

14기 NLP seminar

ToBig's 13기 오진석

Lecture 13

Contextual Word Embeddings

Content

Unit 01 | Reflections on word representations

Unit 02 | Pre-ELMo and ELMO

Unit 03 | ULMfit and onward

Unit 04 | Transformer architectures

Unit 05 | BERT

Unit 01 | Reflections on word representations

Unit 01 | Reflections on word representations

Representations for a word

- ‘단어(word)를 표현(representation)’하는 것으로 워드 임베딩(word embedding)을 통해, 단어를 벡터로 표현함으로써 컴퓨터가 이해할 수 있도록 자연어를 적절히 변환할 수 있음
- 단어를 벡터로 표현하는 방법으로는 Word2vec, GloVe, fastText 등이 있음

Unit 01 | Reflections on word representations

Representations for a word before 2012

[2012년 이전 word vectors]

	POS WSJ (acc.)	NER CoNLL (F1)	
State-of-the-art*	97.24	89.31	Rule-based, 일반적인 ML 방법론
Supervised NN	96.37	81.47	
Unsupervised pre-training followed by supervised NN**	97.20	88.87	
+ hand-crafted features***	97.29	89.59	

Unit 01 | Reflections on word representations

Representations for a word before 2012

[2012년 이전 word vectors]

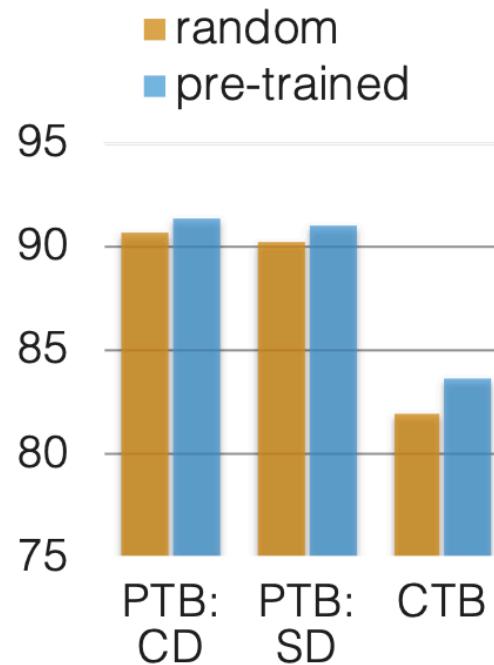
	POS WSJ (acc.)	NER CoNLL (F1)
State-of-the-art*	97.24	89.31
Supervised NN	96.37	81.47
Unsupervised pre-training followed by supervised NN**	97.20	88.87
+ hand-crafted features***	97.29	89.59

- Supervised NN과 word representations 방법인 Word2vec, GloVe 등을 같이 활용한 것
- 약 130,000 단어 사용
- Word window : 11, hidden layer : 100
- 7주 간 학습
- 당시에 일반적인 rule-based보다는 성능 자체는 떨어졌지만 FE를 통해 성장 가능성을 확인할 수 있게됨

Unit 01 | Reflections on word representations

Representations for a word after 2014

- 대부분의 경우 pre-trained word vector를 사용할 때, 보다 높은 성능을 보임



random : random initialize를 시작으로 word vector 생성

Pre-trained : 미리 학습된 word vector를 사용

→ 보다 방대한 양의 데이터로 학습된 결과를 활용하기 때문에 성능 향상에 도움이 됨

Unit 01 | Reflections on word representations

Tips for unknown words with word vectors

- unknown words, <unk>를 처리할 때, 보다 효과적임
 - 데이터 set에서 잘 나타나지 않는 word(5회 이하)
 - train이 아닌 test에서만 나타난 word(OOV)

Unit 01 | Reflections on word representations

Tips for unknown words with word vectors

- unknown words, <unk>를 처리할 때, 보다 효과적임
 - 데이터 set에서 잘 나타나지 않는 word(5회 이하)
→ <unk>를 따로 구별할 수 없으며, 중요한 의미를 가지는 단어를 놓칠 수 있음
 - train이 아닌 test에서만 나타난 word(OOV)

Unit 01 | Reflections on word representations

Tips for unknown words with word vectors

- unknown words, <unk>를 처리할 때, 보다 효과적임
 - 데이터 set에서 잘 나타나지 않는 word(5회 이하)
→ <unk>를 따로 구별할 수 없으며, 중요한 의미를 가지는 단어를 놓칠 수 있음
 - train이 아닌 test에서만 나타난 word(OOV)
- unknown words, <unk>를 처리하는 방법
 - char-level model로 word vector를 생성
 - Pre-trained word vector를 사용
 - random vector를 부여하여 vocab에 추가

Unit 01 | Reflections on word representations

Representations for a word

- ‘단어(word)를 표현(representation)’하는 것으로 워드 임베딩(word embedding)을 통해, 단어를 벡터로 표현함으로써 컴퓨터가 이해할 수 있도록 자연어를 적절히 변환할 수 있음
- 단어를 벡터로 표현하는 방법으로는 Word2vec, GloVe, fastText 등이 있음

Unit 01 | Reflections on word representations

Representations for a word

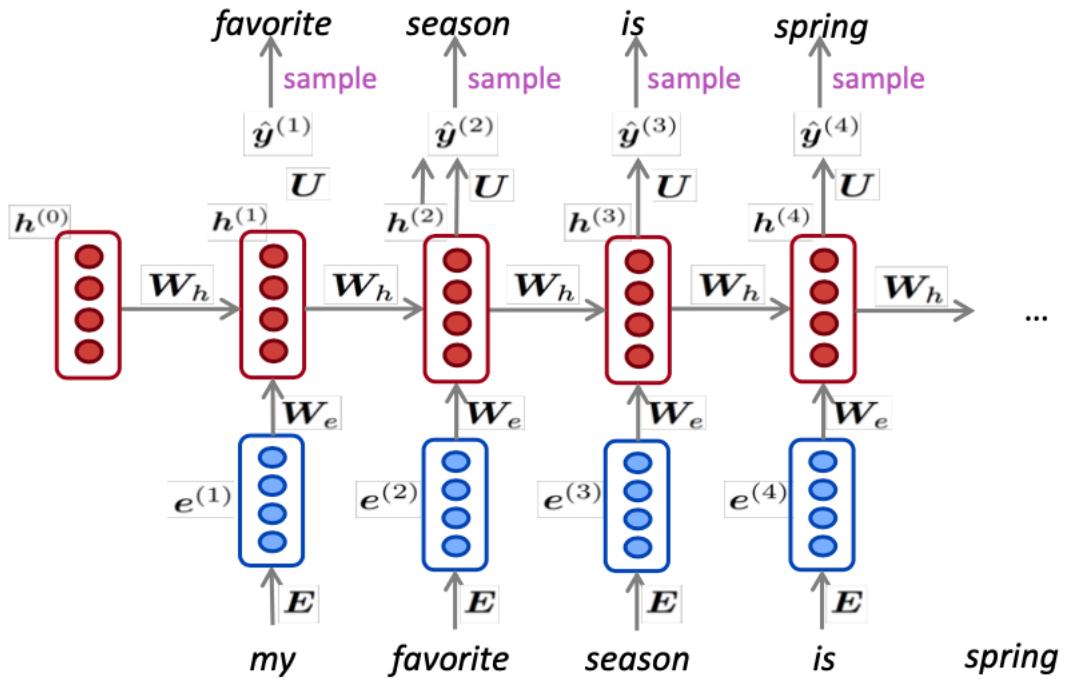
- ‘단어(word)를 표현(representation)’하는 것으로 워드 임베딩(word embedding)을 통해, 단어를 벡터로 표현함으로써 컴퓨터가 이해할 수 있도록 자연어를 적절히 변환할 수 있음
- 단어를 벡터로 표현하는 방법으로는 Word2vec, GloVe, fastText 등이 있음
- 1개의 단어에 1개의 vector로 표현하는 과정에서 문제가 발생함
 - word token이 나타나는 context에 따라 달라지는 word type을 고려하지 못함
 - word의 언어적, 사회적, 의미에 따라 다른 측면을 고려하지 못함

Unit 01 | Reflections on word representations

Representations for a word

- Neural Language Model

- 문장의 sequence를 고려하여 다음 word를 예측하는 모델
- Input은 pre-trained된 word vector 혹은 random initialize 된 word vector
- recurrent layer을 통과함으로써 문장의 sequence가 고려된 context를 유지할 수 있음



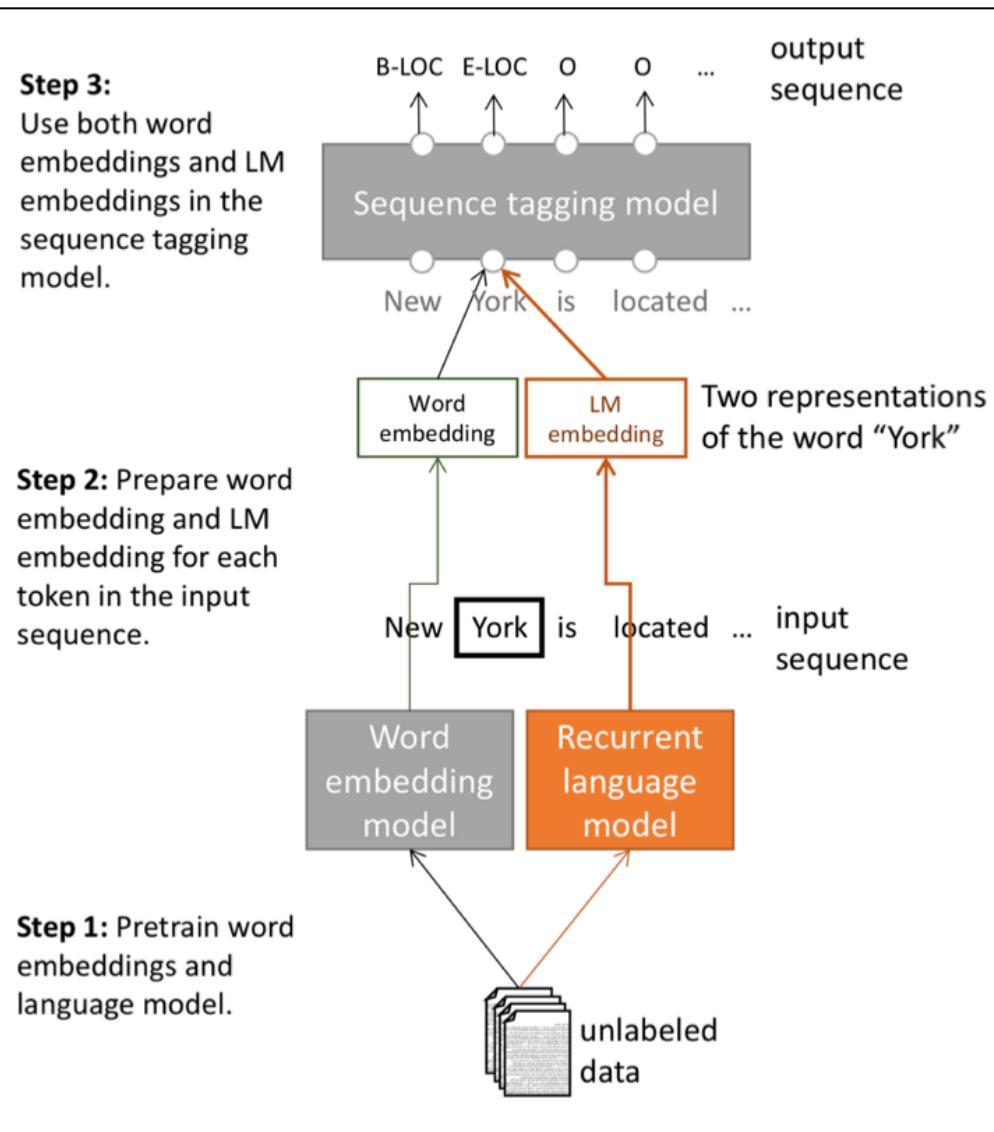
Unit 02 | Pre-ELMo and ELMO

Unit 02 | Pre-ELMo and ELMO

Pre-ELMo - Peters et al. 2017, Semi-supervised sequence tagging with bidirectional language models

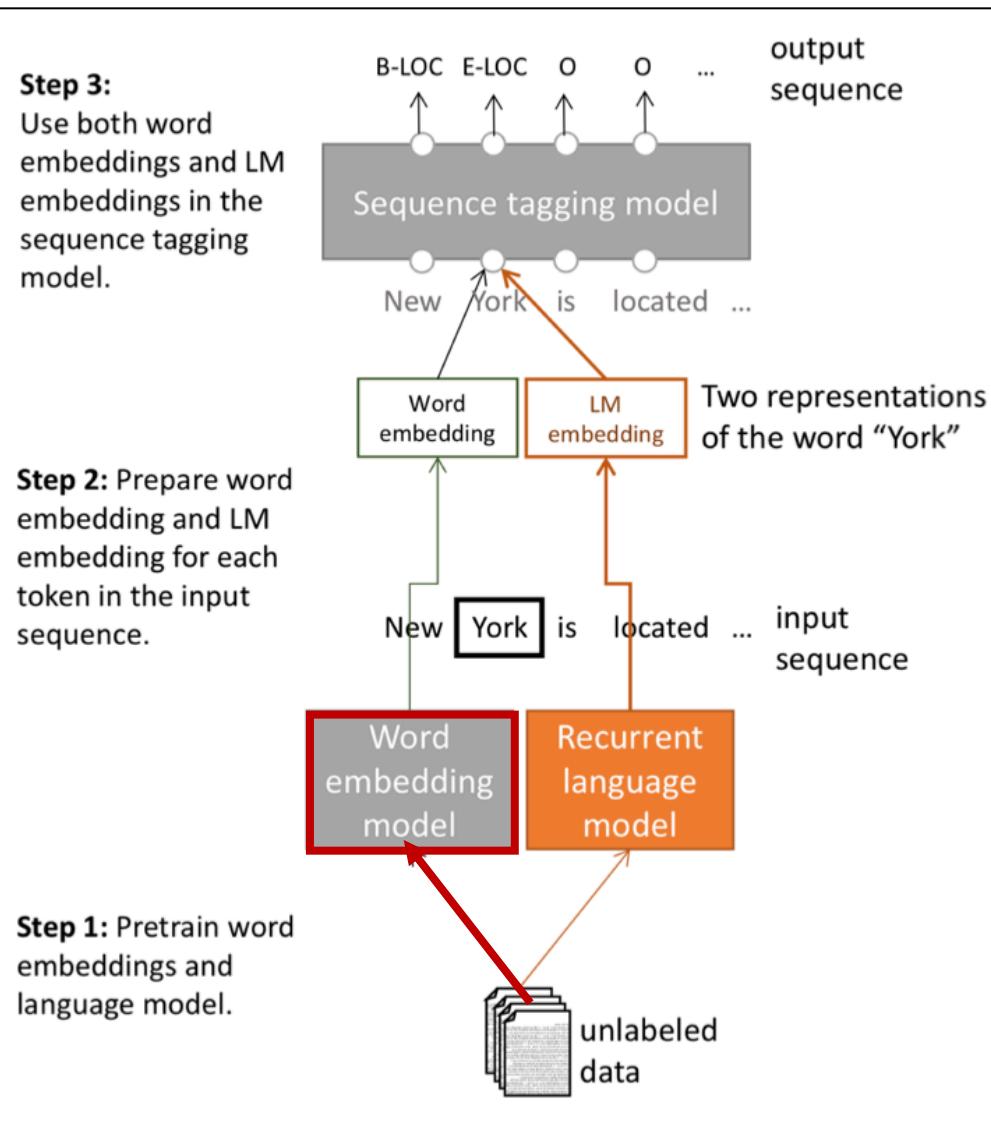
- ELMo 이전 ELMo 저자가 발표한 TagLM을 CS224N에서는 Pre-ELMo로 지칭
- NER과 같은 small task-labeled data에서 RNN을 통해 context가 유지되는 word vector를 찾기 위한 방법
- 많은 데이터가 NLM으로 미리 학습된 word vector를 사용하는 Semi-supervised 접근법

Unit 02 | Pre-ELMo and ELMO



에서는 Pre-ELMo로 지칭
통해 context가 유지되는 word vector를 찾기 위한 방법
사용하는 Semi-supervised 접근법

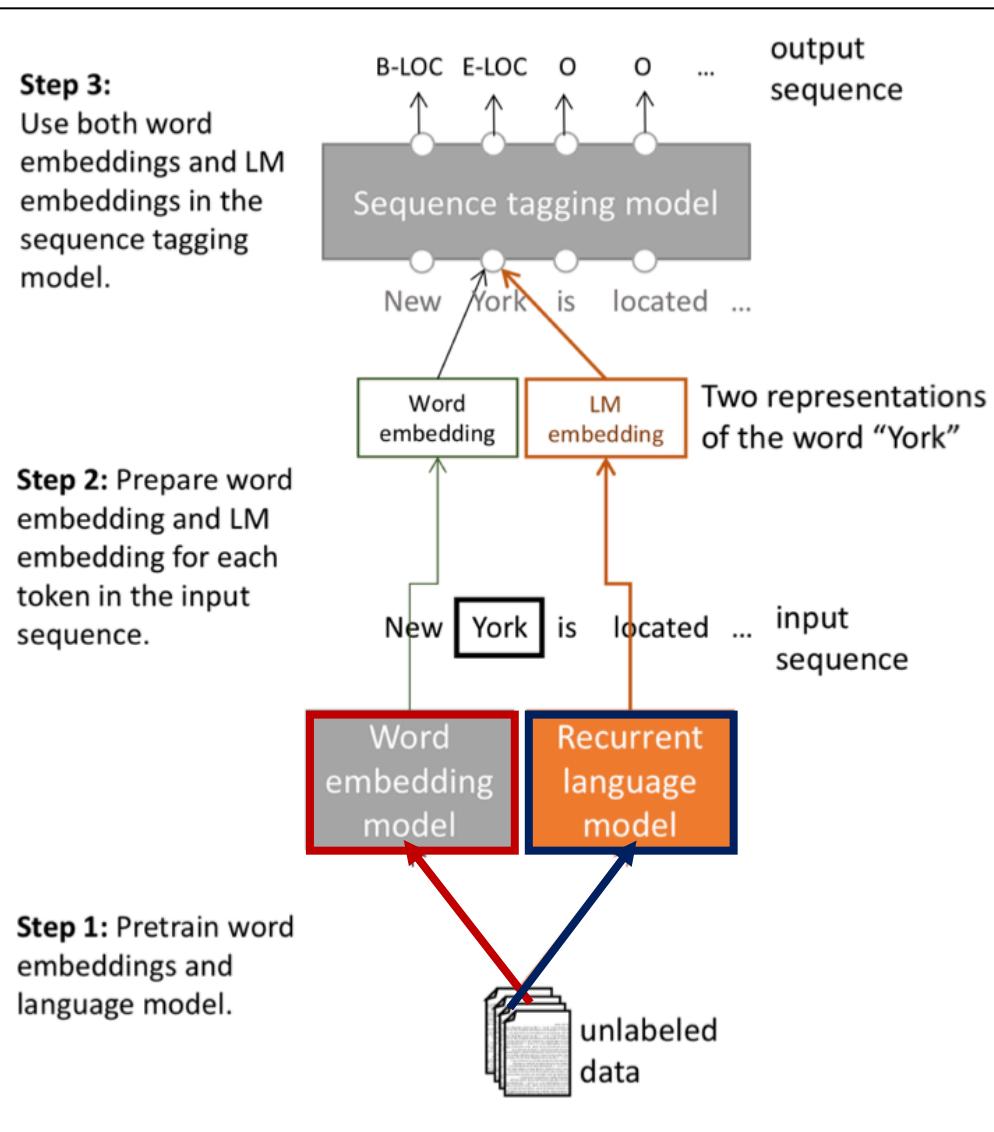
Unit 02 | Pre-ELMo and ELMO



[STEP 1]

- Word2vec과 같은 conventional word embedding model로 word vector를 학습하는 Semi-supervised 접근법

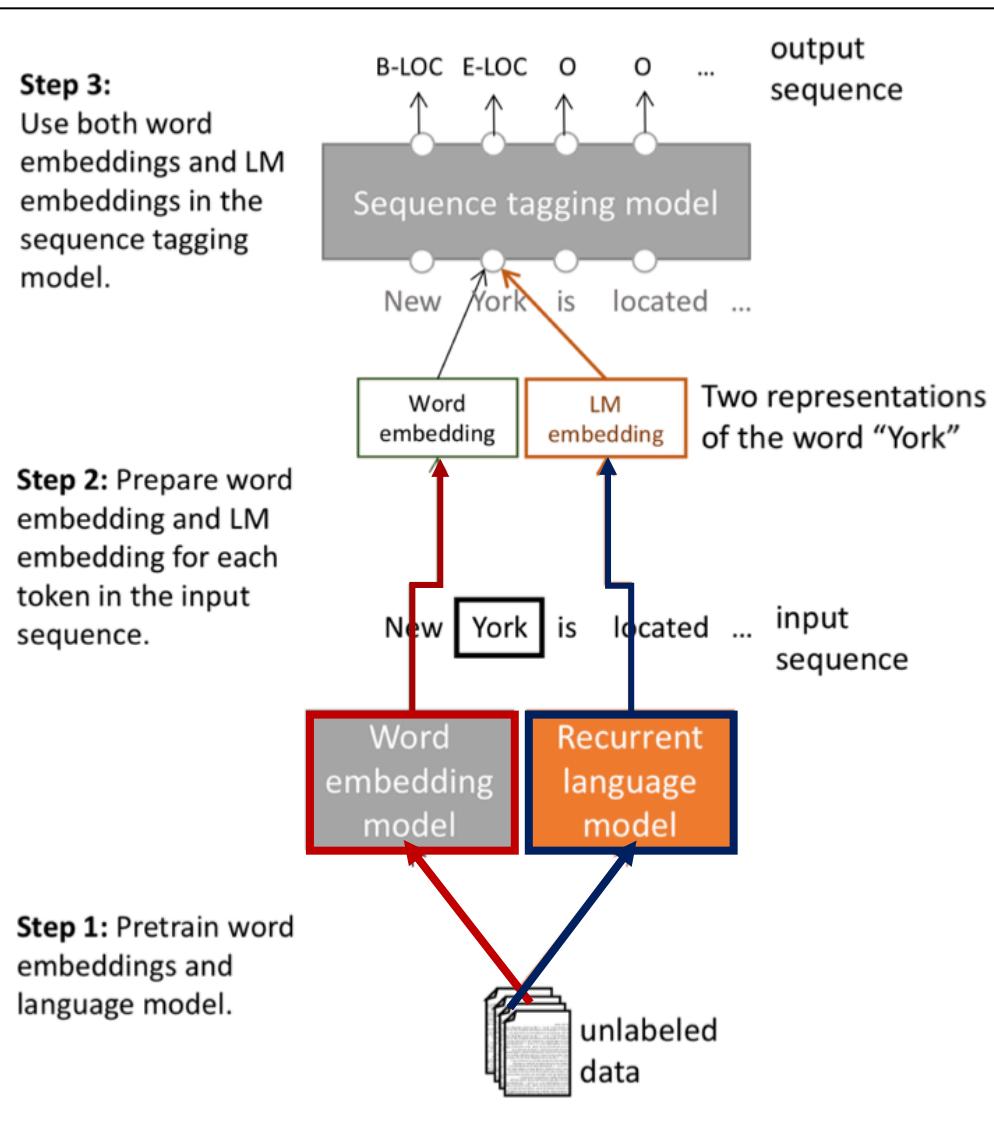
Unit 02 | Pre-ELMo and ELMO



[STEP 1]

- Word2vec과 같은 conventional word embedding model로 word vector를 학습
- 동시에 bi-LSTM과 같은 NLM model로 word vector를 학습

Unit 02 | Pre-ELMo and ELMO



[STEP 1]

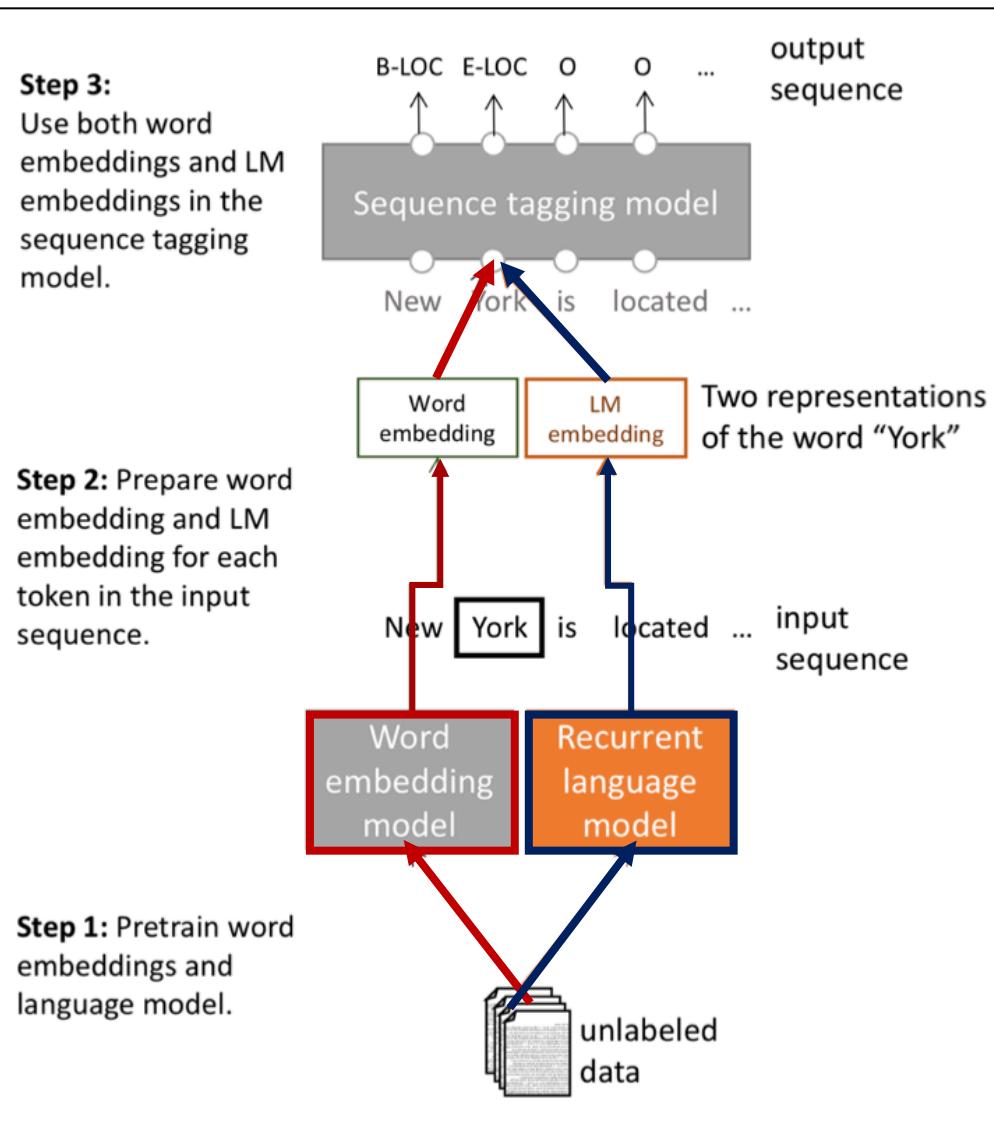
- Word2vec과 같은 conventional word embedding model로 word vector를 학습

- 동시에 bi-LSTM과 같은 NLM model로 word vector를 학습

[STEP 2]

- 2가지 접근법으로 학습된 word embedding을 input으로 사용함

Unit 02 | Pre-ELMo and ELMO



[STEP 1]

- Word2vec과 같은 conventional word embedding model로 word vector를 학습

- 동시에 bi-LSTM과 같은 NLM model로 word vector를 학습

[STEP 2]

- 2가지 접근법으로 학습된 word embedding을 input으로 사용함

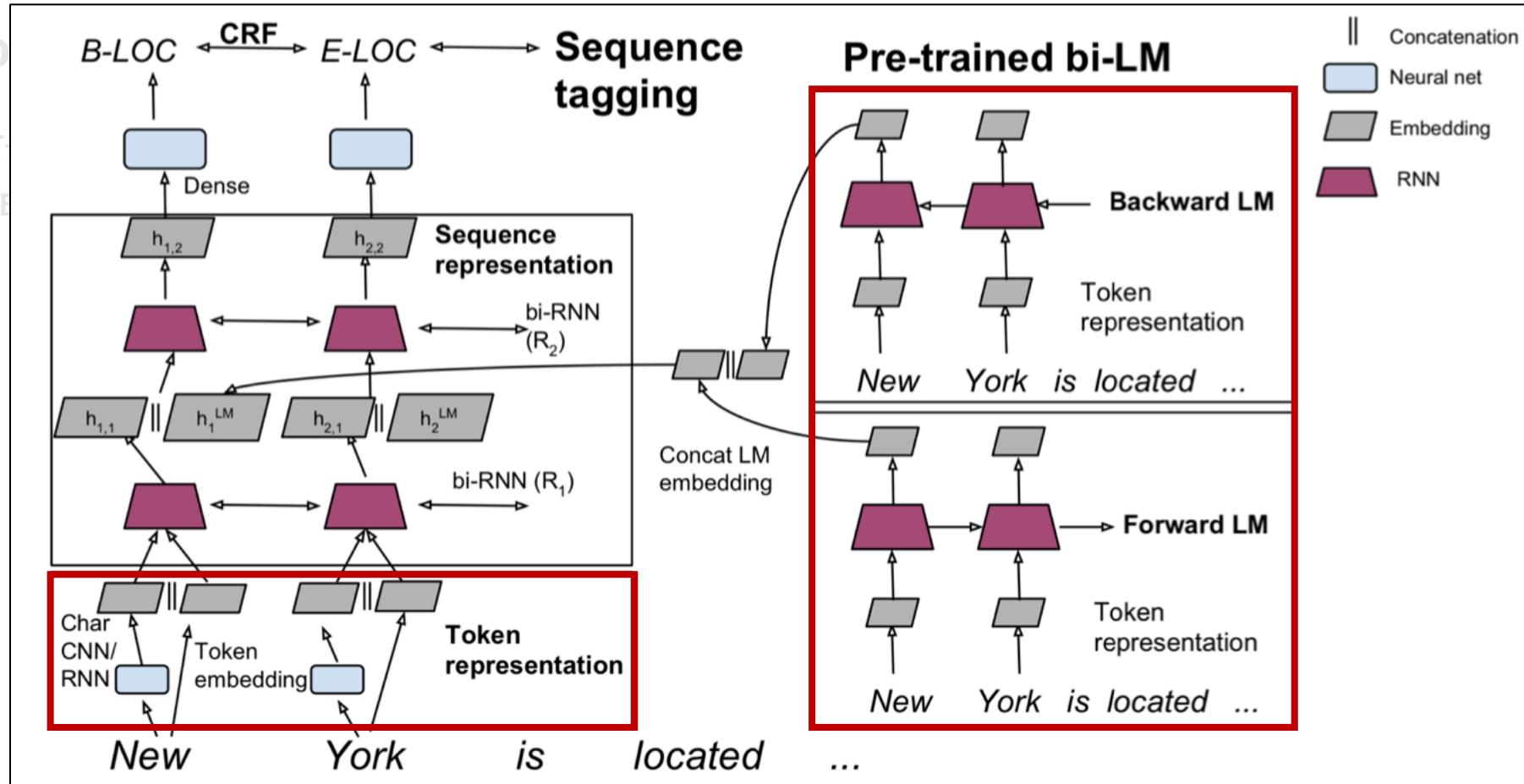
[STEP 3]

- Not only use the word embedding which is context independent, but can use our trained recurrent language model
- context와 상관없는 word embedding과 보유하고 있는 데이터의 context를 고려한 word embedding을 모두 사용할 수 있음

Unit 02 | Pre-ELMo and ELMO

Pre-ELMo

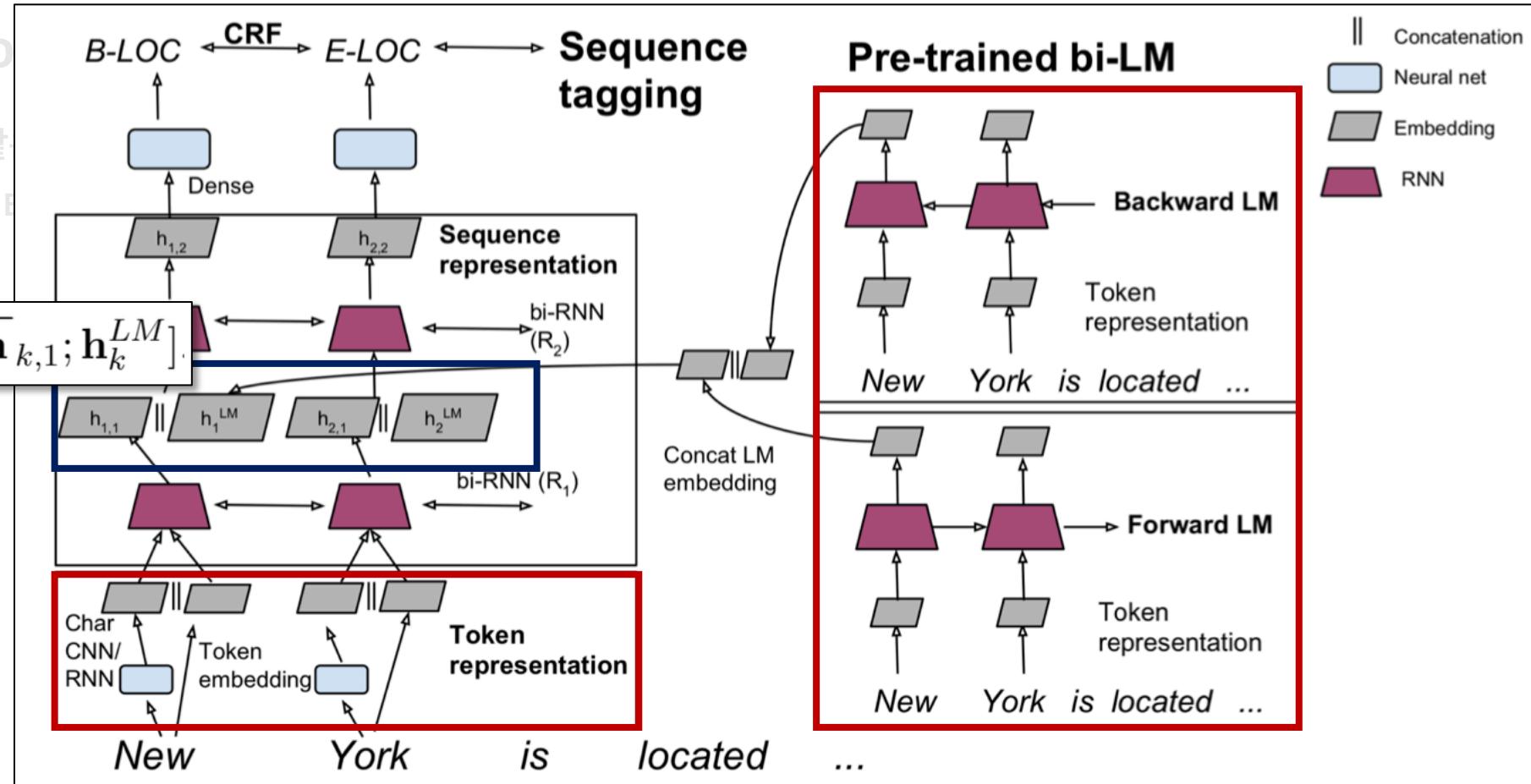
- NER과 같은
- 많은 데일리



Unit 02 | Pre-ELMo and ELMO

Pre-ELMo

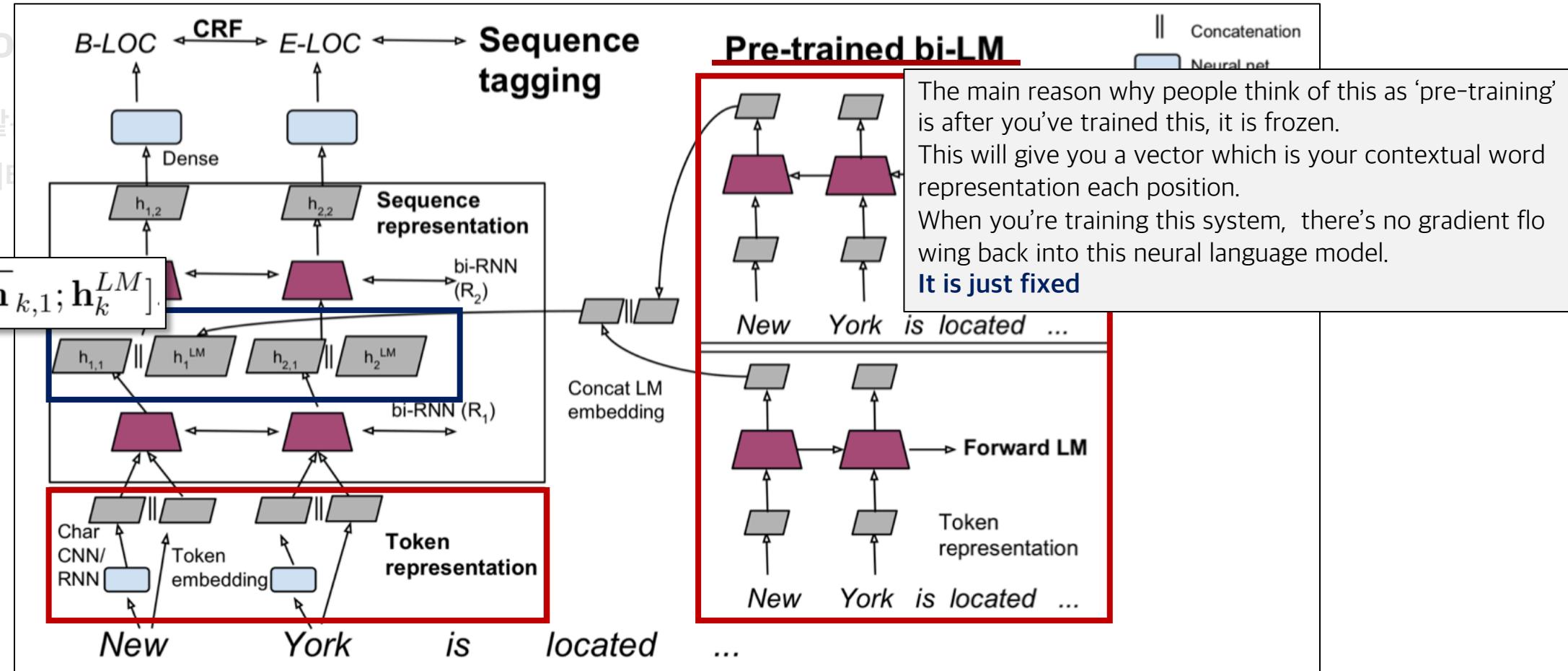
- NER과 같은
- 많은 데이터



Unit 02 | Pre-ELMo and ELMO

Pre-ELMo

- NER과 같은
- 많은 데이터



Unit 02 | Pre-ELMo and ELMO

ELMo - Peters et al. 2018, Deep contextualized word representations

- ELMo, Embeddings from Language Models 는 문장 전체를 사용하여 contextual word vector를 학습하는 모델
- window를 지정함으로써 주변 context만을 고려한 기존 모델과는 차이를 보임
- 전체적인 구조는 pre-ELMo와 유사함

Unit 02 | Pre-ELMo and ELMO

ELMo - Peters et al. 2018, Deep contextualized word representations

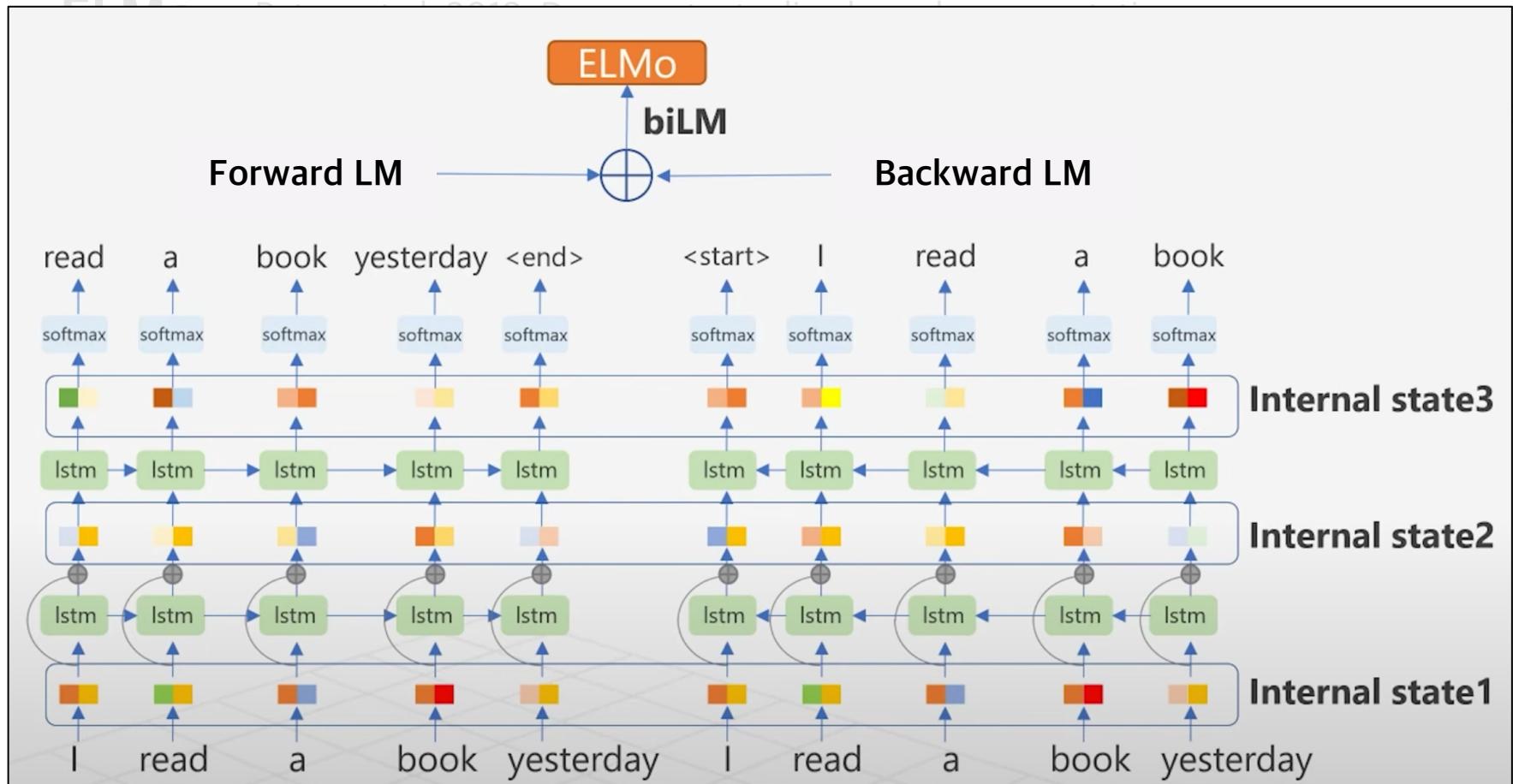
- ELMo, Embeddings from Language Models 는 문장 전체를 사용하여 contextual word vector를 학습하는 모델
- window를 지정함으로써 주변 context만을 고려한 기존 모델과는 차이를 보임
- 전체적인 구조는 pre-ELMo와 유사함

ELMo('Here is your birthday **present**') [2,5] [7,3] [8,3] [9,4] [1,6]



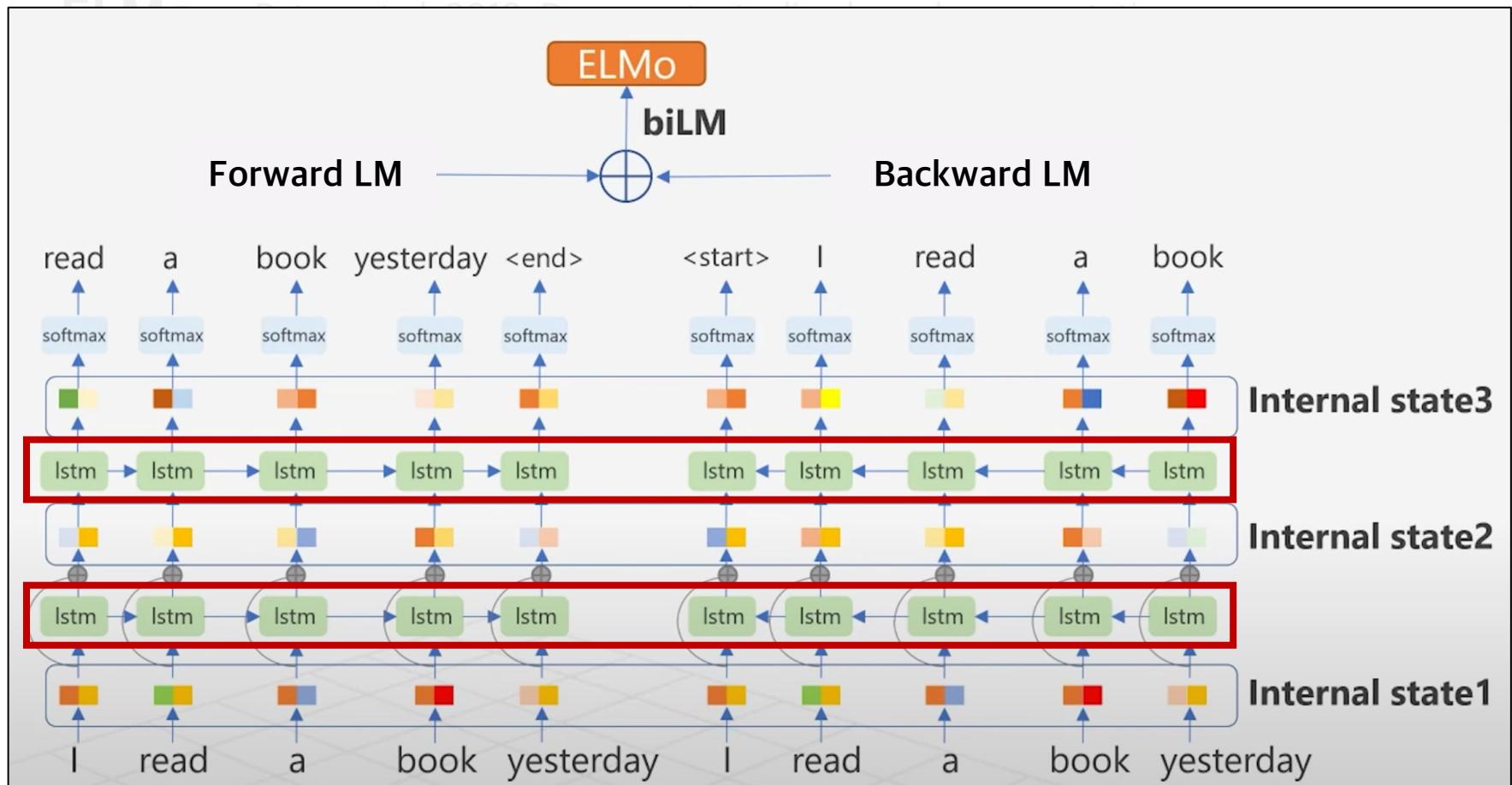
ELMo('Live in **present** not past') [1,8] [6,4] [2,3] [0,9] [4,7]

Unit 02 | Pre-ELMo and ELMO



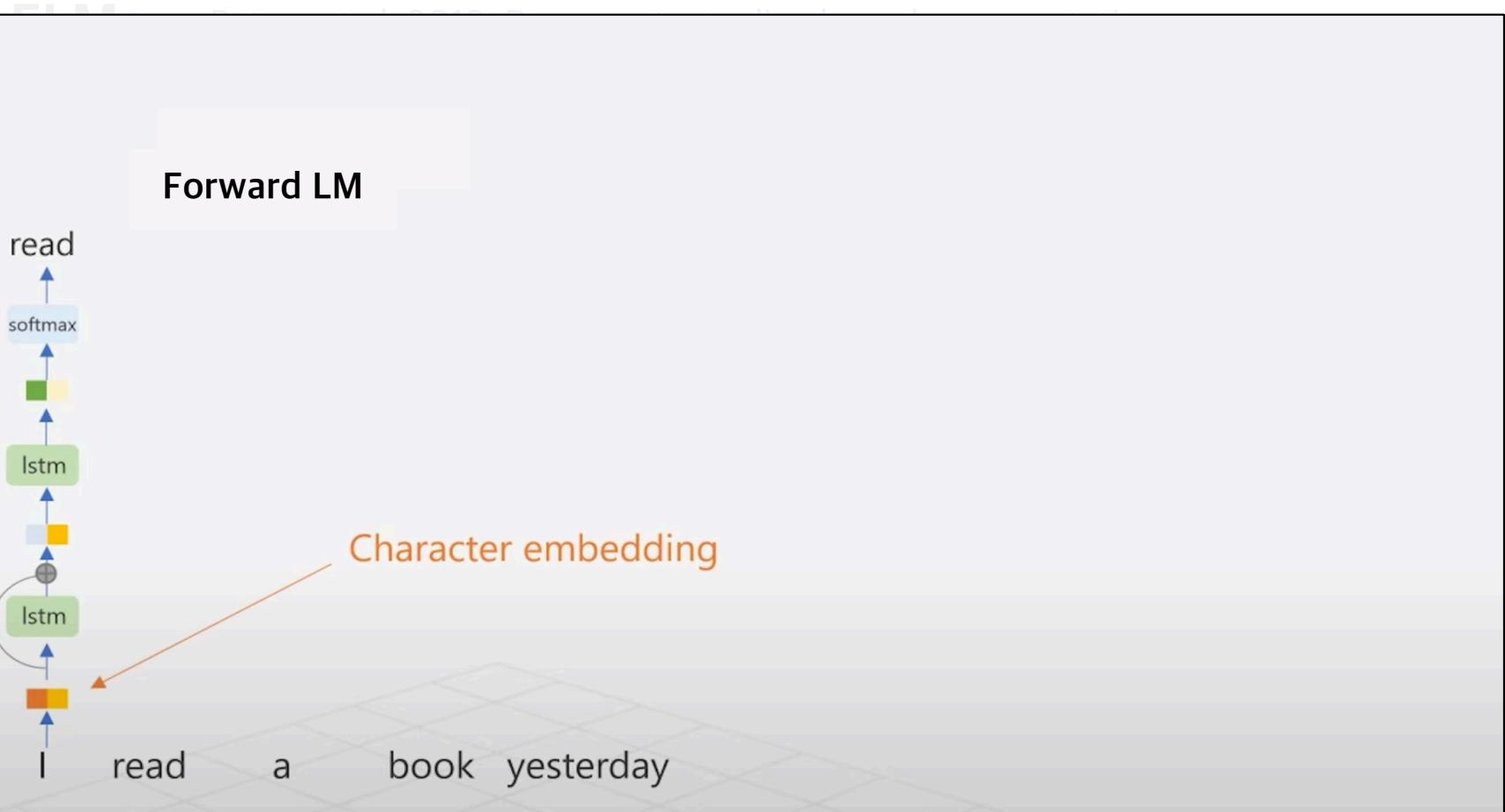
ектор을 학습하는 모델

Unit 02 | Pre-ELMo and ELMO



- 2개의 bi-LSTM layers를 사용

Unit 02 | Pre-ELMo and ELMO



- 2개의 bi-LSTM layers를 사용
- character CNN을 통한 word embedding만을 사용해 1st layer로 입력

Unit 02 | Pre-ELMo and ELMO

Forward LM

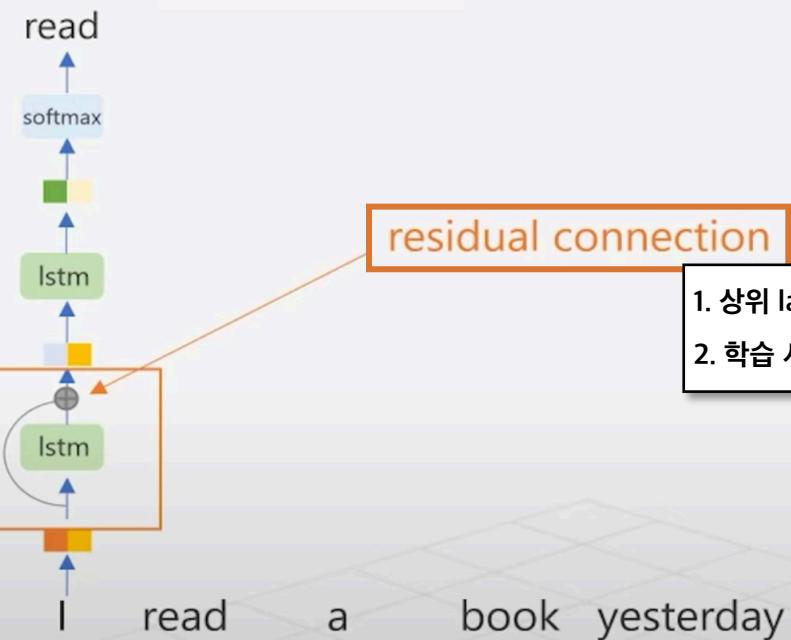


- 1. 최초 embedding은 context에 영향을 받지 않아야 함
- 2. pre-trained word embedding(word2vec, glove)와 비교를 위해 사용하지 않음

- 2개의 bi-LSTM layers를 사용
- character CNN을 통한 word embedding만을 사용해 1st layer로 입력

Unit 02 | Pre-ELMo and ELMO

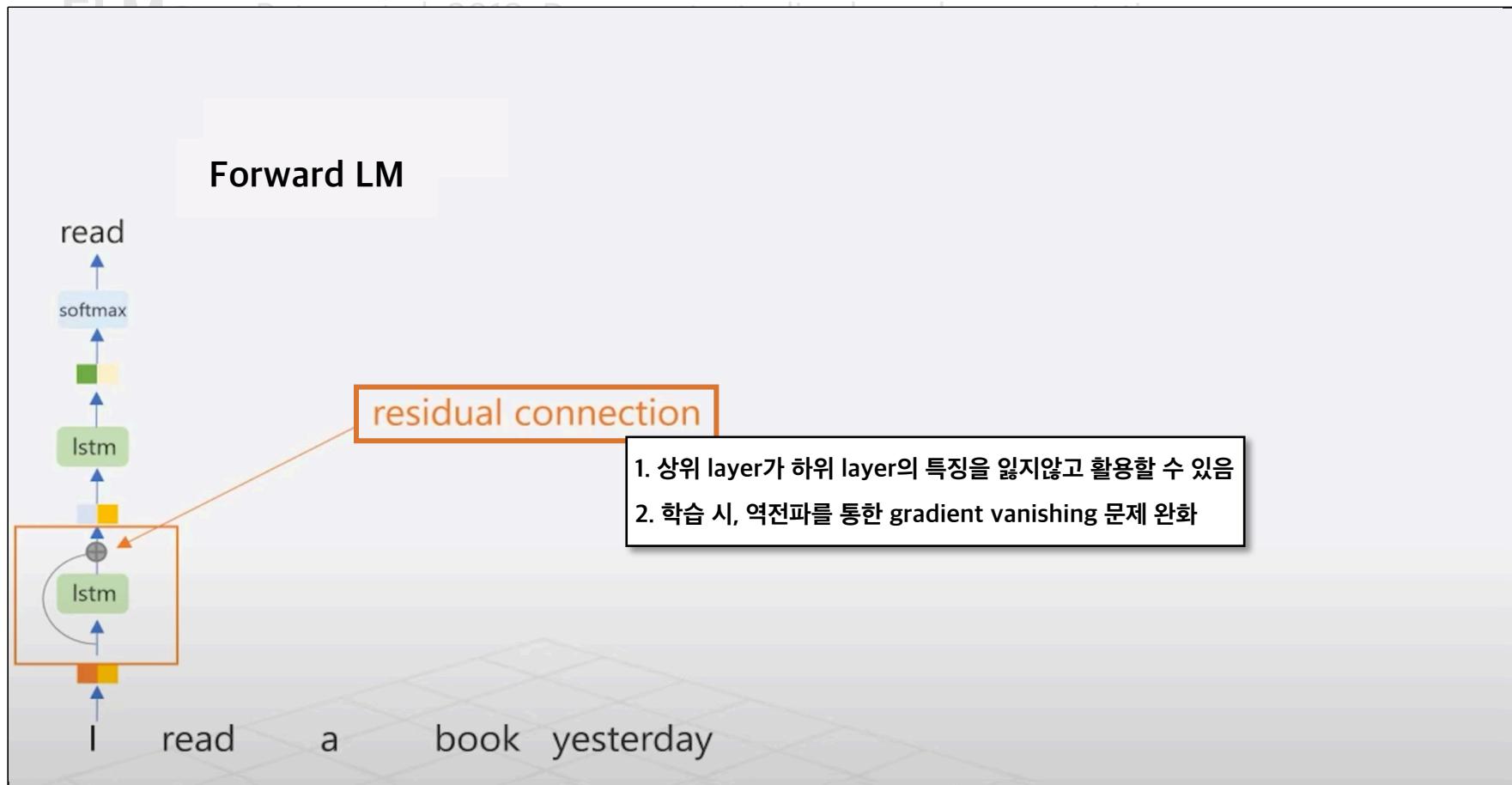
Forward LM



- 1. 상위 layer가 하위 layer의 특징을 잃지 않고 활용할 수 있음
- 2. 학습 시, 역전파를 통한 gradient vanishing 문제 완화

- 2개의 bi-LSTM layers를 사용
- character CNN을 통한 word embedding 만을 사용해 1st layer로 입력
- **residual connection을 사용**

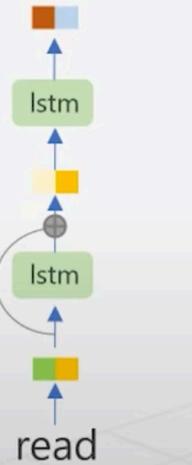
Unit 02 | Pre-ELMo and ELMO



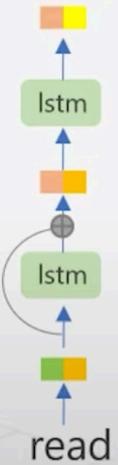
- 2개의 bi-LSTM layers를 사용
- character CNN을 통한 word embedding 만을 사용해 1st layer로 입력
- residual connection을 사용
- Character CNN과 LSTM에 projection을 적용

Unit 02 | Pre-ELMo and ELMO

Forward LM

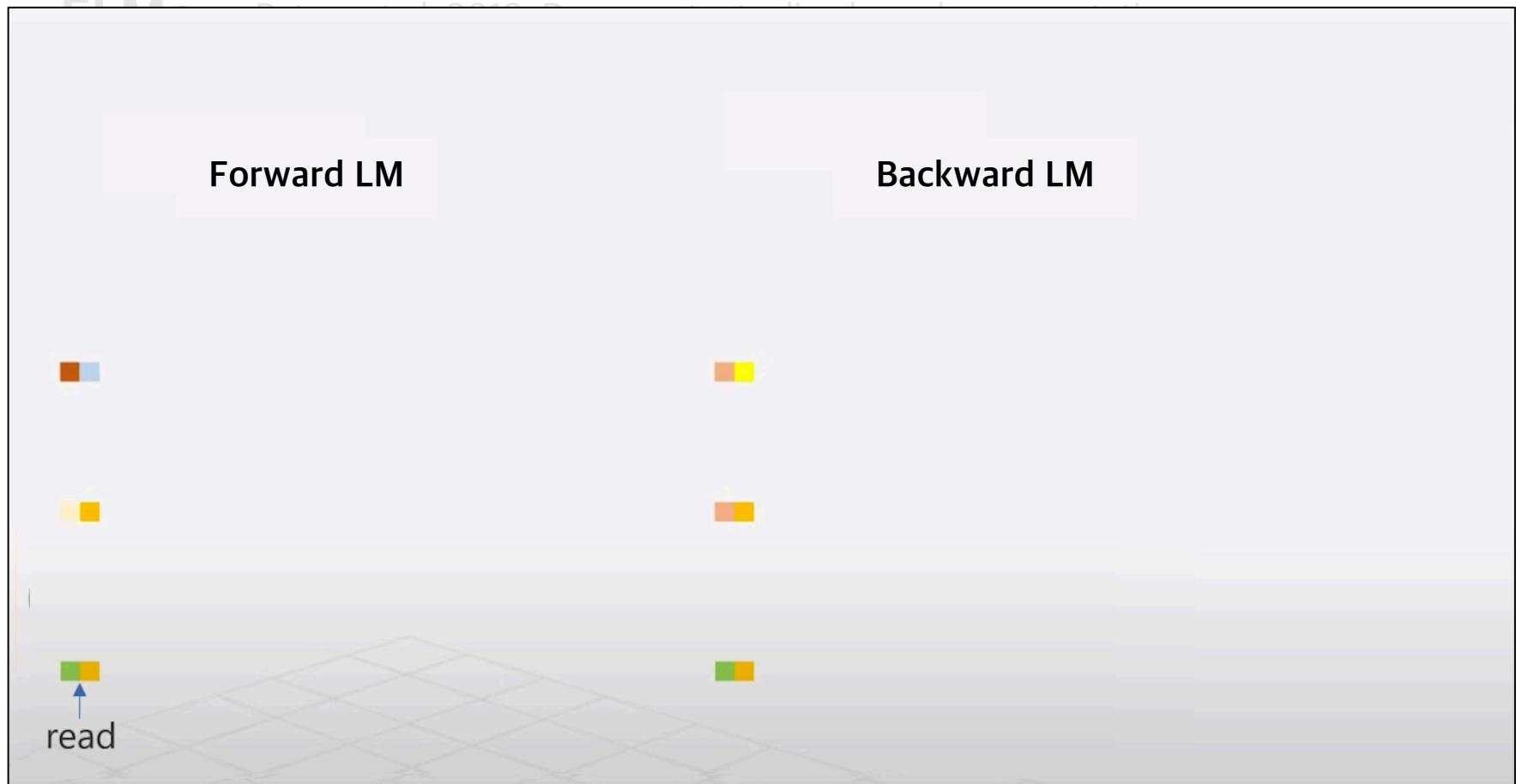


Backward LM



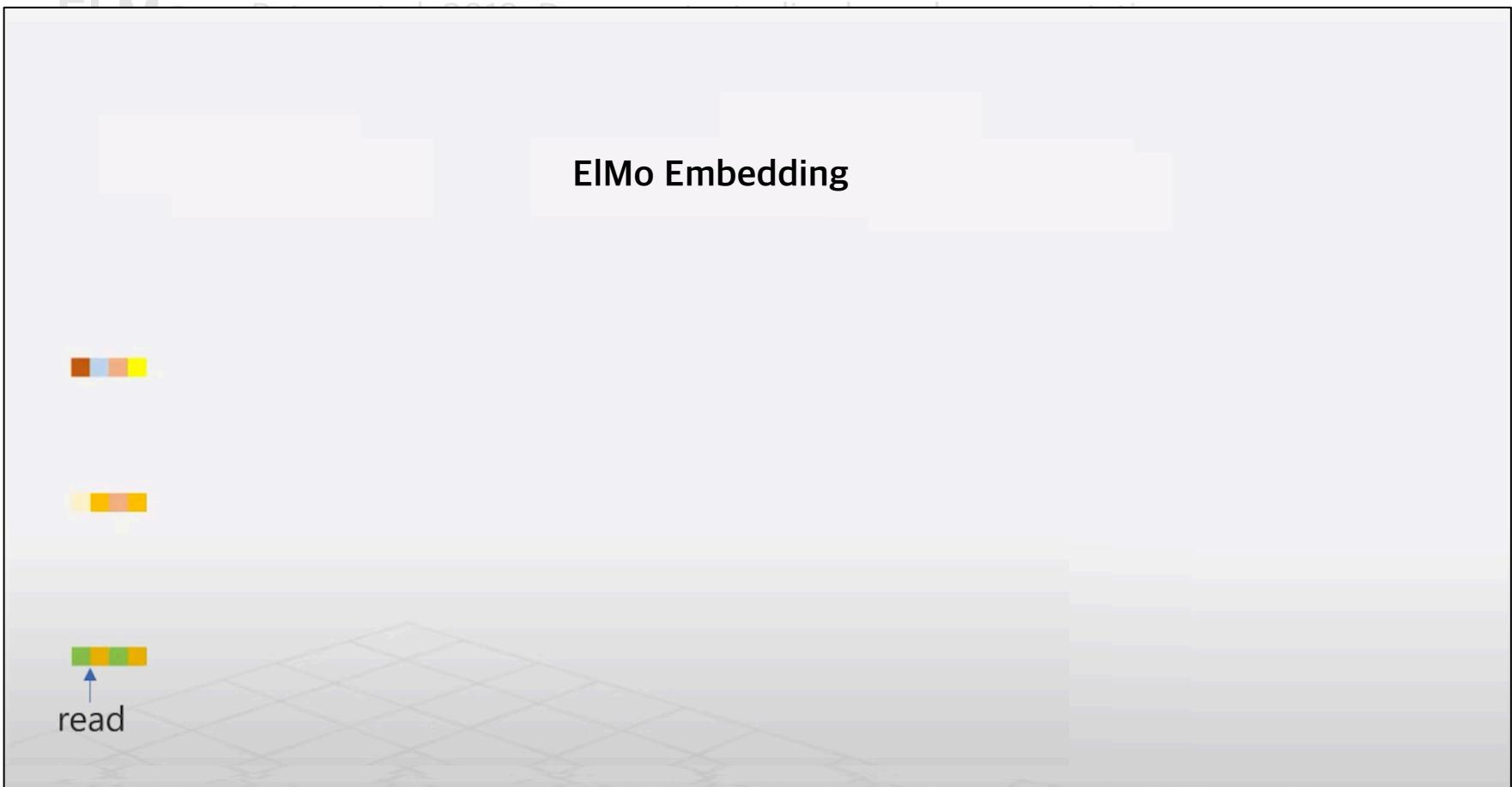
vector를 학습하는 모델

Unit 02 | Pre-ELMo and ELMO



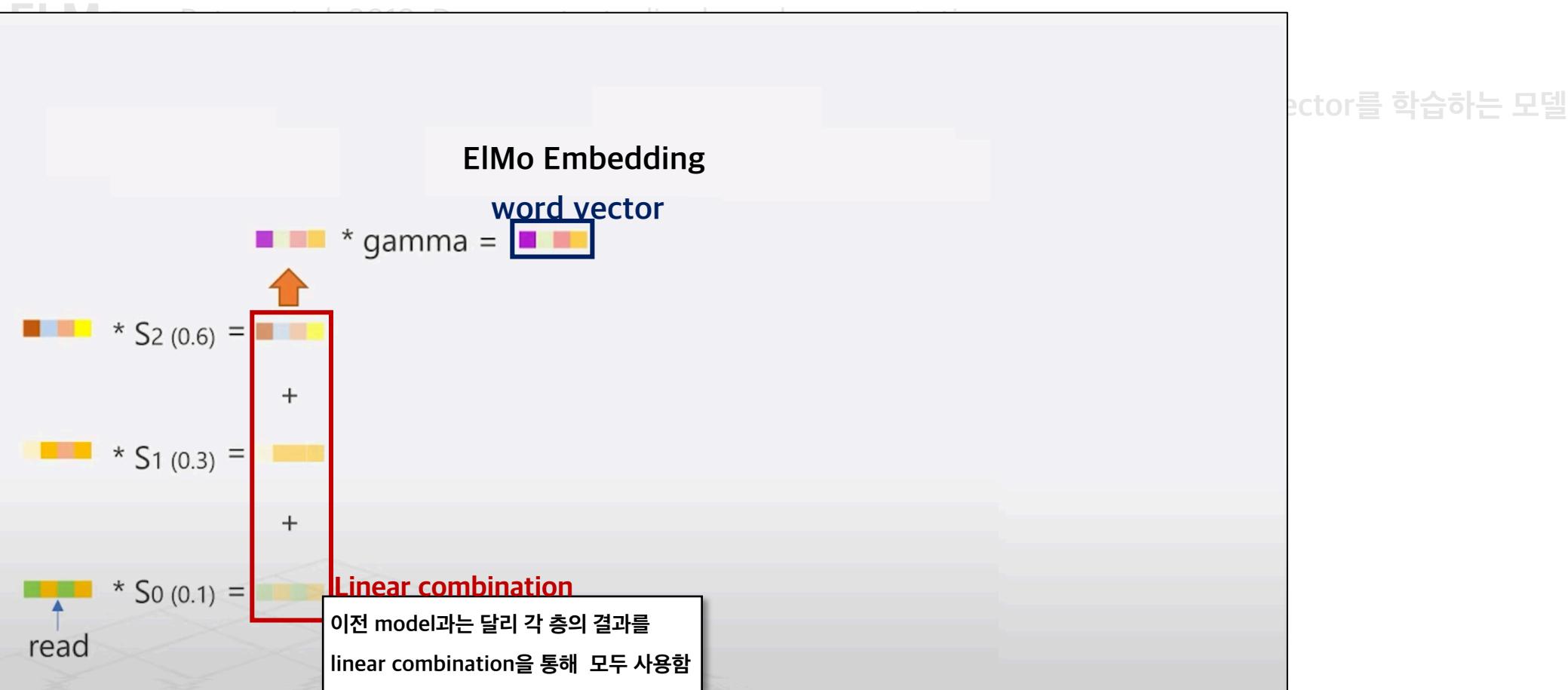
vector를 학습하는 모델

Unit 02 | Pre-ELMo and ELMO

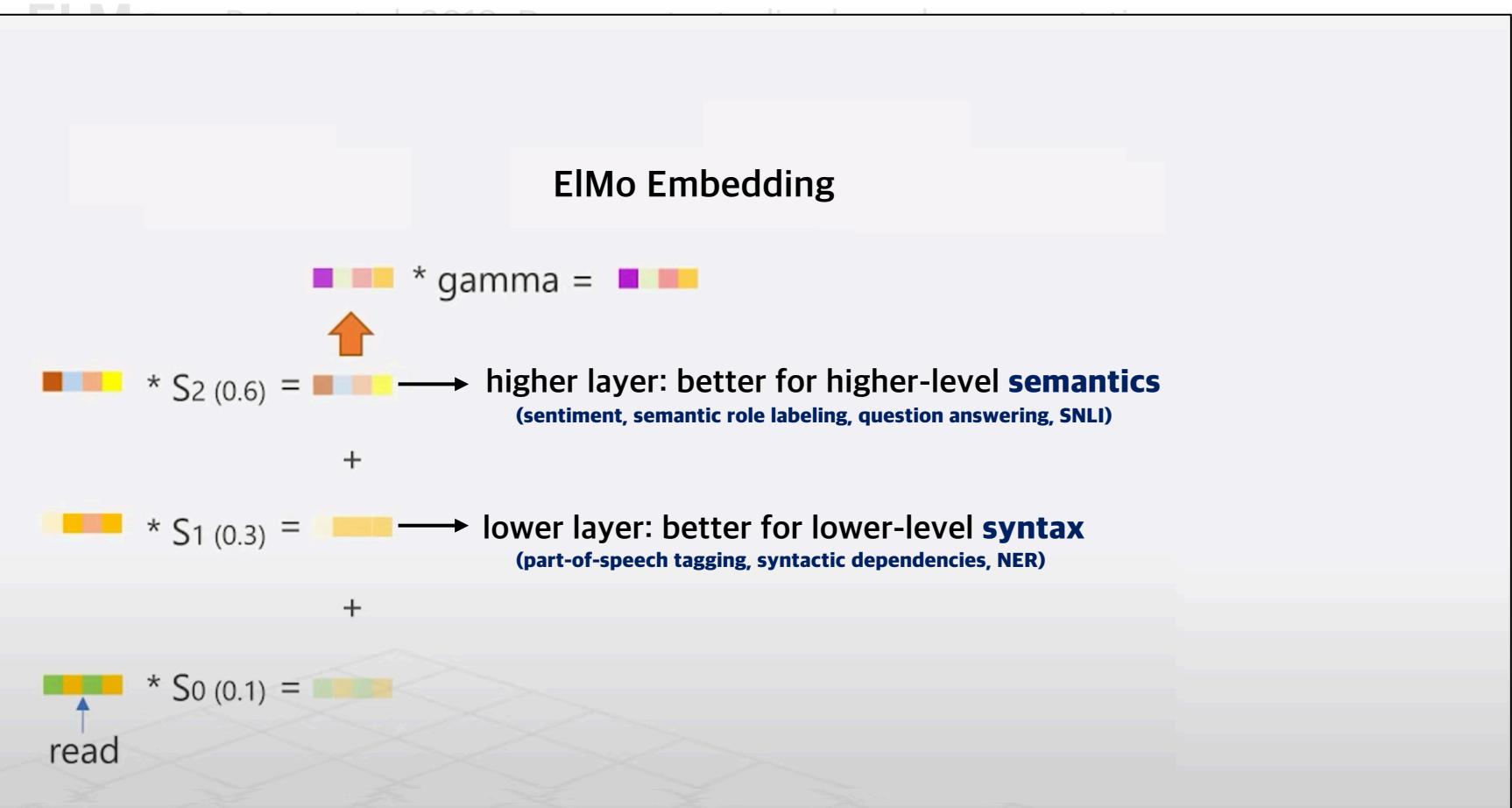


vector를 학습하는 모델

Unit 02 | Pre-ELMo and ELMO



Unit 02 | Pre-ELMo and ELMO



vector를 학습하는 모델

Unit 02 | Pre-ELMo and ELMo

ELMo - Peters et al. 2018, Deep contextualized word representations

- pre-ELMo와 비교하여 NER task에서 0.3%의 성능 향상을 보임
- NER 뿐만 아니라 NLP 대부분의 task에서 가장 높은 성능을 보임

CoNLL 2003 Named Entity Recognition (en news testb)			
Name	Description	Year	F1
ELMo	ELMo in BiLSTM	2018	92.22
TagLM Peters	LSTM BiLM in BiLSTM tagger	2017	91.93
Ma + Hovy	BiLSTM + char CNN + CRF layer	2016	91.21
Tagger Peters	BiLSTM + char CNN + CRF layer	2017	90.87
Ratinov + Roth	Categorical CRF+Wikidata+word cls	2009	90.80
Finkel et al.	Categorical feature CRF	2005	86.86
IBM Florian	Linear/softmax/TBL/HMM ensemble, gazetteer++	2003	88.76
Stanford	MEMM softmax markov model	2003	86.07

TASK	PREVIOUS SOTA	OUR BASELINE	ELMO + BASELINE	INCREASE (ABSOLUTE/RELATIVE)
SQuAD	Liu et al. (2017)	84.4	81.1	85.8
SNLI	Chen et al. (2017)	88.6	88.0	88.7 ± 0.17
SRL	He et al. (2017)	81.7	81.4	84.6
Coref	Lee et al. (2017)	67.2	67.2	70.4
NER	Peters et al. (2017)	91.93 ± 0.19	90.15	92.22 ± 0.10
SST-5	McCann et al. (2017)	53.7	51.4	54.7 ± 0.5

Unit 03 | ULMfit and onward

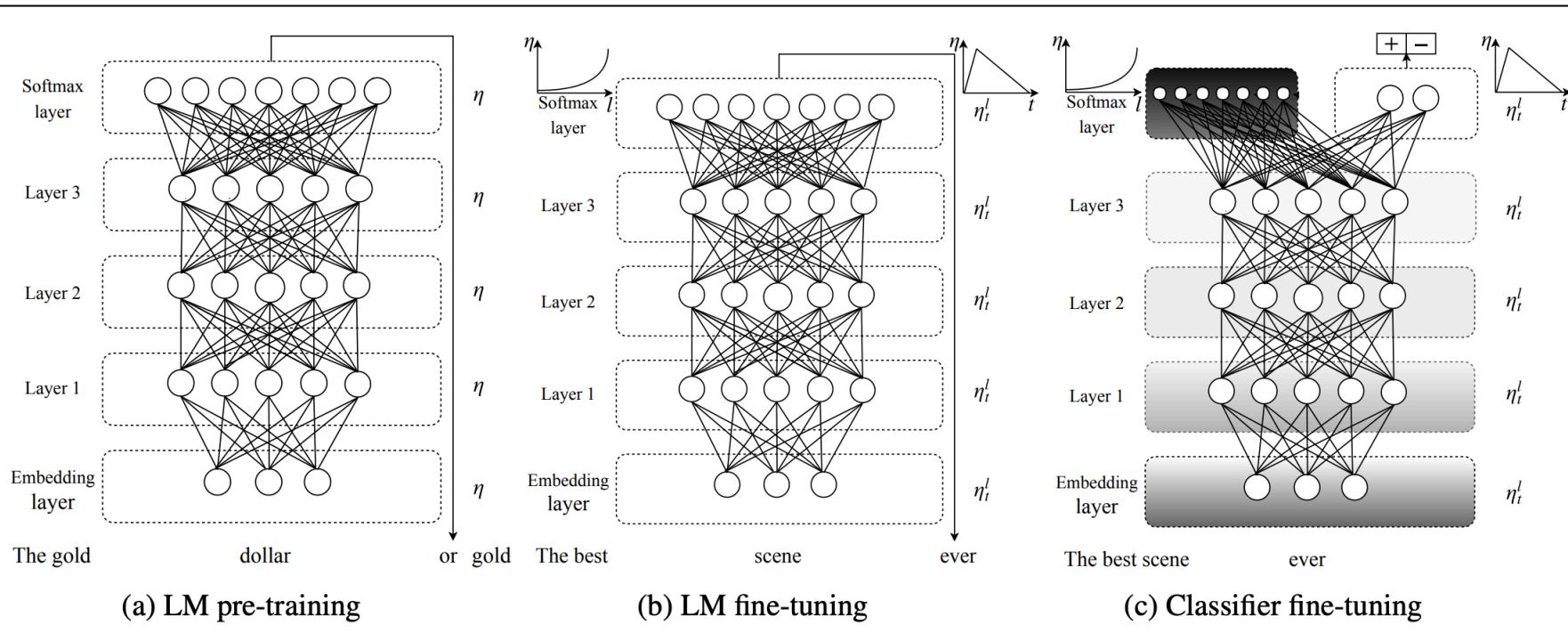
Unit 03 | ULMfit and onward

ULMfit - Howard and Ruder, 2018, Universal Language Model Fine-tuning for Text Classification

- Language model을 사용하여 classifier을 생성하는 모델
- NLP에서 본격적으로 transfer learning을 도입하게 됨
- 1개의 GPU로 학습할 수 있는 정도의 사이즈

Unit 03 | ULMfit and onward

ULMfit - Howard and Ruder, 2018, Universal Language Model Fine-tuning for Text Classification



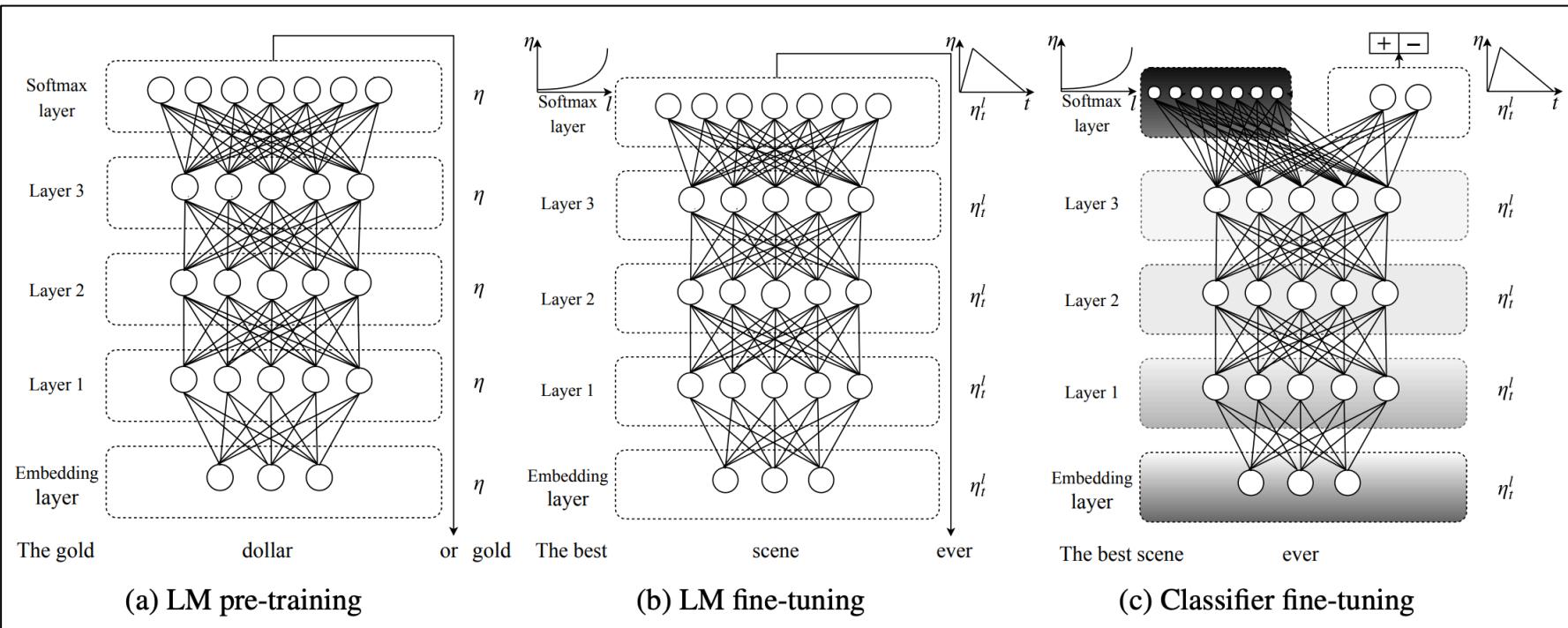
(a) 일반 언어 모델 학습
General-domain Language Model Pre-training

(b) 과제 맞춤형 언어 모델 투닝
Target task Language Model fine-tuning

(c) 과제 분류기 투닝
Target task Classifier fine-tuning

Unit 03 | ULMfit and onward

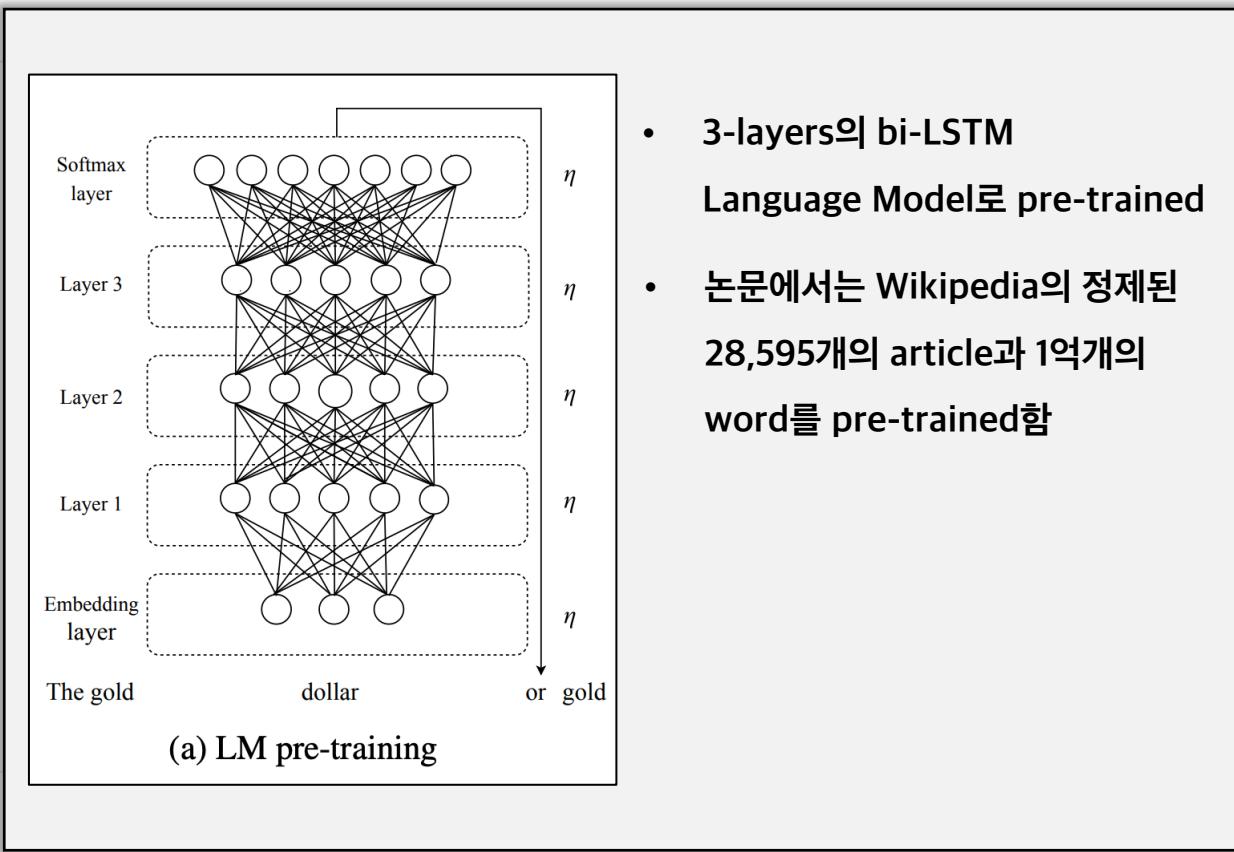
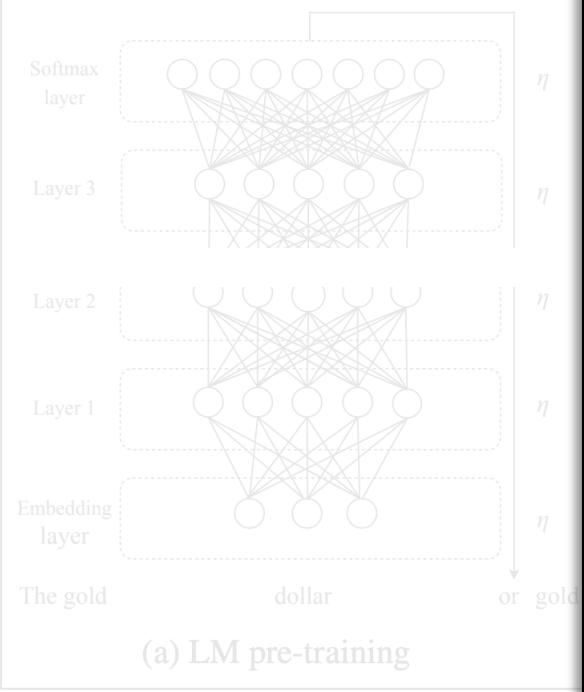
ULMfit - Howard and Ruder, 2018, Universal Language Model Fine-tuning for Text Classification



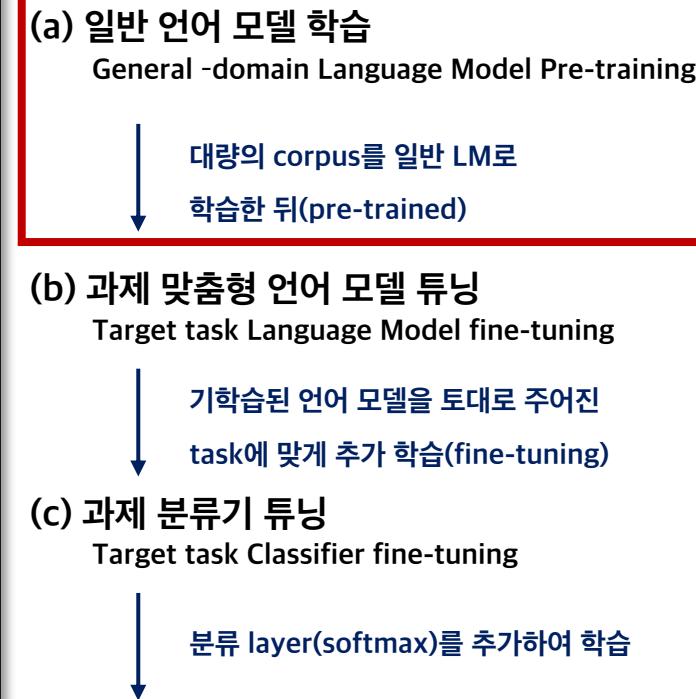
- (a) 일반 언어 모델 학습**
General-domain Language Model Pre-training
- 대량의 corpus를 일반 LM로 학습한 뒤(pre-trained)
- (b) 과제 맞춤형 언어 모델 투닝**
Target task Language Model fine-tuning
- 기학습된 언어 모델을 토대로 주어진 task에 맞게 추가 학습(fine-tuning)
- (c) 과제 분류기 투닝**
Target task Classifier fine-tuning
- 분류 layer(softmax)를 추가하여 학습

Unit 03 | ULMfit and onward

ULMfit - Howard and Ruder, 2018, Universal Language Model Fine-tuning for Text Classification



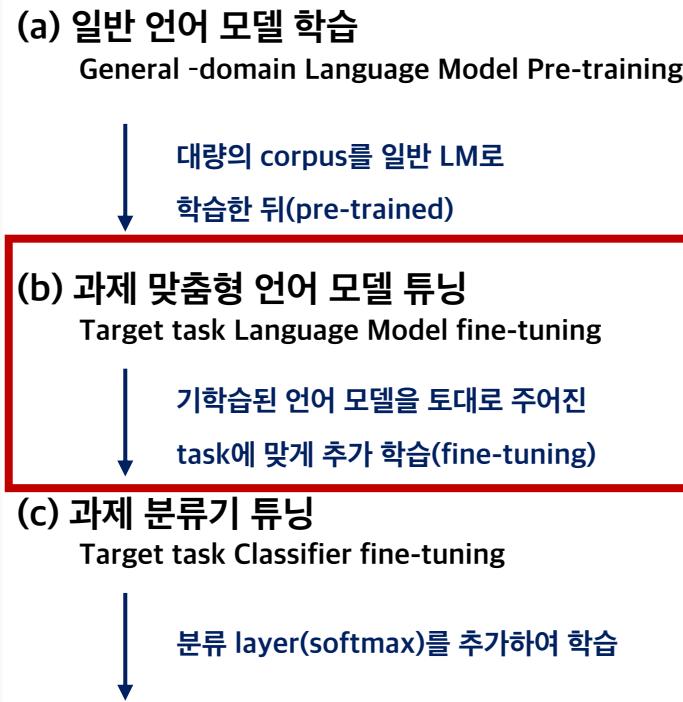
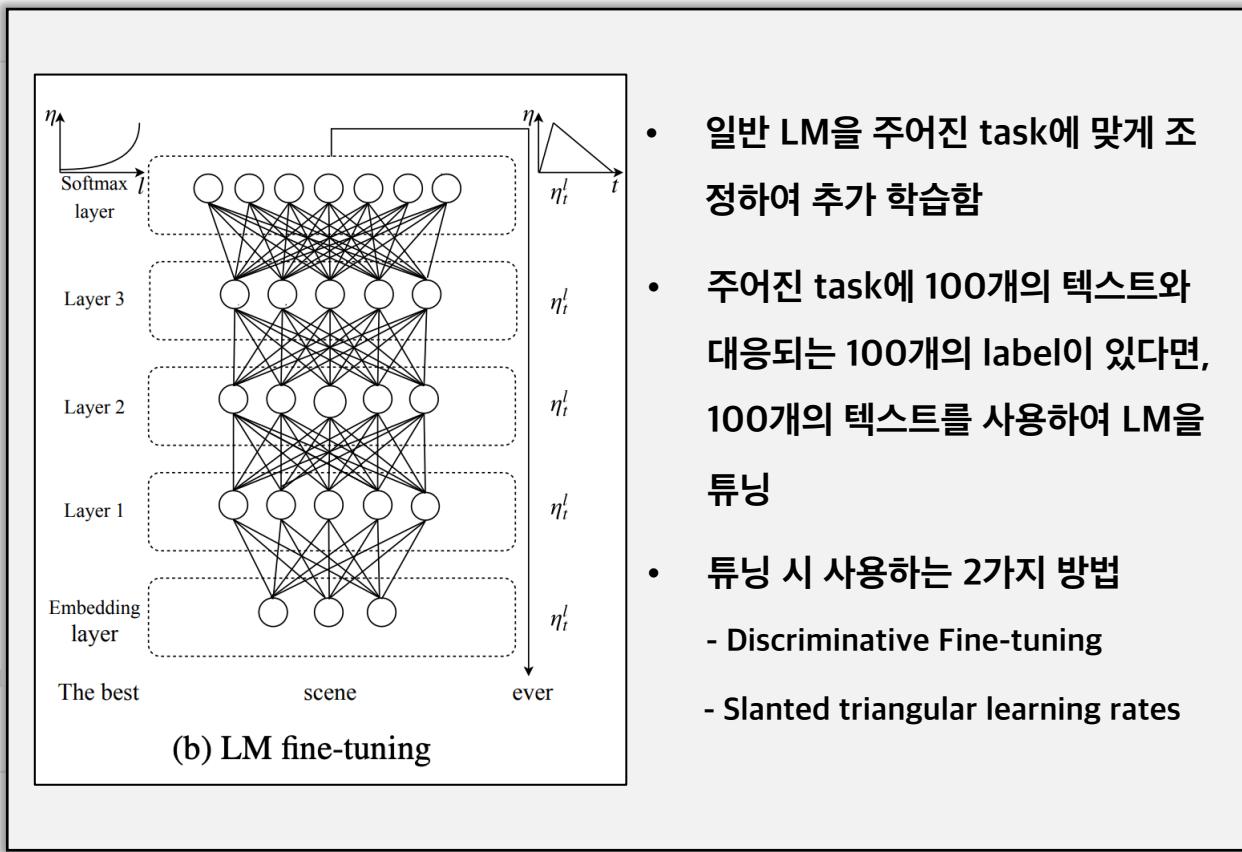
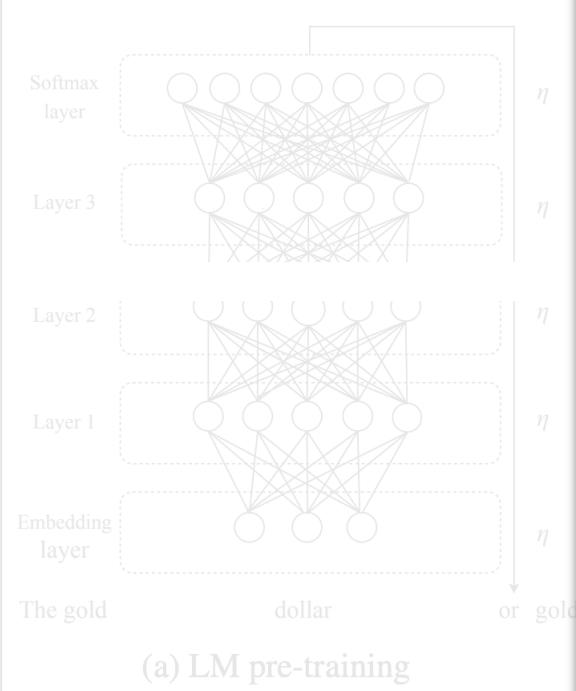
- 3-layers의 bi-LSTM Language Model로 pre-trained
- 논문에서는 Wikipedia의 정제된 28,595개의 article과 1억개의 word를 pre-trained함



Unit 03 | ULMfit and onward

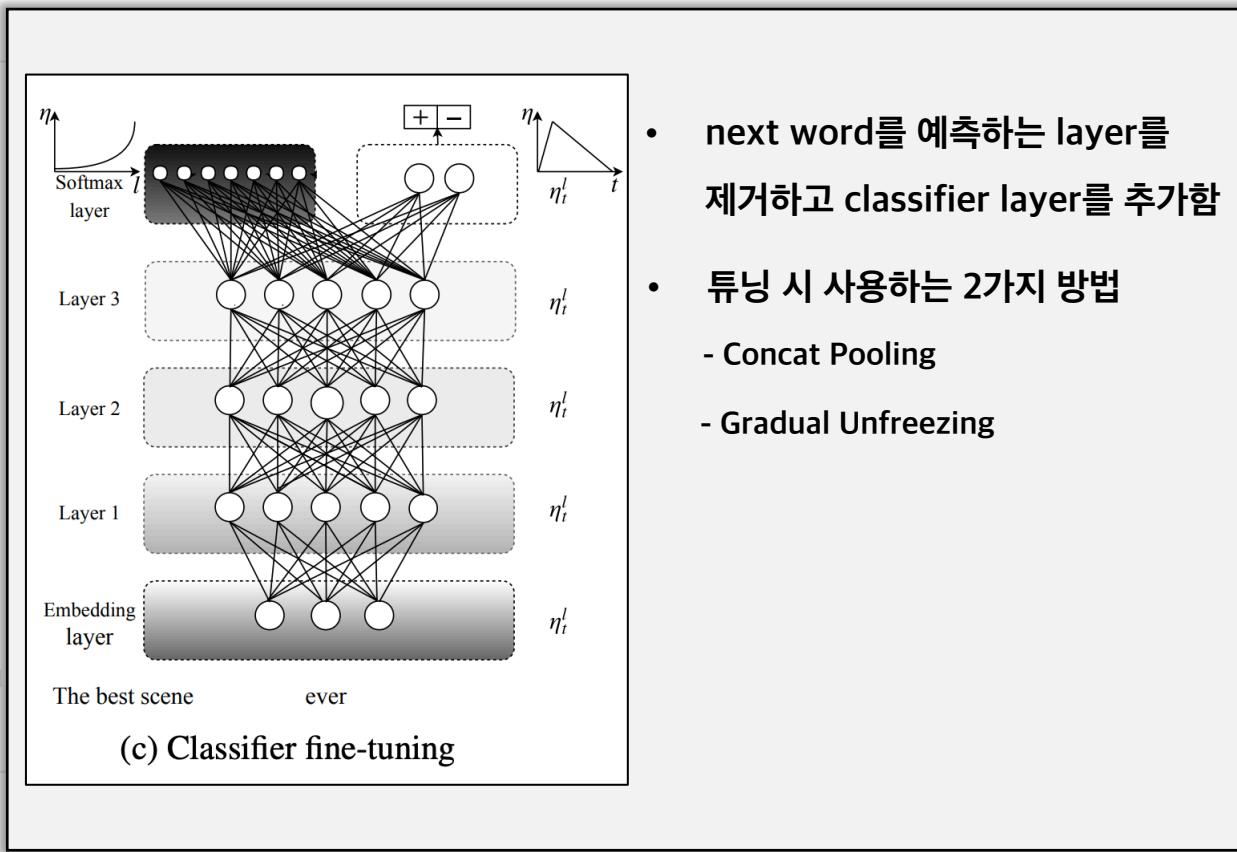
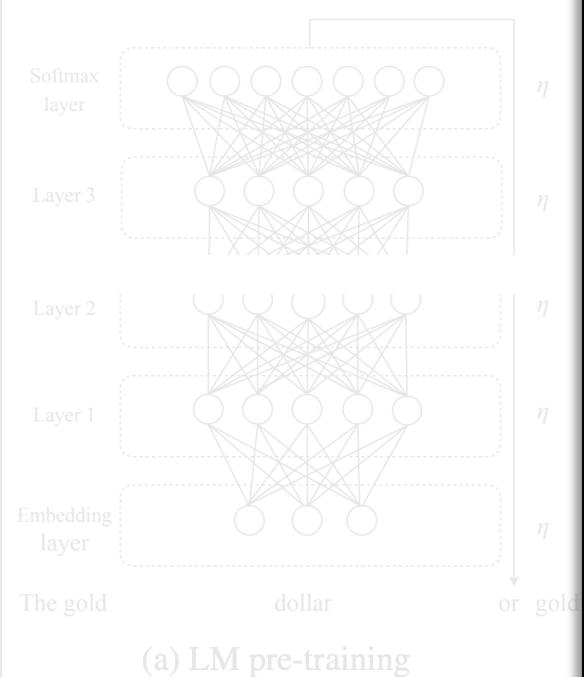
ULMfit

- Howard and Ruder, 2018, Universal Language Model Fine-tuning for Text Classification

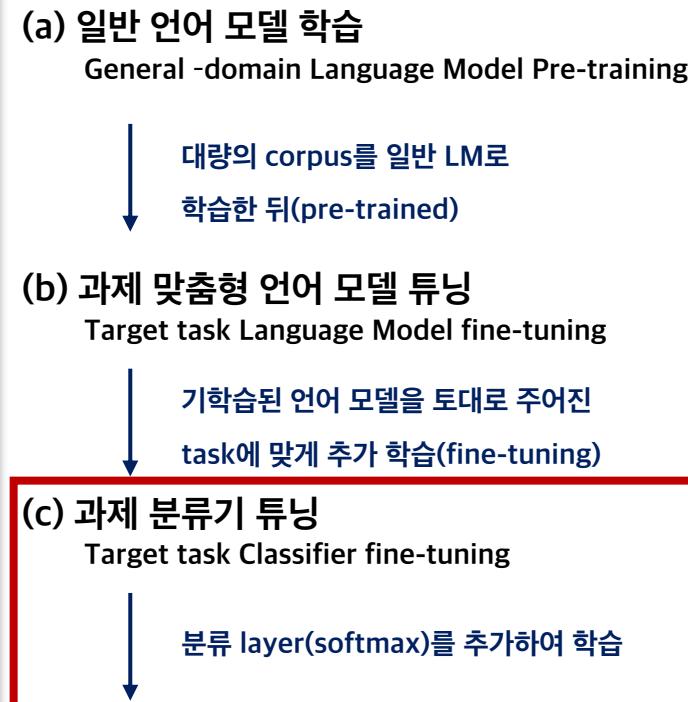


Unit 03 | ULMfit and onward

ULMfit - Howard and Ruder, 2018, Universal Language Model Fine-tuning for Text Classification



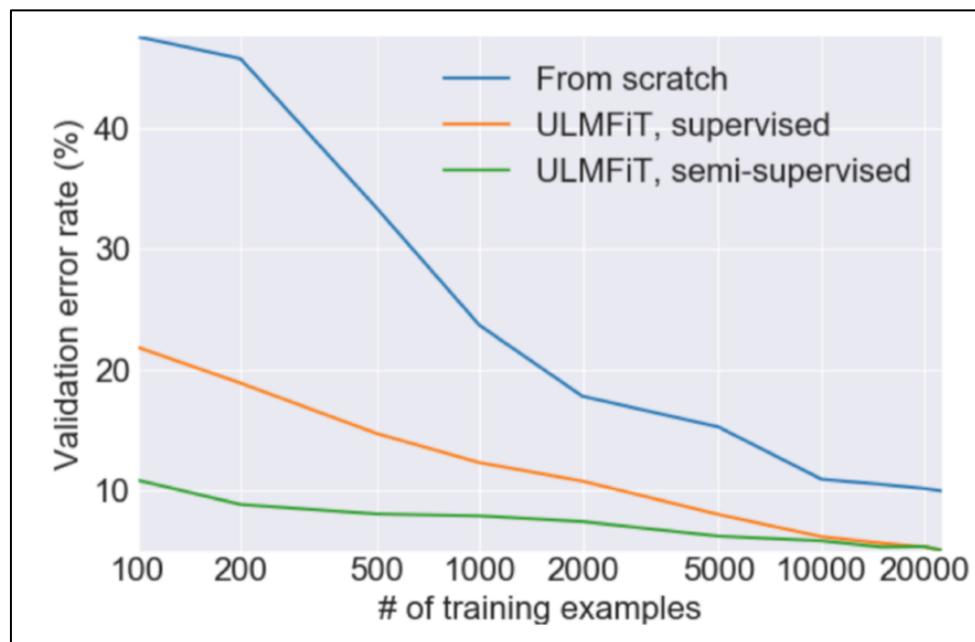
- next word를 예측하는 layer를 제거하고 classifier layer를 추가함
- 튜닝 시 사용하는 2가지 방법
 - Concat Pooling
 - Gradual Unfreezing



Unit 03 | ULMfit and onward

ULMfit - Howard and Ruder, 2018, Universal Language Model Fine-tuning for Text Classification

[ULMfit transfer learning]

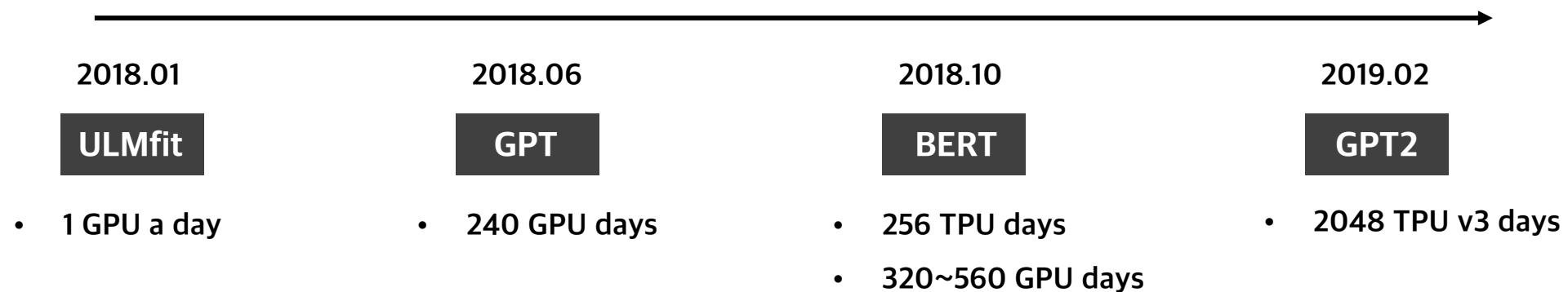


[Text Classifier error rates]

IMDb	Model	Test	IMDb	Model	Test
	CoVe (McCann et al., 2017)	8.2		CoVe (McCann et al., 2017)	4.2
	oh-LSTM (Johnson and Zhang, 2016)	5.9		TBCNN (Mou et al., 2015)	4.0
	Virtual (Miyato et al., 2016)	5.9		LSTM-CNN (Zhou et al., 2016)	3.9
	ULMFiT (ours)	4.6	TREC-6	ULMFiT (ours)	3.6

Unit 03 | ULMfit and onward

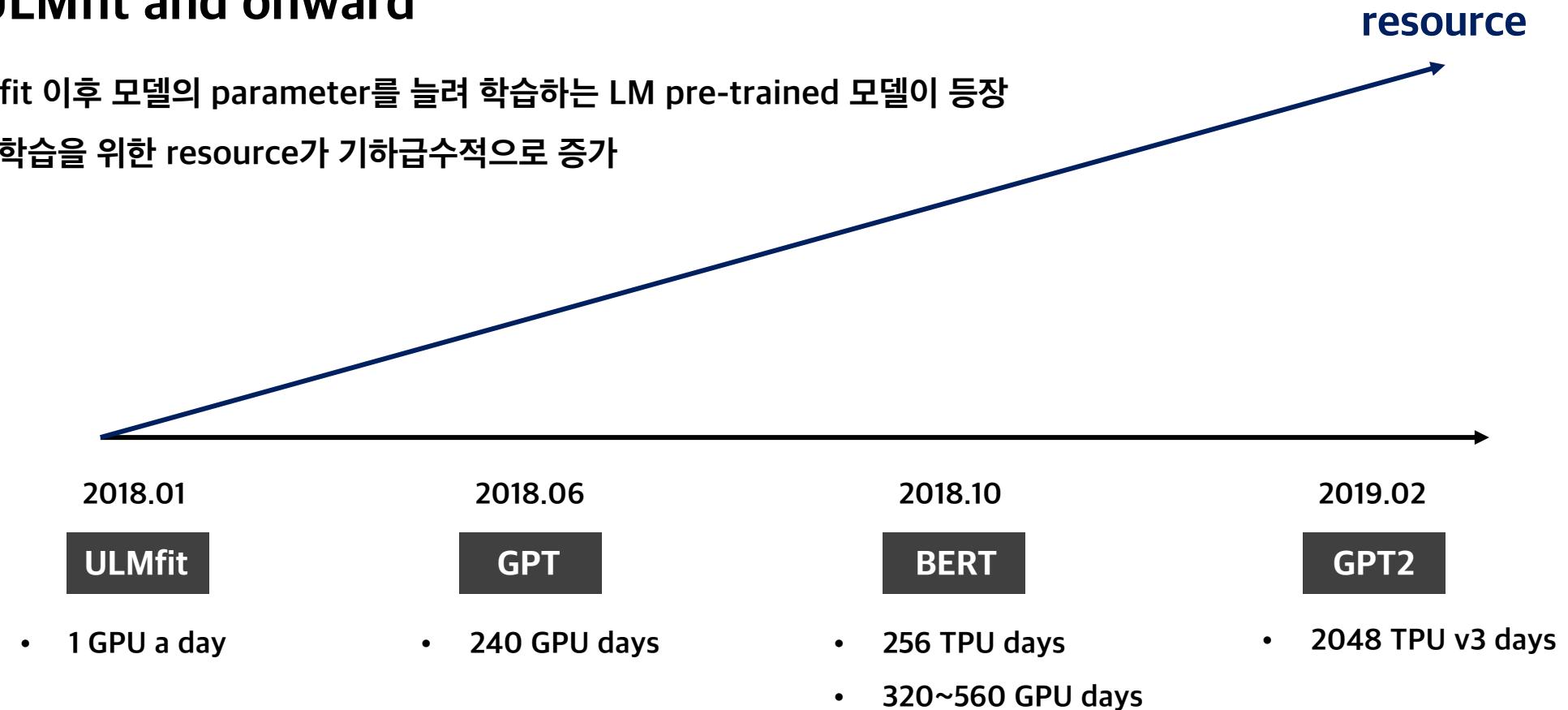
After ULMfit and onward



Unit 03 | ULMfit and onward

After ULMfit and onward

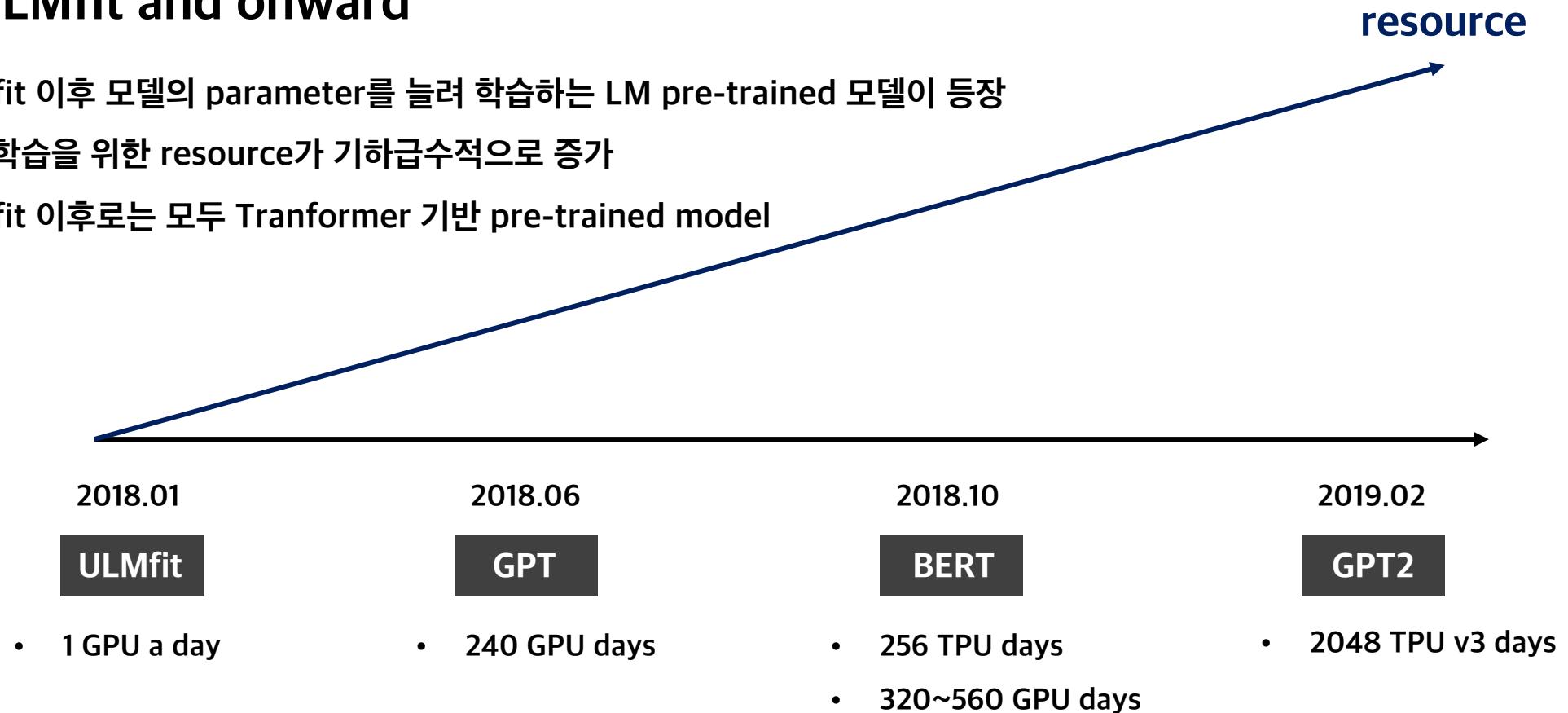
- ULMfit 이후 모델의 parameter를 늘려 학습하는 LM pre-trained 모델이 등장
- 모델 학습을 위한 resource가 기하급수적으로 증가



Unit 03 | ULMfit and onward

After ULMfit and onward

- ULMfit 이후 모델의 parameter를 늘려 학습하는 LM pre-trained 모델이 등장
- 모델 학습을 위한 resource가 기하급수적으로 증가
- ULMfit 이후로는 모두 Transformer 기반 pre-trained model



Unit 04 | Transformer architectures

Unit 04 | Transformer architectures

The Motivation for Transformers

- RNN 계열의 문제점
 - Sequential한 특징때문에 parallelization(병렬 계산)이 불가능
 - Long-term Dependency 문제 해결을 위한 LSTM, GRU 이상의 방법론이 필요함

Unit 04 | Transformer architectures

The Motivation for Transformers

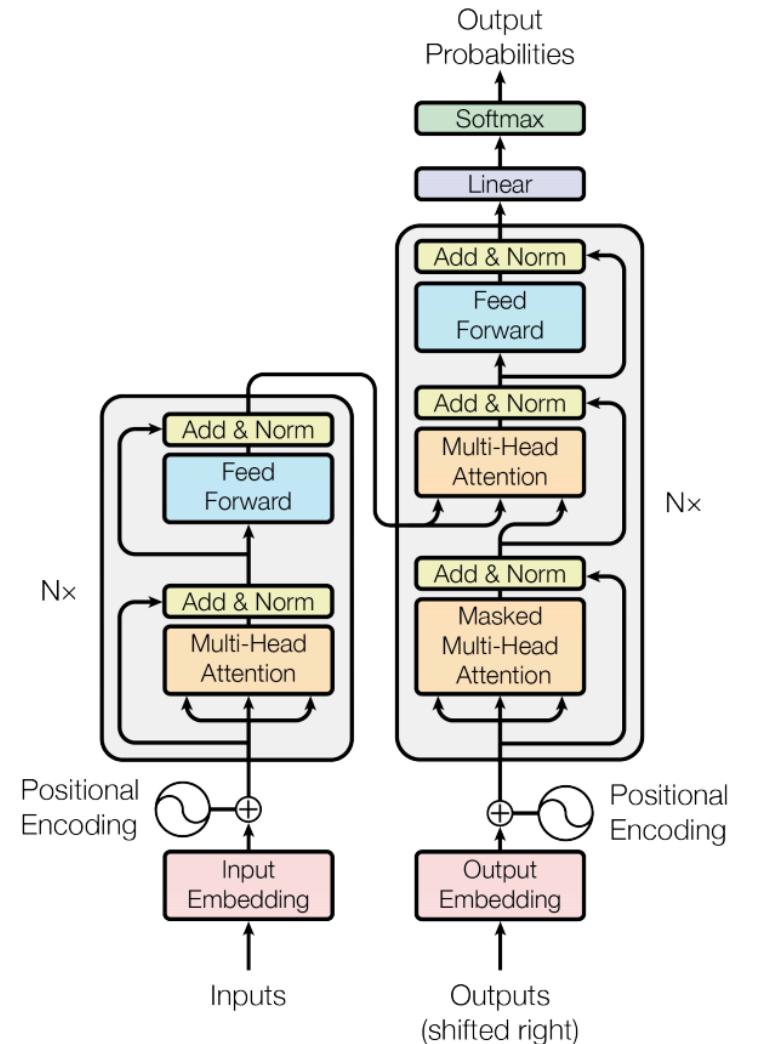
- RNN 계열의 문제점
 - Sequential한 특징때문에 parallelization(병렬 계산)이 불가능
 - Long-term Dependency 문제 해결을 위한 LSTM, GRU 이상의 방법론이 필요함

→ ‘Attention’으로 어떠한 state에도 접근 가능하게 된다면 RNN을 사용할 필요가 없지 않을까?

Unit 04 | Transformer architectures

Transformer Overview

- Non-recurrent seq2seq encoder-decoder model, RNN을 사용하지 않고
Attention 매커니즘이 적용된 seq2seq 모델
- 기계 번역을 위한 input0| encoder로, output0| decoder로 할당됨
- decoder의 특정 time-step의 output0| encoder의 모든 time-step의
output 중 어떤 time-step과 가장 연관성이 있는지에 대해 파악

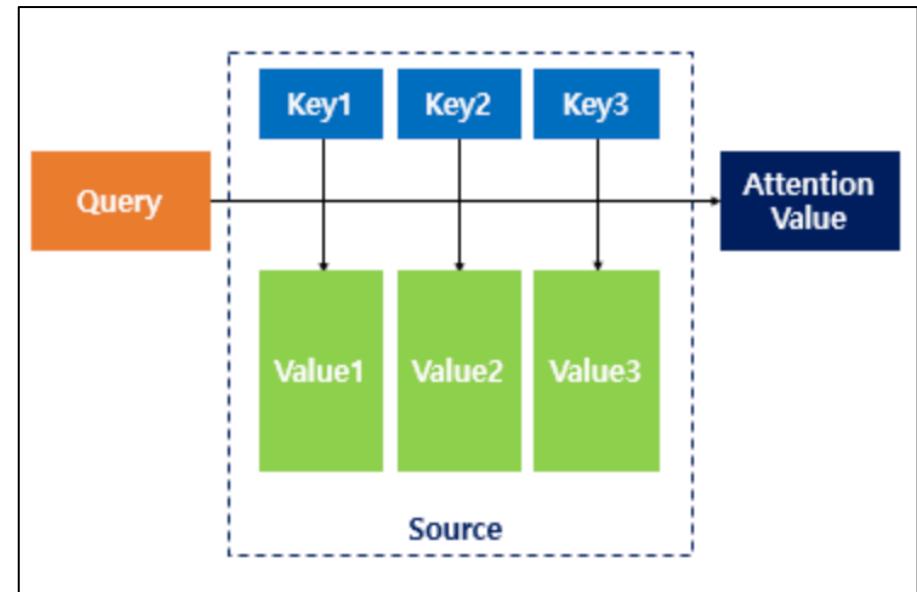


Unit 04 | Transformer architectures

Attention mechanism

- Attention의 구조에는 query, key, value가 존재함
- 주어진 query와 각 key간의 유사도를 구한 후, 유사도와 key를 맵핑하여 value를 매칭해줌
- output으로 key의 value를 weighted sum한 attention value가 반환됨

$$A(q, K, V) = \sum_i \frac{e^{q \cdot k_i}}{\sum_j e^{q \cdot k_j}} v_i$$



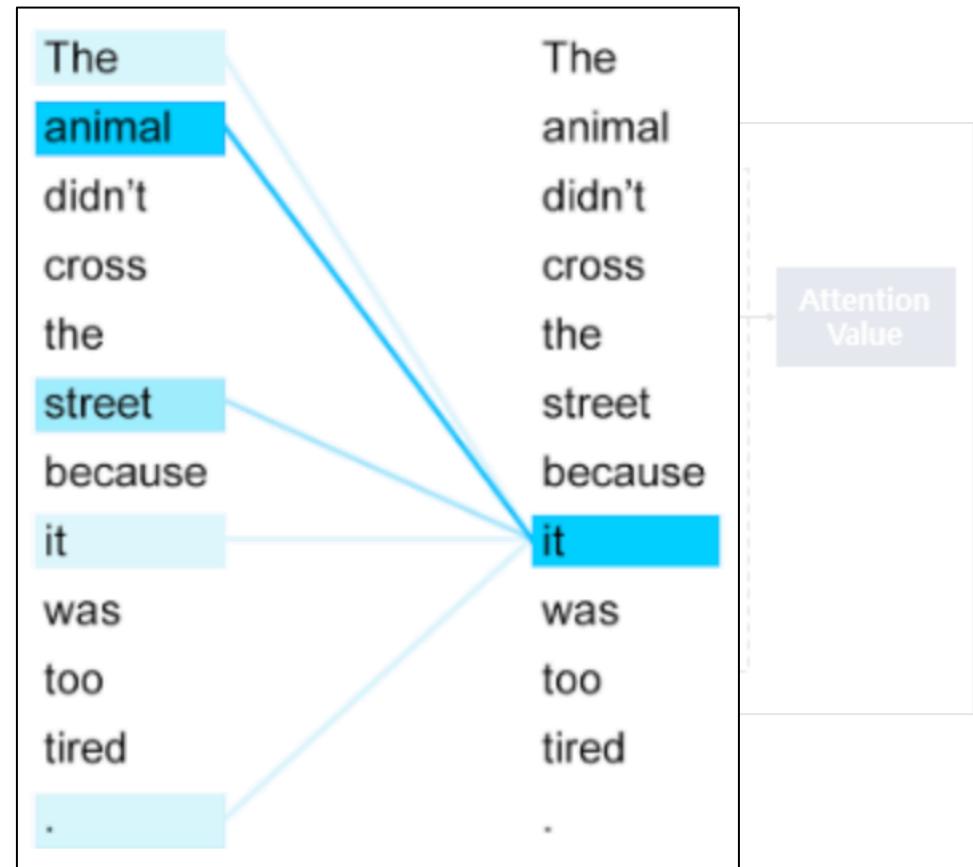
Unit 04 | Transformer architectures

Attention mechanism

- Attention의 구조에는 query, key, value가 존재함
- 주어진 query와 각 key간의 유사도를 구한 후, 유사도와 key를 맵핑하여 value를 매칭해줌
- output으로 key의 value를 weighted sum한 attention value가 반환됨

$$A(q, K, V) = \sum_i \frac{e^{q \cdot k_i}}{\sum_j e^{q \cdot k_j}} v_i$$

- self-Attention: 자기 자신의 query로 attention 과정을 진행하는 것



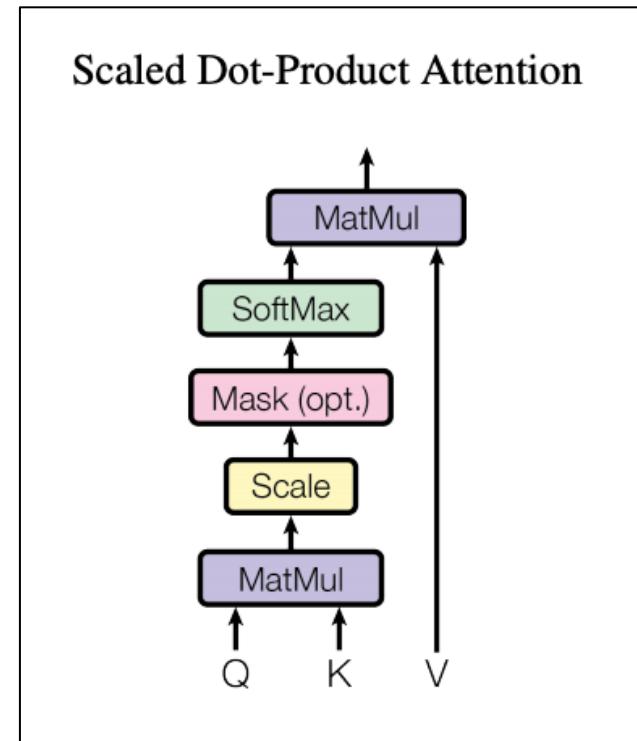
Unit 04 | Transformer architectures

Scaled Dot-Product Attention

- Transformer에서는 이러한 Attention 매커니즘을 ‘Scaled Dot-Product Attention’이라고 함

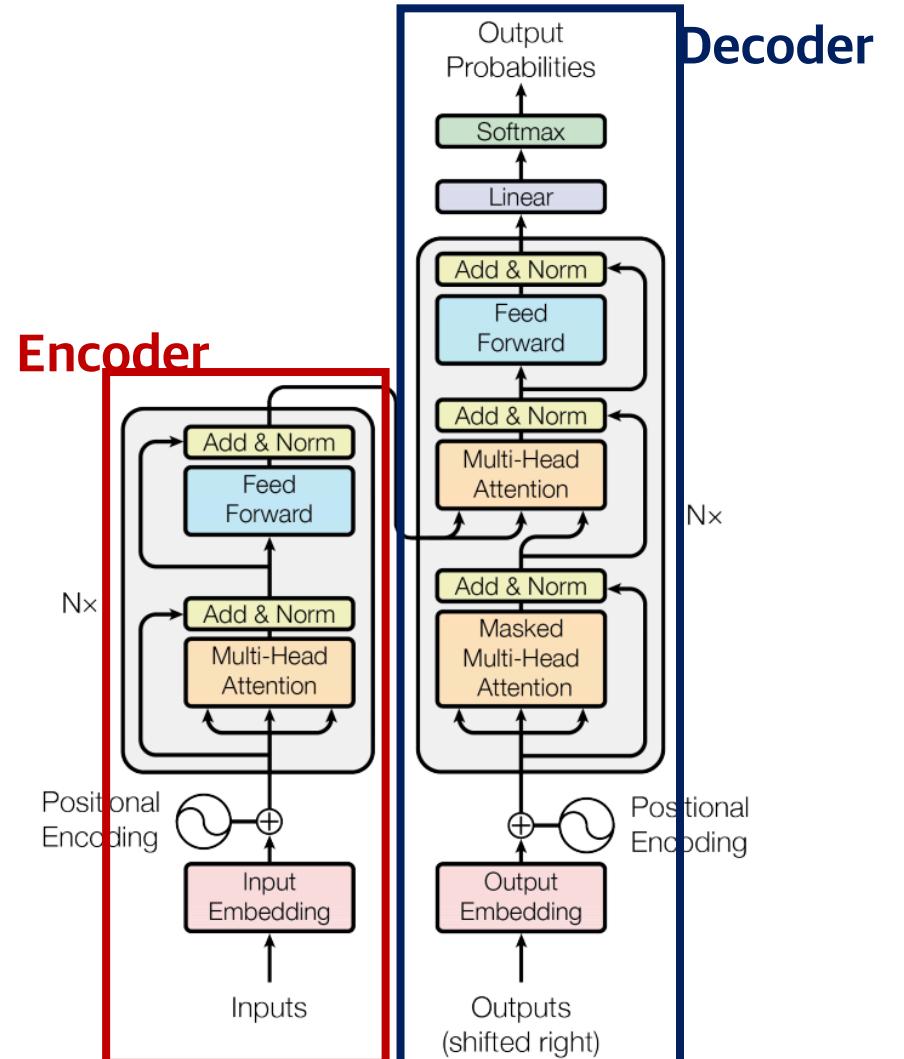
$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

- 모든 query와 key에 대한 dot-product를 계산하고 각각을 $\sqrt{d_k}$ 로 scaling을 해줌
- query와 각 key간의 유사도가 softmax를 거친 후에 value와 다시 곱해 진다면 해당 query에서 가장 높은 값을 가지는 정보를 알 수 있게 됨



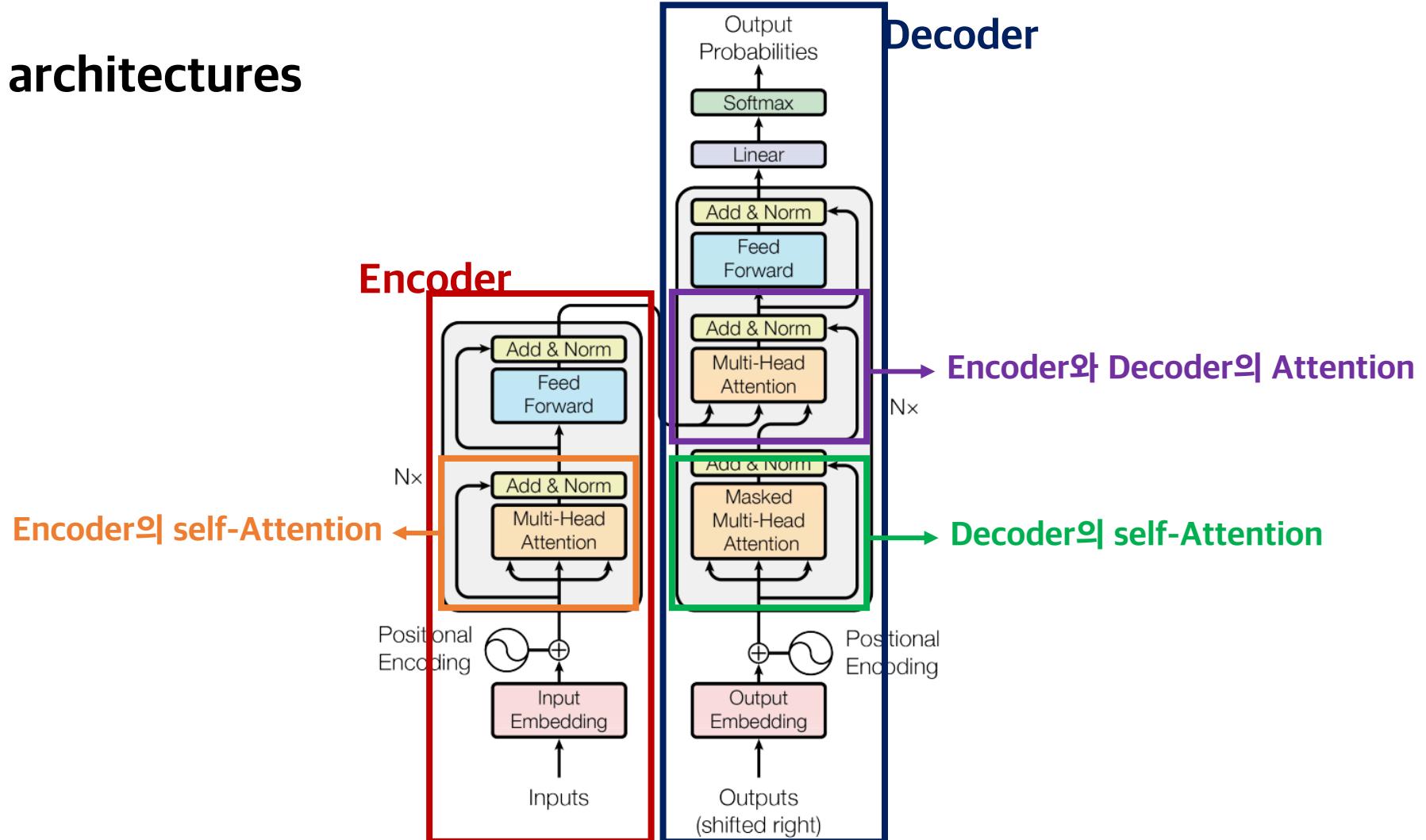
Unit 04 | Transformer architectures

Transformer architectures



Unit 04 | Transformer architectures

Transformer architectures



Unit 04 | Transformer architectures

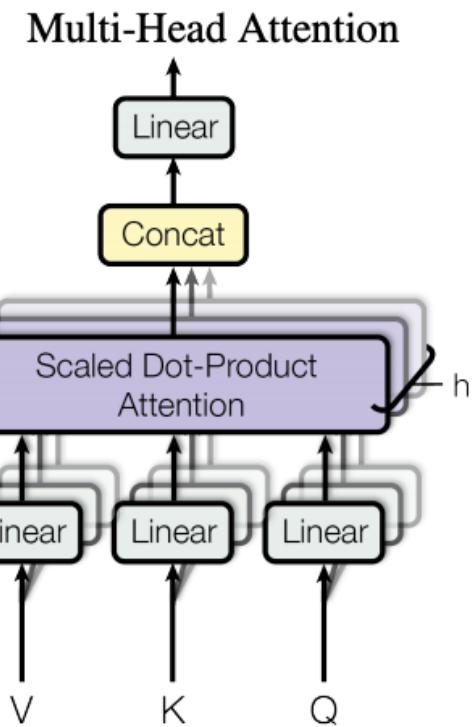
Transformer architectures

- Multi-head Attention

- query에 대한 attention 매커니즘을 여러 head(번) 진행하는 방법으로 동일한 문장을 다른 관점에서 해석하는 관점과 유사함
- 여러 개의 head를 concatenate하고 다시 projection을 수행

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O$$

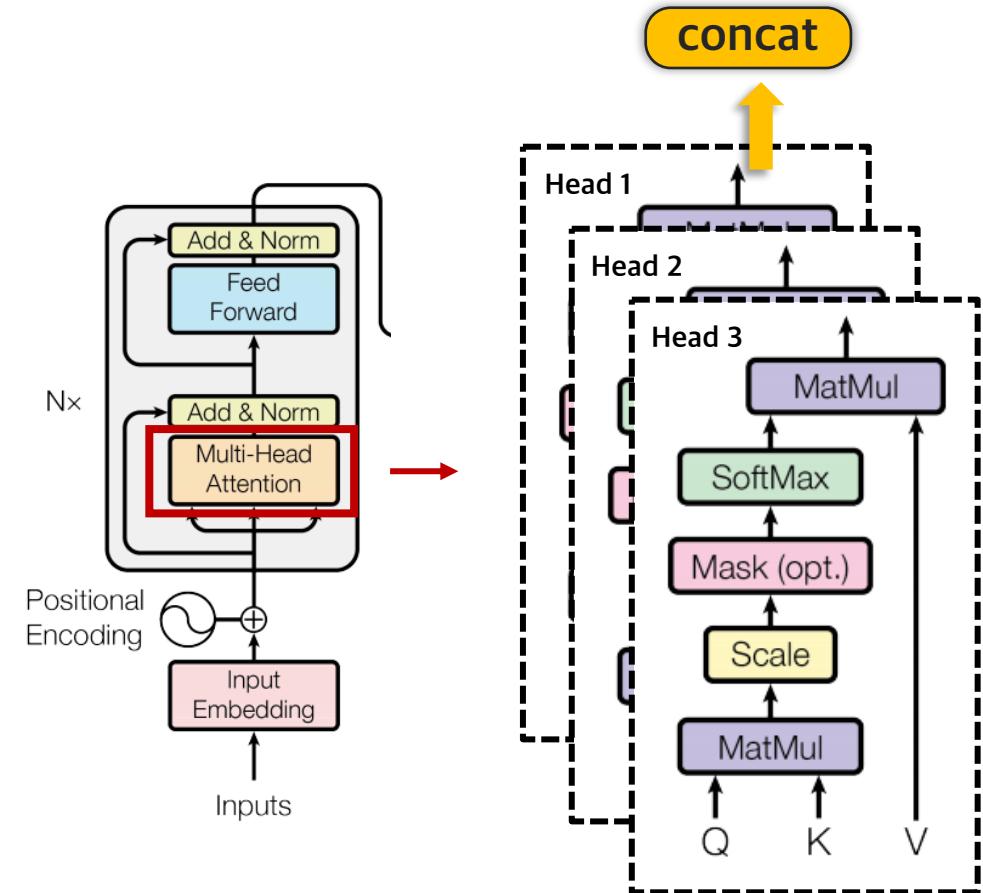
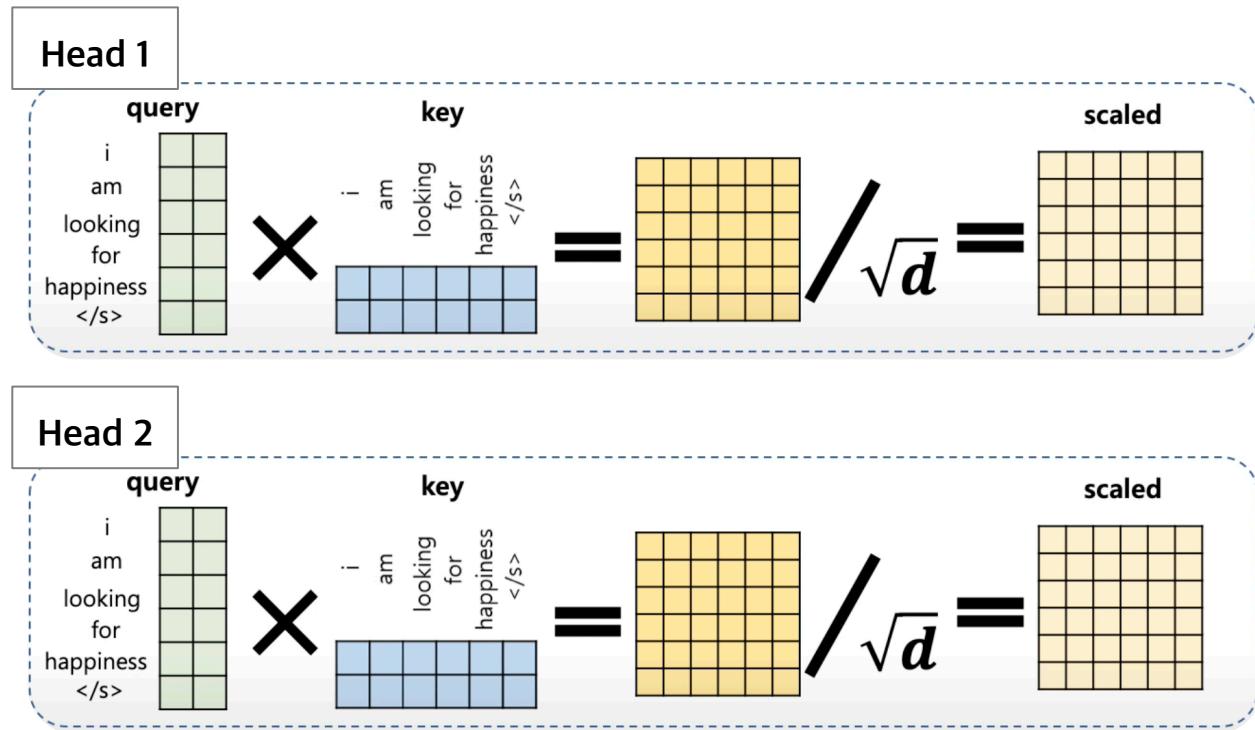
where $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$



Unit 04 | Transformer architectures

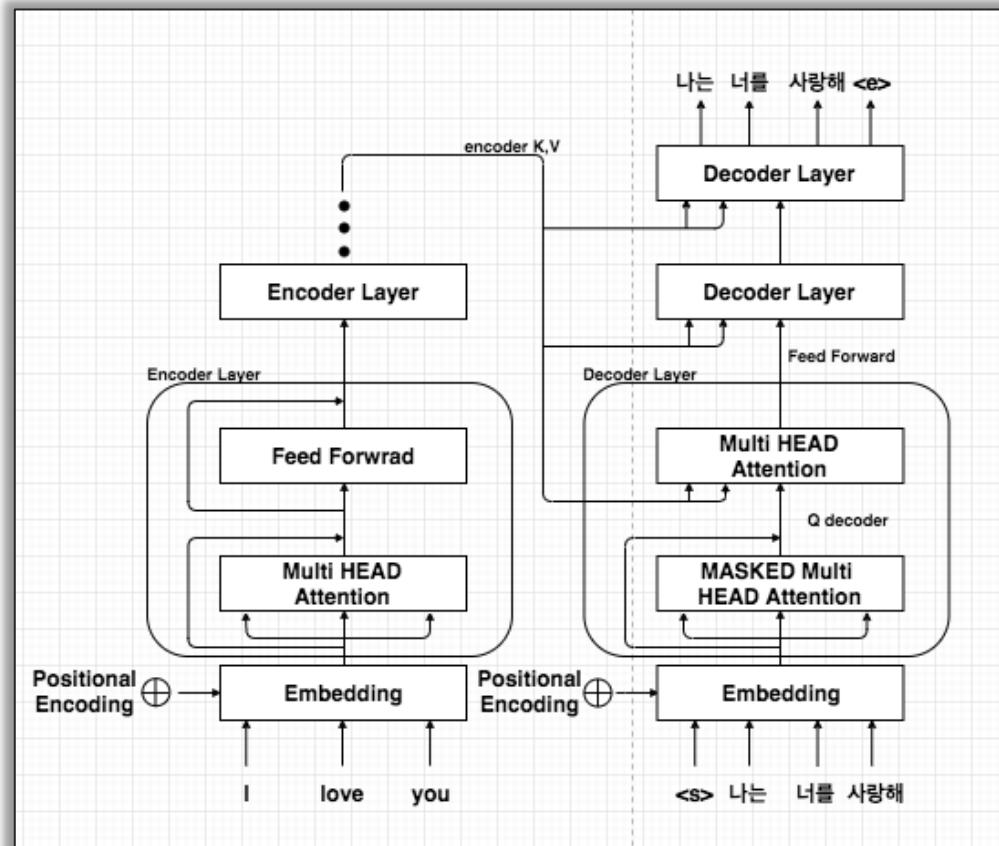
Transformer architectures

- Multi-head Attention



Unit 04 | Transformer architectures

Transformer architectures



Unit 04 | Transformer architectures

Transformer

- Results

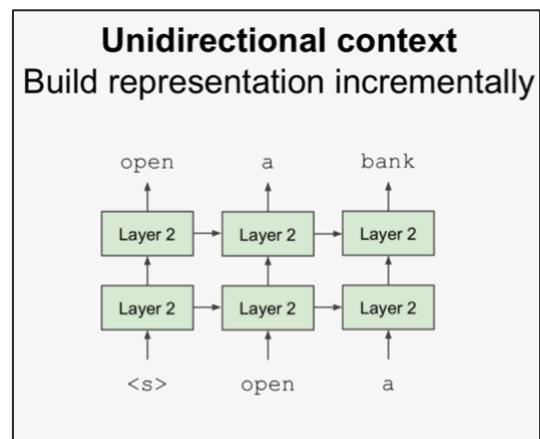
Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	41.29	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1		$3.3 \cdot 10^{18}$
Transformer (big)	28.4	41.8		$2.3 \cdot 10^{19}$

Unit 05 | BERT

Unit 05 | BERT

BERT - Devlin et al. 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

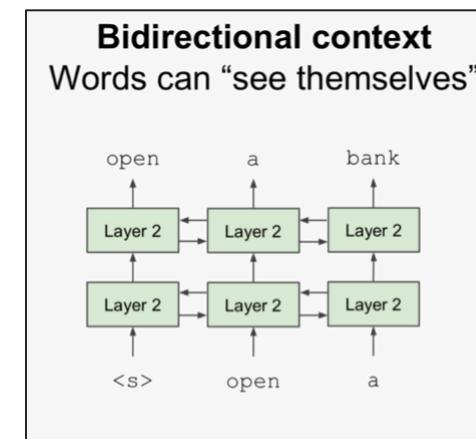
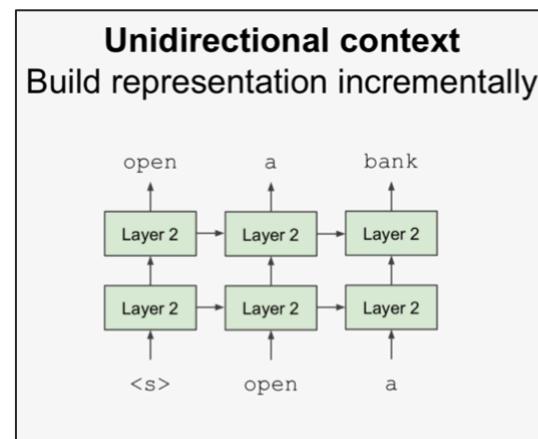
- Transformer의 encoder 부분을 차용하여 Bi-directional Language Model로 구축
- 기존 Language Model에서는 sentence를 left와 right 방향의 context만 확인할 수 있었음



Unit 05 | BERT

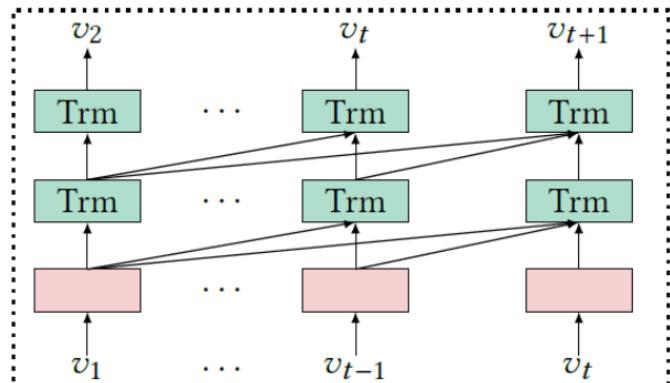
BERT - Devlin et al. 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

- Transformer의 encoder 부분을 차용하여 Bi-directional Language Model로 구축
- 기존 Language Model에서는 sentence를 left와 right 방향의 context만 확인할 수 있었음
- left와 right 방향을 모두 적용했던 기존의 bi-directional한 context는 결국 예측해야하는 word의 정보를 이미 알고 있는 상태가 되버리는 문제점이 있음

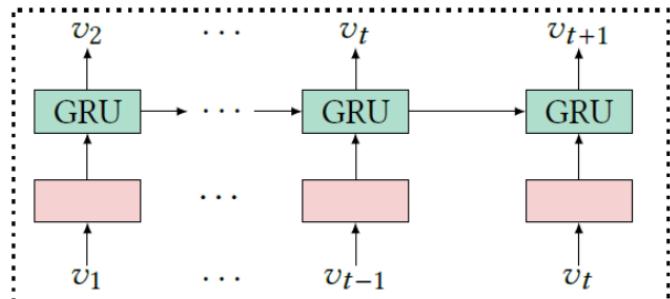


Unit 05 | BERT

BERT - Devlin et al. 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding



(c) SASRec model architecture.



(d) RNN based sequential recommendation methods.

사람의 행동은 예측하기 어렵다

사람의 행동은

단방향 예측

사람의 행동은 하기 어렵다

동사 (제약조건)

양방향 예측

사람의 행동은? → 제약이 없고, 관측되지 않은 외부 요소 존재

양방향 및 복잡한 관계 학습 가능한 모델 필요

Unit 05 | BERT

BERT - Devlin et al. 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

- Masked Language Model을 적용함으로써 context를 찾는 방법을 해결함
- Objective function
 - mask prediction

The men went to the [MASK] to buy a [MASK] of milk

- next sentence prediction

Sentence A: 'The men went to the store.'

Sentence B: 'He bought some snacks.'

Label: IsNextSentence

Sentence A: 'The men went to the store.'

Sentence B: 'Sky is blue.'

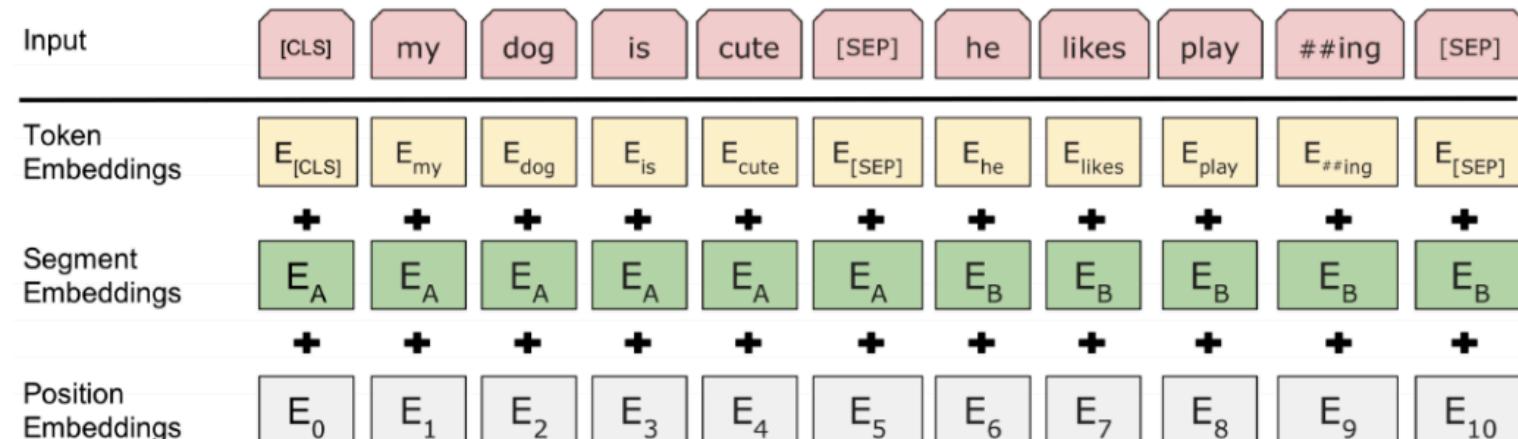
Label: NotNextSentence

Unit 05 | BERT

BERT - Devlin et al. 2018, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

- Sentence Pair encoding

- Word piece embedding 이용
- Next sentence prediction을 위한 segment embedding
- Transformer와 달리 Position 정보를 embedding으로 해결



참고자료

- KoreaUniv DSBA, [\[CS224N\]-13. Contextual Word Representation and Pretraining](#), 이정호
- Jeong Ukjae, [CS224n Lecture 13 Modeling contexts of use: Contextual Representations and Pretraining](#)
- wikidocs, [딥 러닝을 이용한 자연어 처리 입문 - 07\) 엘모\(Embeddings from Language Model, ELMo\)](#)
- Minsuk Heo 허민석, [ELMo \(Deep contextualized word representations\)](#)
- [전이 학습 기반 NLP \(2\): ULMFiT](#), 박성준
- wikidocs, [딥 러닝을 이용한 자연어 처리 입문 - 01\) 어텐션 메커니즘 \(Attention Mechanism\)](#)
- Platfarm tech team, [어텐션 메커니즘과 transformer\(self-attention\)](#)
- 포자랩스의 기술 블로그, [Attention is all you need paper 뽑개기](#)
- KoreaUniv DSBA, [Transformer & BERT](#), 김동화
- KoreaUniv DSBA, [BERT4REC](#), 이정호

Q & A

들어주셔서 감사합니다.