



6. Simple and LSTM Recurrent Neural Networks

0. Language Models

 [5. Language Models and Recurrent Neural Networks](#)

1. The Simple RNN Language Model

RNN(Recurrent Neural Network)?

- 시퀀스 데이터를 모델링하기 위해 등장한 모델
 - 기존 NN 과 다르게 hidden state)를 가지고 있음
- 은닉층의 노드에서 활성화 함수를 통해 나온 결과값을 출력층 방향으로도 보내면서 다시 은닉층 노드의 다음 계산의 입력으로 보내는 특징을 가짐
 - 반복적(recurrent)
 - 직전 시점의 은닉층에서 생성된 hidden states를 다음 시점의 input으로 전달
 - ⇒ “출력 결과는 이전의 계산 결과에 영향을 받는다.”

구조

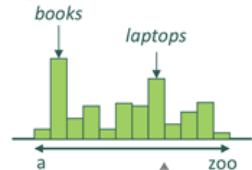
1. The Simple RNN Language Model

↳ recurrent(반복적)

④ output distribution

$$\hat{y}^{(t)} = \text{softmax}(\mathbf{U}h^{(t)} + b_2) \in \mathbb{R}^{|V|}$$

$$\hat{y}^{(4)} = P(x^{(5)} | \text{the students opened their})$$



③ hidden states

$$h^{(t)} = \sigma(\mathbf{W}_h h^{(t-1)} + \mathbf{W}_e e^{(t)} + b_1)$$

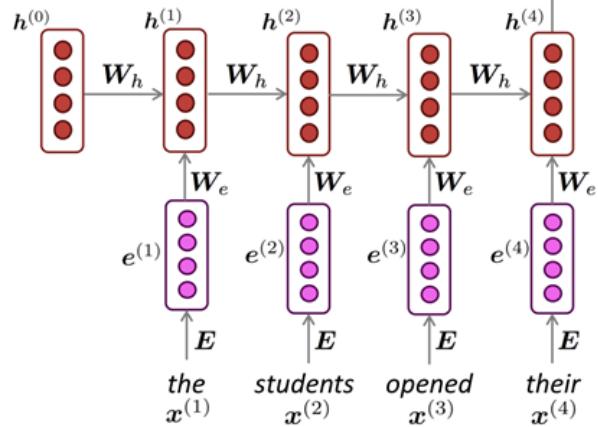
$h^{(0)}$ is the initial hidden state

② word embeddings

$$e^{(t)} = \mathbf{E}x^{(t)}$$

① words / one-hot vectors

$$x^{(t)} \in \mathbb{R}^{|V|}$$



4

- 표기

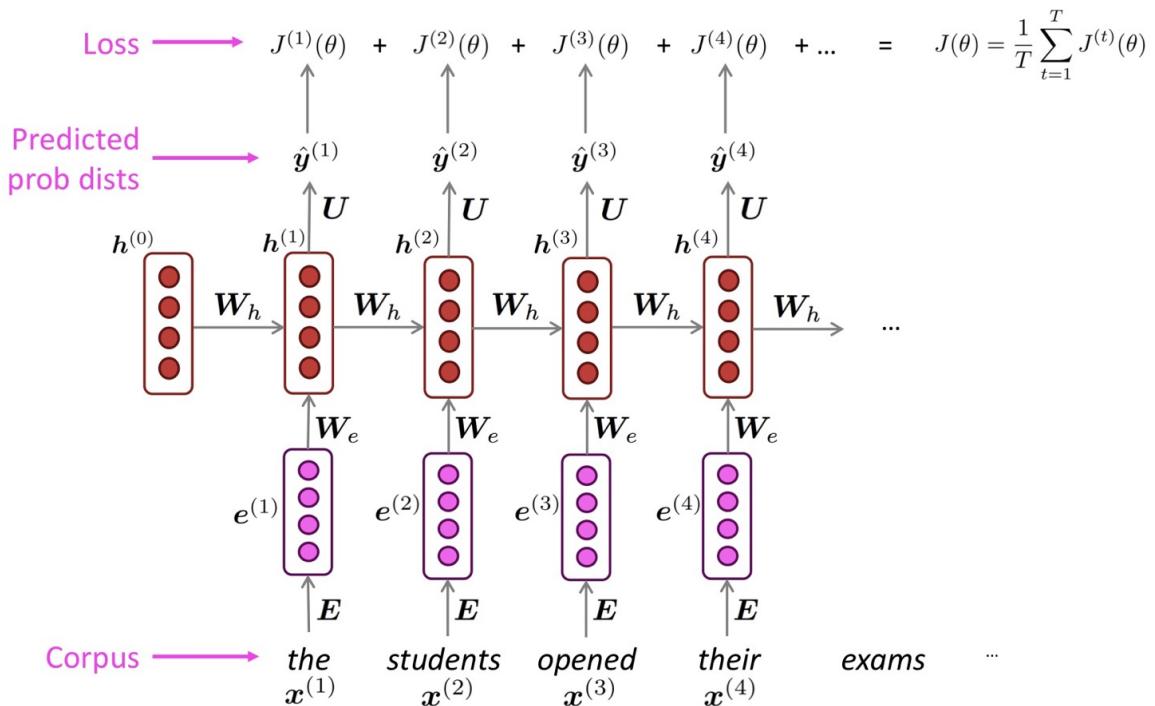
- $x^{(t)}$: t 시간 스텝에서의 입력 벡터, one-hot vector 형태
- $e^{(t)}$: 입력 벡터 $x^{(t)}$ 에 대한 word embedding
- $h^{(t)}$: t 시간 스텝에서 RNN의 기억을 담당하는 hidden state

$$h^{(t)} = \sigma(\mathbf{W}_h h^{(t-1)} + \mathbf{W}_e e^{(t)} + b_1)$$

- \mathbf{W}_e : 입력 x 의 임베딩 e 를 출력 h 로 변환하기 위한 가중치
- \mathbf{W}_h : RNN 출력을 다음 시각의 출력으로 변환하기 위한 가중치
- 편향 b

학습

- 과정



1. 단어들로 이루어진 시퀀스의 corpus를 준비
2. 단어들을 순서대로 RNN에 입력하고 매 단계(t)에 대한 출력분포를 계산
3. t 단계에 대한 손실함수(Cross-Entropy Loss)를 계산
4. 전체 training set에 대한 손실을 구하기 위해 평균값을 구함
 - 전체 corpus에 대한 loss와 기울기 계산은 시간이 많이 걸리므로 실제론 문장이나 문서 단위로 입력을 주기도 함
 - 혹은 SGD를 통해 최적화 하기도 함

- **손실 함수**

- cross-entropy loss
- 예측된 확률($\hat{y}^{(t)}$) vs 실제 확률($y^{(t)}$)

$$J^{(t)}(\theta) = CE(\mathbf{y}^{(t)}, \hat{\mathbf{y}}^{(t)}) = - \sum_{w \in V} \mathbf{y}_w^{(t)} \log \hat{\mathbf{y}}_w^{(t)} = - \log \hat{\mathbf{y}}_{\mathbf{x}_{t+1}}^{(t)}$$

▲ 1 step에서의 loss

$$J(\theta) = \frac{1}{T} \sum_{t=1}^T J^{(t)}(\theta) = \frac{1}{T} \sum_{t=1}^T -\log \hat{\mathbf{y}}_{x_{t+1}}^{(t)}$$

▲ 전체 데이터에 대한 overall loss

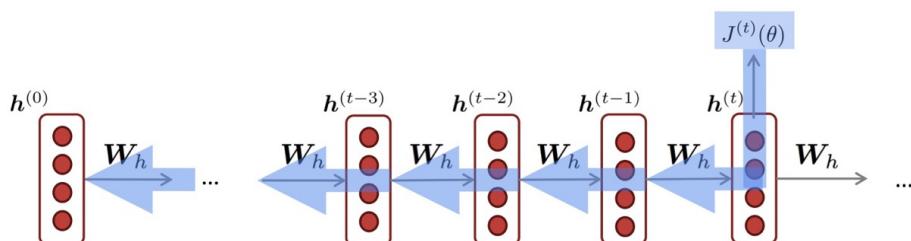
- 그러나 전체 데이터에 대해 loss를 계산하기에는 많은 비용이 소모됨
 - batch 방식을 활용

- **Teacher Forcing**

- RNN 학습 시 실제 정답을 입력으로 사용하여 모델을 훈련시키는 기술
 - 모델이 정확한 정보를 기반으로 학습하도록 함
 - 잘못된 예측으로 인한 오차 누적을 방지
- Teacher forcing은 학습을 안정적으로 하지만 실제 사용 시에는 이전 시간 단계에서의 예측값을 사용해야 함을 유의해야 함
 - 모델의 평가나 실제 사용 시에는 모델의 예측값을 활용해야 함

- **Back Propagation**

- 기존 역전파와 다르게 계산에 사용된 시간, 시점의 수가 영향을 줌
 - 시간에 따른 역전파(BPTT, Backpropagation Through Time)를 사용



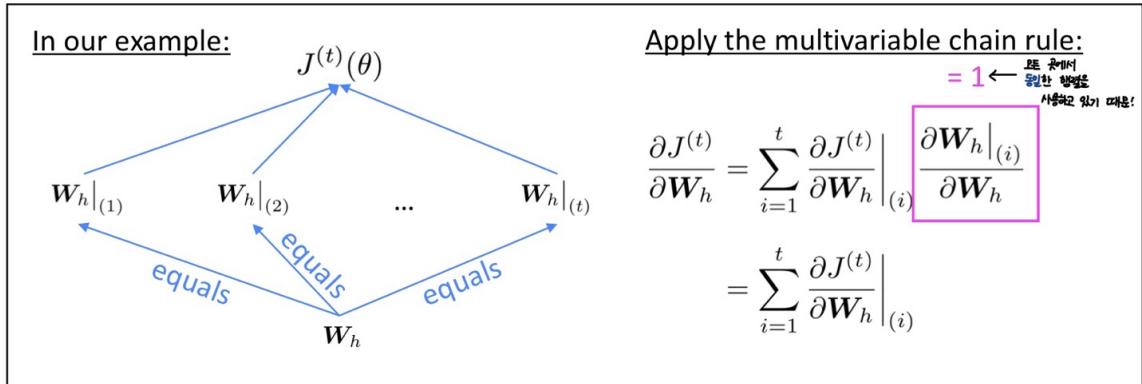
$$\frac{\partial J^{(t)}}{\partial \mathbf{W}_h} = \sum_{i=1}^t \left. \frac{\partial J^{(t)}}{\partial \mathbf{W}_h} \right|_{(i)}$$

Answer: Backpropagate over timesteps $i=t, \dots, 0$, summing gradients as you go.
 This algorithm is called “backpropagation through time” [Werbos, P.G., 1988, *Neural Networks 1*, and others]

- 동일한 가중치 W 를 반복적으로 적용
 - 단어 간 symmetric하지 않았던 NN 기반 LM의 단점을 보완

- Given a multivariable function $f(x, y)$, and two single variable functions $x(t)$ and $y(t)$, here's what the multivariable chain rule says:

$$\underbrace{\frac{d}{dt} f(\mathbf{x}(t), \mathbf{y}(t))}_{\text{Derivative of composition function}} = \frac{\partial f}{\partial \mathbf{x}} \frac{d\mathbf{x}}{dt} + \frac{\partial f}{\partial \mathbf{y}} \frac{d\mathbf{y}}{dt}$$



평가

Perplexity

- LM은 주어진 과거 단어 정보로부터 다음에 출현할 단어의 확률 분포를 출력하는 모델임
 - 대표적인 척도가 Perplexity
 - 출현할 단어의 확률에 대한 역수
- 값이 작을수록 좋은 언어 모델
- The standard evaluation metric for Language Models is perplexity.

$$\text{perplexity} = \prod_{t=1}^T \left(\underbrace{\frac{1}{P_{\text{LM}}(\mathbf{x}^{(t+1)} | \mathbf{x}^{(t)}, \dots, \mathbf{x}^{(1)})}}_{\text{Inverse probability of corpus, according to Language Model}} \right)^{1/T}$$

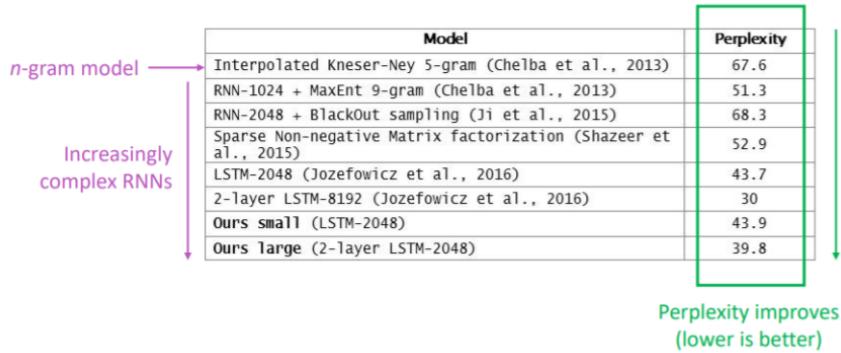
Normalized by
number of words

- This is equal to the exponential of the cross-entropy loss $J(\theta)$:

$$= \prod_{t=1}^T \left(\frac{1}{\hat{\mathbf{y}}_{\mathbf{x}^{t+1}}^{(t)}} \right)^{1/T} = \exp \left(\frac{1}{T} \sum_{t=1}^T -\log \hat{\mathbf{y}}_{\mathbf{x}^{t+1}}^{(t)} \right) = \exp(J(\theta))$$

Lower perplexity is better!

실험 결과



장/단점

장점

- 이전의 정보들을 활용할 수 있음
- 이론적으로는 길이가 긴 timestamp t 에 대해 처리 가능
 - 시퀀스 순서에 맞게 하나씩 입력해주기 때문에 입력의 길이에 제한이 없음
 - 어떤 길이의 텍스트이던 계산 가능
- 매 step마다 동일한 가중치 W 가 적용됨
 - 입력에 따른 모델의 크기가 증가하지 않음
 - 모델의 크기는 W_h 와 W_e 로 고정되어 있음
 - symmetric

단점

- 단어가 하나씩 입력됨
 - 순차적인 계산이 필요
 - recurrent 계산이 느림
- 정보의 손실 문제(기울기 소실 문제)
 - 실제로는 길이가 긴 timestep 에 대해서는 처리하기 어려움
⇒ 먼 곳에 있는 단어 정보를 반영하기는 어려움
 - 중요한 입력과 출력 단계 사이의 거리가 멀어질수록 그 관계를 학습하기 어려워짐



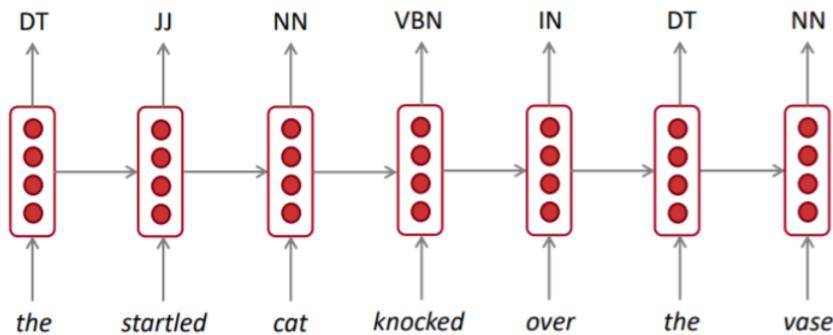
요약

- 언어 모델(Language Model)은 다음 단어를 예측하는 시스템임
- 순환 신경망(Recurrent Neural Network)
 - 어떤 길이의 순차적인 입력을 받음
 - 각 단계에서 동일한 가중치를 적용
 - 선택적으로 각 단계에서 출력을 생성
- 순환 신경망 ≠ 언어 모델

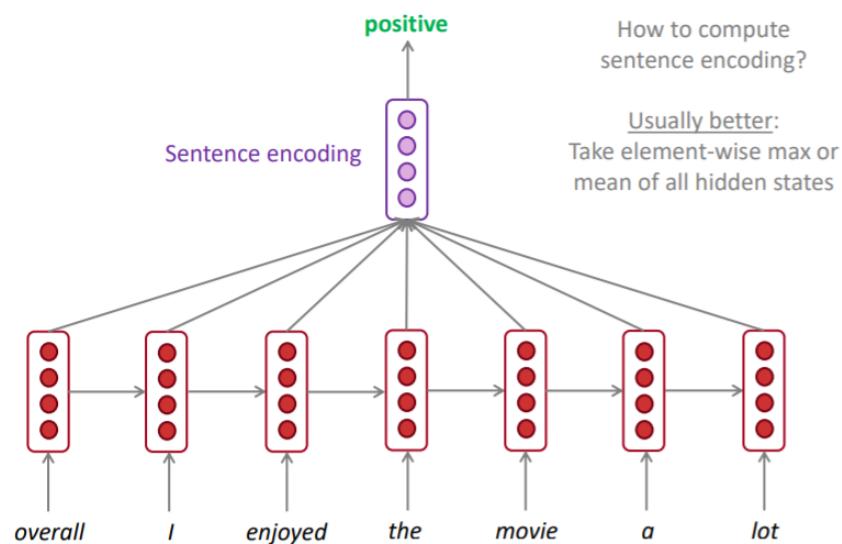
2. Other RNN uses

- RNN의 입출력은 task(목적)에 따라 얼마든지 달라질 수 있음

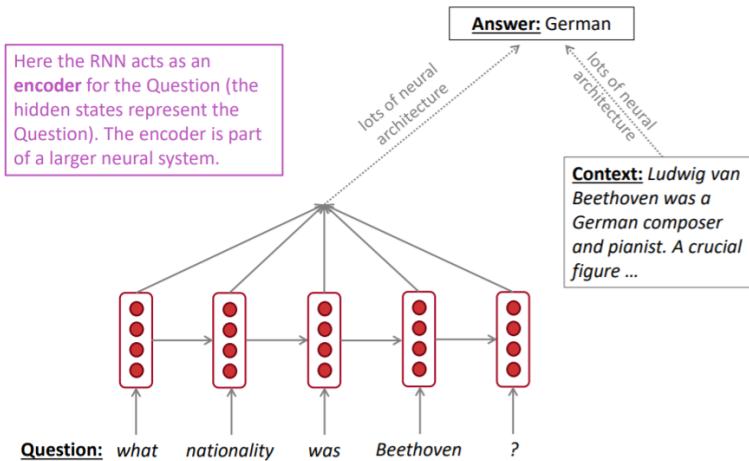
tagging: part-of-speech tagging, named entity recognition



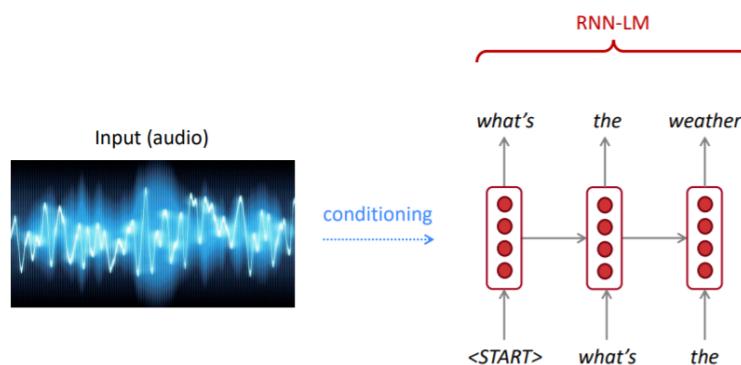
문장 분류/ 감정 분류



encoder module: question answering, machine translation



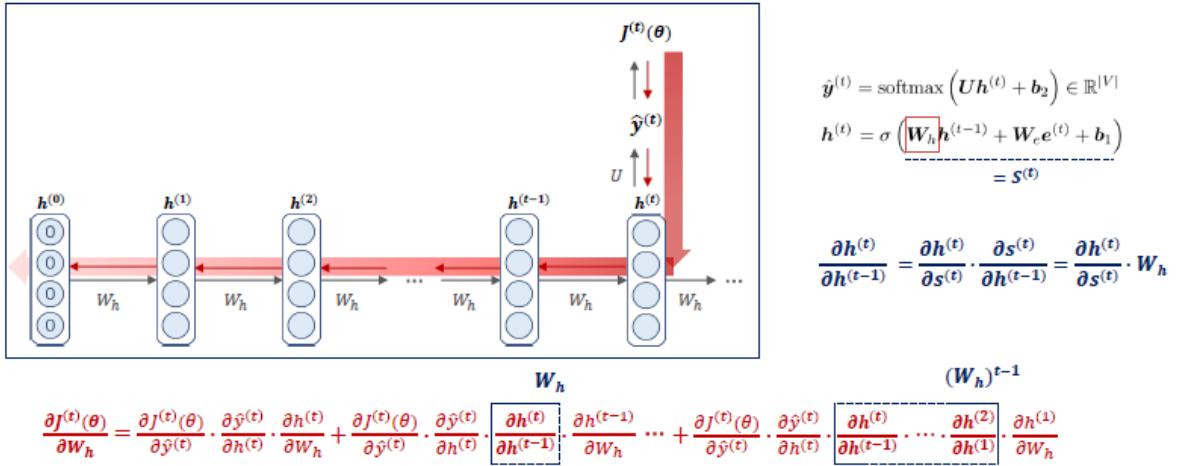
generate text: speech recognition, machine translation, summarization



This is an example of a *conditional language model*.
We'll see Machine Translation in much more detail later.

3. Vanishing and Exploding Gradients Problems

Vanishing/Exploding Gradients



동일한 가중치(W_h) 공유

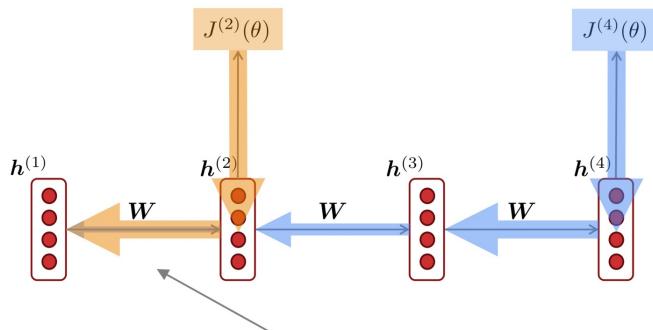
- W_h 가 작을 수록(< 1) 반복적으로 곱해지는 값이 0에 가까워져 gradient vanishing
- W_h 가 클 수록(> 1) 반복적으로 곱해지는 값이 기하급수적으로 커져 gradient exploding

출처: 고려대학교 DSBA 연구실_Seminar 자료

• 왜 문제일까?

◦ Vanishing Gradient

- 멀리 떨어진 곳에서의 기울기 신호는 가까운 곳에서의 기울기 신호보다 훨씬 작기 때문에 손실됨
 - Long Term Dependency(장기 의존성) 문제
 - 모델 가중치는 근처 영향에 대해서만 업데이트되며 장기적인 영향은 고려되지 않음



◦ Exploding Gradient

- 파라미터 update step이 너무 커지는 문제
 - 제대로 된 학습이 이루어지지 못할 수 있음
- 해결) Gradient Clipping

- 기울기가 어떤 임계값보다 크면 SGD를 적용하기 전에 값을 조정

Algorithm 1 Pseudo-code for norm clipping

```

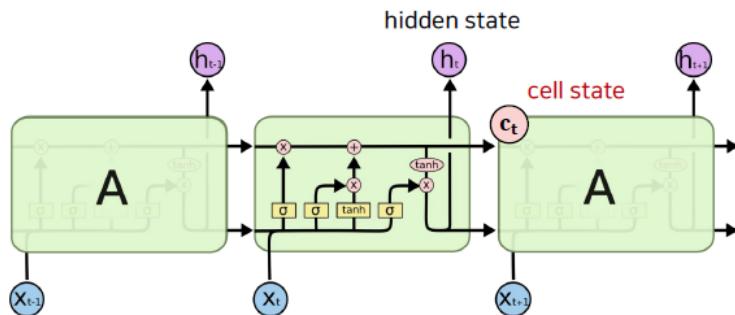
 $\hat{\mathbf{g}} \leftarrow \frac{\partial \mathcal{E}}{\partial \theta}$ 
if  $\|\hat{\mathbf{g}}\| \geq threshold$  then
     $\hat{\mathbf{g}} \leftarrow \frac{threshold}{\|\hat{\mathbf{g}}\|} \hat{\mathbf{g}}$ 
end if

```

4. Long Short-Term Memory RNNs(LSTM)

구조

전체 구조



- hidden state를 통해 short-term memory를 조절하고 cell state를 통해 long term memory 보존
- forget, input, output 3개의 gate를 통해 매 time step의 cell state와 hidden state, input에서 취할 정보의 양을 결정

세부 구조

