HMM의 인자를 학습하는 과정을 거쳐야 하기 때문에 학습 연산 역시 HMM의 구현 시 고려되어야 하는 문제이다. 따라서, HMM을 구현할 때는 아래와 같은 3가지 문제를 풀기 위한 연산을 어떻게 구현할 것인지 생각해야 한다.

- HMM의 parameter가 주어졌을 때, 주어진 observation이 나타날 확률 계산
- HMM의 parameter가 주어졌을 때, 주어진 observation이 나타날 확률이 가장 높은 state의 나열을 계산
- 학습 데이터로부터 HMM의 parameter $\theta = \{\pi, \boldsymbol{A}, \boldsymbol{B}\}$를 학습

첫 번째와 두 번째 문제는 각각 dynamic programming 기반의 forward algorithm과 Viterbi algorithm으로 해결할 수 있다. HMM의 parameter를 학습시키는 마지막 문제는 주로 expectation-maximization algorithm (EM algorithm)의 한 형태는 Baum-Welch algorithm으로 해결한다.

2.1. 주어진 observation이 나타날 확률 (likelihood) 계산: forward algorithm

첫 번째 문제는 HMM의 모든 parameter가 결정되었을 때, 주어진 observation $O = o_1, o_2, \ldots, o_T$가 관측될 확률 $p(O|\theta)$를 계산하는 것이다. 먼저, 첫 번째 observation o_1이 나타날 확률은 [식 1]과 같이 계산된다.

$$p(o_1|\theta) = \sum_{i_1} p(q_{i_1}) p(o_1|q_{i_1}) \qquad (1)$$

그다음, 첫 번째와 두 번째 observation이 연속으로 나타날 확률은 다음과 같다.

$$p(o_1, o_2|\theta) = \sum_{i_1} p(q_{i_1}) p(o_1|q_{i_1}) \sum_{i_2} p(s_{i_2}|q_{i_1}) p(o_2|q_{i_2}) \qquad (2)$$

따라서, 시간 T까지의 observation $O = o_1, o_2, \ldots, o_T$가 나타날 확률은 [식 3]과 같다.

$$p(O|\theta) = \sum_{i_1} \sum_{i_2} \cdots \sum_{i_T} p(q_{i_1}) p(o_1|q_{i_1}) p(q_{i_2}|q_{i_1}) p(o_2|q_{i_2}) \cdots p(q_{i_T}|q_{i_{T-1}}) p(o_T|q_{i_T}) \qquad (3)$$

그러나 [식 3]을 계산하는 것은 $O(2TN^T)$의 막대한 time complexity가 요구된다. 이를 해결하기 위해 HMM은 dynamic programming 기반의 $O(TN^2)$의 time complexity를 갖는 forward algorithm을 이용한다. Forward algorithm에서는 새로운 변수인 $\alpha_t(q_j)$를 아래와 같이 정의한다.

$$\alpha_t(q_j) = p(o_1, o_2, \ldots, o_t, s_t = q_j|\theta) \qquad (4)$$

또한, $\alpha_t(q_j)$는 아래와 같이 재귀적인 형태로 계산할 수 있다.

$$\alpha_t(q_j) = \sum_{i=1}^{N} \alpha_{t-1}(q_i) p(q_j|q_i) p(o_t|q_j) \qquad (5)$$

아래의 [알고리즘 1]은 HMM에서 forward algorithm을 이용하여 [식 3]의 likelihood를 계산하는 과정을 나타낸다.

Algorithm 1: Forward algorithm

Input : initial probabilities π,
transition probabilities T,
emission probabilities E,
observation $O = o_1, o_2, ..., o_T$

Output : likelihood $p(O|\pi, T, E)$

for $j = 1 : N$ **do**
  $\alpha_1(q_j) \leftarrow p(q_j)p(o_1|q_j)$
end
for $t = 2 : T$ **do**
  for $j = 1 : N$ **do**
    $\alpha_t(q_j) \leftarrow \sum_{i=1}^{N} \alpha_{t-1}(q_i)p(q_j|q_i)p(o_t|q_j)$
  end
end
$p(O|\pi, T, E) \leftarrow \sum_{j=1}^{N} \alpha_T(q_j)$
return $p(O|\pi, T, E)$

위의 [그림 3]의 예제에서 지출내역이 1, 3, 2, 1, 2인 (O = 1, 3, 2, 1, 2) 경우, forward algorithm을 이용한 likelihood는 0.00134349로 계산된다.

2.2. 주어진 observation이 나타날 확률이 가장 큰 state 계산: Viterbi algorithm

두 번째 문제는 주어진 observation이 나타날 확률이 가장 state의 연속 $S^* = s_1^* s_2^* \dots s_T^*$를 찾는 것이다. 이러한 과정을 decoding이라고 한다. HMM에서는 앞의 forward algorithm과 유사하게 dynamic programming 기반의 Viterbi algorithm을 이용하여 최적의 state를 찾는다. 아래의 [알고리즘 2]는 viterbi algorithm의 동작을 보여준다.

Algorithm 2: Viterbi algorithm

Input : initial probabilities π,
transition probabilities T,
emission probabilities E,
observation $O = o_1, o_2, ..., o_T$

Output : $s_1^*, s_2^*, ..., s_T^*$

for $j = 1 : N$ **do**
  $v_1(q_j) = p(q_j)p(o_1|q_j)$
end
$s_1^* \leftarrow \arg\max_{q \in Q} p(q_j)p(o_1|q_j)$
for $t = 2 : T$ **do**
  for $j = 1 : N$ **do**
    $v_t(q_j) \leftarrow \max_{q \in Q} v_{t-1}(q)p(q_j|q)p(o_t|q_j)$
    $s_t^* \leftarrow \arg\max_{q \in Q} v_{t-1}(q)p(q_j|q)p(o_t|q_j)$
  end
end
return $s_1^*, s_2^*, ..., s_T^*$

위의 [그림 3]의 예제에서 지출내역이 1, 3, 2, 1, 2인 (O = 1, 3, 2, 1, 2) 경우, Viterbi algorithm을 이용하여 찾은 최적의 state는 study, friend, game, study, game이 된다. 따라서, 우리는 과거의 지출내역이 1, 3, 2, 1, 2인 경우, 가장 높은 확률로 과거 5일동안 공부, 친구 만나기, 게임, 공부, 게임를 했었다고 생각할 수 있다.

2.3. HMM의 parameter 학습: Baum-Welch algorithm

HMM의 parameter 학습은 observation들의 집합인 $\boldsymbol{O} = \{O^{(1)}, O^{(2)}, \ldots, O^{(K)}\}$로부터 최적의 parameter $\theta^* = \{\pi^*, \boldsymbol{A}^*, \boldsymbol{B}^*\}$를 추정하는 것이다.

HMM의 parameter 학습에는 주로 EM algorithm의 한 종류인 Baum-Welch algorithm가 이용된다. 이 알고리즘에서는 먼저 $\boldsymbol{O}$에 포함된 observation들이 나타날 확률을 의미하는 likelihood $p(\boldsymbol{O}, \boldsymbol{S}|\theta)$를 정의한다. 여기에서 $\boldsymbol{S}$는 $\boldsymbol{O}$에 포함된 각 observation에 대해 은닉된 state가 어떻게 변화하였는지를 나타낸다. 만약 $\boldsymbol{S}$를 정확히 알고 있다면, $p(\boldsymbol{O}, \boldsymbol{S}|\theta)$는 아래와 같이 계산된다.

$$p(\boldsymbol{O}, \boldsymbol{S}|\theta) = \prod_{k=1}^{K} (p(s_1^{(k)}) p(o_1^{(k)}|s_1^{(k)}) \prod_{t=2}^{T} p(s_t^{(k)}|s_{t-1}^{(k)}) p(o_t^{(k)}|s_t^{(k)}) \quad (6)$$

양변에 log를 취하면 [식 7]과 같다.

$$\log p(\boldsymbol{O}, \boldsymbol{S}|\theta) = \sum_{k=1}^{K} \left\{ (p(s_1^{(k)}) p(o_1^{(k)}|s_1^{(k)}) + \sum_{t=2}^{T} \log p(s_t^{(k)}|s_{t-1}^{(k)}) + \sum_{t=2}^{T} \log p(o_t^{(k)}|s_t^{(k)}) \right\} \quad (7)$$

그러나 대부분의 경우에는 $\boldsymbol{O}$에 대한 $\boldsymbol{S}$를 모르기 때문에 아래와 같이 log-likelihood를 변형하여 $Q(\theta, \theta')$를 정의한다. [식 8]에서 θ는 현재의 parameter, θ'는 이전의 parameter를 의미한다. 왜 이전의 parameter를 고려하여 식을 정의하는지를 비롯하여 Baum-Welch algorithm이 왜 다음과 같이 전개되는 지를 알기 위해서는 EM algorithm에 대한 이해가 필요하다.

$$Q(\theta, \theta') = \sum_{s_1, s_2, \ldots, s_T} \log\{p(\boldsymbol{O}, \boldsymbol{S}|\theta)\} \, p(\boldsymbol{O}, \boldsymbol{S}|\theta') \quad (8)$$

[식 7]을 [식 8]에 대입하면, Baum-Welch algorithm을 통해 최대화하고자 하는 $Q(\theta, \theta')$는 다음과 같다.

$$Q(\theta, \theta') = \sum_{s_1, s_2, \ldots, s_T} \sum_{k=1}^{K} \left\{ (p(s_1^{(k)}) p(o_1^{(k)}|s_1^{(k)}) + \sum_{t=2}^{T} \log p(s_t^{(k)}|s_{t-1}^{(k)}) + \sum_{t=2}^{T} \log p(o_t^{(k)}|s_t^{(k)}) \right\} p(\boldsymbol{O}, \boldsymbol{S}|\theta') \quad (9)$$

마지막으로, 라그랑주 승수법 (Lagrange multiplier method)를 이용하여 확률의 정의에 의한 제약 조건을 $Q(\theta, \theta')$에 추가하면, 최종 목적 함수 $L(\theta, \theta')$는 [식 10]과 같이 주어진다.

$$L(\theta, \theta') = Q(\theta, \theta') - \lambda_{\pi}\left(\sum_{i=1}^{N} p(q_i) - 1\right) - \sum_{i=1}^{N} \lambda_{T_i}\left(\sum_{j=1}^{N} p(q_j|q_i) - 1\right) - \sum_{i=1}^{N} \lambda_{E_i}\left(\sum_{j=1}^{M} p(y_j|q_i) - 1\right) \quad (10)$$

따라서, HMM의 학습은 $L(\theta, \theta')$를 최대화하는 $\theta^* = \{\pi^*, \boldsymbol{A}^*, \boldsymbol{B}^*\}$를 찾는 것과 같다. $L(\theta, \theta')$을 최대화하는 각각의 parameter들은 아래와 같이 계산된다. 여기에서 $\pi_i = p(q_i), A_{ij} = p(q_j|q_i), B_{ij} = p(y_j|q_i)$를 의미한다.

$$\begin{aligned}\frac{\partial L(\theta,\theta')}{\partial \pi_i} &= \frac{\partial}{\partial \pi_i}\left\{\sum_{s_1,s_2,\cdots,s_T}\sum_{k=1}^{K}\log p(s_1^{(k)})\,p(\boldsymbol{O},\boldsymbol{S}|\theta')\right\} - \lambda_\pi \\ &= \frac{\partial}{\partial \pi_i}\left\{\sum_{j=1}^{N}\sum_{k=1}^{K}\log p(s_1^{(k)}=q_j)\,p(\boldsymbol{O},s_1^{(k)}=q_j|\theta')\right\} - \lambda_\pi \\ &= \sum_{k=1}^{K}\frac{p(\boldsymbol{O},s_1^{(k)}=q_i|\theta')}{p(s_1^{(k)}=q_i)} - \lambda_\pi = 0\end{aligned}$$

$$\therefore\ \pi_i = \frac{\sum_{k=1}^{K} p(\boldsymbol{O},\ s_1^{(k)}=\ q_i|\theta')}{\sum_{j=1}^{N}\sum_{k=1}^{K} p(\boldsymbol{O},\ s_1^{(k)}=\ q_j|\theta')} \qquad (11)$$

그다음, A_{ij}는 [식 12]와 같이 계산된다.

$$\begin{aligned}\frac{\partial L(\theta,\theta')}{\partial A_{ij}} &= \frac{\partial}{\partial A_{ij}}\left\{\sum_{s_1,s_2,\cdots,s_T}\sum_{k=1}^{K}\sum_{t=2}^{T}\log p(s_t^{(k)}|s_{t-1}^{(k)})\,p(\boldsymbol{O},\boldsymbol{S}|\theta')\right\} - \lambda_{A_i} = 0 \\ &\Leftrightarrow \frac{\partial}{\partial \pi_i}\left\{\sum_{j=1}^{N}\sum_{l=1}^{N}\sum_{k=1}^{K}\sum_{t=2}^{T}\log p(s_t^{(k)}=q_l|s_{t-1}^{(k)}=q_j)\,p(\boldsymbol{O},s_{t-1}^{(k)}=q_j,s_t^{(k)}=q_l|\theta')\right\} - \lambda_{A_i} = 0 \\ &\Leftrightarrow \sum_{k=1}^{K}\sum_{t=2}^{T}\frac{p(\boldsymbol{O},s_{t-1}^{(k)}=q_i,s_t^{(k)}=q_j|\theta')}{p(s_t^{(k)}=q_j|s_{t-1}^{(k)}=q_i)} - \lambda_{A_i} = 0\end{aligned}$$

$$\therefore\ A_{ij} = \frac{\sum_{k=1}^{K}\sum_{t=2}^{T} p(s_{t-1}^{(k)}=q_i,\ s_t^{(k)}=q_j|O^{(k)},\theta')}{\sum_{k=1}^{K}\sum_{t=2}^{T} p(s_{t-1}^{(k)}=q_i|O^{(k)},\theta')} \qquad (12)$$

마지막으로, B_{ij}는 [식 13]과 같이 계산된다.

$$\begin{aligned}\frac{\partial L(\theta,\theta')}{\partial B_{ij}} &= \frac{\partial}{\partial A_{ij}}\left\{\sum_{s_1,s_2,\cdots,s_T}\sum_{k=1}^{K}\sum_{t=1}^{T}\log p(o_t^{(k)}|s_t^{(k)})\,p(\boldsymbol{O},\boldsymbol{S}|\theta')\right\} - \lambda_{B_i} = 0 \\ &\Leftrightarrow \frac{\partial}{\partial \pi_i}\left\{\sum_{i=1}^{N}\sum_{k=1}^{K}\sum_{t=1}^{T}\log p(o_t^{(k)}=y_j|s_t^{(k)}=q_i)\,p(\boldsymbol{O},\ s_t^{(k)}=q_i|\theta')\right\} - \lambda_{B_i} = 0 \\ &\Leftrightarrow \sum_{k=1}^{K}\sum_{t=1}^{T}\frac{p(\boldsymbol{O},s_t^{(k)}=q_i|\theta')I(o_t^{(k)}=y_j)}{p(o_t^{(k)}=y_j|s_t^{(k)}=q_i)} - \lambda_{B_i} = 0\end{aligned}$$

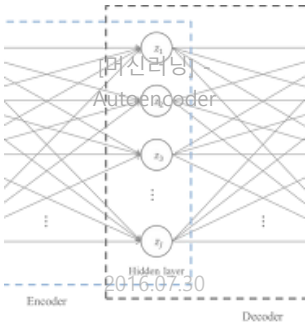
$$\therefore\ B_{ij} = \frac{\sum_{k=1}^{K}\sum_{t=1}^{T} p(s_t^{(k)}=q_i|O^{(k)},\theta')I(o_t^{(k)}=q_j)}{\sum_{k=1}^{K}\sum_{t=1}^{T} p(s_t^{(k)}=q_i|O^{(k)},\theta')} \qquad (13)$$

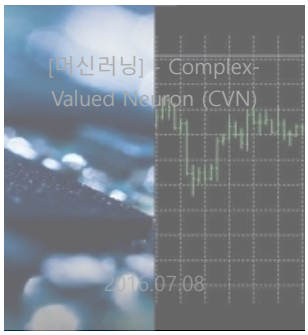
'지능형시스템_' 카테고리의 다른 글

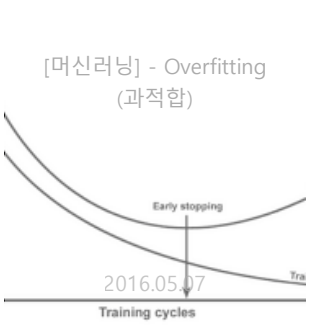
[머신러닝] - RNNs (Recurrent Neural Networks) (3) 2016.10.06
[머신 러닝] - 은닉 마르코프 모델 (Hidden Markov Model, HMM) (11) 2016.09.02
[머신러닝] - Autoencoder (3) 2016.07.30
[머신러닝] - Complex-Valued Neuron (CVN) (0) 2016.07.08
[머신러닝] - Overfitting (과적합) (0) 2016.05.07
[지능형시스템] - Artificial Bee Colony(ABC) Algorithm (0) 2016.04.13

'지능형시스템_' Related Articles

more







11 Comments

FrogM4n 2017.07.20 21:55
좋은글 감사합니다. ML공부하는데 도움 많이 되었습니다!!!

ㅇㅇ 2018.08.20 11:46
좋은 자료 감사합니다. 특정 state의 특정 observation에 대한 emission probability 가 정의되지 않은 경우에는 어떻게 해결하나요 ?

CHML 2018.08.25 11:08 신고

초기에는 emission probability를 랜덤으로 설정하신 후에, (식 13)을 이용하여 학습하시면 됩니다.

theorist17 2018.10.03 00:45
이렇게 알기 쉽게 글을 쓰시다니.. 정말 고마워요~

정말 2020.02.04 15:43
감사합니다! :) :)

Corgilog 2021.06.28 09:26 신고
혹시 위 pseudo-code 출처 알 수 있을까요? 책 같은데 책 이름이 알고 싶어요!

CHML 2021.07.08 20:15 신고
여러 자료를 참고해서 제가 검증하면서 작성한 내용이라 따로 생각나는 출처는 없네요. 죄송합니다.

광장의꽃향기 2021.09.08 23:37 신고
equation (2) 에서 s 가 뭘 뜻하는건가요?

천국의9번째계단 2021.11.08 14:26 신고
지나가다가 대신 답변남깁니다. state 1에서 state2로 넘어가는 transition으로, q2|q1인데 오타가 난거 같네요. 감사합니다.

천국의9번째계단 2021.11.08 14:29 신고
좋은 글 감사합니다. 내용 덧붙여도 될까요? 식 (6)에서 식 (7)으로 넘어가는 과정에서 log를 붙이는 이유는 time sequence가 길어지면, probability의 multiplication으로 인해 forward 값이 매우 작아져 precision excision이 일어나기 때문입니다. 따라서 양번에 log를 취해줌으로써 소실을 방지할 수 있습니다. 감사합니다.

Spike 2022.08.07 20:55
설명을 정말 읽기 좋게 잘 써 주셨네요. 감사합니다. 2.2의 "확률이 가장 state의 연속" 은 "확률이 가장 큰 state의 연속" 의 오타인가요? "연속"이란 말은 영어로는 sequence가 될 것 같은데 한국어로는 뭐라고 써야 할지 좀 애매하네요. "어떤 state들의 순서를 거쳤을 때, 주어진 observation이 관찰될 확률이 가장 높은가" 를 말씀하시는 거죠?

Name Password
여러분의 소중한 댓글을 입력해주세요
Secret Send