Connectionist Temporal Classification : Labelling Unsegmented Sequence Data with Recurrent Neural Networks

Graves, A., Fernández, S., Gomez, F., & Schmidhuber, J. (2006, June). Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning* (pp. 369-376).

Contents

1 Introduction

음성 레이블링

input

target

사과가 먹고 싶다

labeling

0~10ms: 사
10ms~23ms: 과
23ms~30ms:가
30ms~35ms: blank
...

- 음성과 같은 sequence data 에 labeling 하는 것은 매우 어려움.
- labeling 을 하더라도 사람마다 다르게 할 수 있음. (e.g. 0~9ms: 사)

=> 명시적인 alignment (즉, seqeunce에 대한 labeling) 없이도 음성인식 모델을 학습할 수 있는 방법 필요

본 논문에서는 이 방법으로 CTC 제안.
이전에는 HMM이 쓰였음.

2 Background

HMM(Hidden Markov Model)

Markov chain 을 전제로 한 모델로 음소(또는 단어) 시퀀스를 모델링하는데 자주 쓰였음.

Markov chain
: Markov Property (마코프 성질)을 지닌 이산확률과정(discrete-time stochastic process)를 가리킴.
러시아 수학자 마코프가 러시아어 문헌의 글자들의 순서에 관한 모델을 구축하기 위해 제안된 개념.

key idea: 어떤 state q_i가 나타날 확률은 단지 그 이전 상태 q_{i-1} 에만 의존한다.
=> 즉, 어떤 상태에서 다른 상태로의 전이(transition)은 그동안의 상태전이에 대한 이력(history)를 필요로 하지 않고, 바로 **직전상태에서의 전이로 추정 가능!**

수식1 MARKOV PROPERTY(마코프 성질)

$$P(q_i|q_1,\ldots,q_{i-1}) = P(q_i|q_{i-1})$$

2 Background

HMM(Hidden Markov Model)

HMM은 각 상태가 Markov chain을 따르되, 은닉(Hidden)되어 있다고 가정.

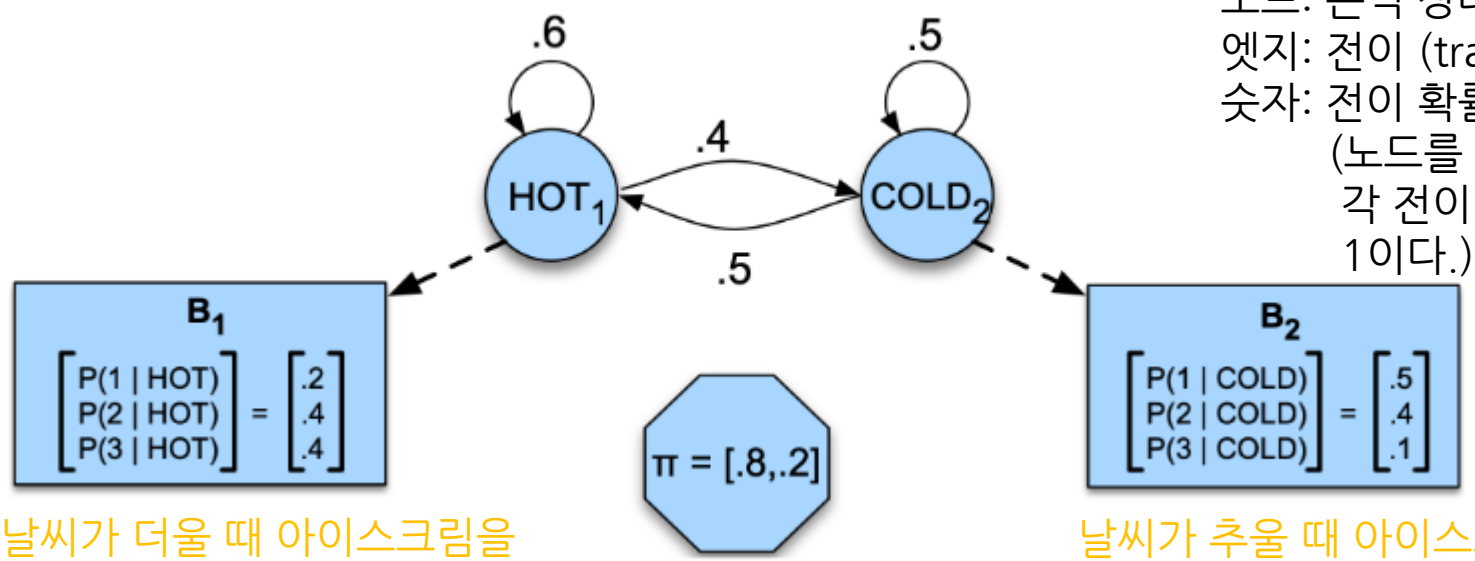
예시: 아이스크림 소비 기록으로 당시 날씨 맞추기

주어진 데이터: 당시의 아이스크림 소비기록

우리가 가진 데이터를 데이터에 true hidden state 라는 노이즈가 낀 형태라고 보는 것.

M/d	total	weather
9/1	3	?
9/2	1	?
9/3	3	?
…	…	…

그림1 HIDDEN MARKOV MODEL



노드: 은닉 상태(hidden state)
엣지: 전이 (transition)
숫자: 전이 확률
(노드를 기준으로 각 전이 확률의 합은 1이다.)

날씨가 더울 때 아이스크림을 1,2,3개 사먹을 확률

날씨가 추울 때 아이스크림을 1,2,3개 사먹을 확률

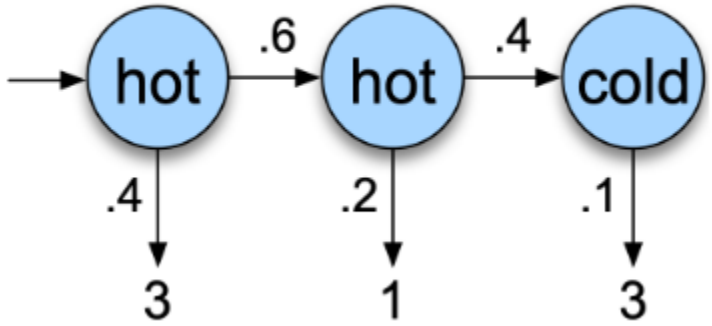
2 Background

예시: 아이스크림 소비 기록으로 당시 날씨 맞추기

주어진 데이터: 당시의 아이스크림 소비기록

M/d	total	weather
9/1	3	?
9/2	1	?
9/3	1	?
…	…	…

그림2 날씨가 [HOT, HOT, COLD]일 때 아이스크림 개수 [3, 1, 3]이 관측될 LIKELIHOOD



수식3 날씨가 [HOT, HOT, COLD]일 때 아이스크림 개수 [3, 1, 3]이 관측될 LIKELIHOOD

$$P(3\quad 1\quad 3, hot\quad hot\quad cold) = P(hot|start) \times P(hot|hot) \times P(cold|hot)$$
$$\times P(3|hot) \times P(1|hot) \times P(3|cold)$$
$$= 0.8 \times 0.6 \times 0.3 \times 0.4 \times 0.2 \times 0.1 = 0.001536$$

2 Background

예시: 아이스크림 소비 기록으로 당시 날씨 맞추기

주어진 데이터: 당시의 아이스크림 소비기록

M/d	total	weather
9/1	3	?
9/2	1	?
9/3	1	?
…	…	…

표1 LIKELIHOOD (NAIVE)

q1	q2	q3	p(3|q1)	p(1|q2)	p(3|q3)	p(q1|q0)	p(q2|q1)	p(q3|q2)	value	P(3 1 3)
hot	hot	hot	0.4	0.2	0.4	0.8	0.6	0.6	0.009216	**0.02856**
hot	hot	cold	0.4	0.2	0.1	0.8	0.6	0.4	0.001536	
hot	cold	hot	0.4	0.5	0.4	0.8	0.4	0.5	0.0128	
hot	cold	cold	0.4	0.5	0.1	0.8	0.4	0.5	0.0032	
cold	hot	hot	0.1	0.2	0.4	0.2	0.5	0.6	0.00048	
cold	hot	cold	0.1	0.2	0.1	0.2	0.5	0.4	0.00008	
cold	cold	hot	0.1	0.5	0.4	0.2	0.5	0.5	0.001	
cold	cold	cold	0.1	0.5	0.1	0.2	0.5	0.5	0.00025	

계산량이 매우 많아짐. 비효율적.

=> 이를 해결하기 위해 동적 프로그래밍(dynamic programming) 기법 사용

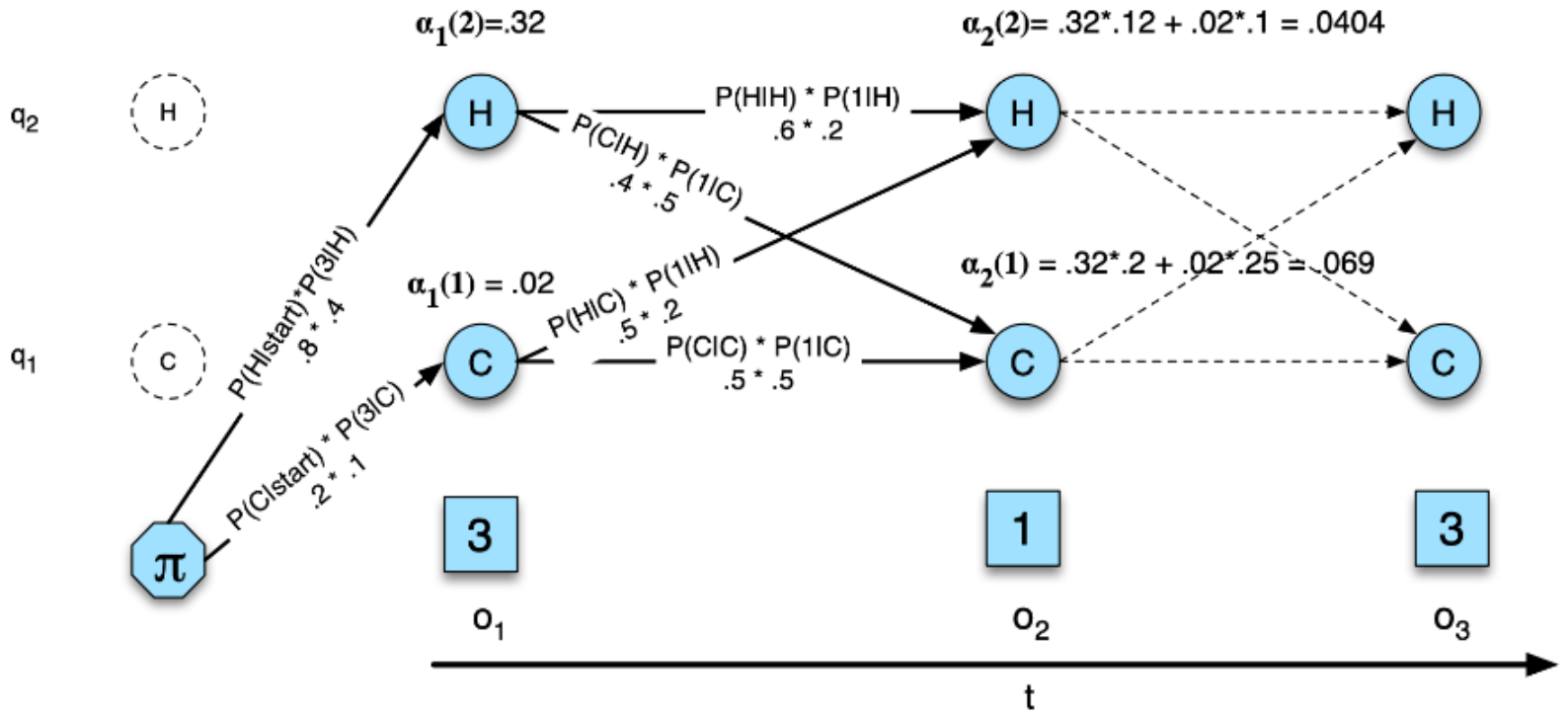
2 Background

예시: 아이스크림 소비 기록으로 당시 날씨 맞추기
주어진 데이터: 당시의 아이스크림 소비기록

Dynamic Programming:
중복되는 계산을 저장했다가 나중에 다시 사용하는 것.

2 Background

그림4 FORWARD ALGORITHM



$\alpha_t(k)$ t: 시점, k: 상태

$\alpha_1(1)$: O_1 이 3개로 관측되었는데, 날씨가 cold 일 확률

$\alpha_2(1)$: O_1 이 3개, O_2가 1개로 관측되었을 때 2의 날씨가 cold일 확률

수식4 FORWARD COMPUTATION

$$\alpha_1(1) = P(cold|start) \times P(3|cold)$$

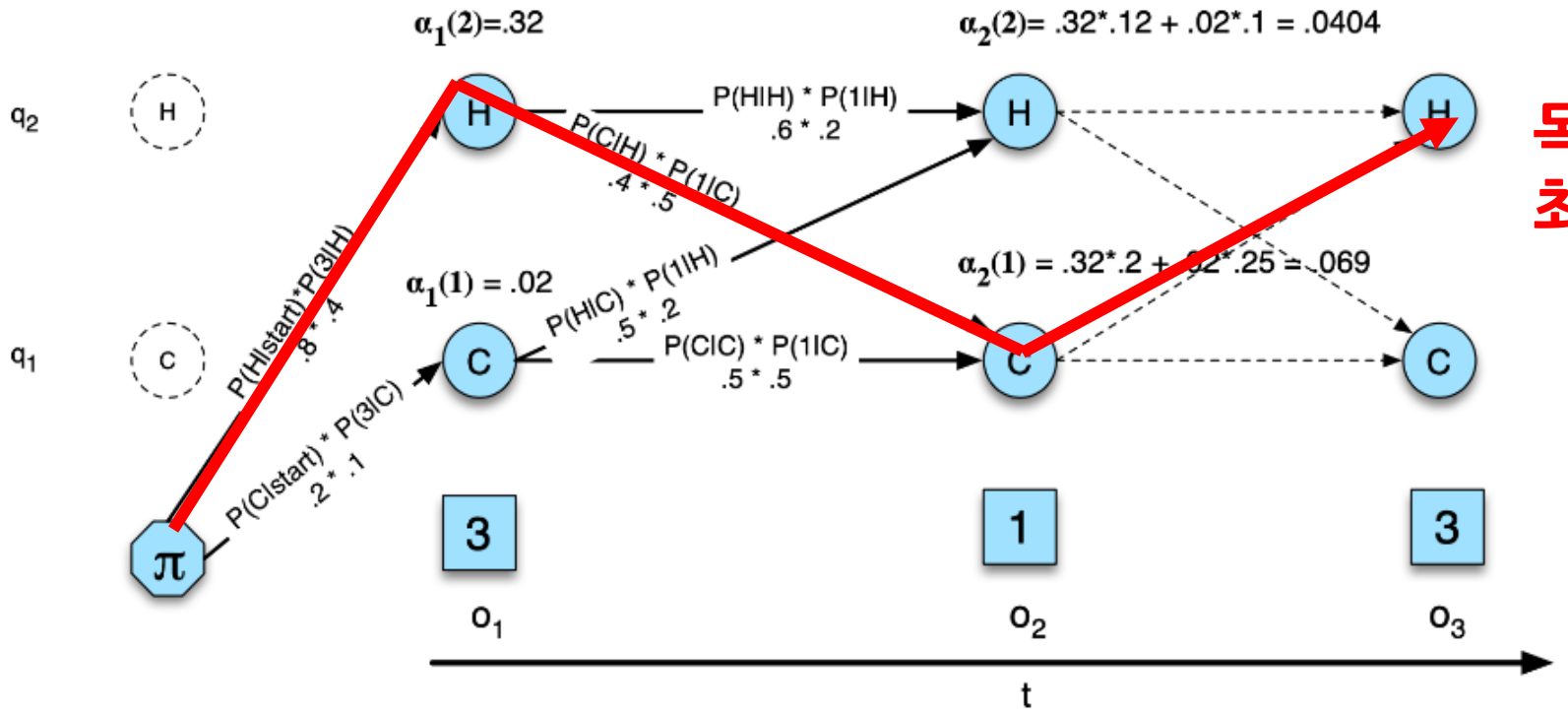
$$\alpha_1(2) = P(hot|start) \times P(3|hot)$$

$$\alpha_2(1) = \alpha_1(1) \times P(cold|cold) \times P(1|cold) + \alpha_1(2) \times P(cold|hot) \times P(1|cold)$$

첫번째에 3개인데 날씨가 cold일 확률 x 두번째에 1개인데 cold일 확률
+ 첫번째에 3개에 날씨가 hot일 확률 x 두번째에 1개인데 cold일 확률

2 Background

그림4 FORWARD ALGORITHM



목표:
최적의 path(경로) 구하기!

$\alpha_t(k)$ t: 시점, k: 상태

$\alpha_1(1)$: O_1 이 3개로 관측되었는데, 날씨가 cold 일 확률

$\alpha_2(1)$: O_1 이 3개, O_2가 1개로 관측되었을 때 2의 날씨가 cold일 확률

수식4 FORWARD COMPUTATION

$$\alpha_1(1) = P(cold|start) \times P(3|cold)$$
$$\alpha_1(2) = P(hot|start) \times P(3|hot)$$
$$\alpha_2(1) = \alpha_1(1) \times P(cold|cold) \times P(1|cold) + \alpha_1(2) \times P(cold|hot) \times P(1|cold)$$

첫번째에 3개인데 날씨가 cold일 확률 x 두번째에 1개인데 cold일 확률
+ 첫번째에 3개에 날씨가 hot일 확률 x 두번째에 1개인데 cold일 확률

2 Background

Decoding

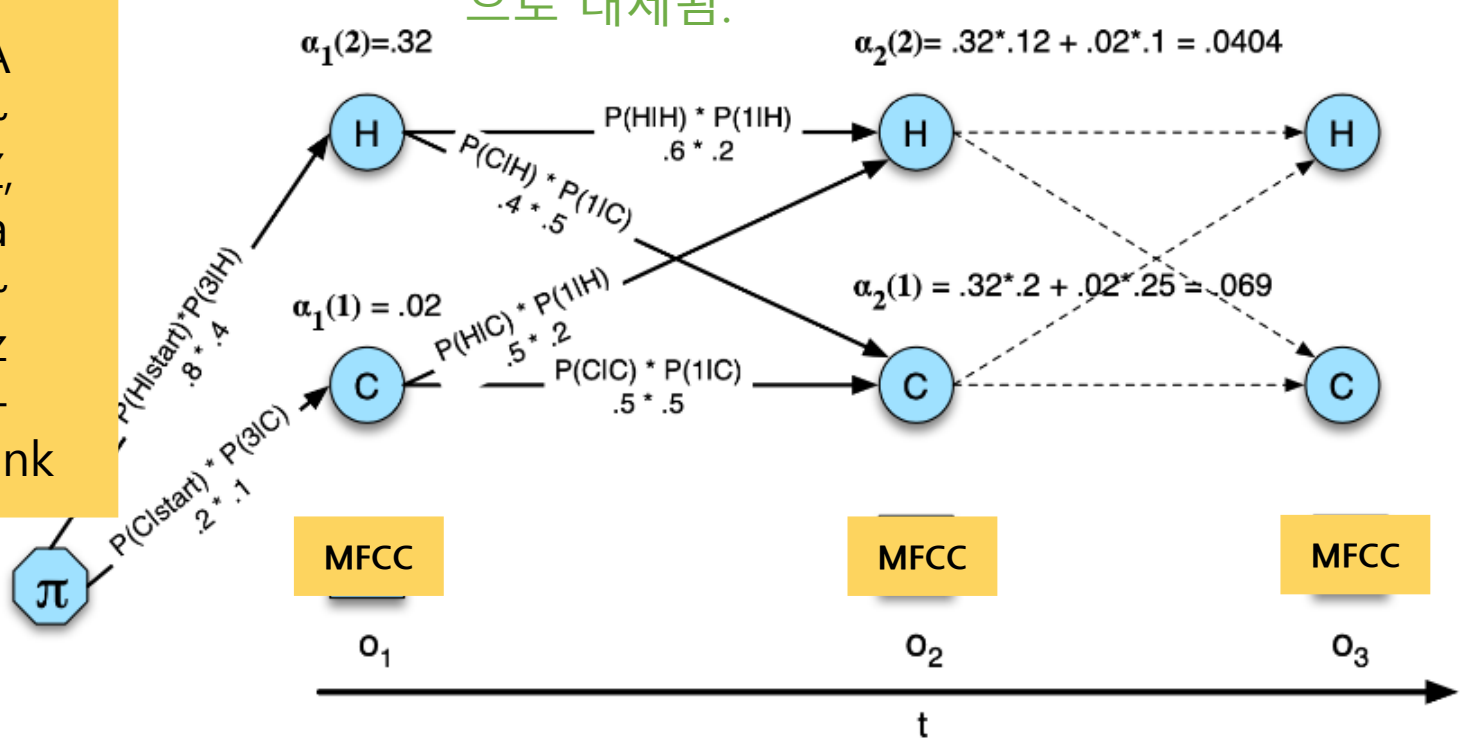
모델과 관측된 시퀀스 O가 주어졌을 때, hidden state 의 best path 를 찾는 과정

그림4 FORWARD ALGORITHM

C, H가 알파벳 으로 대체됨.

Label

A ~ Z, a ~ z + blank



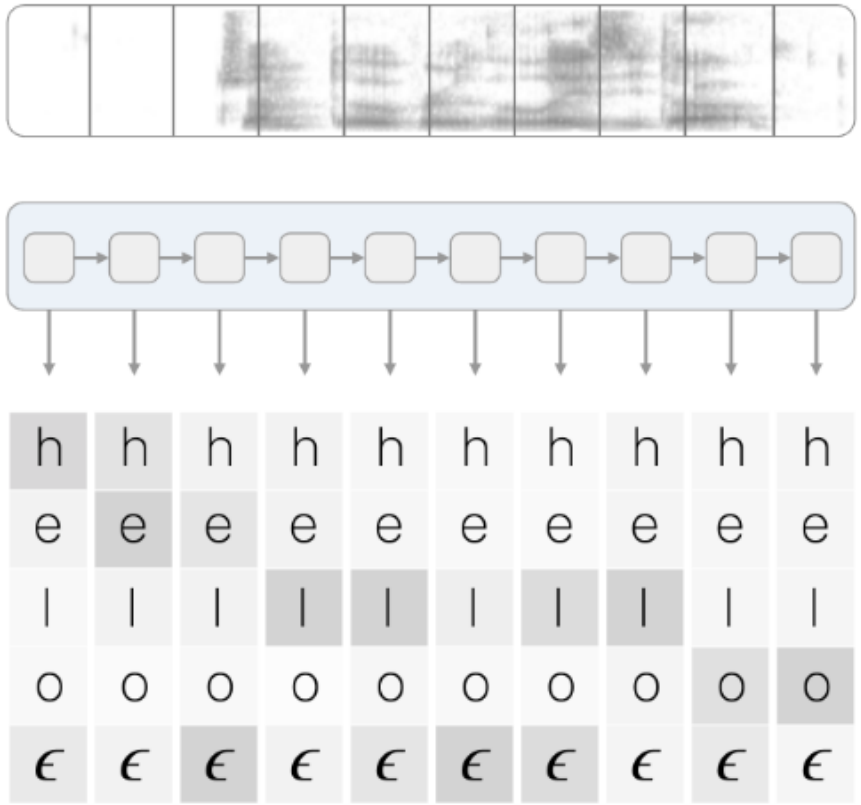
보통 25ms로 나눈 음성 feature

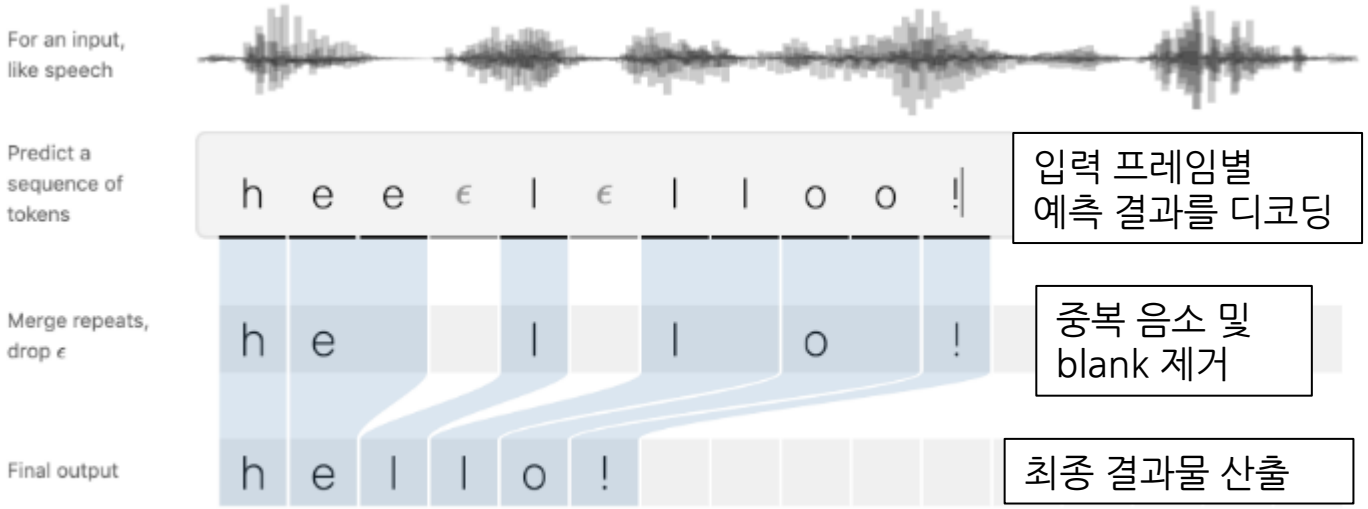
3 CTC: Connectionist Temporal classification

CTC overview

CTC 기법은 시퀀스 분류를 위한 딥러닝 모델 맨 마지막에 loss 및 gradient 계산 레이어로 구현됨.

그림1 CTC INPUT





CTC prediction overview

3 CTC: Connectionist Temporal classification

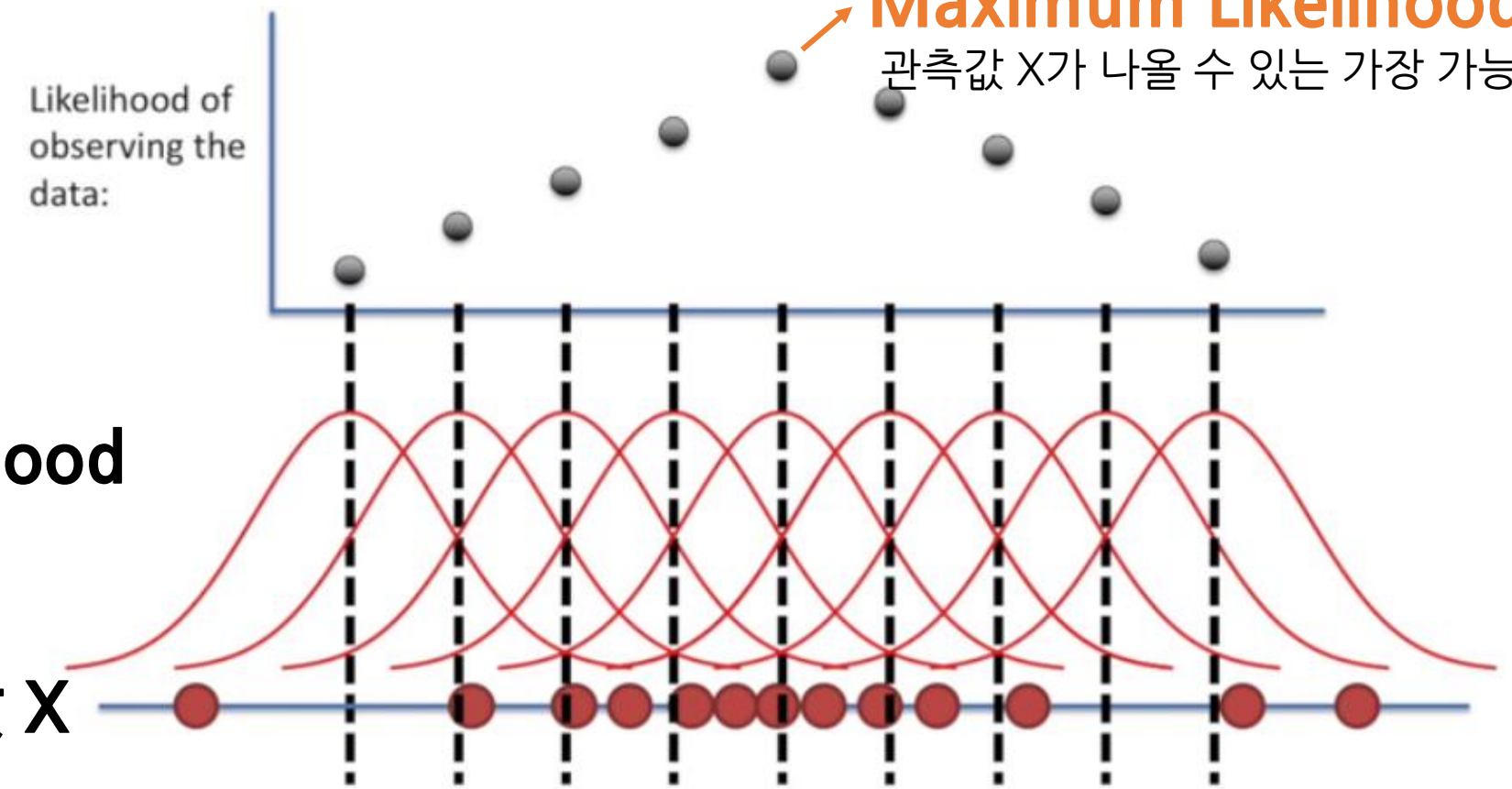
우리의 목표
: 음성 시퀀스 x 가 주어졌을 때 label sequence l 맞추기
즉, likelihood(우도) 를 최대화하는 모델의 parameter 를 찾는 것.

- 입력 시퀀스의 각 프레임에 대해 모두 레이블이 있다면 그냥 Cross entropy 를 최소화하면 됨.
- But! 우리는 입력 시퀀스에 대해 프레임별 레이블이 없음(시간, 인력 등 비용소모, 레이블링의 어려움 등)
- 그래서 likelihood 를 최대화하려고 함. 프레임별로 가장 가능한 분포를 찾으려함.
- likelihood 최대화를 위해 likelihood 값을 구해야 함.
- Forward/Backward Computation 을 구해서 likelihood 값을 구할 수 있다.

https://jjangjjong.tistory.com/41
likelihood: 확률의 확률.(주어진 관측값이 해당 확률분포에서 나왔을 확률)
$likelihood = L$(확률분포D | 관측값 X) (우도, 가능도)
maximum likelihood; **관측값이 나올 확률이 가장 큰 확률분포찾기!**

3 CTC: Connectionist Temporal classification

Maximum Likelihood



Maximum Likelihood

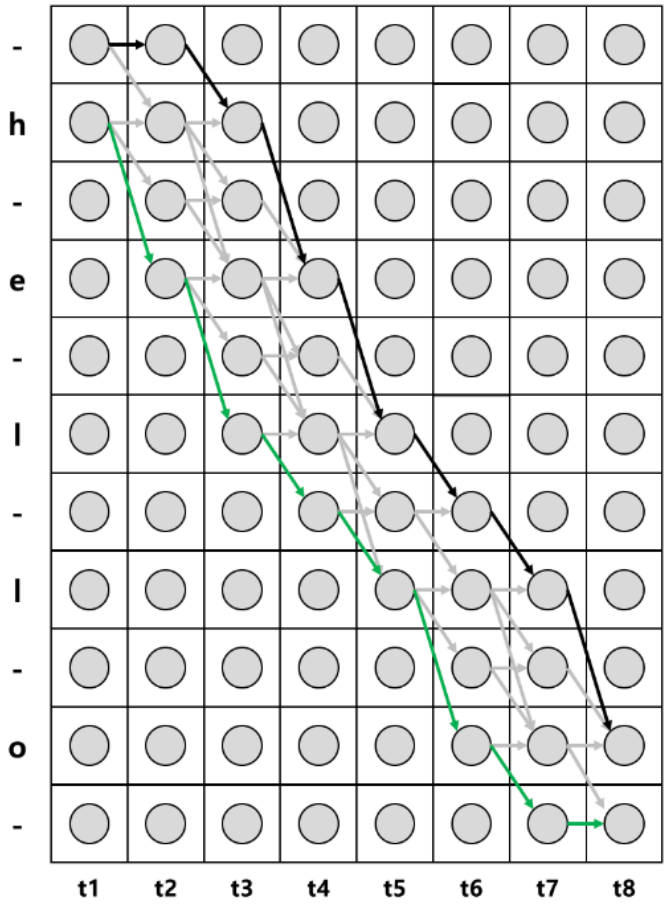
관측값 X가 나올 수 있는 가장 가능한 확률분포

3 CTC: Connectionist Temporal classification

CTC overview

CTC 기법은 시퀀스 분류를 위한 딥러닝 모델 맨 마지막에 loss 및 gradient 계산 레이어로 구현됨.

그림4 ALL POSSIBLE PATHS



정답 레이블 시퀀스 l: h,e,l,l,o
l' : l 의 시작과 끝, 레이블 사이에 blank 추가
- h - e - l - l - o -
따라서 길이는 2|l| + 1 = 11
옆에 그림에서는 편의를 위해 세로축에 l' 만 그렸지만,
l'말고 다른 레이블도 있을 수 있음.

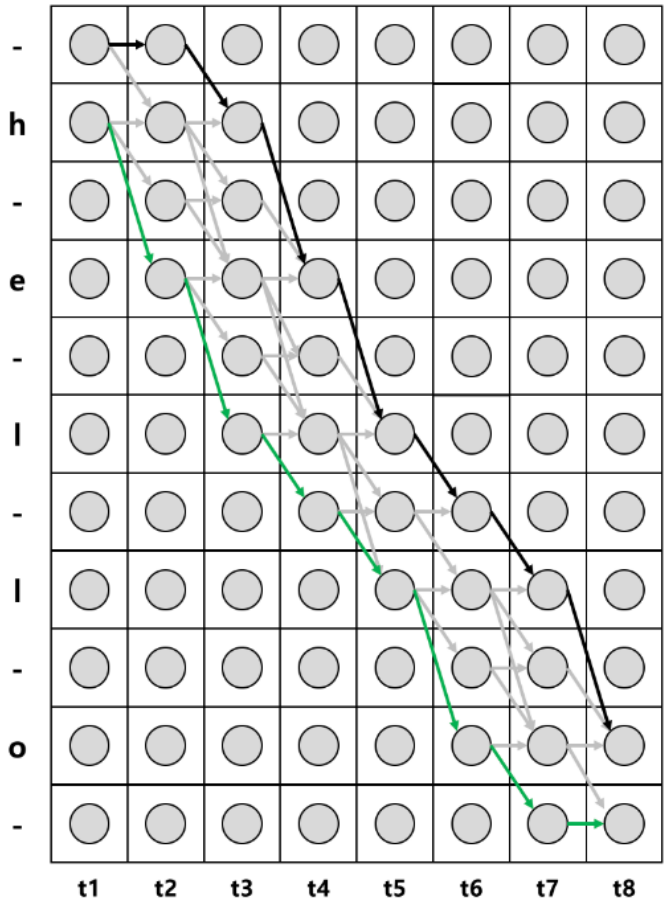
각 동그라미는 개별 확률값을 의미.
예를 들어 t = 2 시점의 상태가 h 일 확률은 y_h^2
같은 시점에 중복된 레이블에 해당하는 y 확률값은 동일.

3 CTC: Connectionist Temporal classification

CTC overview

CTC 기법은 시퀀스 분류를 위한 딥러닝 모델 맨 마지막에 loss 및 gradient 계산 레이어로 구현됨.

그림4 ALL POSSIBLE PATHS



π: paths 중 하나
검정색 실선의 경로를 π 라고 하자.
π = --hel-lo
π_t = h (t=3)

CTC 기법에서는 각 state(동그라미)가 조건부 독립이라고 가정함.
=> x(input) 가 주어졌을 때 π(특정 sequence) 가 나타날 확률

$$p(\pi|x) = \prod_{t=1}^{T} y_{\pi_t}^t$$

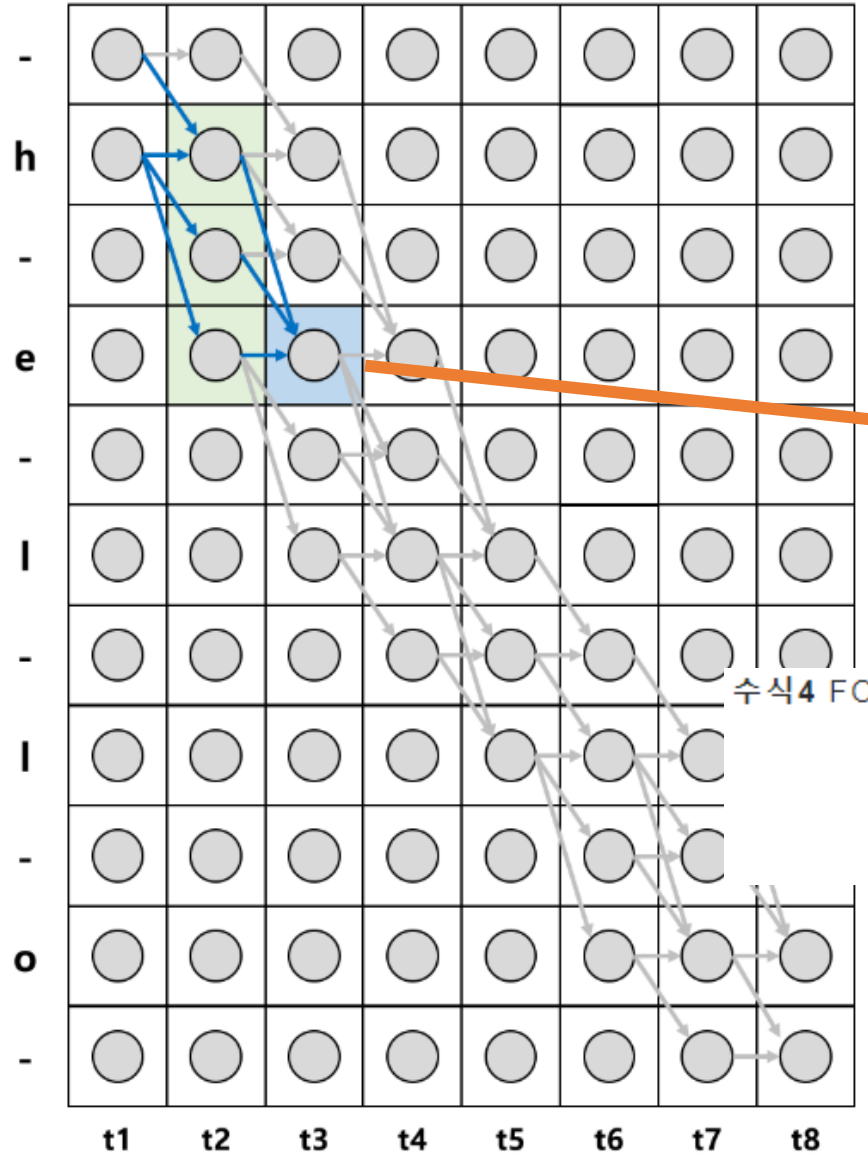
3 CTC: Connectionist Temporal classification

Forward Computation

x(input) 가 주어졌을 때 π(특정 sequence) 가 나타날 확률 $p(\pi|x)$
=> 그렇다면 정답을 구하려면? 정답 레이블 l paths 를 가질 확률은?
$p(l|x)$ 를 계산해야 함.

2 CTC: Connectionist Temporal classification

그림5 FORWARD COMPUTATION EXAMPLE



quence) 가 나타날 확률 $p(\pi|x)$

갈 레이블 l paths 를 가질 확률은?

$p(l|x)$ 를 계산해야 함.

가로 3, 세로 4

$\alpha_3(4)$ 을 구하려면?

t = 3인 시점에 e 가 나타날 확률

t = 3 시점에 e 가 나타나는 경로

-he, hhe, h-e, hee (파란색 화살표)

수식4 FORWARD COMPUTATION EXAMPLE

$$\alpha_3(4) = p(\text{"-he"}|\mathbf{x}) + p(\text{"hhe"}|\mathbf{x}) + p(\text{"h-e"}|\mathbf{x}) + p(\text{"hee"}|\mathbf{x})$$
$$= y_-^1 \cdot y_h^2 \cdot y_e^3 + y_h^1 \cdot y_h^2 \cdot y_e^3 + y_h^1 \cdot y_-^2 \cdot y_e^3 + y_h^1 \cdot y_e^2 \cdot y_e^3$$

t1에 - 일 확률 + t2에 h 일 확률 + t3에 e 일 확률 + ...

일반화

$$\alpha_t(s) = \sum_{\pi \in N^T : \mathcal{B}(\pi_{1:t}) = \mathbf{l}_{1:s}} \prod_{t'=1}^{t} y_{\pi_{t'}}^{t'}$$

3 CTC: Connectionist Temporal classification

Dynamic Programming

$$\alpha_t(s) = \sum_{\pi \in N^T : \mathcal{B}(\pi_{1:t}) = \mathbf{l}_{1:s}} \prod_{t'=1}^{t} y_{\pi_{t'}}^{t'}$$

계산량이 너무 많음!
=> 겹치는 계산은 저장했다가 다시 써먹어보자 => **dynamic programming**

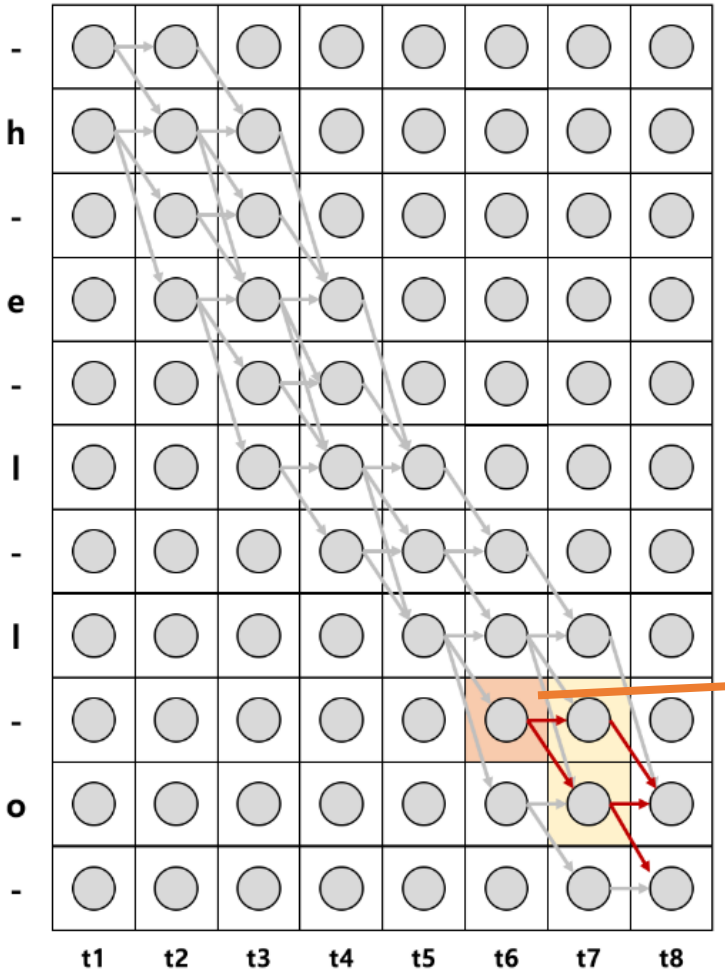
수식6 DYNAMIC PROGRAMMING EXAMPLE

$$\begin{aligned}
\alpha_3(4) &= p(\text{"-he"}|\mathbf{x}) + p(\text{"hhe"}|\mathbf{x}) + p(\text{"h-e"}|\mathbf{x}) + p(\text{"hee"}|\mathbf{x}) \\
&= y_-^1 \cdot y_h^2 \cdot y_e^3 + y_h^1 \cdot y_h^2 \cdot y_e^3 + y_h^1 \cdot y_-^2 \cdot y_e^3 + y_h^1 \cdot y_e^2 \cdot y_e^3 \\
&= \left\{ \left(y_-^1 \cdot y_h^2 + y_h^1 \cdot y_h^2 \right) + \left(y_h^1 \cdot y_-^2 \right) + \left(y_h^1 \cdot y_e^2 \right) \right\} y_e^3 \\
&= \left\{ \alpha_2(2) + \alpha_2(3) + \alpha_2(4) \right\} y_e^3
\end{aligned}$$

3 CTC: Connectionist Temporal classification

Backward Computation

그림9 BACKWARD COMPUTATION EXAMPLE



Backward Computation은
입력 시퀀스의 길이가 T, 타겟 레이블 시퀀스가 l 이라고 할 때, t 시점의 backward probability 는 t:T 시점에 걸쳐 레이블 시퀀스가 s: $|l|$ 일 확률

s는 state. 동그라미

수식8 BACKWARD COMPUTATION EXAMPLE

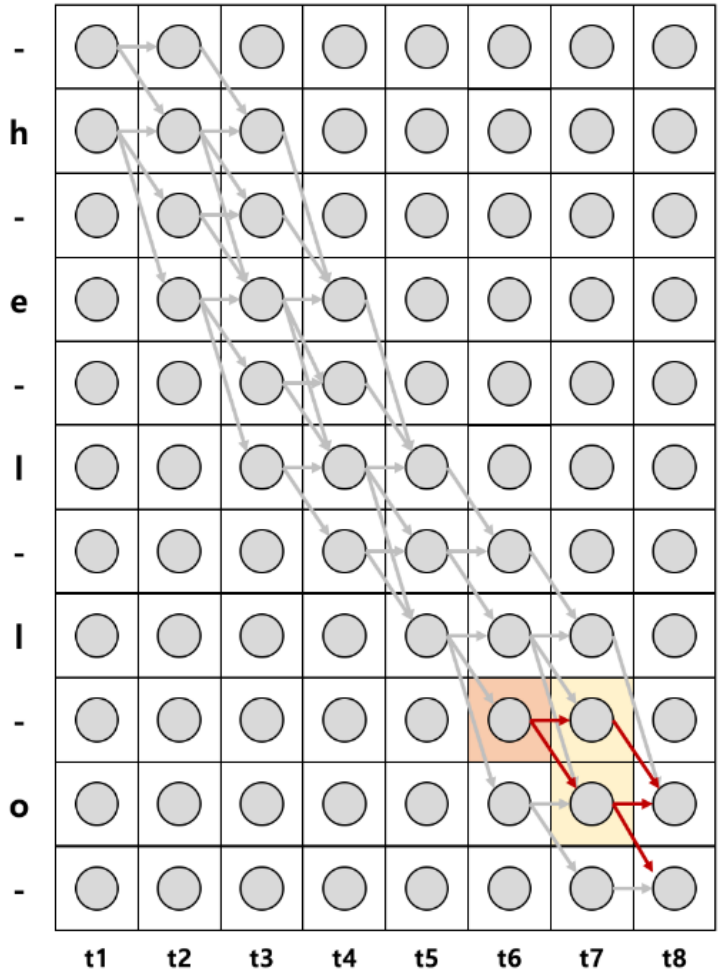
$$\begin{aligned}\beta_6(9) &= p(\text{"--o"}|\mathbf{x}) + p(\text{"-oo"}|\mathbf{x}) + p(\text{"-o-"}|\mathbf{x}) \\ &= y_-^6 \cdot y_-^7 \cdot y_o^8 + y_-^6 \cdot y_o^7 \cdot y_o^8 + y_-^6 \cdot y_o^7 \cdot y_-^8\end{aligned}$$

일반화 $$\beta_t(s) = \sum_{\pi \in N^T : \mathcal{B}(\pi_{t:T}) = \mathbf{l}_{s:|\mathbf{l}|}} \prod_{t'=t}^{T} y_{\pi_{t'}}^{t'}$$

3 CTC: Connectionist Temporal classification

Backward Computation

그림9 BACKWARD COMPUTATION EXAMPLE



$$\alpha_t(s) = \sum_{\pi \in N^T : \mathcal{B}(\pi_{1:t}) = \mathbf{l}_{1:s}} \prod_{t'=1}^{t} y_{\pi_{t'}}^{t'} \qquad \beta_t(s) = \sum_{\pi \in N^T : \mathcal{B}(\pi_{t:T}) = \mathbf{l}_{s:|\mathbf{l}|}} \prod_{t'=t}^{T} y_{\pi_{t'}}^{t'}$$

forward: 1~임의의 state s까지의 경로의 확률
backward: 임의의 state s~끝까지의 경로의 확률

수식10 DYNAMIC PROGRAMMING EXAMPLE

$$\begin{aligned}
\beta_6(9) &= p(\text{"--o"}|\mathbf{x}) + p(\text{"-oo"}|\mathbf{x}) + p(\text{"-o-"}|\mathbf{x}) \\
&= y_-^6 \cdot y_-^7 \cdot y_o^8 + y_-^6 \cdot y_o^7 \cdot y_o^8 + y_-^6 \cdot y_o^7 \cdot y_-^8 \\
&= \left\{ \left(y_-^7 \cdot y_o^8 \right) + \left(y_o^7 \cdot y_o^8 + y_o^7 \cdot y_-^8 \right) \right\} y_-^6 \\
&= \left\{ \beta_7(9) + \beta_7(10) \right\} y_-^6
\end{aligned}$$

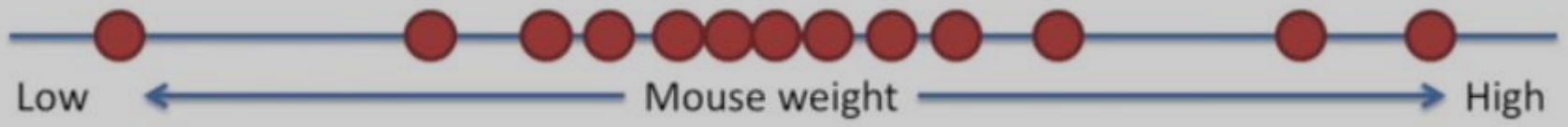
3 CTC: Connectionist Temporal classification

우리의 목표
: 음성 시퀀스 x 가 주어졌을 때 label sequence l맞추기
즉, likelihood(우도) 를 최대화하는 모델의 parameter 를 찾는 것.

- 입력 시퀀스의 각 프레임에 대해 모두 레이블이 있다면 그냥 Cross entropy 를 최소화하면 됨.
- But! 우리는 입력 시퀀스에 대해 프레임별 레이블이 없음(시간, 인력 등 비용소모, 레이블링의 어려움 등)

- 그래서 likelihood 를 최대화하려고 함.
- likelihood 최대화를 위해 likelihood 값을 구해야 함.
- Forward/Backward Computation 을 구해서 likelihood 값을 구할 수 있다.

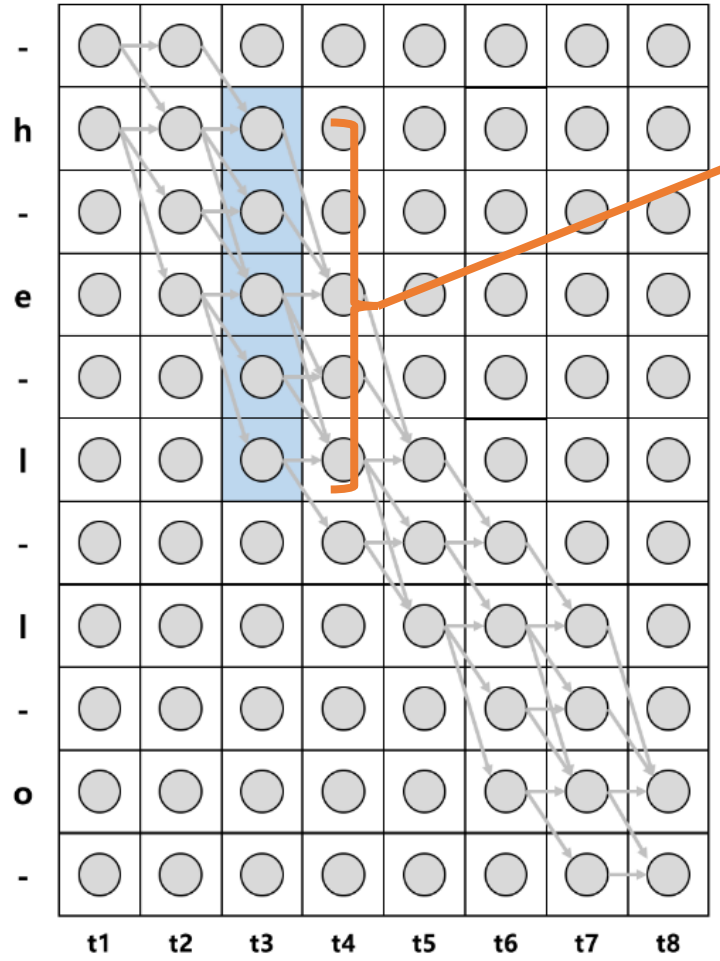
https://jjangjjong.tistory.com/41
likelihood: 확률의 확률.(주어진 관측값이 해당 확률분포에서 나왔을 확률)
$likelihood = L$(확률분포D | 관측값 X) (우도, 가능도)
maximum likelihood; **관측값이 나올 확률이 가장 큰 확률분포찾기!**



3 CTC: Connectionist Temporal classification

likelihood 구하기

그림11 LIKELIHOOD COMPUTATION EXAMPLE

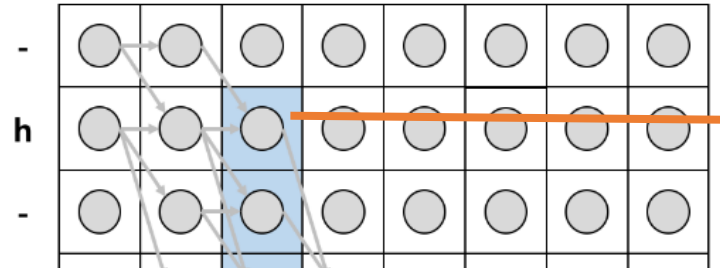


t=3 인 시점에 h를 지나는 경로들의 확률

3 CTC: Connectionist Temporal classification

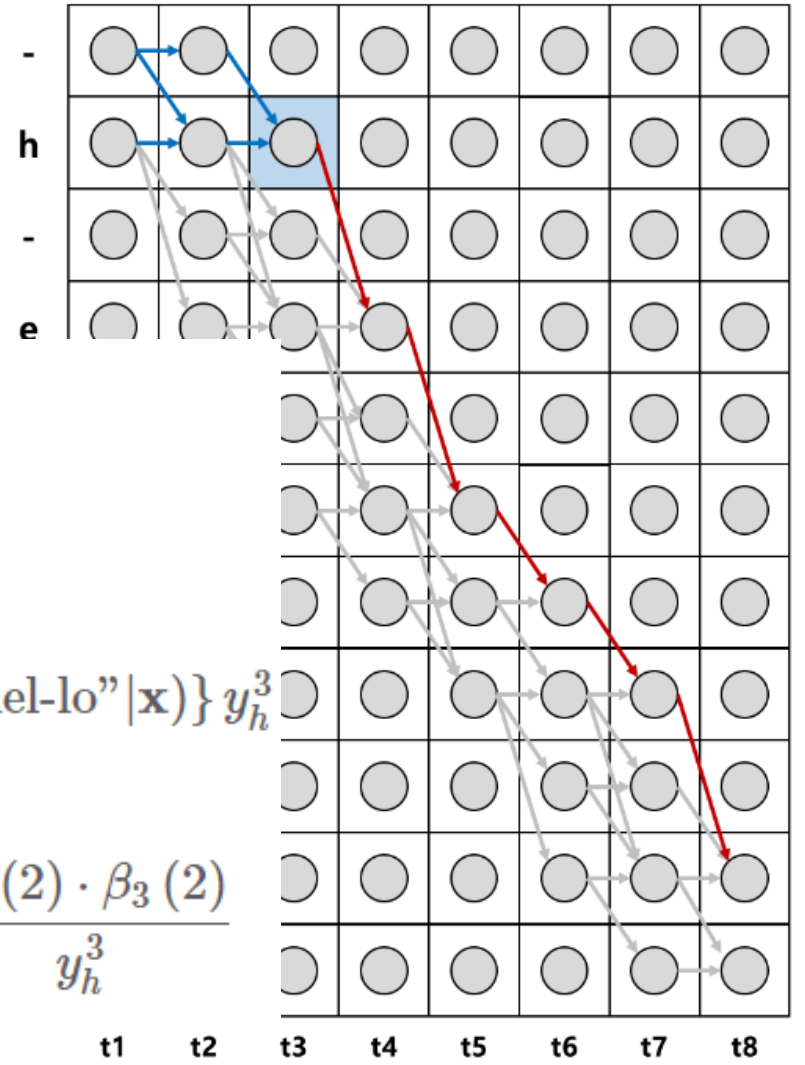
likelihood 구하기

그림11 LIKELIHOOD COMPUTATION EXAMPLE



특정 시점의 확률 구하기

그림10 COMPLETE PATH CALCULATION EXAMPLE



수식14 FORWARD/BACKWARD PROBABILITY EXAMPLE

$$\begin{aligned}\alpha_3(2)\cdot\beta_3(2) =& y_{-}^{1}\cdot y_{-}^{2}\cdot y_{h}^{3}\cdot y_{h}^{3}\cdot y_{e}^{4}\cdot y_{l}^{5}\cdot y_{-}^{6}\cdot y_{l}^{7}\cdot y_{o}^{8} \\ &+ y_{-}^{1}\cdot y_{h}^{2}\cdot y_{h}^{3}\cdot y_{h}^{3}\cdot y_{e}^{4}\cdot y_{l}^{5}\cdot y_{-}^{6}\cdot y_{l}^{7}\cdot y_{o}^{8} \\ &+ y_{h}^{1}\cdot y_{h}^{2}\cdot y_{h}^{3}\cdot y_{h}^{3}\cdot y_{e}^{4}\cdot y_{l}^{5}\cdot y_{-}^{6}\cdot y_{l}^{7}\cdot y_{o}^{8} \\ =& \{p(\text{"--hel-lo"}|\mathbf{x}) + p(\text{"-hhel-lo"}|\mathbf{x}) + p(\text{"hhhel-lo"}|\mathbf{x})\}\, y_{h}^{3}\end{aligned}$$

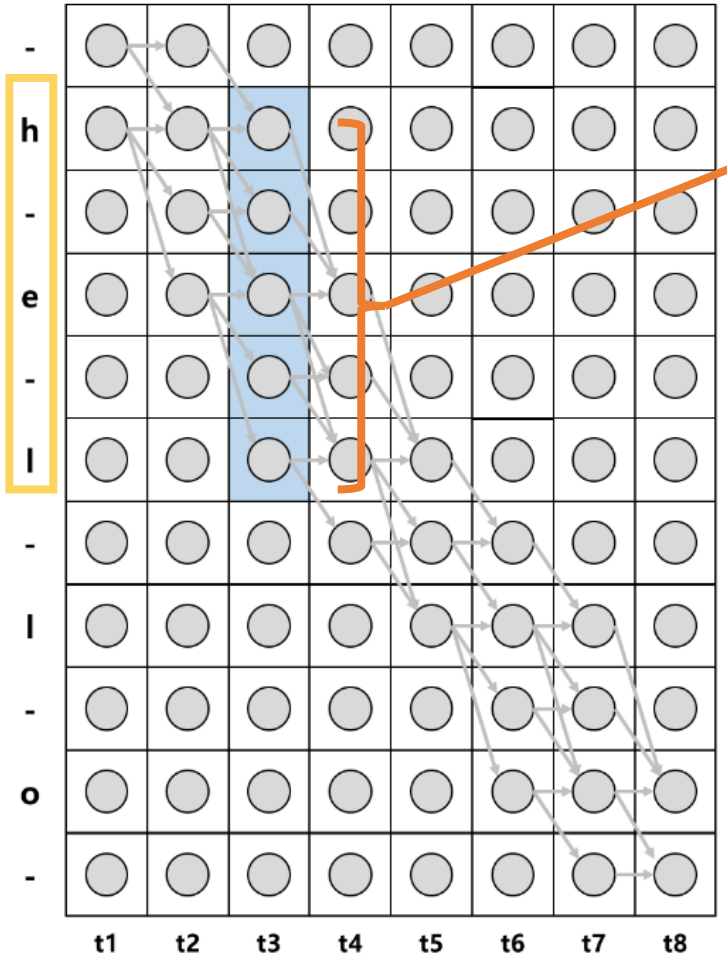
수식15 COMPLETE PATH PROBABILITY EXAMPLE

$$p(\text{"--hel-lo"}|\mathbf{x}) + p(\text{"-hhel-lo"}|\mathbf{x}) + p(\text{"hhhel-lo"}|\mathbf{x}) = \frac{\alpha_3(2)\cdot\beta_3(2)}{y_{h}^{3}}$$

3 CTC: Connectionist Temporal classification

likelihood 구하기

그림11 LIKELIHOOD COMPUTATION EXAMPLE



수식15 COMPLETE PATH PROBABILITY EXAMPLE

$$p(\text{"--hel-lo"}|\mathbf{x}) + p(\text{"-hhel-lo"}|\mathbf{x}) + p(\text{"hhhel-lo"}|\mathbf{x}) = \frac{\alpha_3(2) \cdot \beta_3(2)}{y_h^3}$$

t=3 인 시점에 h를 지나는 경로들의 확률

수식16 LIKELIHOOD COMPUTATION EXAMPLE

$$p(\text{"hello"}|\mathbf{x}) = \frac{\alpha_3(h) \cdot \beta_3(h)}{y_h^3} + \frac{\alpha_3(-) \cdot \beta_3(-)}{y_-^3} + \frac{\alpha_3(e) \cdot \beta_3(e)}{y_e^3} + \frac{\alpha_3(-) \cdot \beta_3(-)}{y_-^3} + \frac{\alpha_3(l) \cdot \beta_3(l)}{y_l^3}$$

편의를 위해 s의 인덱스 대신 h, e 등 문자로 표시

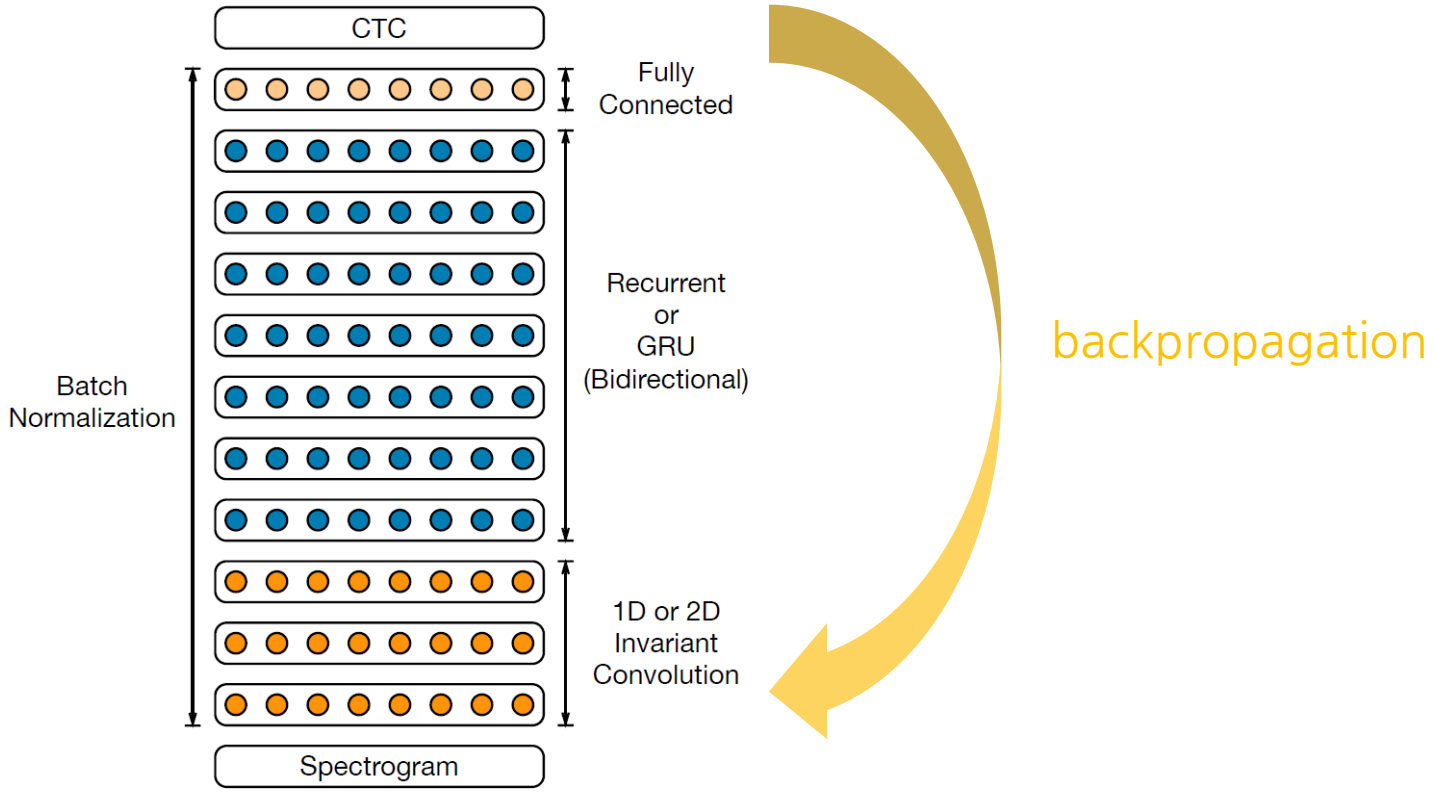
수식17 LIKELIHOOD COMPUTATION 일반화

$$p(\mathbf{l}|\mathbf{x}) = \sum_{s=1}^{|\mathbf{l}'|} \frac{\alpha_t(s) \cdot \beta_t(s)}{y_{\mathbf{l}'_s}^t}$$

3 CTC: Connectionist Temporal classification

Gradient Computation

우리는 likelihood $p(l|x)$ 를 최대화하는 모델 parameter 를 찾고자 함.
=> 이를 위해 likelihood 에 대한 gradient 를 구해야 한다. likelihood가 최대화되는 지점 찾기
이 gradient 를 **모델 전체 학습 파라미터**에 backpropagation 하는 것이 CTC 적용 모델의 train.



4 Experiments

Temporal classification Task

- TIMIT 음성 말뭉치의 음성 레이블링에 대해 CTC의 성능을 HMM 및 HMM-RNN 하이브리드 모두의 성능과 비교.
- => i.e TIMIT 테스트 세트의 발화에 가능한 최저 라벨 오류율을 제공하는 음소 시퀀스로 주석을 다는 작업

System	LER
Context-independent HMM	38.85 %
Context-dependent HMM	35.21 %
BLSTM/HMM	33.84 ± 0.06 %
Weighted error BLSTM/HMM	31.57 ± 0.06 %
CTC (best path)	31.47 ± 0.21 %
CTC (prefix search)	30.51 ± 0.19 %

The CTC network used an extended BLSTM architecture with peepholes and forget gates (Gers et al., 2002), 100 blocks in each of the forward and backward hidden layers, hyperbolic tangent for the input and output cell activation functions and a logistic sigmoid in the range $[0, 1]$ for the gates.

Reference

- speech recognition 관련 자료 정리 블로그:
  https://ratsgo.github.io/speechbook/docs/neuralam/ctc
- likelihood 참고 자료 : https://jjangjjong.tistory.com/41