

○ Softmax 사용하는 이유 21.F.7

(1) 작은 값은 더 작게 큰 값은 더 크게 만들기 위해

(2) likelihood는 음수값이 나올 수 있음
→ 총합을 1로 만들기가 어려움

(3) softmax는 shift에 invariant

(4) softmax를 이용하여 back propagation 시
수식이 간단해짐

Sigmoid는 데이터 x 의 절대값이 커질수록 derivative가 0으로 수렴하는 문제가 존재함

→ vanishing gradient 문제 발생 가능

tanh는 sigmoid 보다 derivative의 범위가 넓음
but derivative가 거의 항상 1보다 작음

→ 마찬가지로 vanishing gradient 문제 발생 가능

ReLU는 $x < 0$ 일 때 derivative가 0인 문제 존재

$$\frac{\partial L}{\partial w_{ij}} = \sum_{i=1}^I \frac{\partial L_t}{\partial w_{ij}}$$

$$\hat{y}_t^{(i)} = \sigma\left(\sum_{j=1}^{d_h} w_{ij} h_t^{(j)}\right)$$

$$L_t = \frac{1}{2} (y_t^{(i)} - \hat{y}_t^{(i)})^2$$

$$\begin{aligned} \frac{\partial L_t}{\partial w_{ij}} &= \frac{\partial L_t}{\partial \hat{y}_t^{(i)}} \frac{\partial \hat{y}_t^{(i)}}{\partial w_{ij}} \\ &= -(y_t^{(i)} - \hat{y}_t^{(i)}) \cdot \hat{y}_t^{(i)} (1 - \hat{y}_t^{(i)}) h_t^{(i)} \end{aligned}$$

왜 해설과 다르지? ...

$$\frac{\partial L_t}{\partial \hat{y}_t^{(i)}} = -(y_t^{(i)} - \hat{y}_t^{(i)})$$

$$\frac{\partial \hat{y}_t^{(i)}}{\partial w_{ij}} = \hat{y}_t^{(i)} (1 - \hat{y}_t^{(i)}) h_t^{(i)}$$

○ Activation function으로 sigmoid 또는 ReLU 대신 tanh 사용하는 이유 21.F.8

$$\text{Sigmoid} = \frac{1}{1 + e^{-x}}$$

[0, 1]로 대응됨

$$\tanh = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

[-1, 1]로 대응됨

$$\text{ReLU} = \begin{cases} ax & (x \geq 0, a \in \mathbb{R}) \\ 0 & (x < 0) \end{cases}$$

○ Viterbi training 21.F.9

현재의 모델로 Viterbi algorithm에 따라
전체 학습 자료에 대해 segmentation 수행

모델 파라미터 업데이트

$$\prod_{r=1}^R \Pr(O^r | M)$$

가 더 이상 증가하지 않을 때까지 과정 반복

c.f) Viterbi algorithm for segmentation

$$\Phi_j(t) = \max_{1 \leq i \leq N} [\Phi_i(t-1) a_{ij}] b_j(O_t)$$

$$\Pr_{\max}(O | M) = \max_{1 \leq i \leq N} [\Phi_i(T) a_{iN}]$$

예) 한글에 음소 40개의 모델 'ㅏ' 모델 M 평균: μ_0 분산: σ_0^2
정규분포 가정

학습 데이터 corpus

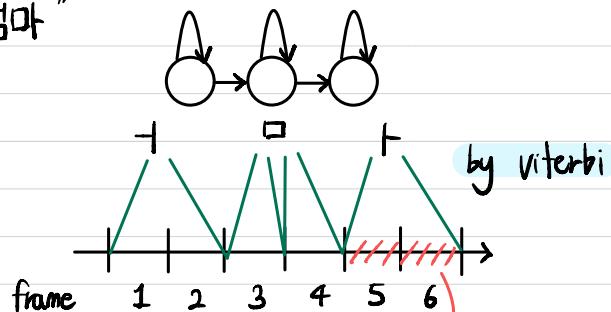
"엄마" "사랑" ...

(*)

각 단어 데이터에 관해
Viterbi로 각 음소에 대한 segmentation 수행

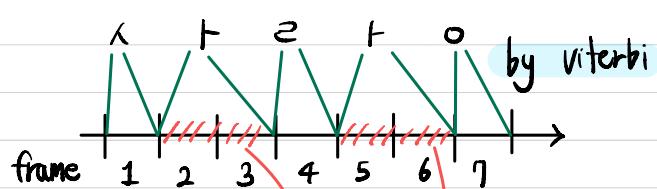
40개의 모든 음소 state를 갖는 HMM

"엄마"



O^1 첫 번째 'ㅏ'로
segment된 frame의 vector
(1차원 vector 가정) $v_s^{(1)}, v_b^{(1)}$

"사랑"



O^2 두 번째 'ㅏ'로
segment된 frame의 vector $v_s^{(2)}, v_b^{(2)}$

O^3 세 번째 'ㅏ'로
segment된 frame의 vector $v_s^{(3)}, v_b^{(3)}$

새로운 'ㅏ' 모델 M' 구하기 17-2 M.8

$$\mu_0' = \frac{v_s^{(1)} + v_b^{(1)} + v_s^{(2)} + v_b^{(2)} + \dots}{(\text{모든 학습 데이터에 관해 'ㅏ'가 발화된 횟수})}$$

$$\sigma_0'^2 = \frac{(v_s^{(1)} - \mu_0')^2 + (v_b^{(1)} - \mu_0')^2 + (v_s^{(2)} - \mu_0')^2 + \dots}{(\text{모든 학습 데이터에 관해 'ㅏ'가 발화된 횟수})}$$

M'
평균: μ_0'
분산: $\sigma_0'^2$

$$\hat{a}_{ij} = \frac{i \rightarrow j \text{로 이동한 추정 횟수}}{\text{[에서부터 이동한 전체 추정 횟수]}}$$

Expectation maximization algorithm

$$\text{if } \prod_{t=1}^R \Pr(O^t | M') \uparrow \\ = \Pr(O^1 | M') \Pr(O^2 | M') \dots$$

이전 대비 증가하면 M 대신 M'으로 교체 후
(*) 부터 반복

더 이상 $\prod_{t=1}^R \Pr(O^t | M)$ 증가 안할 때까지

다른 음소 모델도 이와 같이 수행

o bottleneck problem 21.F.12

고정된 크기의 hidden state (context vector)에
이전의 모든 time에서의 입력 sequence 정보가
encoding 되므로 앞선 time에서의 정보가 소실되는
short-term memory 문제 발생

o seq2seq with attention 21.F.12 21.M.6

목적 decoder의 각 시점에서 encoder의
입력 sequence의 특정 부분에 주목하도록
encoder로 직접 연결

decoder hidden state
encoder의 모든 hidden state) dot product

각 time별로의 가중치 구하고 → softmax
이를 가지고 encoder의 모든 hidden state에 관한
가중합 → attention output

decoder의 hidden state와 concatenate

해결하고자 한 문제

(1) seq2seq에서 하나의 고정된 크기로
입력 sequence의 정보 압축
→ context vector에서 앞 시간의 정보를

제대로 반영하지 못하는 정보 손실

(2) RNN에서의 vanishing gradient 문제

(3) decoding 수행시 입력 sequence에서
중요한 부분에 주목

21.F.14

o Transformer에서 해결하고자 한 RNN의 문제점

(1) short term memory 정보 손실
(2) 병렬 처리 불가 이전 time till dependent
sequential 처리

(3) 학습 시 BPTT로 인한 Vanishing gradient 문제 발생

21.F.15

o 일반적인 attention과 self-attention의 차이

attention-based model

입력 sequence와 target sequence의 관계

self-attention

input 간의 관계

o Positional encoding 사용 이유 21.F.16

RNN에서는 recurrent unit으로 자연스럽게 순서 고려

self-attention에서는 위치 정보 이용 X

입력 순서 바꿔도 attention 결과 바뀌지 X

input sequence의 각 token의 position 정보 부여

○ Multi-head attention의 차원 21.F.20

40 dim MFCC 8개의 head
head의 dim : 5

$$W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{d \times \frac{d}{h}} \quad W^O \in \mathbb{R}^{h \times d}$$

Multi Head Attention(X)

$$= (\text{head}_1 \oplus \text{head}_2 \oplus \dots \oplus \text{head}_h) W^O$$

$$Q = X W_i^Q \quad K = X W_i^K \quad V = X W_i^V$$

$\text{head}_i = \text{Self Attention}(Q, K, V)$

21.F.22

○ Positional encoding vector들의 dot similarity의 최대값

$$d_{\text{model}} = 256$$

$$\text{PE}(\text{pos}, 2i) = \sin\left(\frac{\text{pos}}{10000^{2i/d_{\text{model}}}}\right)$$

$$\text{PE}(\text{pos}, 2i+1) = \cos\left(\frac{\text{pos}}{10000^{2i/d_{\text{model}}}}\right)$$

$$\begin{aligned} & \text{PE pos}^T \text{PE}(\text{pos} + \Delta p) \\ &= \left(\sin\left(\frac{\text{pos}}{10000^{2i/d}}\right), \cos\left(\frac{\text{pos}}{10000^{2i/d}}\right), \dots \right) \begin{pmatrix} \sin\left(\frac{\text{pos} + \Delta p}{10000^{2i/d}}\right) \\ \cos\left(\frac{\text{pos} + \Delta p}{10000^{2i/d}}\right) \\ \vdots \end{pmatrix} \end{aligned}$$

$$= \sum_{i=0}^{127} \sin\frac{\text{pos}}{10000^{2i/d}} \sin\frac{\text{pos} + \Delta p}{10000^{2i/d}} + \cos\frac{\text{pos}}{10000^{2i/d}} \cos\frac{\text{pos} + \Delta p}{10000^{2i/d}}$$

$$W_i = \frac{1}{10000^{2i/d}}$$

$$= \sum_{i=0}^{127} \sin(W_i \cdot \text{pos}) \sin(W_i \cdot \text{pos} + W_i \cdot \Delta p) + \cos(W_i \cdot \text{pos}) \cos(W_i \cdot \text{pos} + W_i \cdot \Delta p)$$

$$= \sum_{i=0}^{127} \sin(W_i \cdot \text{pos}) \{ \sin(W_i \cdot \text{pos}) \cos(W_i \cdot \Delta p) + \cos(W_i \cdot \text{pos}) \sin(W_i \cdot \Delta p) \} + \cos(W_i \cdot \text{pos}) \{ \cos(W_i \cdot \text{pos}) \cos(W_i \cdot \Delta p) - \sin(W_i \cdot \text{pos}) \sin(W_i \cdot \Delta p) \}$$

$$= \sum_{i=0}^{127} (\sin^2(W_i \cdot \text{pos}) + \cos^2(W_i \cdot \text{pos})) \cos(W_i \cdot \Delta p) = \sum_{i=0}^{127} \cos(W_i \cdot \Delta p)$$

$$\text{if } \Delta p = 0, \text{ max: } 128$$

○ Masked self-attention의 예시와 이유 21.F.18

예시) Language model task

이유) inference time에서 decoder는 앞의 단어를 예측할 때 그 뒤의 단어 정보를 알 수 없으므로 뒤의 단어 정보 참고하는 것을 막기 위해

○ Viterbi training의 특징 21.F.10

매 iteration마다 alignment 재추정 해야 함

HMM과 결합되어 생성 모델

하나의 most probable path 만을 alignment로 사용
→ 학습 초기 alignment 영향에 큰 영향 받음

○ 자주 쓰이는 sampling rate과 그 이유 21. M. 20

음악 저장에 자주 사용되는 sampling rate 44.1kHz

Nyquist theorem에 의해 원래 sampling rate의 절반인

22,050Hz 까지 복원 가능

인간의 가청주파수 범위는 20Hz ~ 22,050Hz

음성 인식에 자주 사용되는 sampling rate 16kHz

Nyquist theorem에 의해 원래 sampling rate의 절반인

8,000Hz 까지 복원 가능

사람의 발성의 최대 주파수는 8,000Hz

○ 음성이 quasi-stationary 한 이유 21. M. 13

사람의 성대를 하나의 발성 기관으로 보면

발성 기관의 물리적인 한계로 인해

짧은 시간의 관점에서는 미래의 발화 예측 가능

보통 20ms에서 음성은 quasi-stationary 함

○ 음소의 정의 21. M. 12

다른 소리와 구별되어 언어 사용자가 인식할 수 있는

소리의 최소 단위 예) 영어의 발음 기호

c.f.) 형태소: 의미의 최소 단위

18-2. M. 6

○ 활성 합수 미분 21. M. 8 21. M. 9 21. M. 10

$$\text{Sigmoid } \sigma(x) = \frac{1}{1+e^{-x}} \in [0, 1]$$

$$\begin{aligned} \frac{\partial}{\partial x} \sigma(x) &= \frac{-(-e^x)}{(1+e^{-x})^2} \\ &= \frac{1+e^{-x}-1}{(1+e^{-x})(1+e^{-x})} \\ &= \frac{1}{1+e^{-x}} - \frac{1+e^{-x}-1}{1+e^{-x}} \\ &= \sigma(x)(1-\sigma(x)) \end{aligned}$$

$$x=0 \text{에서의 미분값 } \frac{1}{2}(1-\frac{1}{2}) = \frac{1}{4}$$

$$\sigma(0) = \frac{1}{2} \quad \lim_{x \rightarrow \infty} \sigma(x) = 1 \quad \lim_{x \rightarrow -\infty} \sigma(x) = 0$$

$$\tanh h \quad \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

$$\begin{aligned} \frac{\partial}{\partial x} \tanh(x) &= \frac{(e^x + e^{-x})(e^x + e^{-x}) - (e^x - e^{-x})(e^x - e^{-x})}{(e^x + e^{-x})^2} \\ &= 1 - \frac{e^x - e^{-x}}{e^x + e^{-x}} \cdot \frac{e^x - e^{-x}}{e^x + e^{-x}} \\ &= 1 - \{\tanh(x)\}^2 \end{aligned}$$

$$x=0 \text{에서의 미분값 } 1 - 0^2 = 1$$

○ HMM의 기본 문제 세 가지와 21.M.

각 문제를 해결하기 위한 알고리즘

state의 이동 경로가 hidden \rightarrow non-deterministic

(1) Evaluation problem (인식)

인식 관측열이 발생할 확률 계산

인식 결과를 어떻게 결정할 것인지

모델이 해당 관측열로 인식 결과를 생성할 확률

모델 파라미터는 주어진 상황에서 관측 열의 생성확률을

최대로 하는 모델 선정

알고리즘: Total likelihood method, forward algorithm

(2) Segmentation

관측열이 어떤 상태전이를 거쳐서 발생했는지를 추정

DH frame마다 어떠한 state에 대응되는지

전체 음성 신호 중 특정 단어를 찾아주는 segmentation 필요

전체 학습 자료에서 모델별 학습에 필요한 자료 자동 추출

알고리즘: Viterbi algorithm

(3) Learning Problem 학습

HMM의 파라미터 추정 모델 학습 시 필요

Expectation maximization algorithm에 기초

알고리즘: Viterbi training, Baum-Welch algorithm

○ Expectation maximization algorithm 21.M. 1

학습 데이터에 대한 현재 모델의 생성확률 최대화

$$\prod_{r=1}^R \Pr(O^r | M) \uparrow$$

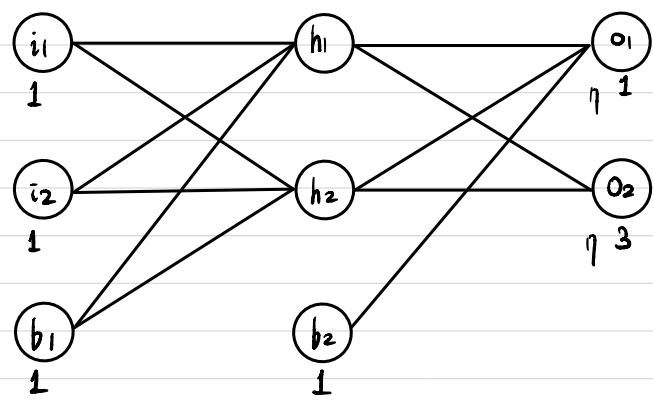
R: 전체 학습 데이터에서 모델 M에 대응되는 데이터의 수

$$\prod_{r=1}^R \Pr(O^r | M) \text{ 생성확률이 최대화 되도록}$$

모델 파라미터 수정

학습 자료로부터 모델의 파라미터 업데이트는 자동으로 진행

○ Backpropagation 18-2. M. 7



모든 weight 1로 초기화

activation ReLU $\sigma = \max(0, z)$ 사용

learning rate $\eta = 0.5$

MSE loss

한 번 update 된 $W(i_1, h_1)$ 구하기

$$\begin{aligned} \text{net}_{h_1} &= W_{(i_1, h_1)} \cdot i_1 + W_{(i_2, h_1)} \cdot i_2 \\ &\quad + b_1 \cdot b_{(b_1, h_1)} \\ &= 1 \cdot 1 + 1 \cdot 1 + 1 \cdot 1 = 3 \end{aligned}$$

$$\text{out}_{h_1} = \sigma(\text{net}_{h_1}) = 3$$

$$\begin{aligned} \text{net}_{h_2} &= W_{(i_1, h_2)} \cdot i_1 + W_{(i_2, h_2)} \cdot i_2 \\ &\quad + b_1 \cdot b_{(b_1, h_2)} \\ &= 1 \cdot 1 + 1 \cdot 1 + 1 \cdot 1 = 3 \end{aligned}$$

$$\text{out}_{h_2} = \sigma(\text{net}_{h_2}) = 3$$

$$\begin{aligned} \text{net}_{o_1} &= W_{(h_1, o_1)} \cdot \text{out}_{h_1} + W_{(h_2, o_1)} \cdot \text{out}_{h_2} \\ &\quad + b_2 \cdot b_{(b_2, o_1)} \\ &= 1 \cdot 3 + 1 \cdot 3 + 1 \cdot 1 = 7 \end{aligned}$$

$$\text{out}_{o_1} = \sigma(\text{net}_{o_1}) = 7$$

$$\begin{aligned} \text{net}_{o_2} &= W_{(h_1, o_2)} \cdot \text{out}_{h_1} + W_{(h_2, o_2)} \cdot \text{out}_{h_2} \\ &\quad + b_2 \cdot b_{(b_2, o_2)} \\ &= 1 \cdot 3 + 1 \cdot 3 + 1 \cdot 1 = 7 \end{aligned}$$

$$\text{out}_{o_2} = \sigma(\text{net}_{o_2}) = 7$$

$$\begin{aligned} E &= \frac{1}{2} (\text{target}_{o_1} - \text{auto}_{o_1})^2 + \frac{1}{2} (\text{target}_{o_2} - \text{auto}_{o_2})^2 \\ &= E_1 + E_2 \end{aligned}$$

$$\frac{\partial E}{\partial w_{(i_1, h_1)}} = \frac{\partial E_1}{\partial \text{net}_{h_1}} \frac{\partial \text{net}_{h_1}}{\partial w_{(i_1, h_1)}}$$

$$= \frac{\partial E_1}{\partial \text{out}_{h_1}} \frac{\partial \text{out}_{h_1}}{\partial \text{net}_{h_1}} \frac{\partial \text{net}_{h_1}}{\partial w_{(i_1, h_1)}} = 10 \cdot 1 \cdot 1 = 10$$

$$\frac{\partial E}{\partial w_{(i_2, h_1)}} = \frac{\partial E_1}{\partial \text{out}_{h_1}} \frac{\partial \text{out}_{h_1}}{\partial \text{net}_{h_1}} \frac{\partial \text{net}_{h_1}}{\partial w_{(i_2, h_1)}}$$

$$+ \frac{\partial E_2}{\partial \text{out}_{h_2}} \frac{\partial \text{out}_{h_2}}{\partial \text{net}_{h_2}} \frac{\partial \text{net}_{h_2}}{\partial w_{(i_2, h_1)}} = 6 \cdot 1 \cdot 1 + 4 \cdot 1 \cdot 1 = 10$$

$$\frac{\partial E_1}{\partial \text{out}_{o_1}} = -(\text{target}_{o_1} - \text{auto}_{o_1}) = -(1 - 7) = 6$$

$$\frac{\partial \text{out}_{o_1}}{\partial \text{net}_{o_1}} = 1$$

$$\frac{\partial \text{net}_{o_1}}{\partial \text{out}_{h_1}} = W_{(h_1, o_1)} = 1$$

$$\frac{\partial E_2}{\partial \text{out}_{o_2}} = -(\text{target}_{o_2} - \text{auto}_{o_2}) = -(3 - 7) = 4$$

$$\frac{\partial \text{out}_{o_2}}{\partial \text{net}_{h_1}} = 1$$

$$\frac{\partial \text{net}_{o_2}}{\partial \text{out}_{h_1}} = W_{(h_1, o_2)} = 1$$

$$\frac{\partial \text{out}_{h_1}}{\partial \text{net}_{h_1}} = 1 \quad \frac{\partial \text{net}_{h_1}}{\partial w_{(i_1, h_1)}} = i_1 = 1$$

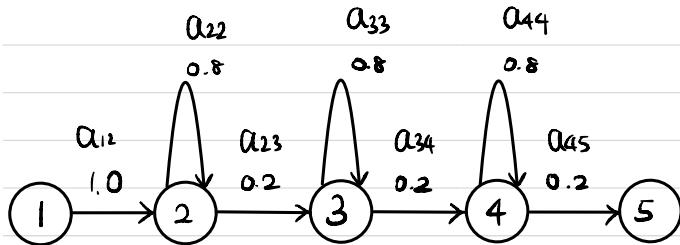
$$W_{(i_1, h_1)} = W_{(i_1, h_1)} - \eta \cdot \frac{\partial E}{\partial w_{(i_1, h_1)}}$$

$$= 1 - 0.5 \cdot 10 = 1 - 5 = -4$$

$\therefore -4$

18-2.M.9

Forward algorithm & Viterbi algorithm



O_t	$b_2(O_t)$	$b_3(O_t)$	$b_4(O_t)$
a	0.4	0.2	0.1
b	0.2	0.4	0.1
c	0.1	0.1	0.2
d	0.1	0.1	0.4
e	0.2	0.2	0.2

$$\begin{aligned}\alpha_4(O_3) &= (\alpha_3(O_2) \times a_{34}) \times b_4(O_3) \\ &= (0.4 \times 0.2 \times 0.4) \times 0.2 \times 0.2 \\ &= 0.032 \times 0.2 \times 0.2 = 0.00128\end{aligned}$$

$$\begin{aligned}\alpha_2(O_4) &= (\alpha_2(O_3) \times a_{22}) \times b_2(O_4) \\ &= (0.4 \times 0.8 \times 0.2 \times 0.8 \times 0.1) \times 0.8 \times 0.1 \\ &= 0.00512 \times 0.8 \times 0.1 = 0.0004096\end{aligned}$$

$$\begin{aligned}\alpha_3(O_4) &= (\alpha_2(O_3) \times a_{23} + \alpha_3(O_3) \times a_{33}) \times b_3(O_4) \\ &= [(0.4 \times 0.8 \times 0.2 \times 0.8 \times 0.1) \times 0.2 \\ &\quad + \{(0.4 \times 0.8 \times 0.2) \times 0.2 + (0.4 \times 0.2 \times 0.4) \times 0.8\} \times 0.1 \\ &\quad \times 0.8] \times 0.1 \\ &= (0.00512 \times 0.2 + 0.00284 \times 0.8) \times 0.1 \\ &= 0.0013312\end{aligned}$$

$$\begin{aligned}\alpha_4(O_4) &= (\alpha_3(O_4) \times a_{34} + \alpha_4(O_3) \times a_{44}) \times b_4(O_4) \\ &= [\{(0.4 \times 0.8 \times 0.2) \times 0.2 + (0.4 \times 0.2 \times 0.4) \times 0.8\} \times 0.1 \times 0.2 \\ &\quad + (0.4 \times 0.2 \times 0.4) \times 0.2 \times 0.2 \times 0.8] \times 0.4 \\ &= (0.00284 \times 0.2 + 0.00128 \times 0.8) \times 0.4 \\ &= 0.0007168\end{aligned}$$

4	0.0	0.0	0.0	0.00128	0.0007168
3	0.0	0.0	0.022	0.00384	0.0013312
2	0.0	0.4	0.064	0.00512	0.0004096
1	1.0	0.0	0.0	0.0	0.0

state time 0 1 2 3 4

$$\begin{aligned}\alpha_2(O_1) &= (\alpha_1(O_0) \times a_{12}) \times b_2(O_1) \\ &= (1 \times 1.0) \times 0.4 = 0.4\end{aligned}$$

$$\begin{aligned}\alpha_2(O_2) &= (\alpha_2(O_1) \times a_{22}) \times b_2(O_2) \\ &= (0.4 \times 0.8) \times 0.2 = 0.064\end{aligned}$$

$$\begin{aligned}\alpha_3(O_2) &= (\alpha_2(O_2) \times a_{23}) \times b_3(O_2) \\ &= (0.4 \times 0.2) \times 0.4 = 0.032\end{aligned}$$

$$\begin{aligned}\alpha_2(O_3) &= (\alpha_2(O_2) \times a_{22}) \times b_2(O_3) \\ &= (0.4 \times 0.8 \times 0.2) \times 0.8 \times 0.1 = 0.00512\end{aligned}$$

$$\begin{aligned}\alpha_3(O_3) &= (\alpha_2(O_2) \times a_{23} + \alpha_3(O_2) \times a_{33}) \times b_3(O_3) \\ &= \{(0.4 \times 0.8 \times 0.2) \times 0.2 + (0.4 \times 0.2 \times 0.4) \times 0.8\} \times 0.1 \\ &= (0.064 \times 0.2 + 0.022 \times 0.8) \times 0.1 \\ &= 0.00384\end{aligned}$$

$$\text{Total } Pr(O|M) = 0.0007168 \times 0.2 = 0.00014336$$

By Viterbi, $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 4 \rightarrow 5$

c.f) HMM의 4부

State

transition probability

output probability

initial state probability

○ 음성 인식의 문제 정의와 핵심

マイ크를 통해 입력받은 음성이 주어졌을 때

가장 확률이 높은 문자열(문장) 출력

$$\arg \max_w P(w|X)$$

$W = \{w_1, w_2, \dots, w_U\}$: U개의 단어 sequence

$X = \{x_1, x_2, \dots, x_T\}$: T개의 frame의 음성 데이터

주로 20ms의 크기의 frame(윈도우)

1초에 50개의 frame으로 나눠짐

MFCC feature에 의해 T개의 13차 벡터 sequence

End-to-End 관점

$x_1, \dots, x_T \rightarrow w_1, \dots, w_U$ 로의 번역 문제

가능한 X의 개수가 무한대이므로

$P(w|X)$ 를 직접 구할 수 X

DNN-WFST에서의 관점

By Bayes rule,

$$\arg \max_w P(w|X) = \arg \max_w \frac{P(w) P(X|w)}{P(X)}$$

$P(X)$: $13 \times T$ 벡터 공간에서 한 점이 나올 확률

$P(X)$ 는 모두 동일하다고 가정하면

$$\arg \max_w P(X|w) \propto P(w)$$

음성 인식의 핵심 component

$P(w)$: 언어 모델

$P(X|w)$: 음향 모델

$\arg \max_w$: 디코딩 네트워크

어휘 (인식 가능한 단어의 set)

c) End-to-End의 장단점

장점 SOTA 성능 (transformer)

음성 파일 - transcription 만으로 학습

전혀 모르는 언어도 음성 인식기 제작 가능

단점 외부 지식을 실시간으로 반영할 방법 X (학습 시간 소요)

대용량 텍스트 corpus를 음성 인식기에 직접 반영 어려움

복잡한 구조, 파라미터 computation power ↑

ML 기반 학습

재현 실험 안 될 수 있음 (모델 초기값에 의존)

c) MFCC feature 추출 과정

1 short [T] [320]

1

16kHz sampling rate 각 frame 당 320개의 샘플

음성 신호 저장 시 필요한 파라미터: Sampling rate

Sample & byte 4 (intensity)

20ms 단위의 window (frame)

2 FFT

(Fast Fourier Transformation)

&

Magnitude

Spectrum

2

모든 신호는 $\alpha \cdot \sin(\beta \cdot \pi)$ 의 합의 형태로 표현 가능

time domain data \rightarrow frequency domain data

amplitude ↑

spectrum

frequency

but 다시 복원할 수는 X

(1) 위치를 알 수 X

(2) fundamental frequency의 정수배가 되는 sine파

3 Mel-filterbank

&

Log

spectrogram: time과 frequency domain이

각 축을 이루고, amplitude를 색으로 표현한 것

4 Discrete Cosine Transformation

3

float [T][13]

음성의 정보 (spectral envelop) 화자의 정보 (spectral detail)

$$X(n) = V(n) * e(n)$$

spectrum

FFT

convolution이 핵심으로

log 쓰임으로써 e(n) 제거 가능

$$\log(X(n)) = \log(V(n)) + \log(e(n))$$

각 window 별로 13차 MFCC feature 추출

왜? 음성은 quasi-stationary

4

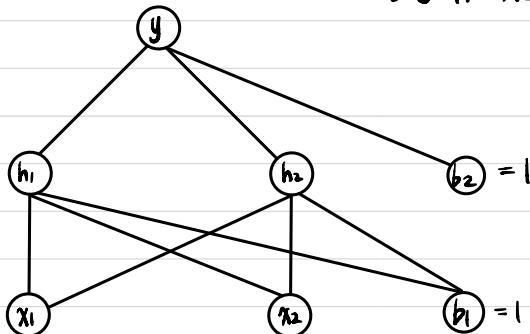
작은 차원의 데이터로 envelop을 구할 때 사용

음성 인식에 있어서 필요한 정보를 담은 13차 vector 생성
(spectrum의 전방적인 모양을 나타내는)

○ XNOR을 위한 neural net design

x_1	x_2	y
0	0	1
0	1	0
1	0	0
1	1	1

$$\sigma(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}$$



$$h_1 = \sigma(W(x_1, h_1) * x_1 + W(x_2, h_1) * x_2 + W(b_1, h_1) * b_1)$$

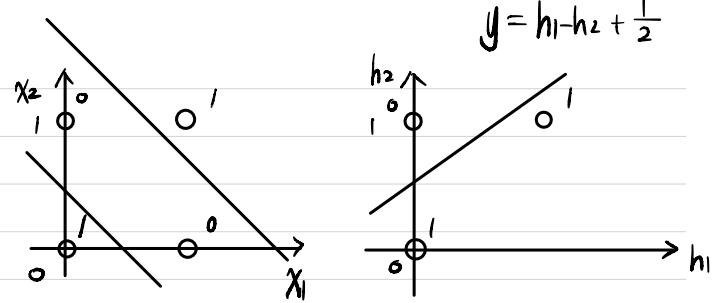
$$h_2 = \sigma(W(x_1, h_2) * x_1 + W(x_2, h_2) * x_2 + W(b_2, h_2) * b_2)$$

$$y = \sigma(W(h_1, y) * h_1 + W(h_2, y) * h_2 + W(b_2, y) * b_2)$$

$$W(x_1, h_1) = 1 \quad W(x_2, h_1) = 1 \quad W(b_1, h_1) = -\frac{3}{2}$$

$$W(x_1, h_2) = 1 \quad W(x_2, h_2) = 1 \quad W(b_2, h_2) = -\frac{1}{2}$$

$$W(h_1, y) = 1 \quad W(h_2, y) = -1 \quad W(b_2, y) = \frac{1}{2}$$



$$h_2 = -h_1 + \frac{3}{2} \Rightarrow h_1 + h_2 - \frac{3}{2} = 0$$

$$x_1=1, x_2=1 \quad 1+1-\frac{3}{2}>0$$

$$x_1=1, x_2=0 \quad 1+0-\frac{3}{2}<0$$

$$x_1=0, x_2=1 \quad 0+1-\frac{3}{2}<0$$

$$x_1=0, x_2=0 \quad 0+0-\frac{3}{2}<0$$

$$h_2 = -h_1 + \frac{1}{2} \Rightarrow h_1 + h_2 - \frac{1}{2} = 0$$

$$x_1=0, x_2=0 \quad 0+0-\frac{1}{2}<0$$

$$x_1=0, x_2=1 \quad 0+1-\frac{1}{2}>0$$

$$x_1=1, x_2=0 \quad 1+0-\frac{1}{2}>0$$

$$x_1=1, x_2=1 \quad 0+1-\frac{1}{2}>0$$

$$h_2 = h_1 + \frac{1}{2} \Rightarrow h_1 - h_2 + \frac{1}{2} = 0$$

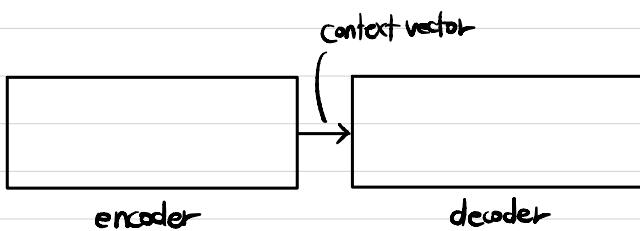
$$h_1=1, h_2=1 \Rightarrow 1-1+\frac{1}{2}>0$$

$$h_1=0, h_2=0 \Rightarrow 0-0+\frac{1}{2}>0$$

$$h_1=0, h_2=1 \Rightarrow 0-1+\frac{1}{2}<0$$

21.M.4

- RNN 기반 seq2seq에서 context vector 생성



encoder의 각 time t에서의 hidden state

$$h_t = \tanh(W_{xh} x_t + W_{hh} h_{t-1})$$

Context vector: encoder의 맨 마지막 time T에서

나오는 고정된 크기의 hidden state로 나오는 output

입력 sequence의 모든 time에 대한 정보 encoding

decoder의 맨 처음의 어떤 hidden state인

h_0 으로 들어감

21.M.5

- entropy 구하기 (entropy H(성적))

GPA	공부여부	성적
H	T	A
H	F	B
M	T	B
M	T	B
M	F	B
M	F	C
L	T	C
L	F	D

$$\text{entropy } H(\text{성적}) = \sum_{i \in \text{성적}} p_i \log\left(\frac{1}{p_i}\right)$$

$$= \left(\frac{1}{8} \log \frac{1}{8} + \frac{1}{2} \log \frac{1}{2} + \frac{1}{4} \log \frac{1}{4} + \frac{1}{8} \log \frac{1}{8} \right)$$

$$= \frac{3}{8} \log 2 + \frac{1}{2} \log 2 + \frac{2}{4} \log 2 + \frac{3}{8} \log 2$$

$$= \left(\frac{3}{8} + \frac{4}{8} + \frac{4}{8} + \frac{3}{8} \right) \log 2 = \frac{14}{8} \log 2 = \frac{7}{4}$$

17-2.M.6

- Forward algorithm의 α 와 시간복잡도

Dynamic Programming 이용

$$\alpha_j(t) = \left[\sum_{i=1}^N \alpha_i(t-1) a_{ij} \right] b_j(0t)$$

$\alpha_j(t)$: State j 에서 time t 에서의 output $0t$ 를 냈을 확률

a_{ij} : state i 에서 j 로 이동할 transition 확률

$b_j(0t)$: state j 에서 0

N : 가능한 state의 수

$$O(N \cdot T \cdot 2N) = O(N^2T)$$

↳ table 초기화 ↳ 각 entry마다 N 개의 이전 시간의 state 고려

- c.f) Total Likelihood Method

$$Pr(O|M) = \sum_S Pr(O, S|M)$$

S: state의 가능한 모든 이동 경로

$$O(N^T \cdot 2T) = O(N^T \cdot T)$$

↳ 각 이동 경로마다 확률 2T 개 곱하는 연산
T 시간 동안 가능한 모든 시간의 이동 경로

17-2.M.5

- 1초에 음성이 가질 수 있는 가능한 패턴의 수

44.1kHz sampling rate 가정

한 샘플 당 2byte의 intensity

$$1\text{초} \times 44,100\text{Hz} \times 2B = 88,200 \text{ B}$$

$$88,200 \text{ B} \times 8\text{bit}/1\text{B} = 88,200 \times 8 \text{ bit}$$

1초에 $2^{88,200 \times 8}$ 개의 패턴 존재