자연어처리 실습

NLP Laboratory

이지현 jihyunlee@postech.ac.kr

강석훈 sh.kang@postech.ac.kr

이원준 lee1jun@postech.ac.kr

조정훈 jeonghuncho@postech.ac.kr

Natural Language Processing

자연어 처리 (NLP; Natural Language Processing)

- 인공지능의 한 분야로 사람의 언어를 컴퓨터가 알아듣도록 처리하는 역할

- 자연어 처리의 최종 목표는 컴퓨터가 사람의 언어를 이해하고 여러 문제를 수행할 수 있도록 하는 것

- 응용분야: 감성 분석, 기계 번역, 대화 시스템 (챗봇), 요약 등

NLP는 왜 어려울까?

1. 모호성

차를 마시러 공원에 가던 차 안에서 나는 그녀에게 차였다.

K: I was in the car going to the park for tea, and I got dumped by her.

N: I was dumped by her on my way to the park for tea.

G: In the car on the way to the park for tea, I was dumped by her.

M: In the car on the way to the park for tea, I was the car to her

NLP는 왜 어려울까?

1. 모호성

나는 철수를 안 때렸다.

해석 1: 철수는 맞았지만, 때린 사람이 나는 아니다.
해석 2: 나는 누군가를 때렸지만 ,그게 철수는 아니다.
해석 3: 나는 누군가를 때린 적도 없고, 철수도 맞은 적이 없다.

선생님은 울면서 돌아오는 우리를 위로했다.

해석 1: (선생님은 울면서) 돌아오는 우리를 위로했다.
해석2: 선생님은 (울면서 돌아오는 우리를) 위로했다

NLP는 왜 어려울까?

2. 다양한 표현

고양이가 앉아 있다.

고양이가 킥보드 위에 앉아 있다.

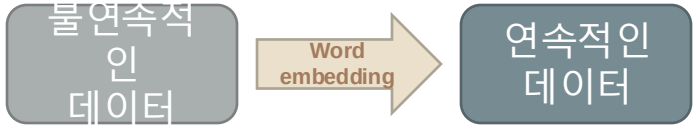
전동 킥보드 위에 있는 흰색 고양이

킥보드를 타려고 하는 흰색과 검정색이 섞여 있는 고양이

킥보드에 앉아 카메라를 보는 고양이

NLP는 왜 어려울까?

3. 불연속적 데이터



- 차원의 저주

  모든 단어를 symbol로 고려하면, 어휘의 크기만큼의 차원이 존재

  Sparseness 문제를 해결하기 위해단어를 적절하게 segmentation 해야함

  적절한 word embedding을 통해 dimension reduction을 하여 문제 해결 가능

- 노이즈와 정규화

  모든 분야에서 노이즈를 신호로부터 적절히 분리해내는 일은 매우 중요

  이미지의 경우 RGB 값 중 하나가 바뀌어도 전체의 시각적 의미에는 변화 X

  하지만 단어는 불연속적인 symbol -> 단어가 살짝만 바뀌어도 문장의 의미가 완전히 변화

NLP는 왜 어려울까?

4. 한국어 자연어 처리의 어려움

- 교착어
  어순이 중요시되는 영어, 중국어와 달리 어근에 접사가 붙어 의미와 문법적 기능 부여
  “잡다”-> 잡다, 잡히다, 잡히시다, 잡히셨다, 잡았다, 잡겠다, 잡더라, 잡혔다
  -> 하나의 어근에서 비롯된 비슷한 의미의 단어가 생성
  -> 추가적인 분절 (segmentation)을 통해 같은 어근에서 생겨난 단어 처리
- 띄어쓰기
  추가적인 분절을 통해 띄어쓰기를 정제해주는 과정이 필요
- 평서문과 의문문
  한국어는 의문문과 평서문이 같은 형태의 문장 구조를 가짐 -> 음성 인식의 경우 어려움을 겪음
- 주어 생략
  영어는 기본적으로 명사가 중요시 되지만 한국어는 동사를 중요시하고 주어가 자주 생략됨 -> 문맥 파악에 어려움

Text Pre-processing

토큰화 (Tokenization)

주어진 코퍼스에서 token이라 불리는 단위로 나누는 작업

단어 토큰화 (Word Tokenization)

- 토큰의 기준을 단어로 하는 경우 word tokenization라고 하며 여기서 단어는 단어 단위 외에도 단어구, 의미를 갖는 문자열로 간주되기도 함

Time is an illusion. Lunchtime double so!

-> “Time”, “is”, “an”, “illusion”, “Lunchtime”, “double”, “so”

토큰화 (Tokenization)

Don't be fooled by the dark sounding name, Mr. Jone's Orphanage is as cheery as cheery goes for a pastry shop.

Don’t와 Jone’s 의 토큰화?

-> Don’t, Don t, Don’t, Do n’t, Jone’s, Jone s, Jone, Jones

```
from nltk.tokenize import word_tokenize
from nltk.tokenize import WordPunctTokenizer
from tensorflow.keras.preprocessing.text import text_to_word_sequence
```

토큰화 (Tokenization)

고려해야할 사항

- 구두점이나 특수 문자 단순 제외 주의
  마침표는 문장의 경계를 알 수 있는데 도움이 되므로 단어를 뽑아낼 때 마침표를 제외하지 않을 수 있음
  Ph.D 와 같이 단어 자체에 구두점을 갖고 있는 경우, 숫자 사이의 컴마가 들어가 있는 경우 등

- 줄임말과 단어 내에 띄어쓰기가 있는 경우
  영어권 언어의 ‘는 압축된 단어를 다시 펼치는 역할을 하기도 함 (we’re -> we are)

토큰화 (Tokenization)

문장 토큰화 (Sentence Tokenization)

토큰의 단위가 문장일 경우 단순히 ? . ! 를 기준으로 문장을 잘라낼 수도 있지만..

IP 192.168.56.31 서버에 들어가서 로그 파일 저장해서 aaa@gmail.com로 결과 좀 보내줘. 그 후 점심 먹으러 가자.

Since I'm actively looking for Ph.D. students, I get the same question a dozen times every year.

정제 (Cleaning), 정규화 (Normalization)

정제(cleaning) : 갖고 있는 코퍼스로부터 노이즈 데이터를 제거

정규화(normalization) : 표현 방법이 다른 단어들을 통합시켜서 같은 단어로 만들어줌

- 규칙에 기반한 표기가 다른 단어들의 통함

- **대, 소문자 통합**
- **불필요한 단어의 제거 (특수 문자, 등장 빈도가 적은 단어 등)**
- **정규 표현식**
  얻어낸 코퍼스에서 노이즈 데이터의 특징을 잡아낼 수 있다면, 정규 표현식을 통해 제거할 수 있는 경우가 많음 (참고: https://wikidocs.net/21703)

Word Embedding

Word Embedding

단어를 벡터로 표현하는 방법으로, 단어를 밀집된 표현 (Dense representation)으로 변환

- 희소 표현 (Sparse Representation)

  원-핫 인코딩을 통해 나온 원-핫 벡터들은 표현하고자 하는 단어의 인덱스의 값만 1이고 나머지는 전부 0으로 표현됨.

  이렇게 벡터 또는 행렬의 값이 대부분 0으로 표현되는 방법이 희소 표현 (Sparse Representation)

  단어의 개수가 늘어나면 벡터의 차원이 한없이 커지며 단어 집합이 클수록 고차원의 벡터가 됨

  ex) 10,000개의 단어

  강아지 = [0 0 0 0 1 0 0 0 0 … 0 … 0]

  => 공간적 낭비

Word Embedding

- 밀집 표현 (Dense Representation)

  벡터의 차원을 단어 집합의 크기로 상정하지 않음

  사용자가 설정한 값으로 모든 단어의 벡터 표현의 차원을 맞춤

  -> element들이 0, 1이 아닌 실수 값을 갖게 됨

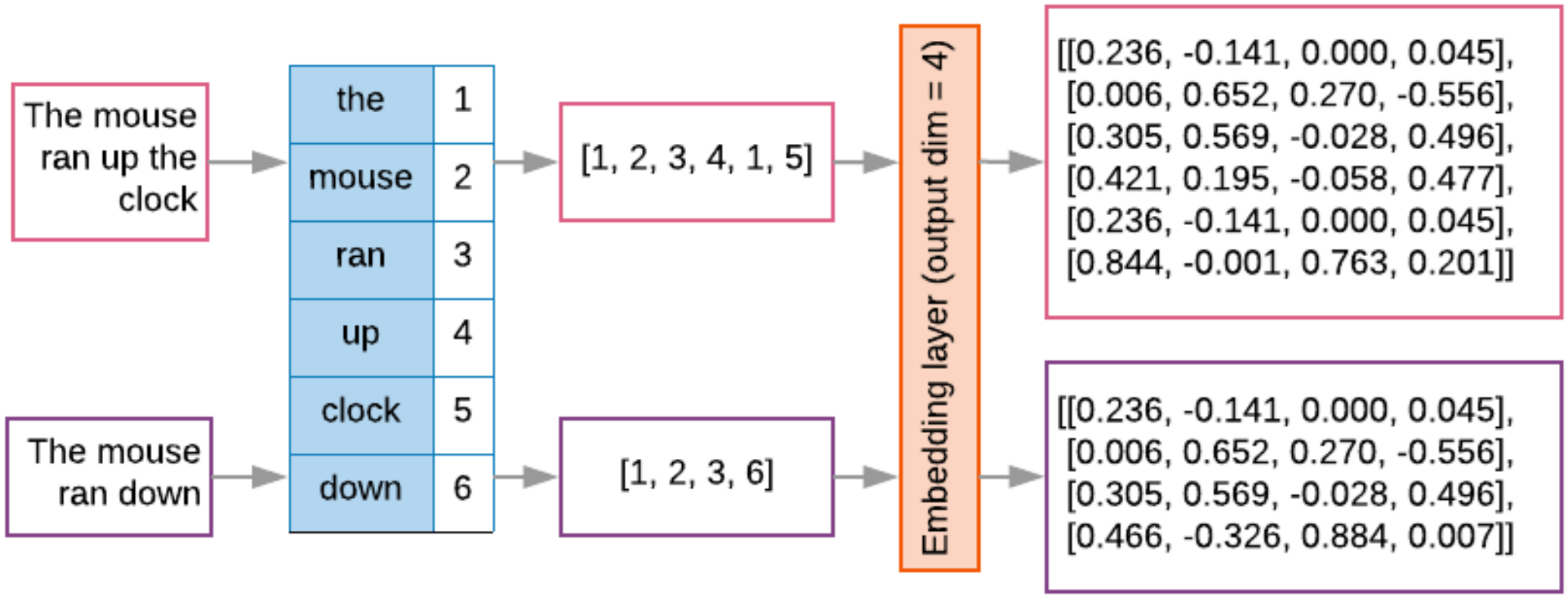
  ex) dim = 128, 강아지 = [0.2 1.8 1.1 …]

- Word Embedding

  단어를 dense vector의 형태로 표현하는 방법

	One-hot Vec.	Embedding Vec.
차원	고차원	저차원
다른 표현	Sparse Vector	Dense Vector
표현 방법	수동	훈련 데이터로부터 학습
값의 타입	1, 0	실수

Word Embedding



Language Model

언어 모델 (Language Model)

언어라는 현상을 모델링하고자 단어 sequence (문장)에 확률을 할당하는 모델

통계를 이용한 방법과 인공 신경망을 이용한 방법으로 구분

이전 단어(토큰)들이 주어졌을 때, 다음 단어(토큰)를 예측하도록 하는 것

ex) 기계 번역: P(나는 버스를 **탔다**) > P(나는 버스를 **태운다**)

오타 교정: P(**달려갔다**|선생님이 교실로 부리나케) > P(**잘려갔다**|선생님이 교실로 부리나케)

음성 인식: P(나는 **메론을** 먹는다) > P(나는 **메롱을** 먹는다)

통계적 언어 모델 (Statistical Language Model)

하나의 단어 w, 단어 시퀀스 W, n개의 단어가 등장하는 W의 확률

$$P(W) = P(w_1, w_2, w_3, w_4, w_5, \ldots, w_n)$$

다음 단어 등장 확률

$$P(w_n|w_1, \ldots, w_{n-1})$$

전체 단어 시퀀스 W의 확률

$$P(W) = P(w_1, w_2, w_3, w_4, w_5, \ldots w_n) = \prod_{i=1}^{n} P(w_i|w_1, \ldots, w_{i-1})$$

*Chain Rule

$$P(x_1, x_2, x_3 \ldots x_n) = P(x_1)P(x_2|x_1)P(x_3|x_1, x_2)\ldots P(x_n|x_1 \ldots x_{n-1})$$

통계적 언어 모델 (Statistical Language Model)

카운트 기반의 접근

$$P(w_3|w_1\ w_2) = \frac{\text{count}(w_1, w_2, w_3)}{\text{count}(w_1, w_2)}$$

-> Sparsity Problem 발생

Test 하고자 하는 단어 배열이 train corpus에 없다면 이 단어 시퀀스에 대한 확률은 0이 됨

=> 이 문제를 완화하기 위해 n-gram, smoothing 등 적용 가능

통계적 언어 모델 (Statistical Language Model)

N-gram Language Model

이전에 등장한 모든 단어를 고려하는 것이 아니라, n개의 단어만 고려하는 방법

⇒Corpus에서 카운트하지 못하는 경우 감소

3-gram: $P(w_n|w_{n-1}w_{n-2}\dots w_1) \approx P(w_n|w_{n-1}w_{n-2})$

An adorable little boy is spreading smiles

unigrams : an, adorable, little, boy, is, spreading, smiles

bigrams : an adorable, adorable little, little boy, boy is, is spreading, spreading smiles

trigrams : an adorable little, adorable little boy, little boy is, boy is spreading, is spreading smiles

4-grams : an adorable little boy, adorable little boy is, little boy is spreading, boy is spreading smiles

통계적 언어 모델 (Statistical Language Model)

N-gram Language Model 한계

- Sparsity Problem이 여전히 존재
- 적합한 n 찾기

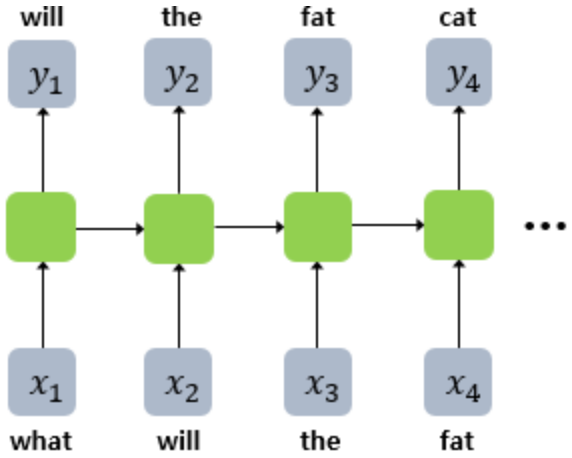
Large n은 언어 모델의 성능을 높일 수 있지만
실제 훈련 코퍼스에서 해당 n-gram을 카운트 할 수 있는 확률이 적어져 sparsity problem 심화, 모델 size가 커짐

Small n은 훈련 코퍼스에서 카운트는 잘 되겠지만 근사의 정확도는 현실의 확률 분포와 멀어짐
=> 최대 5를 넘기지 않는 것을 권장

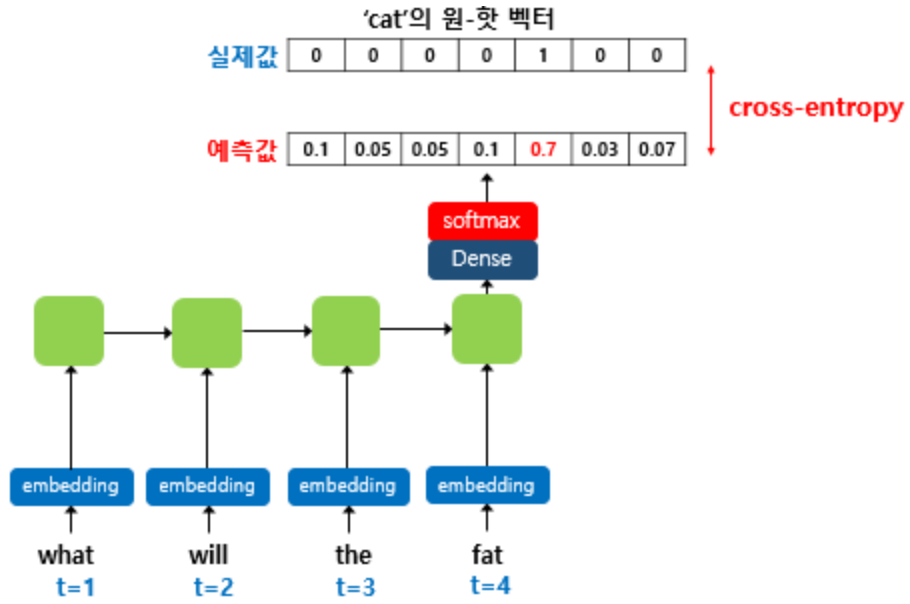
Recurrent Neural Network (RNN)

Sequence to Sequence 모델

“What will the fat cat sit on”



이전 시점의 출력을 현재 시점의 입력으로 받음

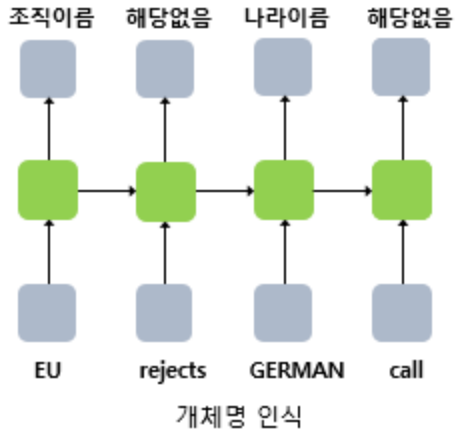


훈련 과정에서는 sequence를 모델의 입력으로 넣으면,

다음 sequence를 예측하도록 훈련

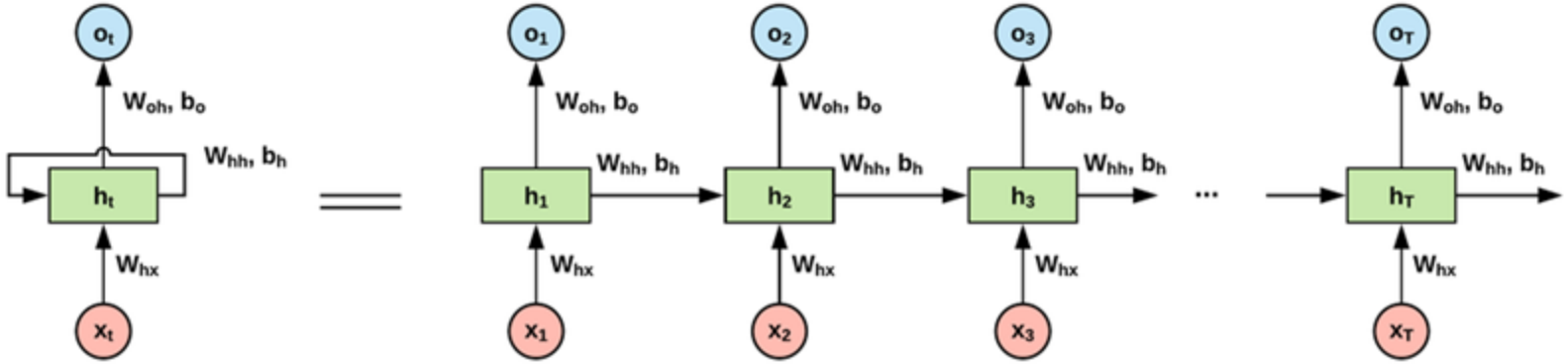
Recurrent Neural Network (RNN)

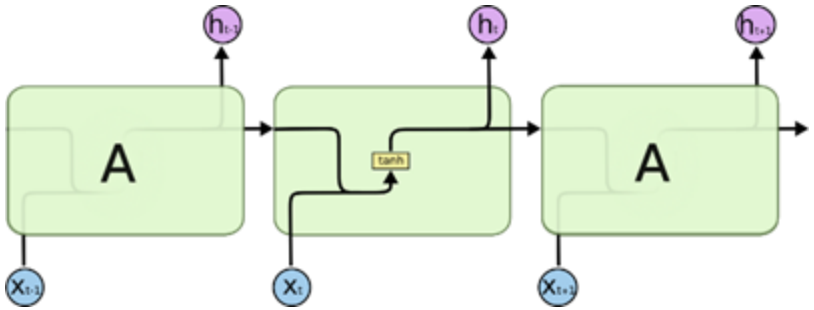
e.g.
스팸 메일 분류,
감정분류,



e.g.
개체명인식,
Anomaly detection,

Recurrent Neural Network (RNN)





$$h_t = \tanh(h_{t-1}W_h + x_tW_x + b)$$

장점

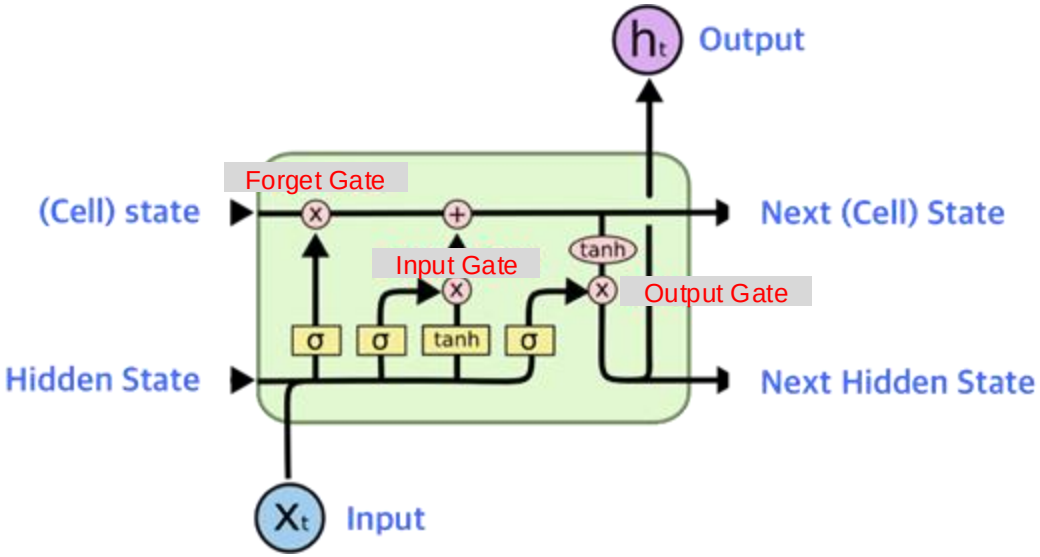
모델이 간단하고 어떤 길이의 sequential data라도 처리할 수 있음

단점

병렬화 불가능 (벡터가 순차적으로 입력되기 때문)

Exploiding Gradient, Vanishing Gradient

Long Term Short Memory (LSTM)

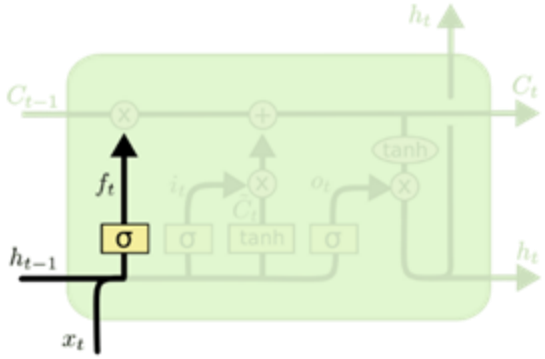


Vanishing Gradient Problem 을 해결하기 위해 등장

- Forget gate: 과거 정보를 얼마나 유지할 것인가?
- Input gate: 새로 입력된 정보는 얼마나 활용할 것인가?
- Output gate: 두 정보를 계산하여 나온 출력 정보를 얼만큼 넘겨줄 것인가?

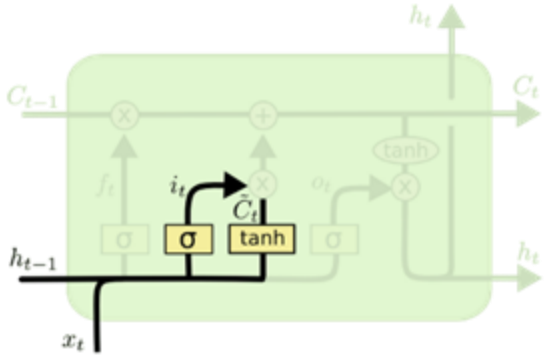
Long Term Short Memory (LSTM)

Forget Gate Layer



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

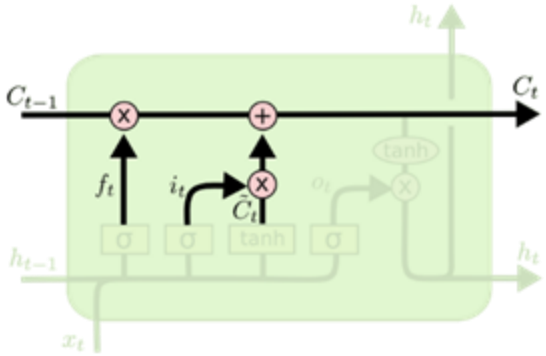
Input Gate Layer



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

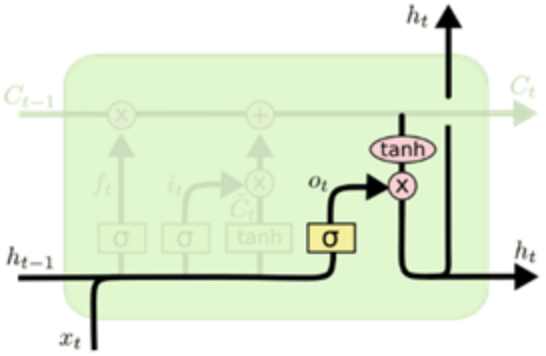
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Update cell state



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Output Gate Layer



$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

NLP
평가 안내

1. Kaggle Competition

Text Classification task

Kaggle 가입 진행, 구글 시트에 기입

2. Test

총 20문제 (객관식)

Introduction

Text Generation & Generative Pre-trained Transformer

Machine Translation & Speech

Tokenization & Embedding