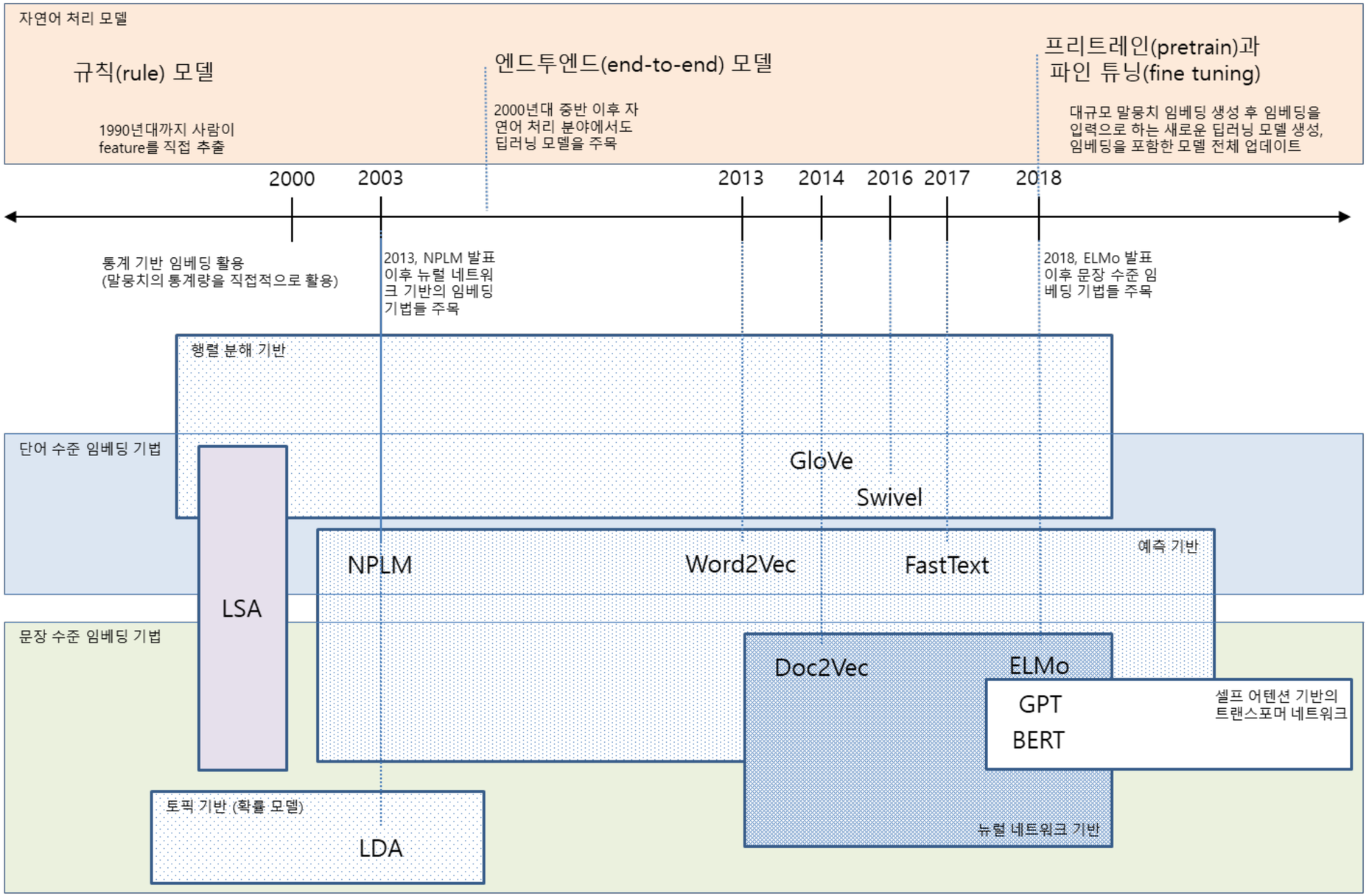
언어 모델
과거, 현재 그리고 미래

임베딩 기법의 역사와 종류



참고서적 : 한국어 임베딩 (이기창 저)

Models

- word2vec, Glove, fastText,…
- Semi-supervised Sequence Learning - 2015, Google
- Deep contextualized word representations (ELMo) - 2018, Allen AI
- Universal Language Model Fine-tuning for Text Classification (ULMFiT) - 2018, fast.ai
- Improving Language Understanding by Generative Pre-Training - 2018, OpenAI
- BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding - 2018, Google AI Language
- Many more to come … etc…

과거의 언어 모델

언어 모델의 개념과, BERT가 탄생하기 이전의 언어모델들을 살펴보겠습니다

모델이란?

언어 모델 (Language Model, LM)

[IT용어]모형(model)

1. ① 어떤 상황이나 물체 등 연구 대상 주제를 도면이나 사진 등 화상을 사용하거나 수식이나 악보와 같은 기호를 사용하여 표현한 것. 표현 양식에 따라서 화상 모형(graphical model)이나 기호 모형(symbolic model)이라고 하고, 특히 수학적 기호와 수식을 사용하여 표현한 것을 수학적 모형이라고 한다. 분자(molecule)의 도식 모형, 우주의 물질 분포의 수학적 모형, 기업 경영의 표 계산(숫자적) 모형 등이 있다. 모형을 변경하거나 조작하여 그것이 변형, 수정 또는 조건의 변화에 의해 어떻게 달라지는가를 알아낼 수 있다.

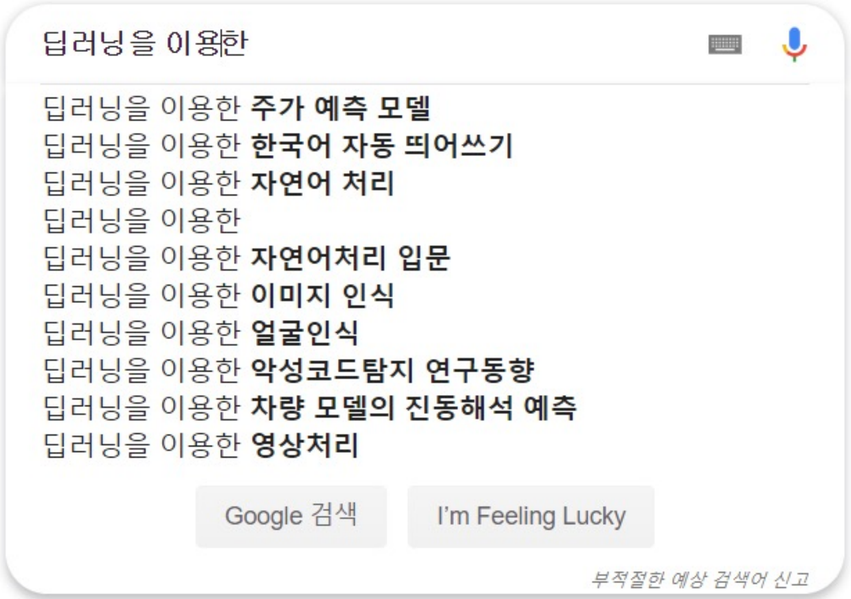
- 모델의 종류
  - 일기예보 모델, 데이터 모델, 비즈니스 모델, 물리 모델, 분자 모델 등

- 모델의 특징
  - **자연 법칙을 컴퓨터로 모사함**으로써 시뮬레이션이 가능
  - 이전 state를 기반으로 미래의 state를 예측할 수 있음
    (e.g. 습도와 바람 세기 등으로 내일 날씨 예측)
  - 즉, 미래의 state를 올바르게 예측하는 방식으로 모델 학습이 가능함

모델이란?

언어 모델 (Language Model, LM)

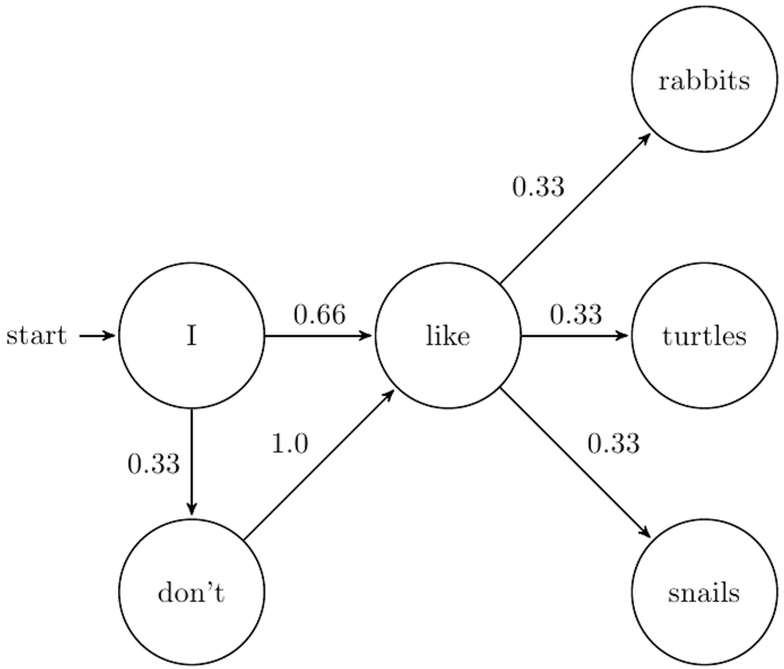
- **'자연어'의 법칙을 컴퓨터로 모사**한 모델 → 언어 '**모델**'
- 주어진 단어들로부터 그 **다음에 등장한 단어의 확률을 예측**하는 방식으로 학습 (이전 state로 미래 state를 예측)
- 다음의 등장할 단어를 잘 예측하는 모델은 그 **언어의 특성이 잘 반영된 모델**이자, **문맥**을 잘 계산하는 좋은 언어 모델



Markov 확률 기반의 언어 모델

언어 모델 (Language Model, LM)

- 마코프 체인 모델 (Markov Chain Model)
- 초기의 언어 모델은 다음의 단어나 문장이 나올 확률을 통계와 단어의 *n*-gram을 기반으로 계산
- 딥러닝 기반의 언어모델은 **해당 확률을 최대로 하도록 네트워크를 학습**



	I	don't	like	rabbits	turtles	snails
I	0	0.33	0.66	0	0	0
don't	0	0	1.0	0	0	0
like	0	0	0	0.33	0.33	0.33
rabbits	0	0	0	0	0	0
turtles	0	0	0	0	0	0
snails	0	0	0	0	0	0

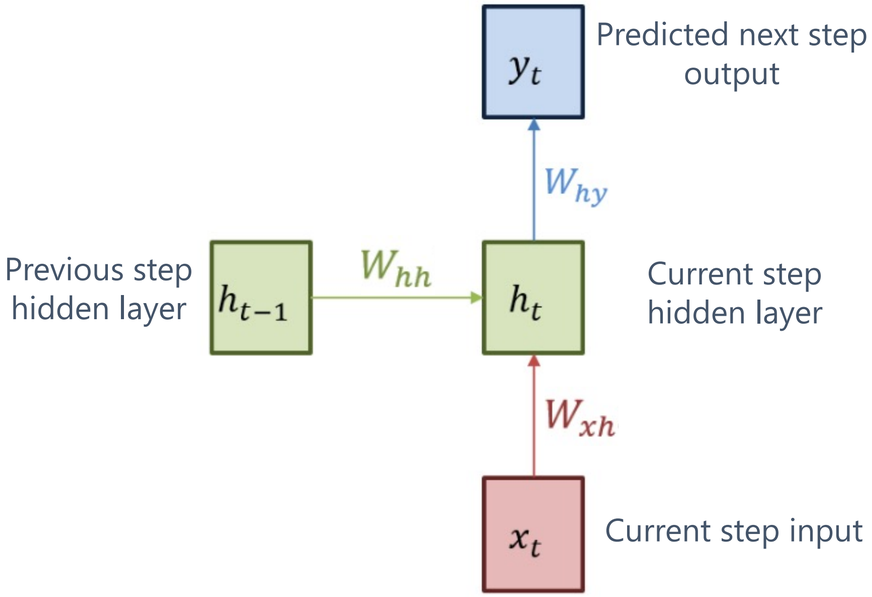
"I like rabbits" → 0.66 * 0.33 = 0.22
"I like turtles" → 0.66 * 0.33 = 0.22
"I don't like snails" → 0.33 * 1.0 * 0.33 = 0.11

Recurrent Neural Network (RNN) 기반의 언어 모델

언어 모델 (Language Model, LM)

- RNN은 히든 노드가 방향을 가진 엣지로 연결돼 순환구조를 이루는(directed cycle) 인공신경망의 한 종류
- 이전 state 정보가 다음 state를 예측하는데 사용됨으로써, 시계열 데이터 처리에 특화

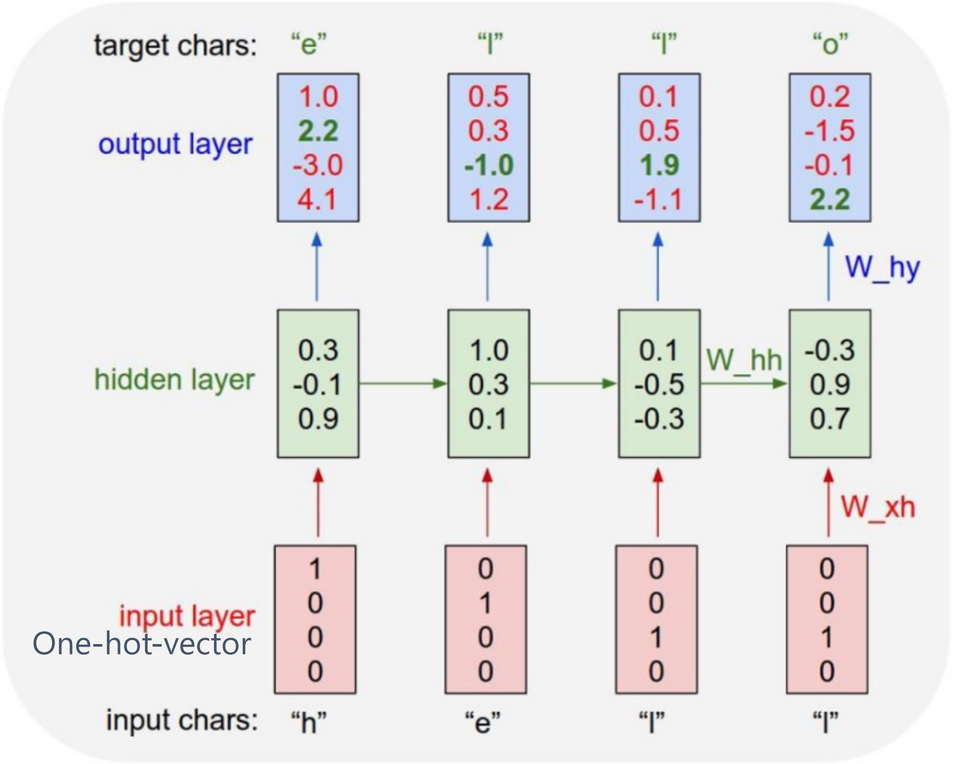
기본적인 RNN의 구조



$$y_t = W_{hy} h_t + b_y$$

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

다음 문자를 학습하는 RNN encoder

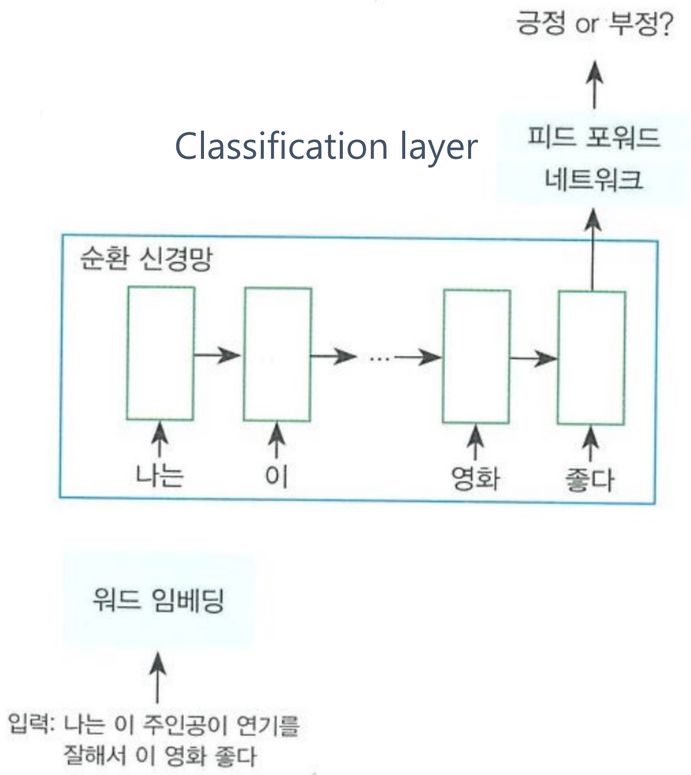


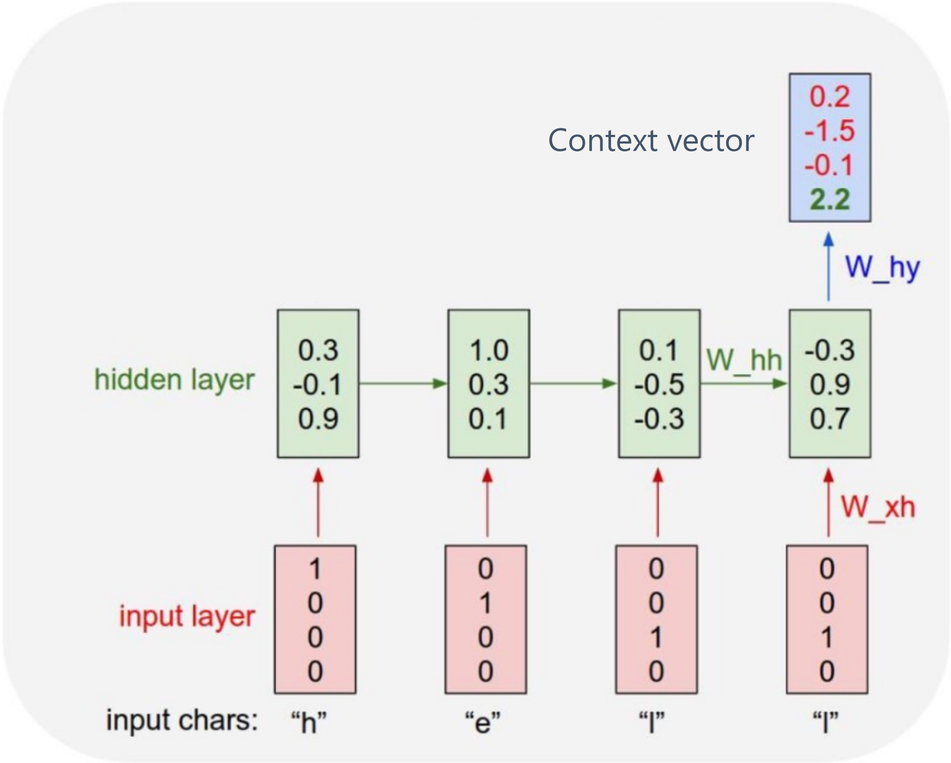
* RNN와 LSTM을 이해해보자!
(https://ratsgo.github.io/natural%20language%20processing/2017/03/09/rnnlstm/)

RNN 언어 모델을 이용한 Application

언어 모델 (Language Model, LM)

- 마지막 출력은 앞선 단어들의 '문맥 ' 을 고려해서 만들어진 최종 출력 vector → Context vector
- 출력된 context vector 값에 대해 classification layer를 붙이면 문장 분류를 위한 신경망 모델



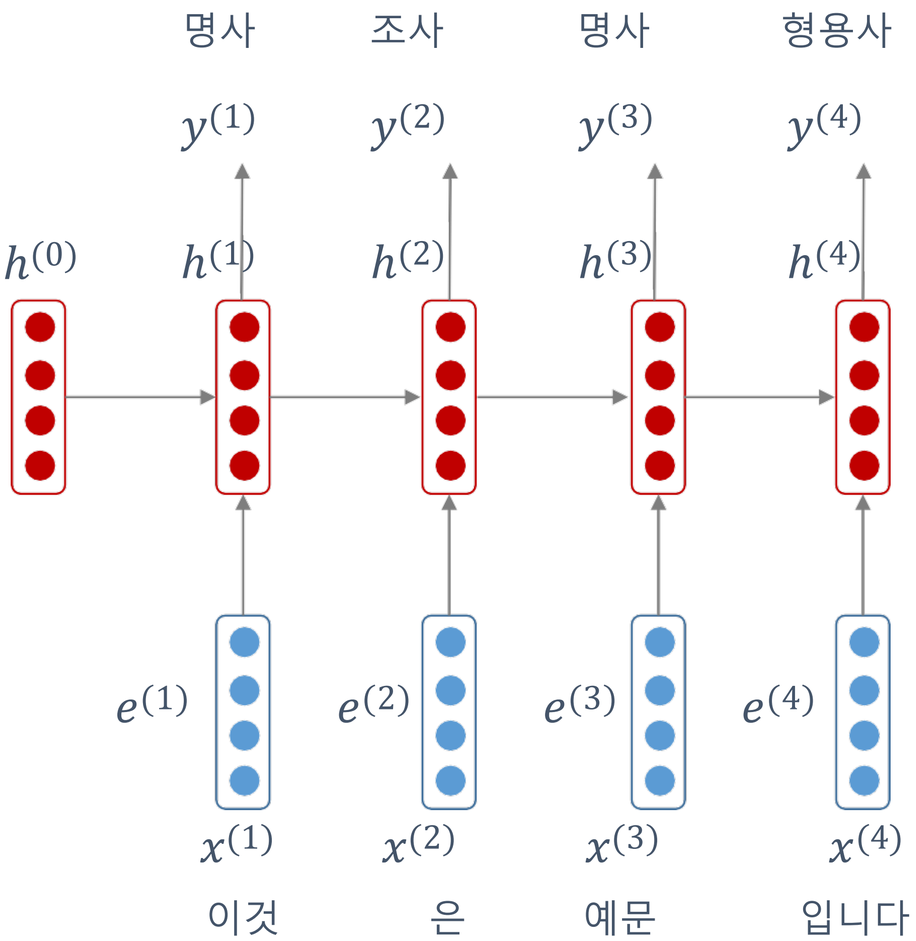


* 텐서플로와 머신러닝으로 시작하는 자연어처리
(http://www.kyobobook.co.kr/product/detailViewKor.laf?ejkGb=KOR&mallGb=KOR&barcode=9791158391379&orderClick=LAG&Kc=)

RNN based Tagging

언어 모델 (Language Model, LM)

- RNN 은 sequential labeling 에 이용될 수 있습니다.

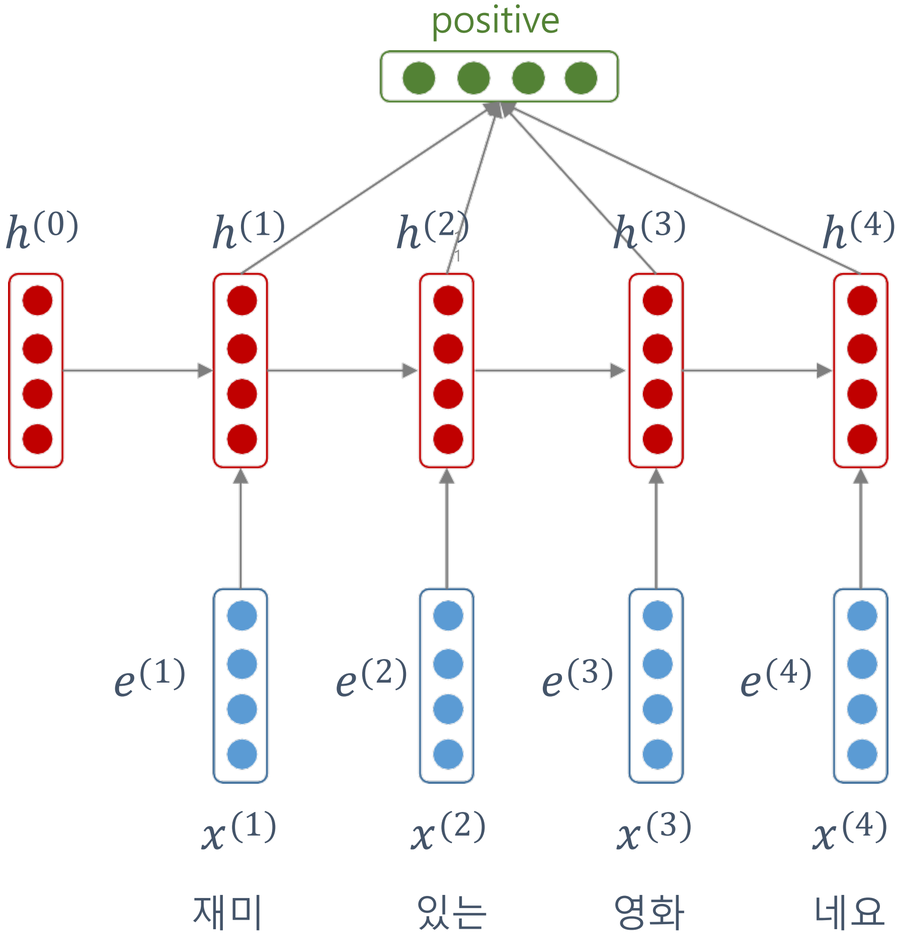


http://web.stanford.edu/class/cs224n/

RNN based Sentence Classification

언어 모델 (Language Model, LM)

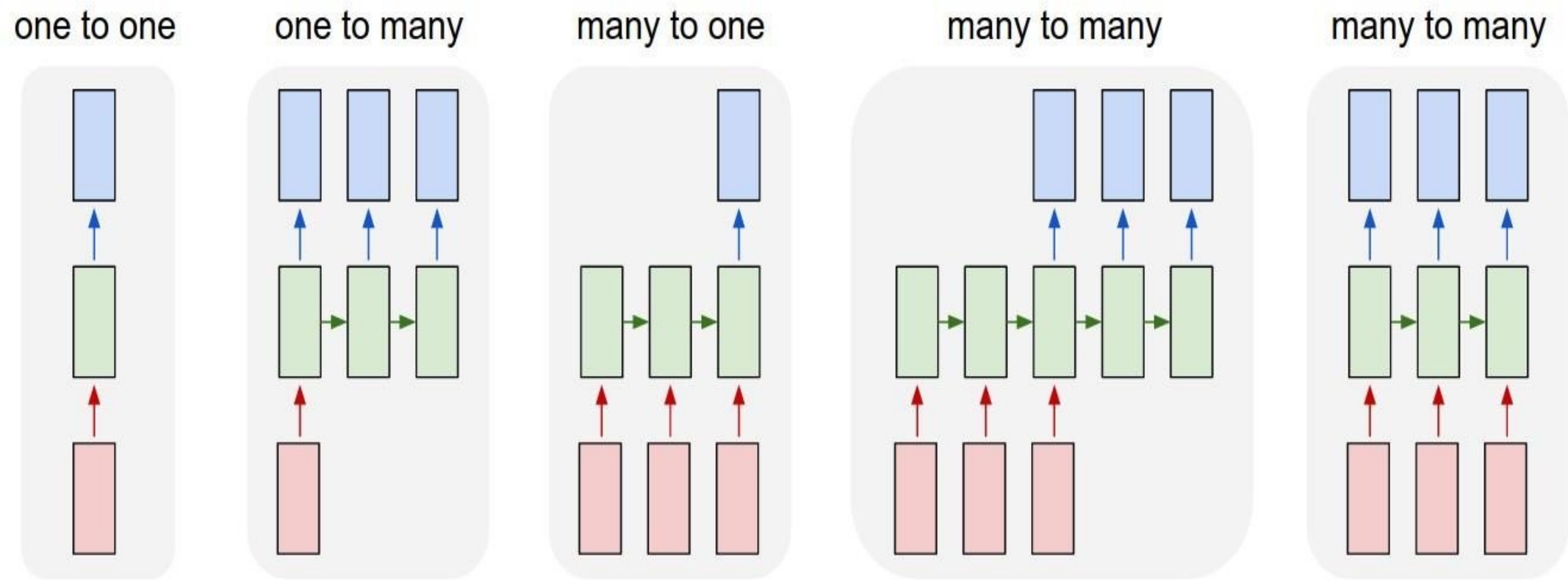
- 마지막 hidden states vector 를 이용하여 sentence classification 을 할 수 있습니다.
- Hidden vectors의 element-wise max or mean을 이용하여 문장 벡터를 만들 수 있습니다.



RNN 언어 모델을 이용한 Application

언어 모델 (Language Model, LM)

- 마지막 출력은 앞선 단어들의 문맥을 고려해서 만들어진 최종 출력 vector → context vector
- 출력된 context vector 값에 대해 classification layer를 붙이면 문장 분류를 위한 신경망 모델

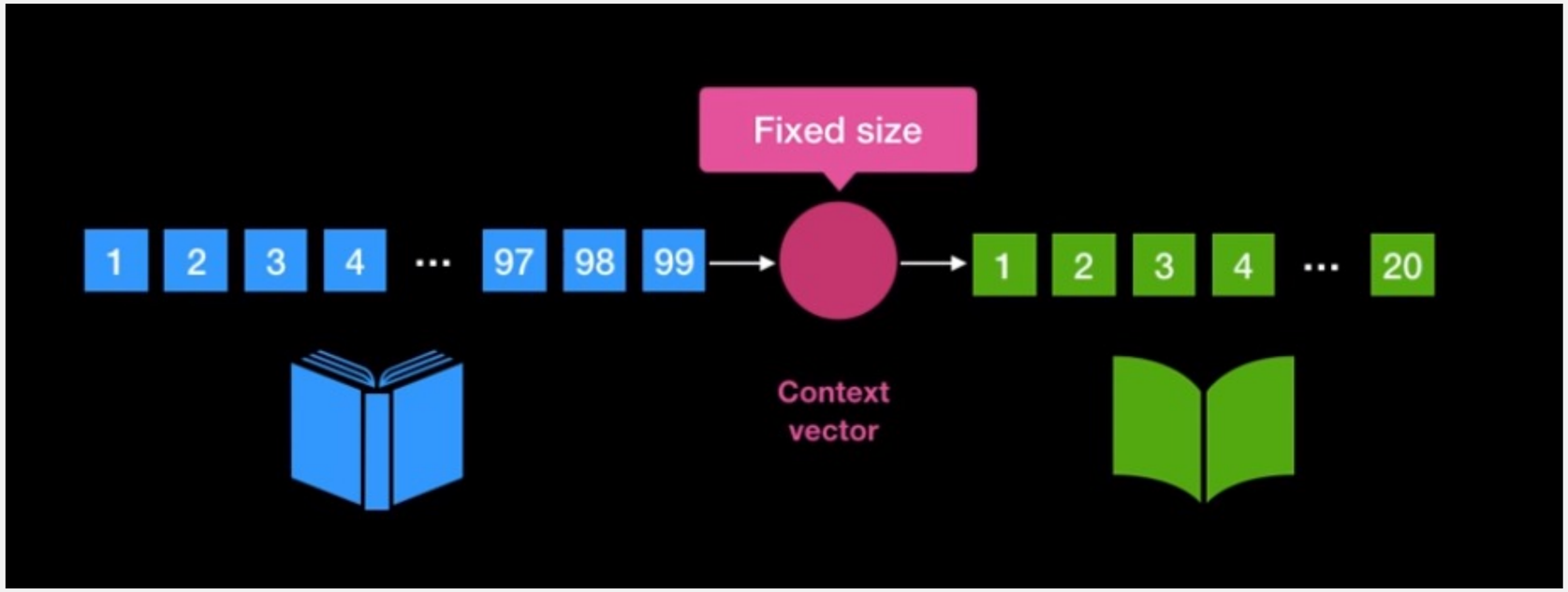


http://karpathy.github.io/2015/05/21/rnn-effectiveness/

RNN 의 구조적 문제점

언어 모델 (Language Model, LM)

- 입력 sequence의 길이가 매우 긴 경우, 처음에 나온 token에 대한 정보가 희석
- 고정된 context vector 사이즈로 인해 긴 sequence에 대한 정보를 함축하기 어려움
- 모든 token이 영향을 미치니, 중요하지 않은 token도 영향을 줌



* [딥러닝 기계번역] 시퀀스 투 시퀀스 + 어텐션 모델
(https://www.youtube.com/watch?v=WsQLdu2JMgI)

Attention의 탄생!

Gated Recurrent Unit (GRU)

- RNN 은 gradient descent vanishing problem 이 발생합니다.
  - RNN 은 멀리 떨어진 단어 간의 연관성도 학습하려는 모델입니다.

    (language model 에서의 $x^{(1)}$ 과 $x^{(6)}$ 처럼)
  - $x^{(1)}$ 과 $x^{(6)}$ 이 연결되기 위해서는 여러 번의 gradient 를 거칩니다.

$$y^{(6)} = g\left(Uh^{(6)}\right)$$

$$h^{(6)} = f(W_h \cdot h^{(5)} + W_e e^{(6)})$$

$$\dots$$

$$h^{(1)} = f\left(W_h \cdot h^{(0)} + W_e e^{(1)}\right)$$

- Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078.

Gated Recurrent Unit (GRU)

- RNN 은 gradient descent vanishing problem 이 발생합니다.
  - 본래는 long dependency 를 학습하기 위해 제안된 모델이지만,
    사실상 local dependency 밖에 학습할 수 없습니다.
- 근본적인 이유는

  (1) input 정보를 선택적으로 hidden 에 넣지 못하였으며

  (2) 불필요한 문맥을 버리지 못했기 때문입니다.

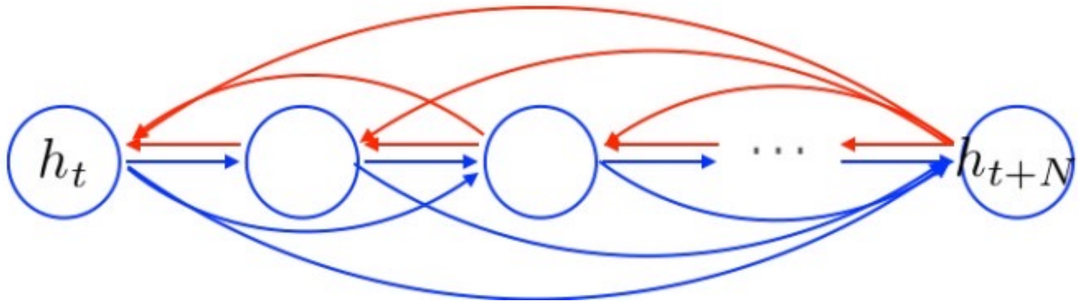
Gated Recurrent Unit (GRU)

- LSTM, GRU 는 hidden states 에 필요한 정보만을 저장하여 long dependency 를 더 잘 학습하기 위한 방법입니다.

Gated Recurrent Unit (GRU)

- GRU 나 LSTM 과 같은 gated 방법이 long dependency 를 학습 할 수 있는 이유는 "아마도" gates 에 의하여 멀리 떨어진 step 간에 shortcut 이 생기기 때문일 것이라 짐작합니다.

- Perhaps we can create *adaptive* shortcut connections.
- Let the net prune unnecessary connections *adaptively*.



Long Short-Term Memory (LSTM)

- Long Short-Term Memory (LSTM) 은 GRU 보다 먼저 제안된 방법입니다.
  - GRU 의 목표가 “비싼 계산 비용이 드는 LSTM 의 기능은 유지하면서 빠르게 학습할 수 있는 가벼운 cell 을 만드는 것” 이었습니다.

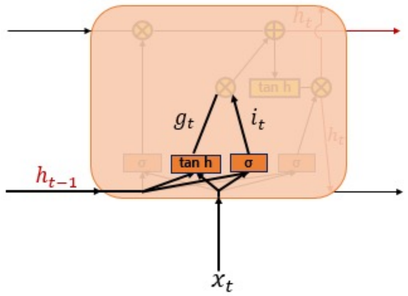
Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735-1780.

Long-Short Term Memory (LSTM)

언어 모델 (Language Model, LM)

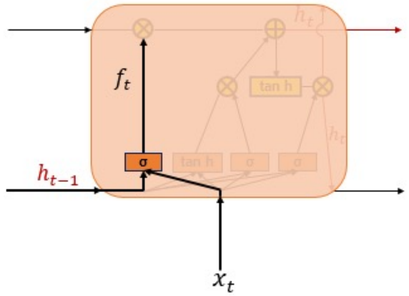
- RNN의 경우, 입력이 길어질수록 앞의 정보가 뒤로 충분히 전달되지 못하는 문제 발생
  -> 장기 의존성 문제(Long-Term Dependencies problem)
- 문장에서는 제일 처음에 온 단어가 중요한 역할을 할 수도 있음
- 이런 문제를 해결해주기 위하여, 고안된 모델이 LSTM
- 불필요한 기억을 지우고, 기억해야할 것들을 정함

Input Gate



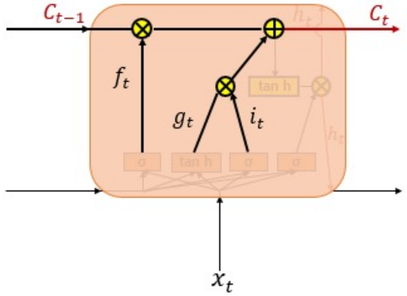
$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i)$$
$$g_t = tanh(W_{xg}x_t + W_{hg}h_{t-1} + b_g)$$

Forget Gate



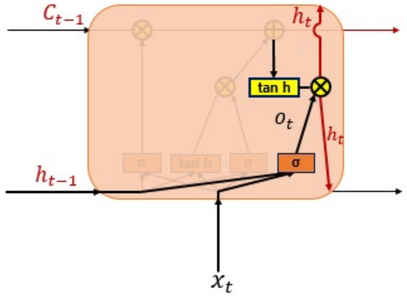
$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f)$$

Cell State



$$C_t = f_t \circ C_{t-1} + i_t \circ g_t$$

Output Gate & Hidden State

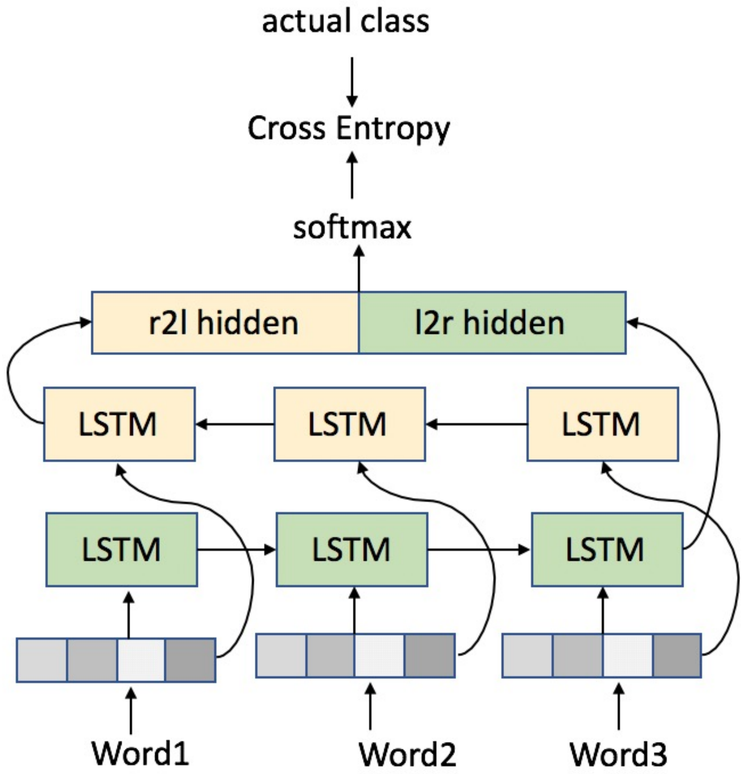


$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o)$$
$$h_t = o_t \circ tanh(c_t)$$

Bi-directional LSTM

언어 모델 (Language Model, LM)

- 텍스트 데이터는 정방향 추론 뿐만이 아니라, 역방향 추론도 유의미한 결과가 존재
- LSTM을 2층을 쌓고, 서로 반대방향에서 연산 후, 붙이는 방식으로 합쳐줌
- 병렬처리가 적용되면 속도는 거의 동일하고, 성능은 높이는 결과를 얻을 수 있음



ELMo (Embedding from Language Models)

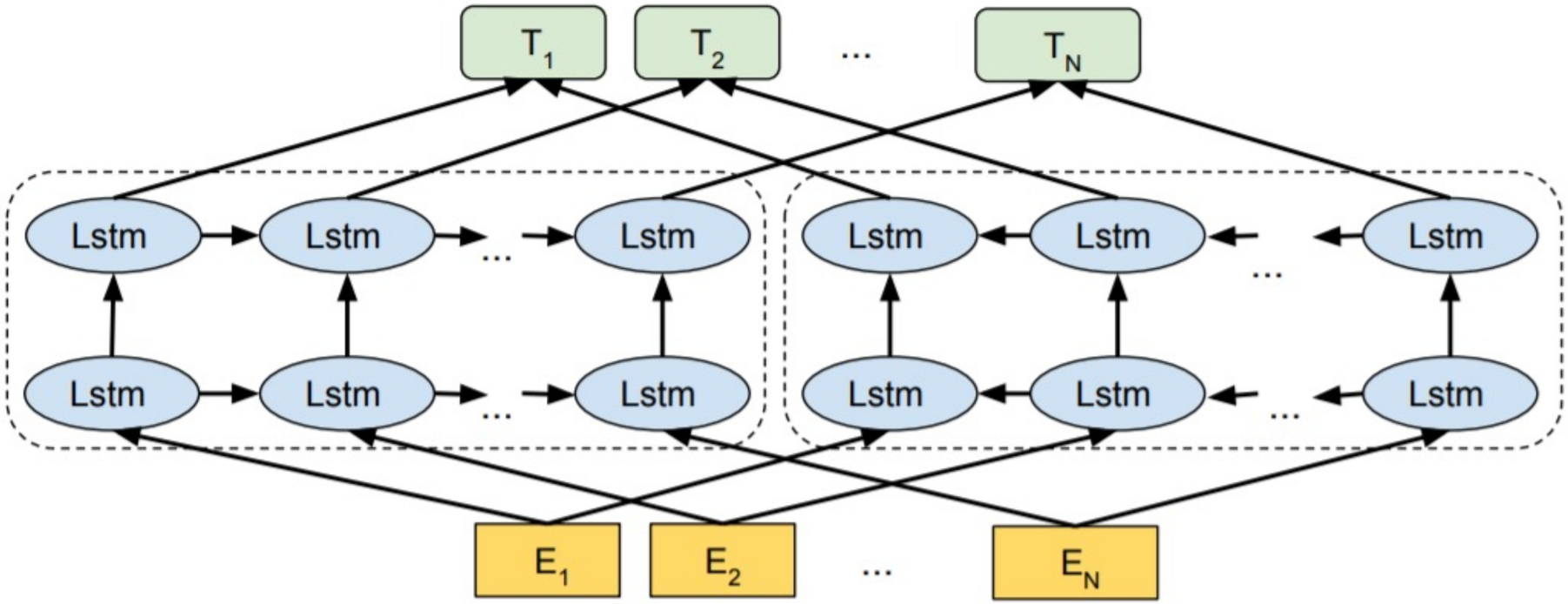
- ELMo 는 2 Layer LSTM 을 이용하는 word embedding 입니다.
  - 각 layers 의 hidden vectors 결합을 embedding vector 로 이용합니다.
- ELMo 는 단어의 문맥을 표현할 수 있습니다.
  - Word2Vec 은 단어가 문맥과 상관없이 고정된 벡터를 지닙니다.
  - 앞/뒤, 문장 전체의 단어를 고려하는 hidden vectors 를 단어 벡터로 이용함으로써 문맥을 표현할 수 있습니다.
- 단어 t_k의 임베딩 벡터는 모든 hidden vectors 의 선형 결합입니다.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Embeddings from Language Model (ELMo)

언어 모델 (Language Model, LM)

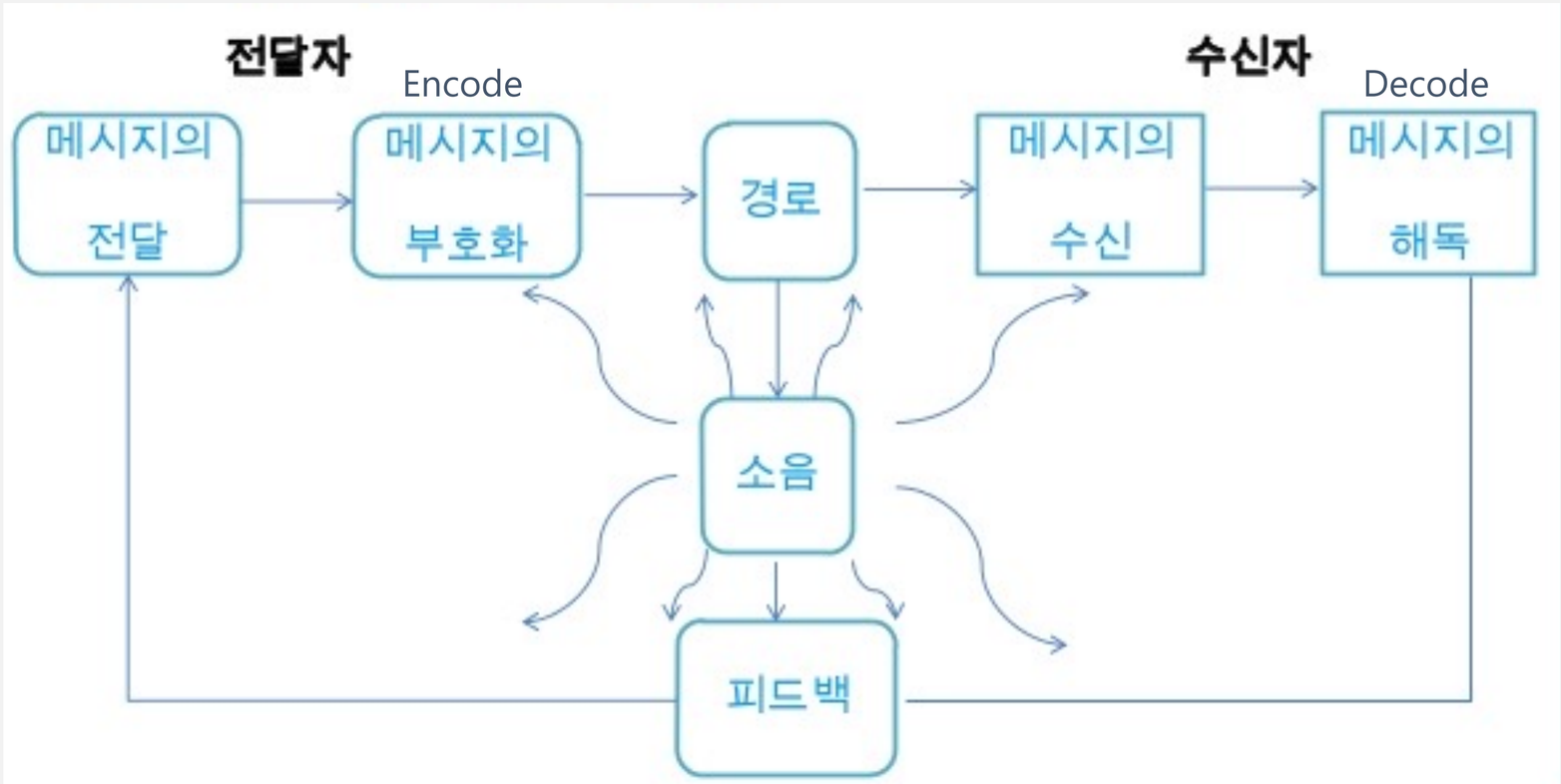
- 문맥을 반영한 워드 임베딩 기법
- 글자가 같은 단어도 다른 뜻을 가지는 경우가 있음
- Pre-trained 모델의 시작



Encoder - Decoder

언어 모델 (Language Model, LM)

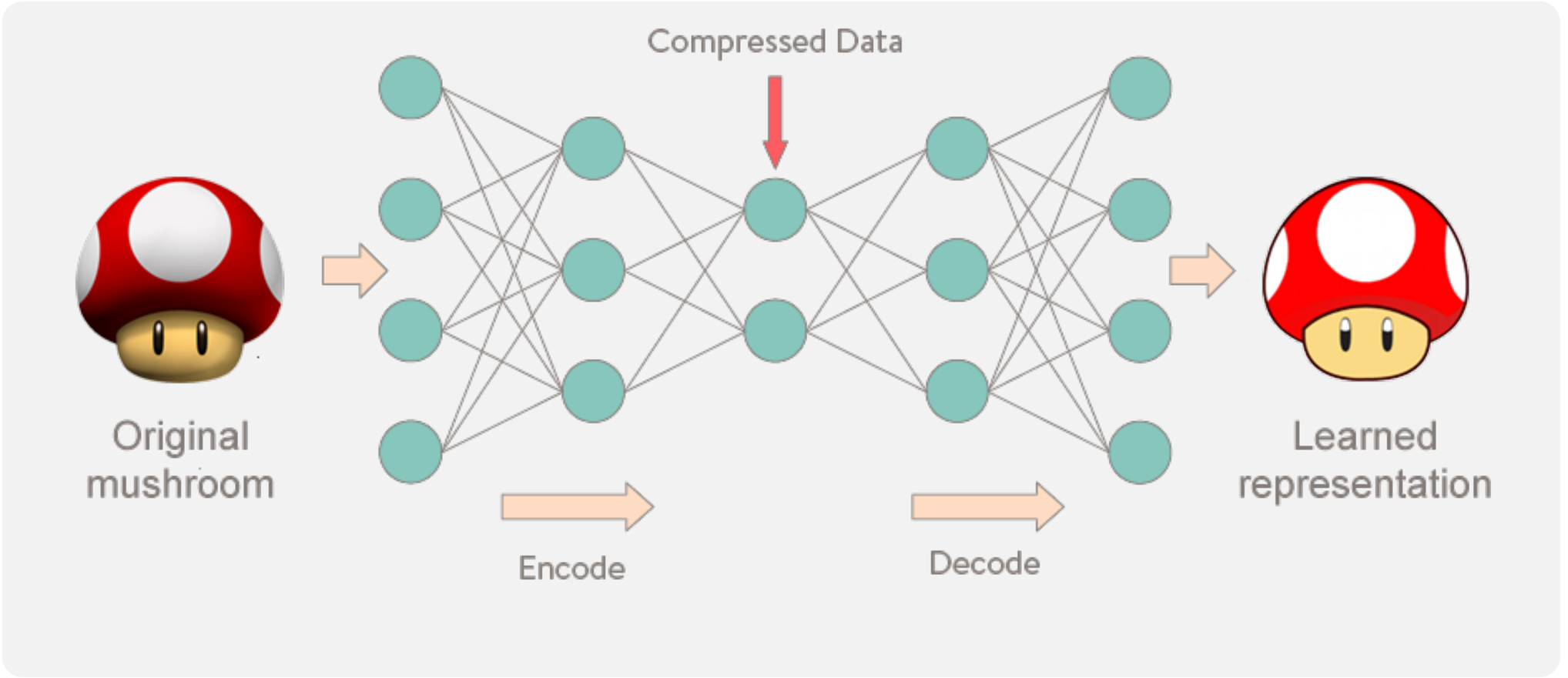
- RNN을 통해서 잘 encoding 된 언어를 다시 decoding을 하면?



Encoder - Decoder

언어 모델 (Language Model, LM)

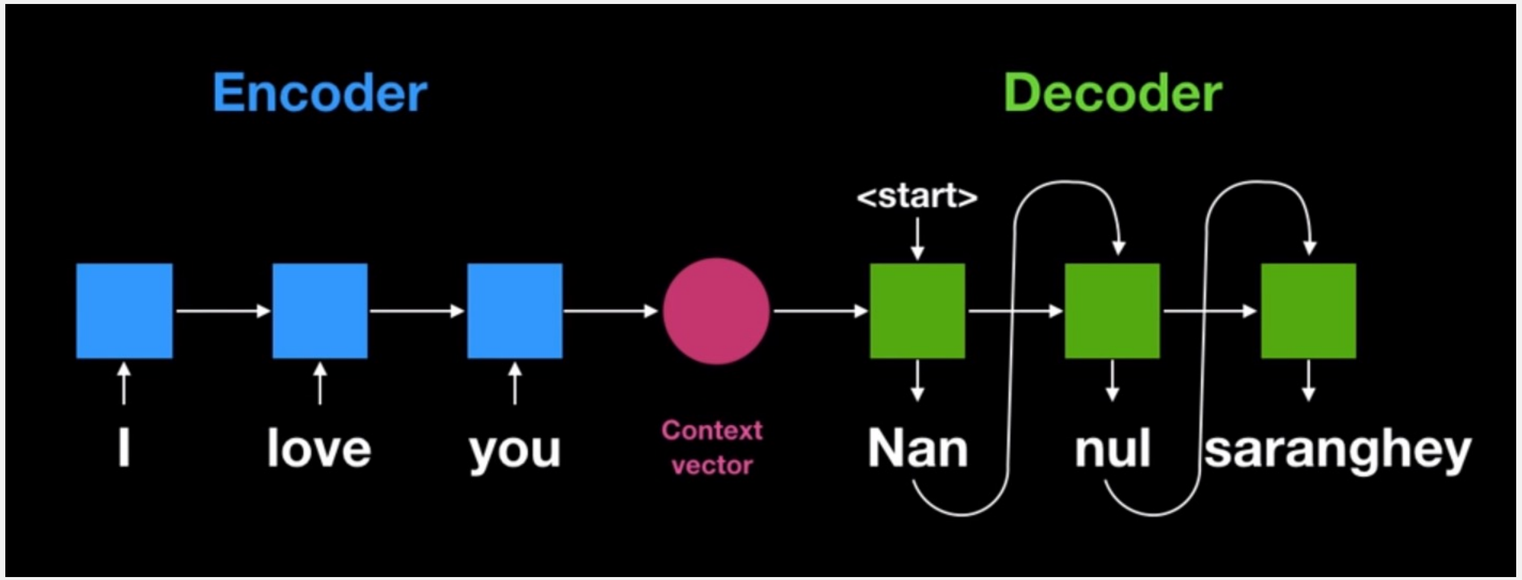
- RNN을 통해서 잘 encoding 된 언어를 다시 decoding을 하면?



RNN 모델 기반의 Seq2Seq

언어 모델 (Language Model, LM)

- Seq2Seq (Sequence to Sequence)
- Encoder layer: RNN 구조를 통해 Context vector 를 획득
- Decoder layer: 획득된 Context vector를 입력으로 출력을 예측

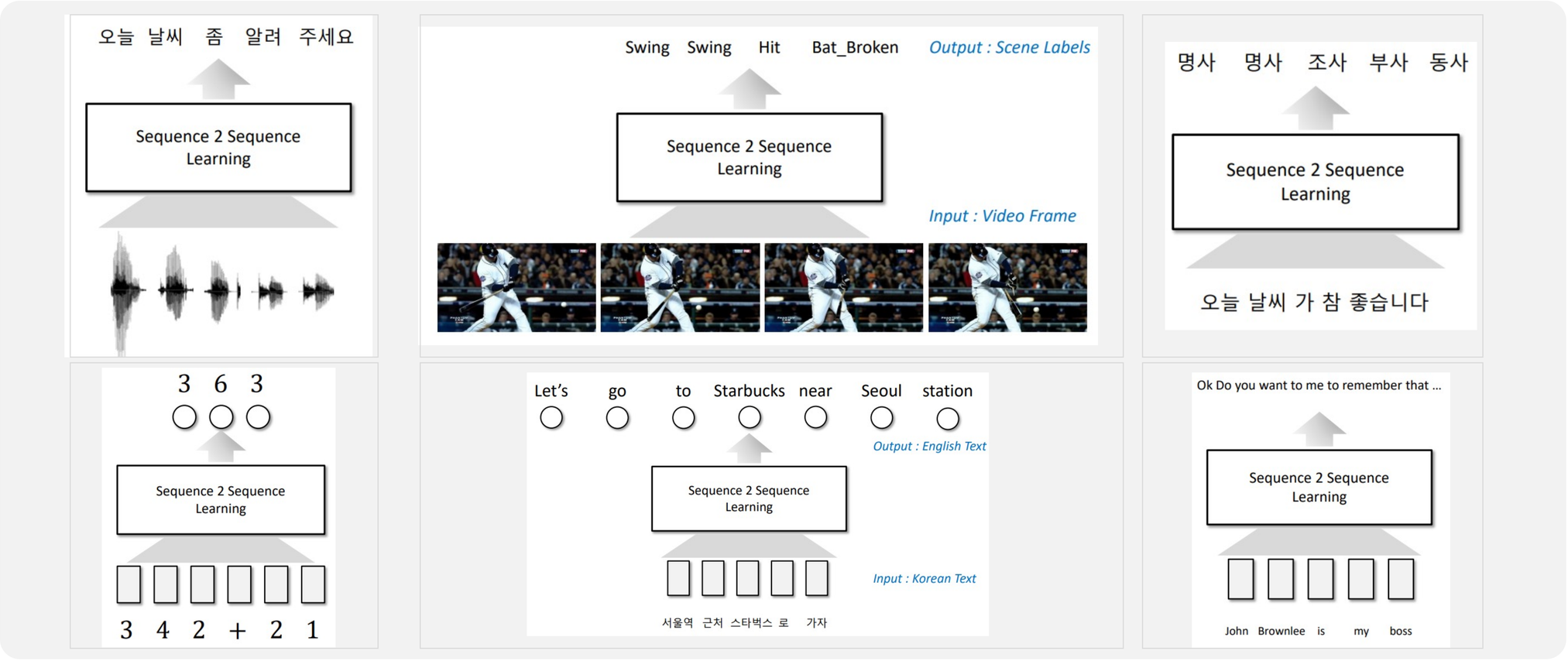


* [딥러닝 기계번역] 시퀀스 투 시퀀스 + 어텐션 모델 (https://www.youtube.com/watch?v=WsQLdu2JMgI)

Seq2Seq Applications

언어 모델 (Language Model, LM)

- Seq2Seq (Sequence to Sequence)
- Encoder layer: Context vector 를 획득
- Decoder layer: 획득된 Context vector를 입력으로 다음 state를 예측



* T아카데미 16차 토크온 세미나, 딥러닝을 활용한 자연어 처리 기술 실습
(https://tacademy.skplanet.com/live/player/onlineLectureDetail.action?seq=123)

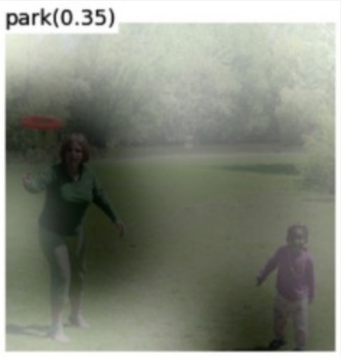
현재의 언어 모델

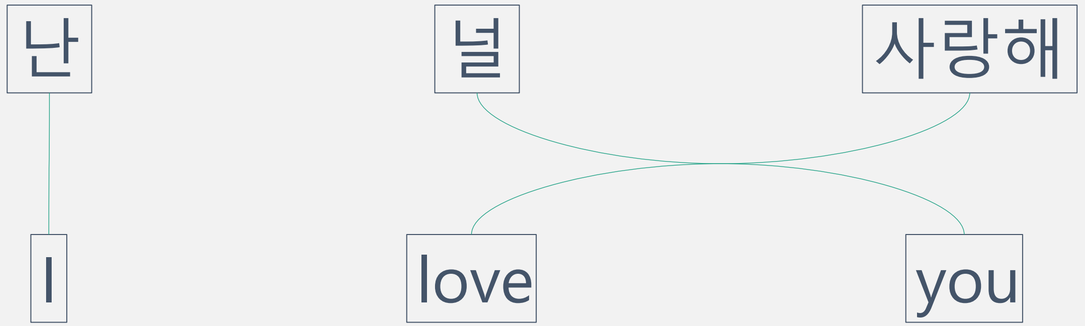
지금의 BERT를 있게 해준 Attention, Transformer를 살펴보겠습니다

Attention 모델

언어 모델 (Language Model, LM)

- 인간이 정보처리를 할 때, 모든 sequence를 고려하면서 정보처리를 하는 것이 아님
- 인간의 정보처리와 마찬가지로, 중요한 feature는 더욱 중요하게 고려하는 것이 Attention의 모티브

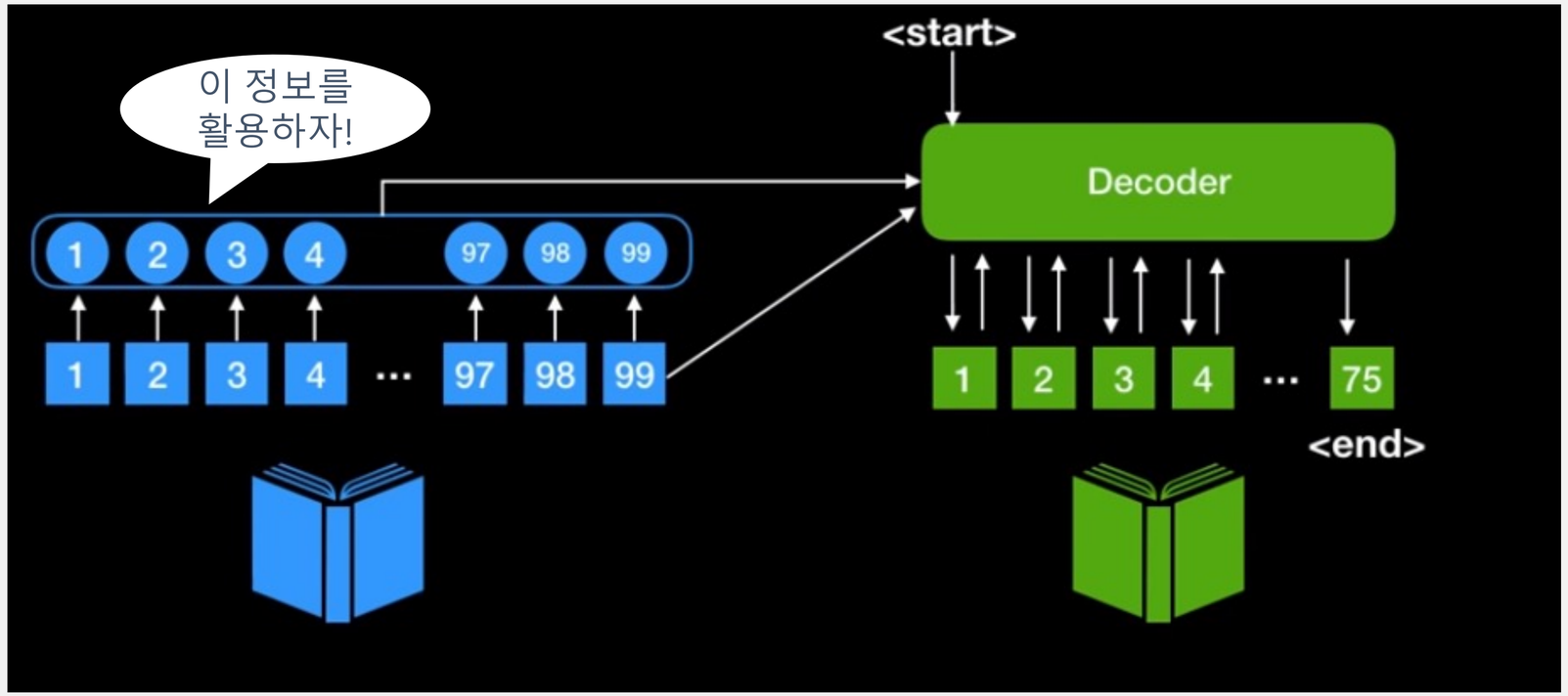




Attention 모델

언어 모델 (Language Model, LM)

- 기존 Seq2Seq에서는 RNN의 최종 output인 Context vector만을 활용
- Attention에서는 인코더 RNN 셀의 각각 output을 활용
- Decoder에서는 매 step마다 RNN 셀의 output을 이용해 dynamic하게 Context vector를 생성

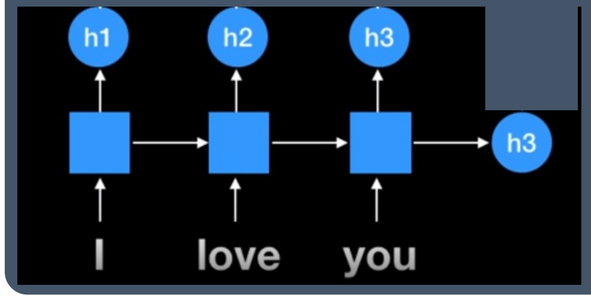


* [딥러닝 기계번역] 시퀀스 투 시퀀스 + 어텐션 모델
(https://www.youtube.com/watch?v=WsQLdu2JMgI)

Attention 모델

언어 모델 (Language Model, LM)

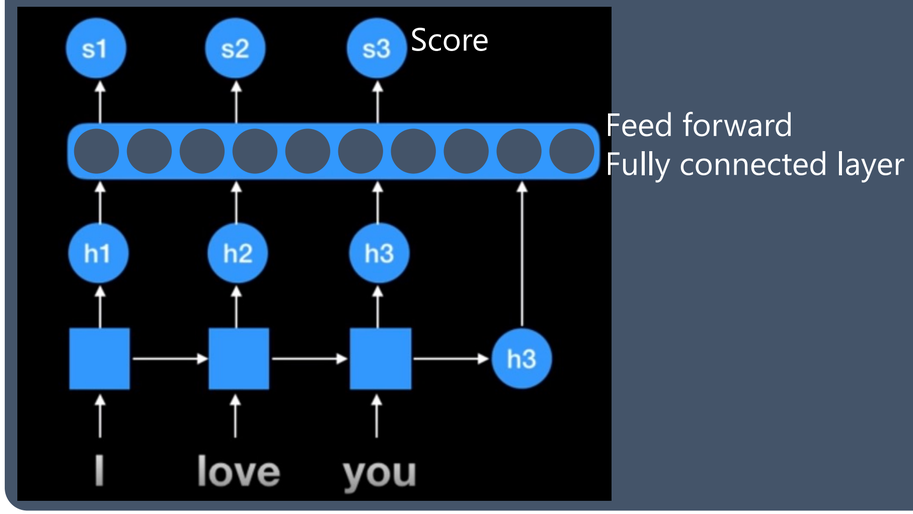
- 기존 RNN 모델에서 시작



Attention 모델

언어 모델 (Language Model, LM)

- 기존 RNN 모델에서 시작
- RNN 셀의 각 output들을 입력으로 하는 Feed forward fully connected layer
- 해당 layer의 output을 각 RNN 셀의 Score로 결정

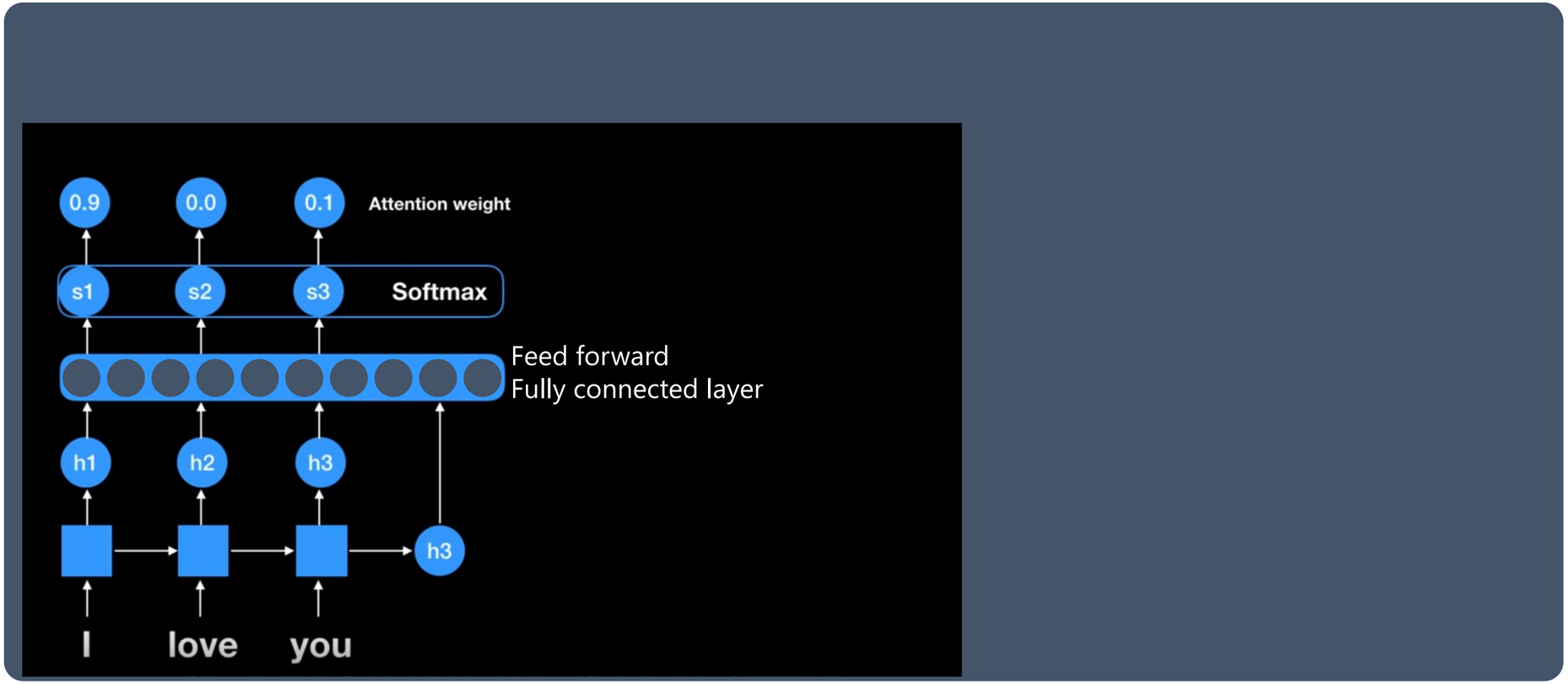


* [딥러닝 기계번역] 시퀀스 투 시퀀스 + 어텐션 모델
(https://www.youtube.com/watch?v=WsQLdu2JMgI)

Attention 모델

언어 모델 (Language Model, LM)

- 출력된 score에 Softmax를 취함으로써 0-1 사이의 값으로 변환
- 해당 값을 Attention weight로 결정

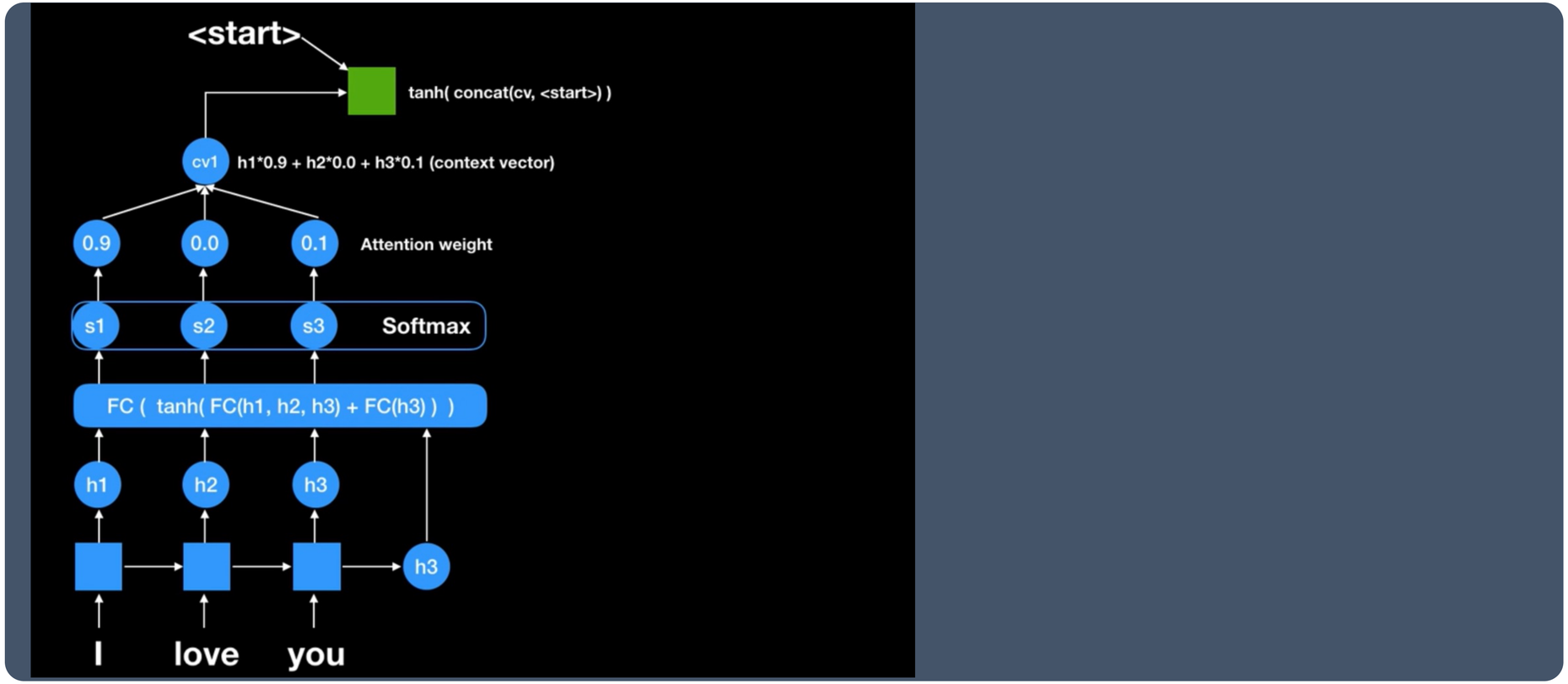


* [딥러닝 기계번역] 시퀀스 투 시퀀스 + 어텐션 모델
(https://www.youtube.com/watch?v=WsQLdu2JMgI)

Attention 모델

언어 모델 (Language Model, LM)

- 출력된 score에 Softmax를 취함으로써 0-1 사이의 값으로 변환
- 해당 값을 Attention weight로 결정
- Attention weight와 hidden state를 곱해서 Context vector 획득

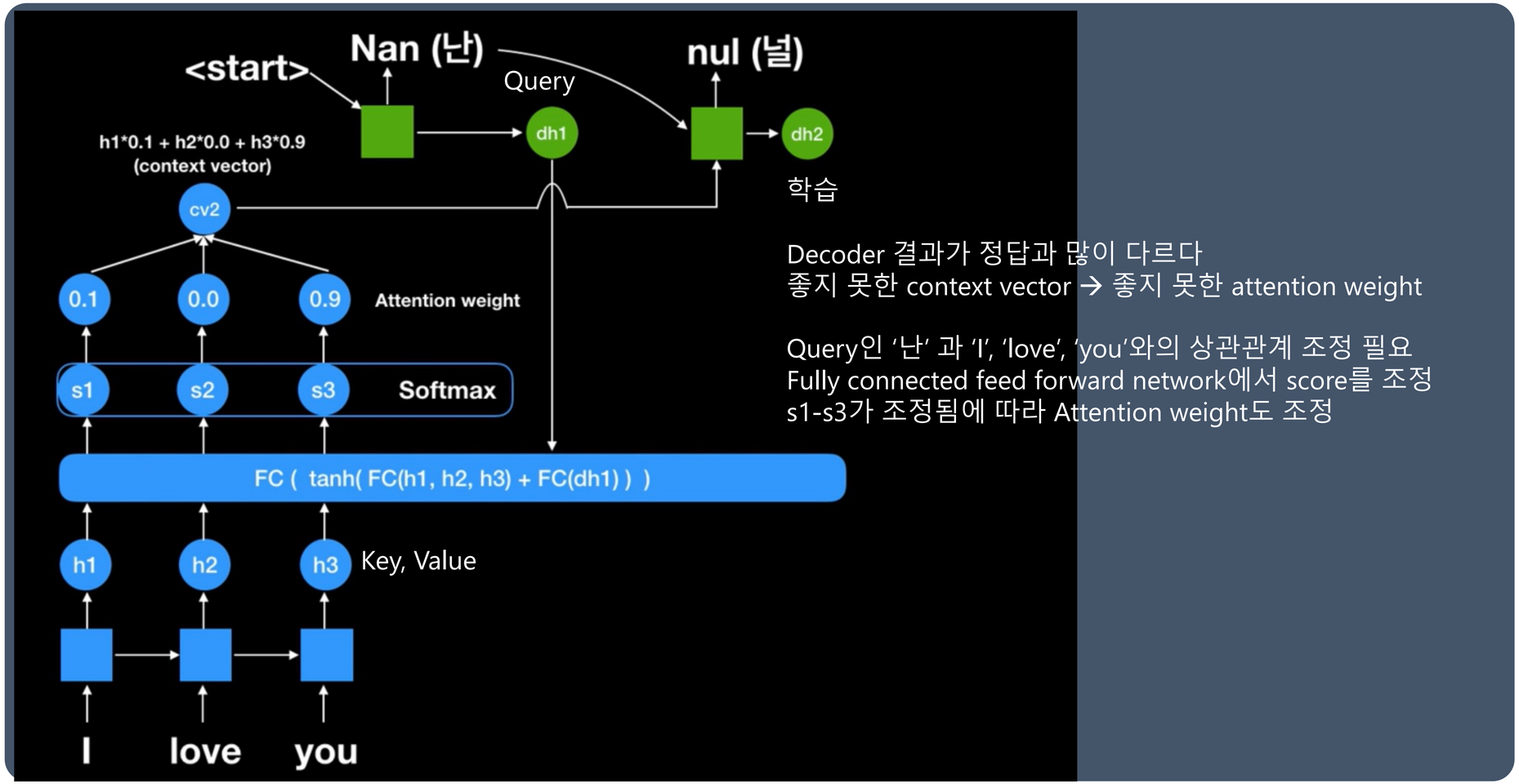


* [딥러닝 기계번역] 시퀀스 투 시퀀스 + 어텐션 모델
(https://www.youtube.com/watch?v=WsQLdu2JMgI)

Attention 모델

언어 모델 (Language Model, LM)

- Decoder의 hidden state가 attention weight 계산에 영향을 줌

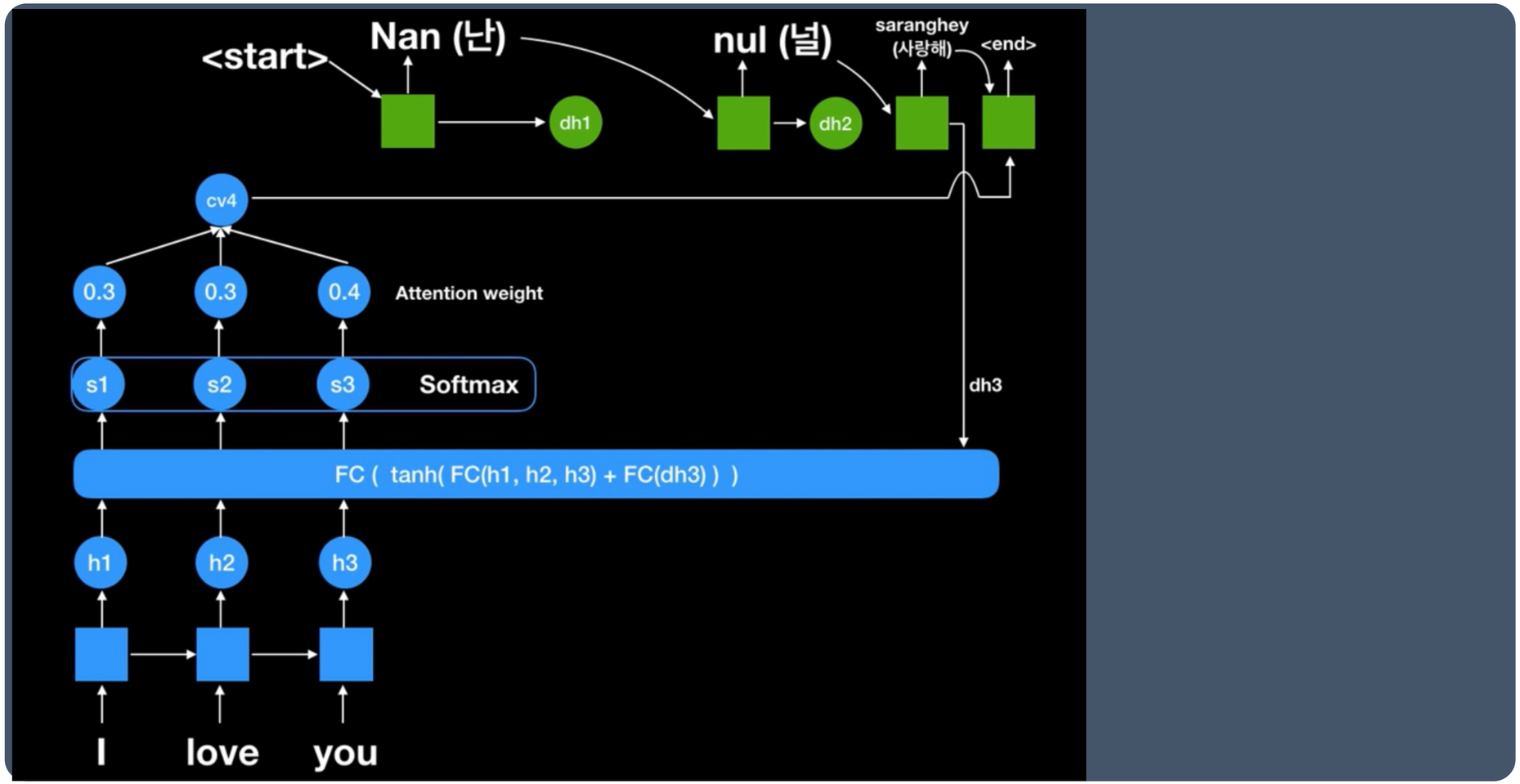


* [딥러닝 기계번역] 시퀀스 투 시퀀스 + 어텐션 모델
(https://www.youtube.com/watch?v=WsQLdu2JMgI)

Attention 모델

언어 모델 (Language Model, LM)

- Decoder의 hidden state가 attention weight 계산에 영향을 줌

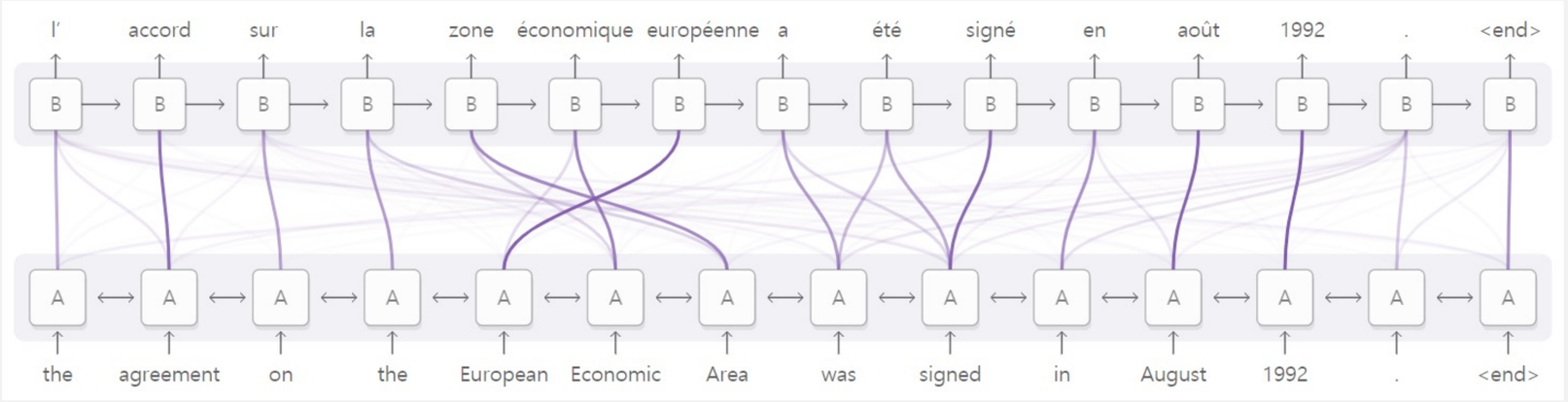


* [딥러닝 기계번역] 시퀀스 투 시퀀스 + 어텐션 모델
(https://www.youtube.com/watch?v=WsQLdu2JMgI)

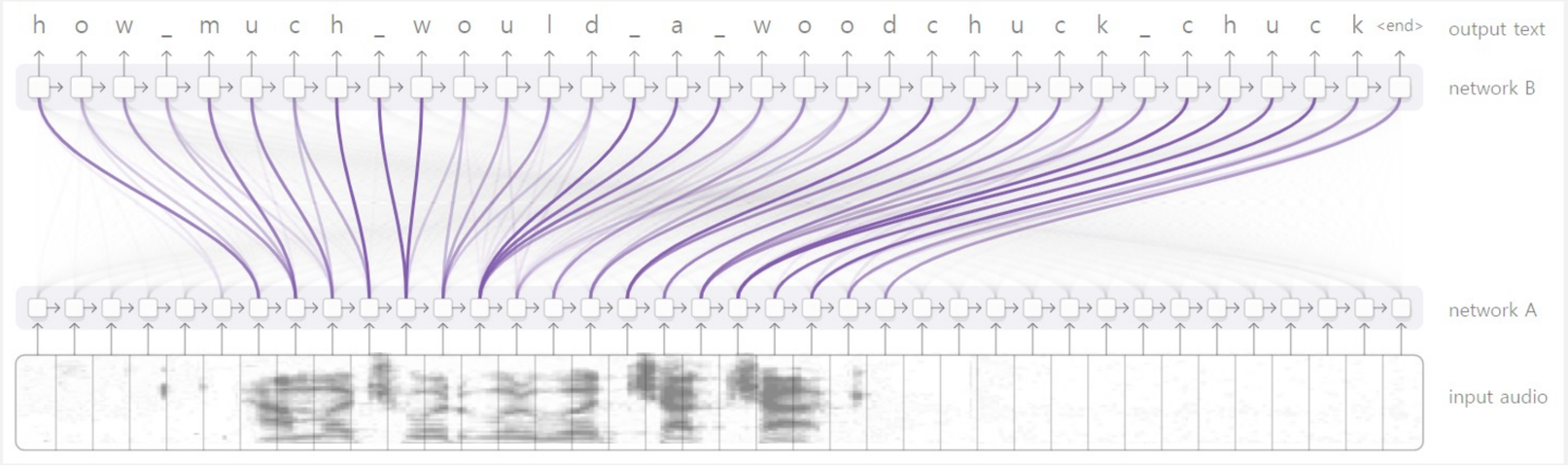
Attention 모델을 이용한 딥러닝의 시각화

언어 모델 (Language Model, LM)

- Attention for neural machine translation (NMT)



- Attention for speech to text (STT)

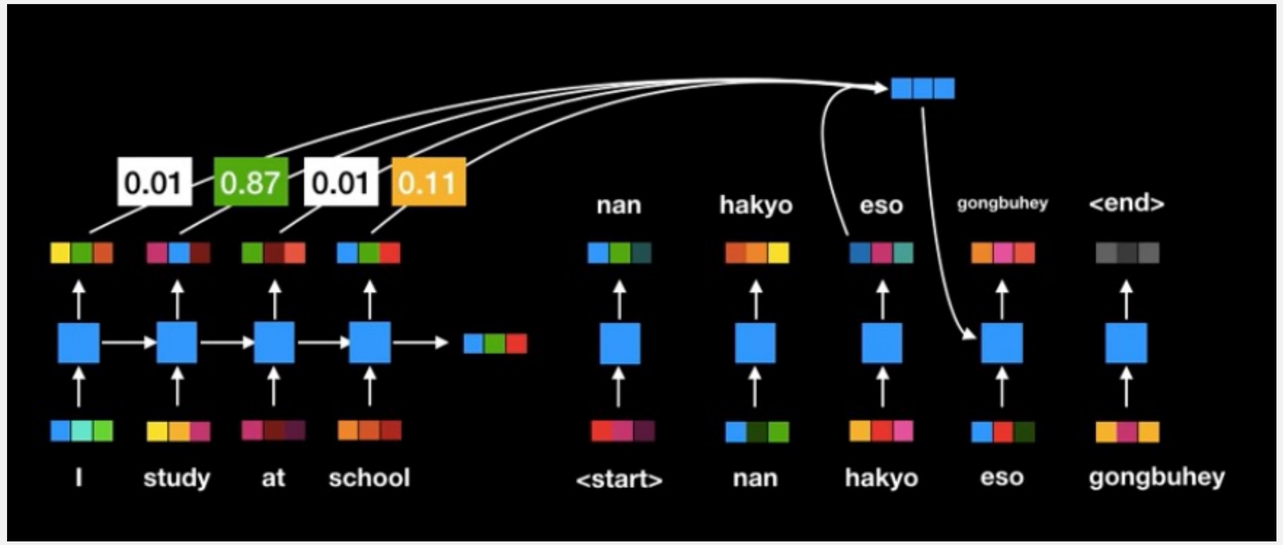


https://distill.pub/2016/augmented-rnns/#attentional-interfaces

Attention 모델

언어 모델 (Language Model, LM)

- 문맥에 따라 동적으로 할당되는 encode의 Attention weight로 인한 dynamic context vector를 획득
- 기존 Seq2Seq의 encoder, decoder 성능을 비약적으로 향상시킴

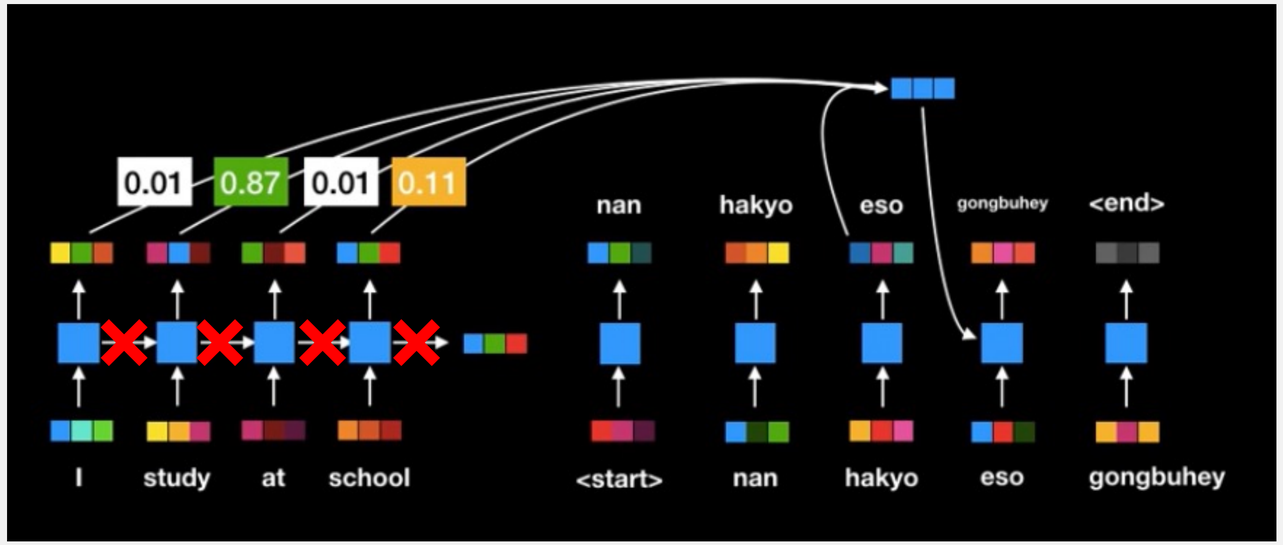


* 트랜스포머 (어텐션 이즈 올 유 니드)
(https://www.youtube.com/watch?v=mxGCEWOxfe8)

Attention 모델

언어 모델 (Language Model, LM)

- 문맥에 따라 동적으로 할당되는 encode의 Attention weight로 인한 dynamic context vector를 획득
- 기존 Seq2Seq의 encoder, decoder 성능을 비약적으로 향상시킴
- 하지만, 여전히 RNN이 순차적으로 연산이 이뤄짐에 따라 연산 속도가 느림



그냥 RNN을 없애는건 어떨까?

* 트랜스포머 (어텐션 이즈 올 유 니드)
(https://www.youtube.com/watch?v=mxGCEWOxfe8)

Transformer (Attention is all you need)

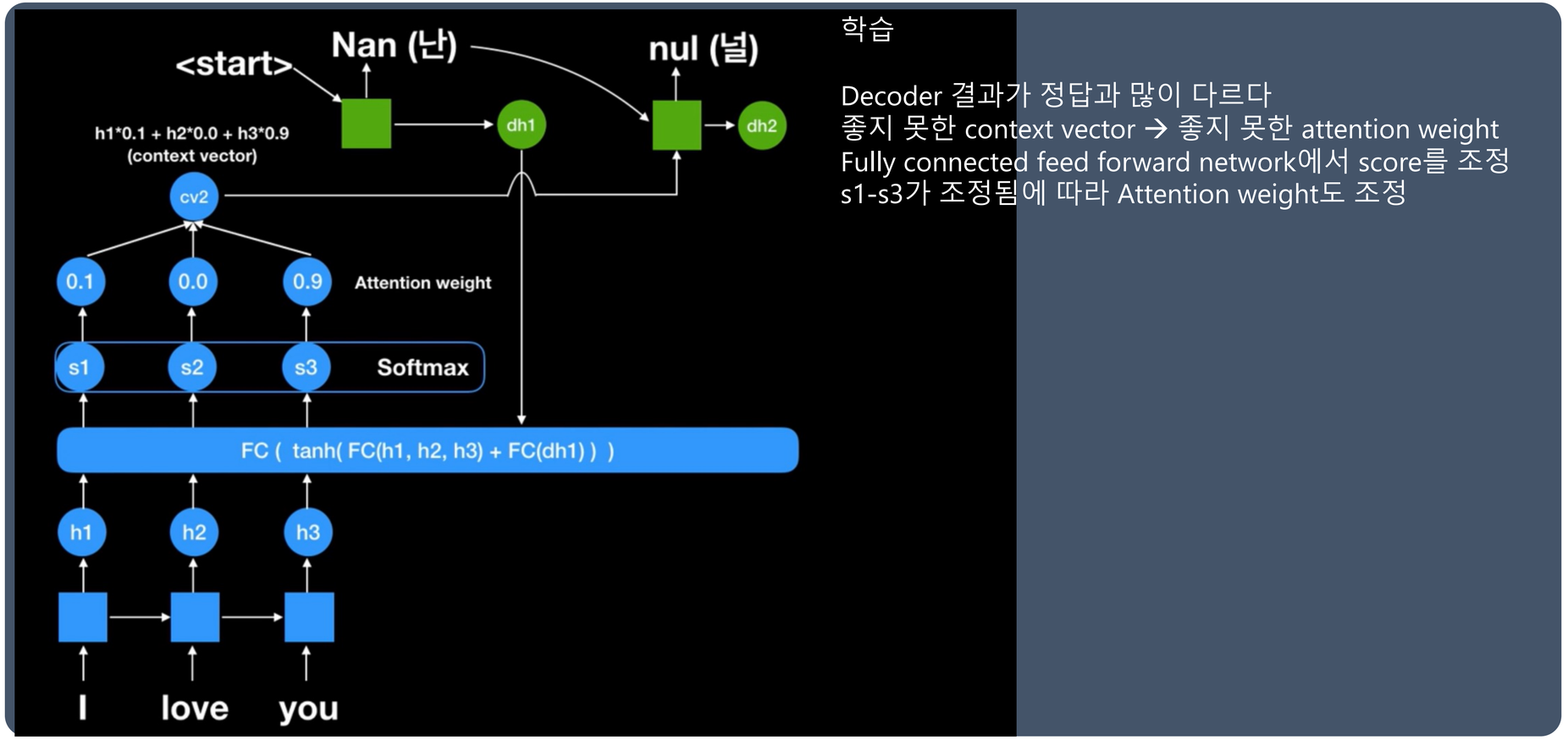
- Why self-attention
  - RNN 계열은 각 layer 별로 parameters 가 많습니다.
  - RNN 계열은 h_i 계산을 위하여 h_{i-1} 가 필요합니다. 순차적인 계산이 필요하므로 병렬화 (parallelization) 이 어렵습니다.
  - 여전히 LSTM 도 long dependency 는 잘 학습되지 않습니다.
    - 두 단어가 연결되기 위한 path 가 깁니다.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*(pp. 6000-6010).

Self-attention 모델

언어 모델 (Language Model, LM)

- Attention is all you need!
- RNN을 encoder와 decoder에서 제거



학습

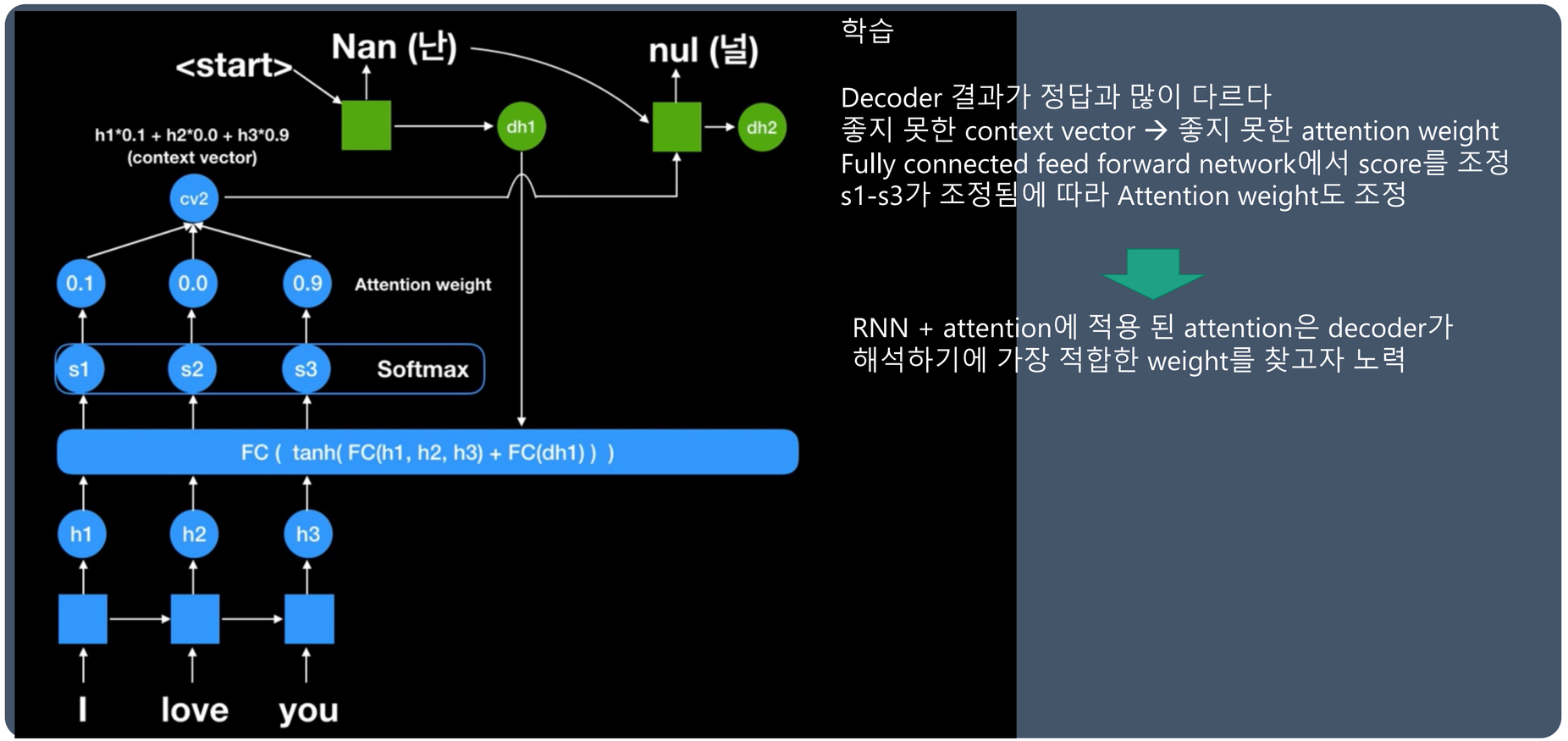
Decoder 결과가 정답과 많이 다르다
좋지 못한 context vector → 좋지 못한 attention weight
Fully connected feed forward network에서 score를 조정
s1-s3가 조정됨에 따라 Attention weight도 조정

* [딥러닝 기계번역] 시퀀스 투 시퀀스 + 어텐션 모델
(https://www.youtube.com/watch?v=WsQLdu2JMgI)

Self-attention 모델

언어 모델 (Language Model, LM)

- Attention is all you need!
- RNN을 encoder와 decoder에서 제거



학습

Decoder 결과가 정답과 많이 다르다
좋지 못한 context vector → 좋지 못한 attention weight
Fully connected feed forward network에서 score를 조정
s1-s3가 조정됨에 따라 Attention weight도 조정

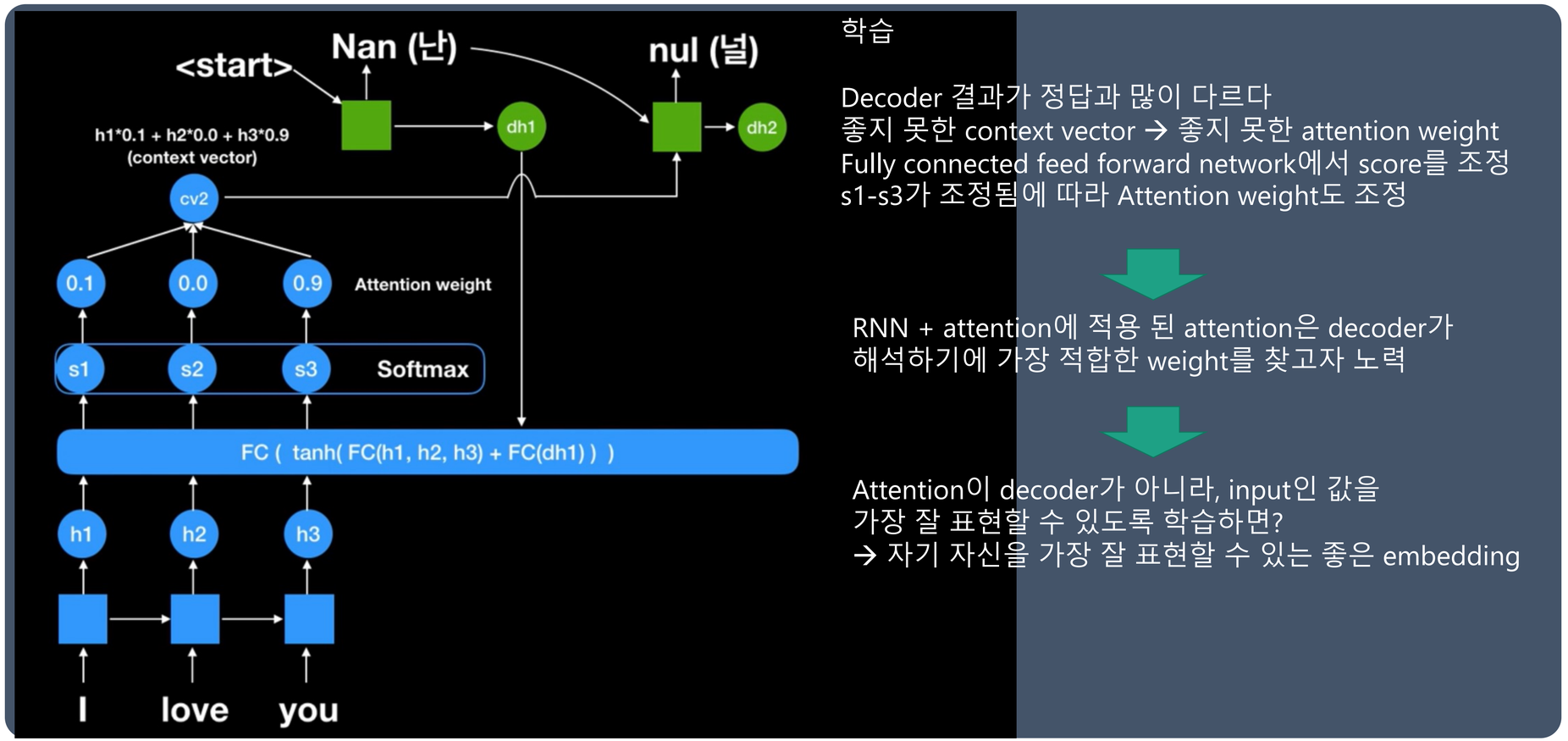
RNN + attention에 적용 된 attention은 decoder가
해석하기에 가장 적합한 weight를 찾고자 노력

* [딥러닝 기계번역] 시퀀스 투 시퀀스 + 어텐션 모델
(https://www.youtube.com/watch?v=WsQLdu2JMgI)

Self-attention 모델

언어 모델 (Language Model, LM)

- Attention is all you need!
- RNN을 encoder와 decoder에서 제거



학습

Decoder 결과가 정답과 많이 다르다
좋지 못한 context vector → 좋지 못한 attention weight
Fully connected feed forward network에서 score를 조정
s1-s3가 조정됨에 따라 Attention weight도 조정

RNN + attention에 적용 된 attention은 decoder가
해석하기에 가장 적합한 weight를 찾고자 노력

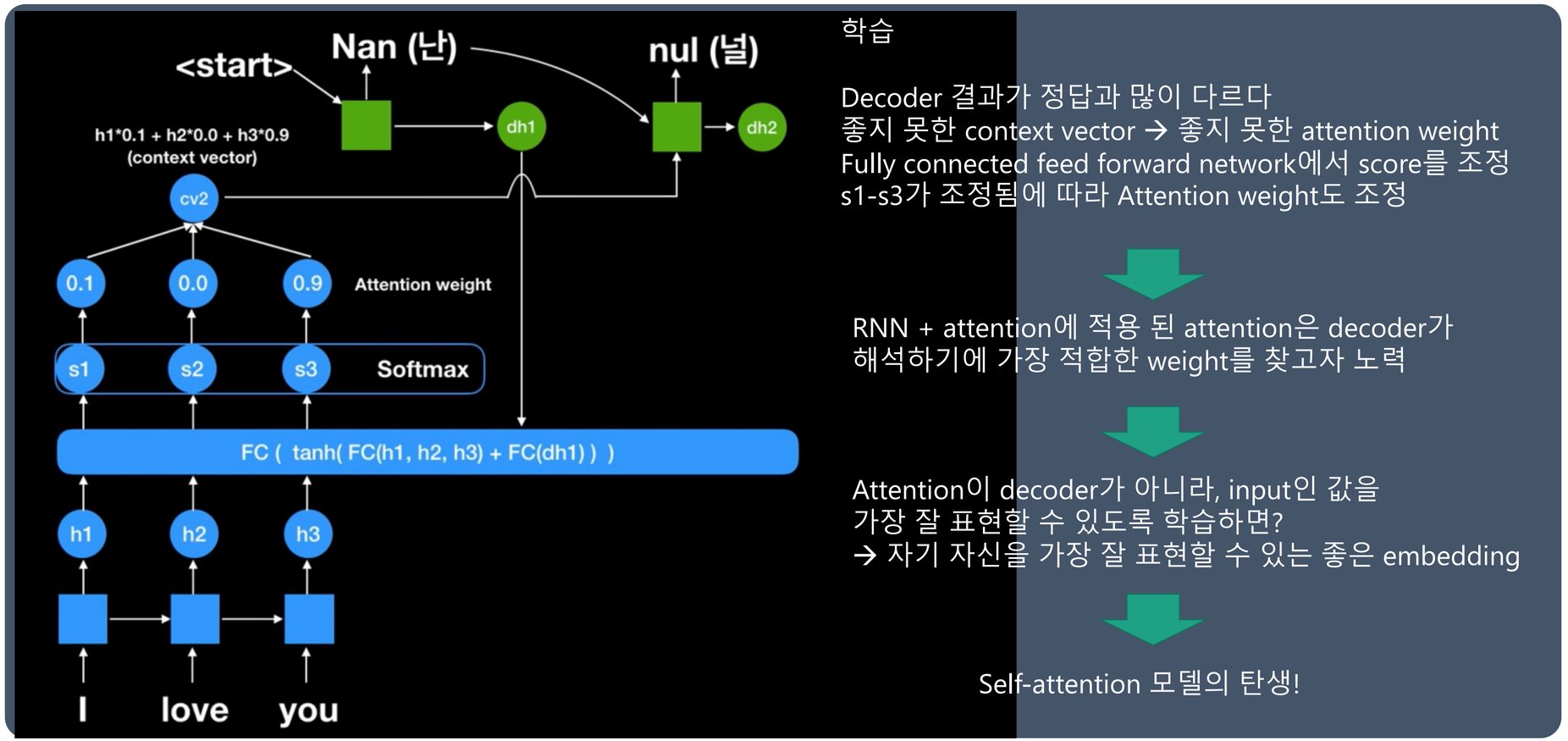
Attention이 decoder가 아니라, input인 값을
가장 잘 표현할 수 있도록 학습하면?
→ 자기 자신을 가장 잘 표현할 수 있는 좋은 embedding

* [딥러닝 기계번역] 시퀀스 투 시퀀스 + 어텐션 모델
(https://www.youtube.com/watch?v=WsQLdu2JMgI)

Self-attention 모델

언어 모델 (Language Model, LM)

- Attention is all you need!
- RNN을 encoder와 decoder에서 제거



학습

Decoder 결과가 정답과 많이 다르다
좋지 못한 context vector → 좋지 못한 attention weight
Fully connected feed forward network에서 score를 조정
s1-s3가 조정됨에 따라 Attention weight도 조정

RNN + attention에 적용 된 attention은 decoder가
해석하기에 가장 적합한 weight를 찾고자 노력

Attention이 decoder가 아니라, input인 값을
가장 잘 표현할 수 있도록 학습하면?
→ 자기 자신을 가장 잘 표현할 수 있는 좋은 embedding

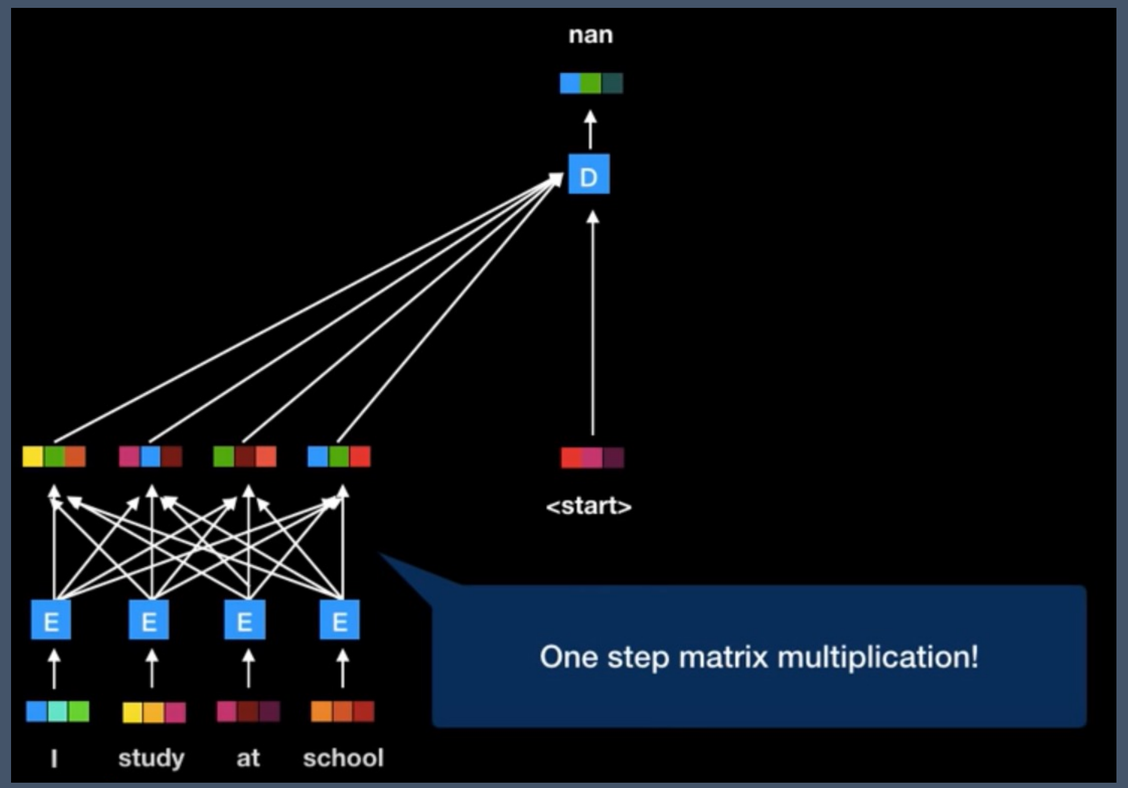
Self-attention 모델의 탄생!

* [딥러닝 기계번역] 시퀀스 투 시퀀스 + 어텐션 모델
(https://www.youtube.com/watch?v=WsQLdu2JMgI)

Self-attention 모델

언어 모델 (Language Model, LM)

- Attention is all you need!
- RNN을 encoder와 decoder에서 제거

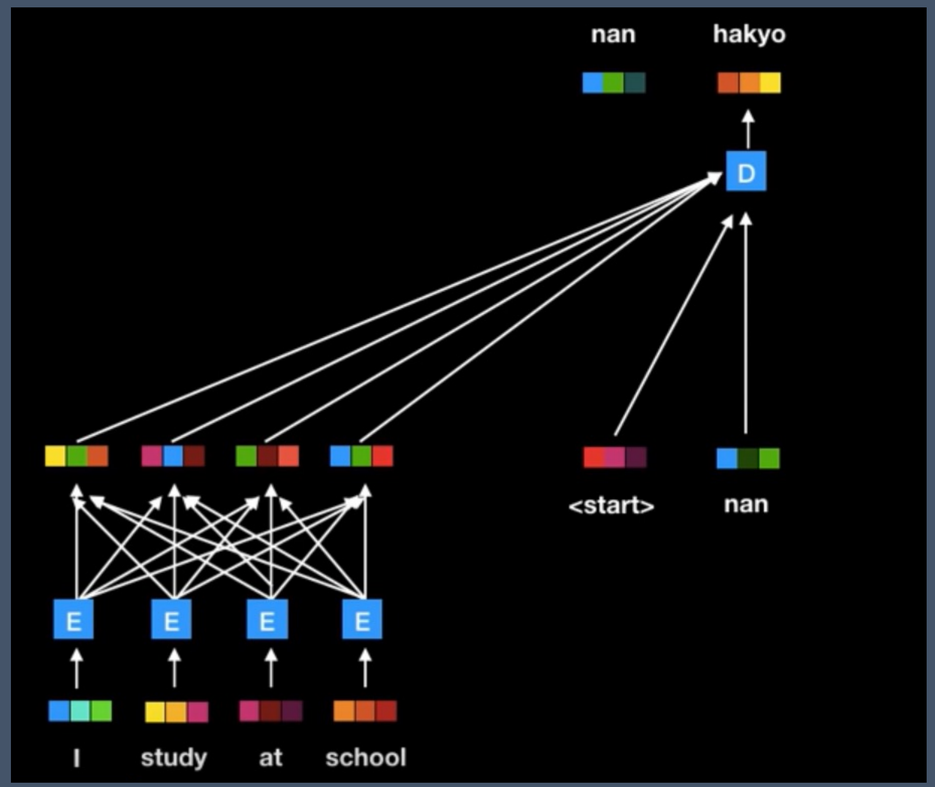


* [딥러닝 기계번역] 시퀀스 투 시퀀스 + 어텐션 모델
(https://www.youtube.com/watch?v=WsQLdu2JMgI)

Self-attention 모델

언어 모델 (Language Model, LM)

- Attention is all you need!
- RNN을 encoder와 decoder에서 제거

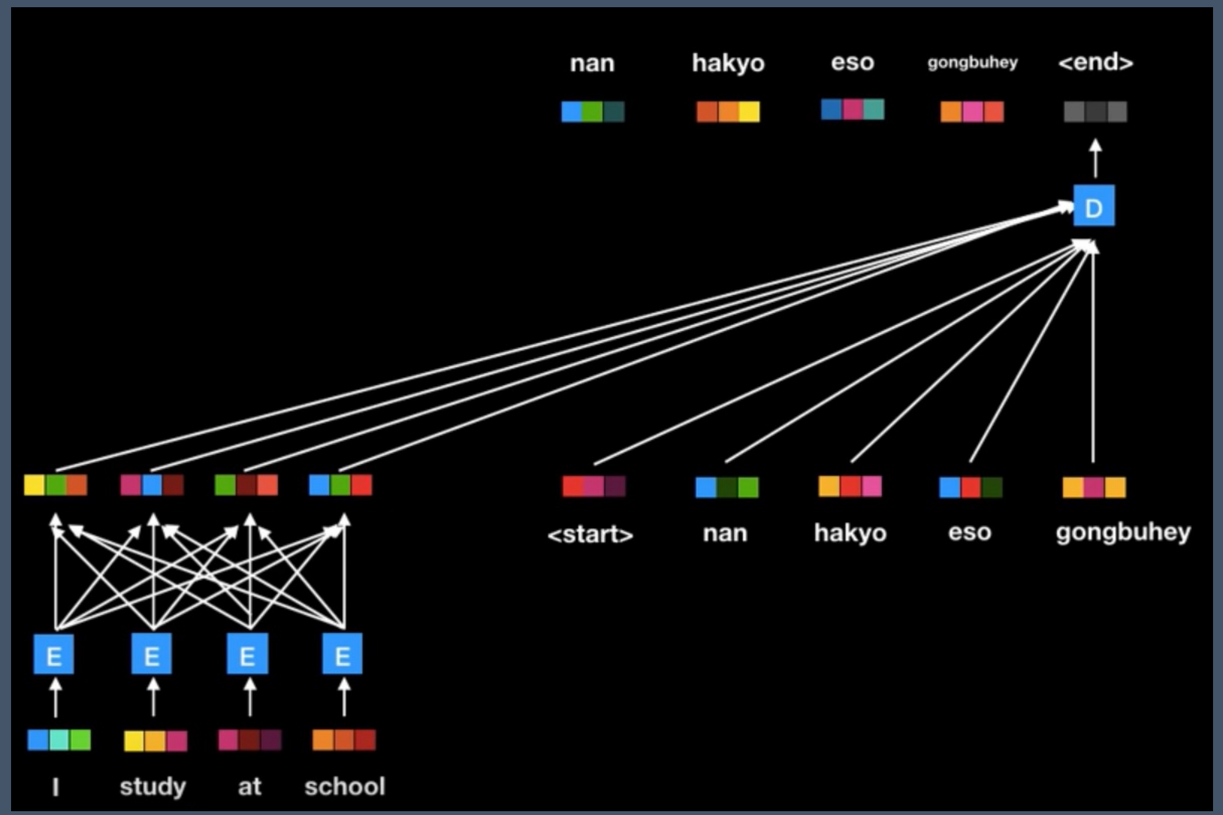


* [딥러닝 기계번역] 시퀀스 투 시퀀스 + 어텐션 모델
(https://www.youtube.com/watch?v=WsQLdu2JMgI)

Self-attention 모델

언어 모델 (Language Model, LM)

- Attention is all you need!
- RNN을 encoder와 decoder에서 제거

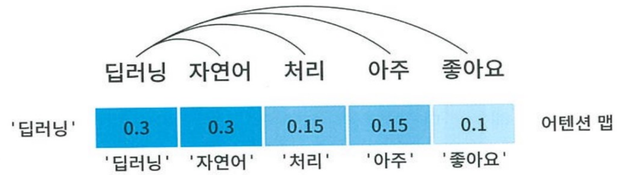


* [딥러닝 기계번역] 시퀀스 투 시퀀스 + 어텐션 모델
(https://www.youtube.com/watch?v=WsQLdu2JMgI)

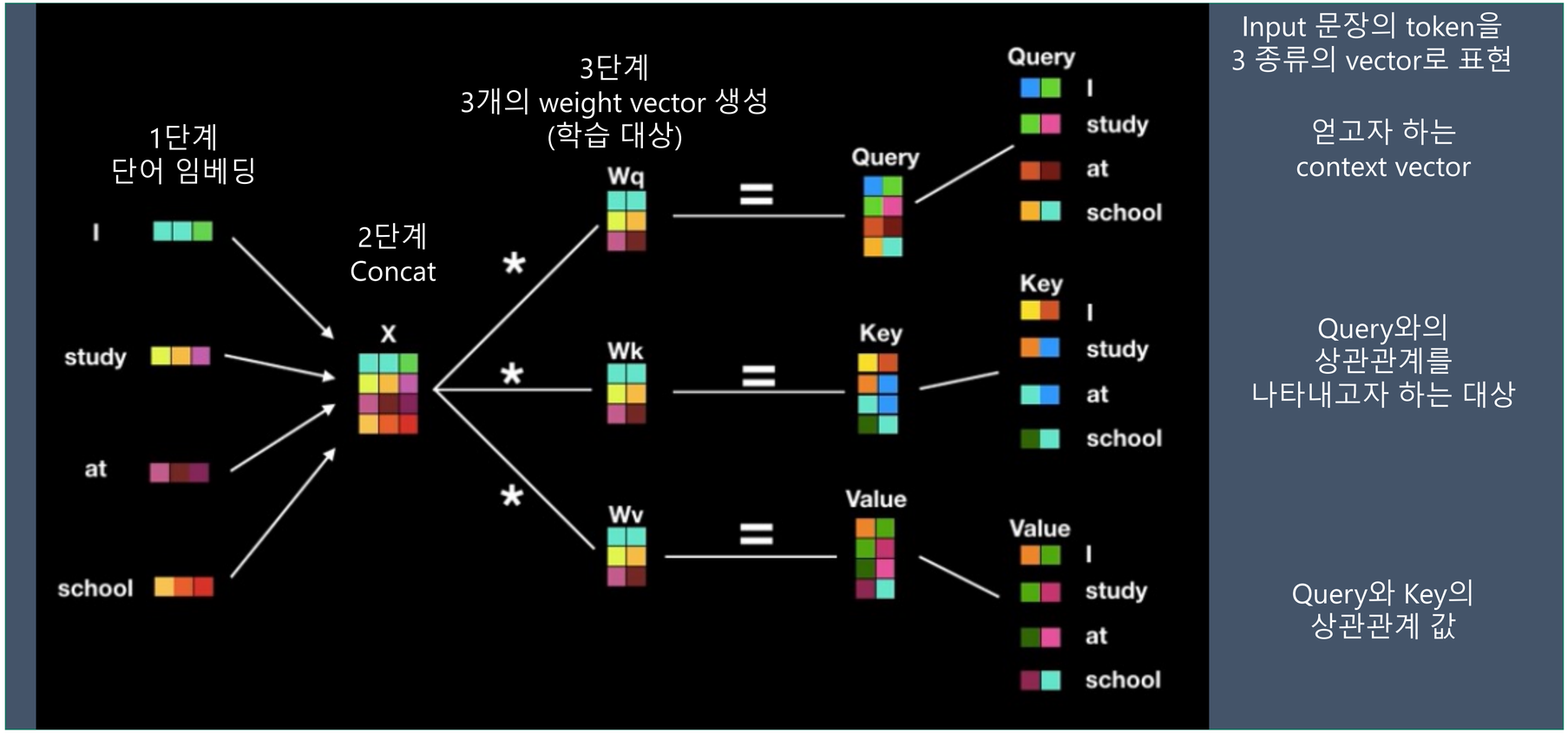
Self-attention 모델

언어 모델 (Language Model, LM)

목표



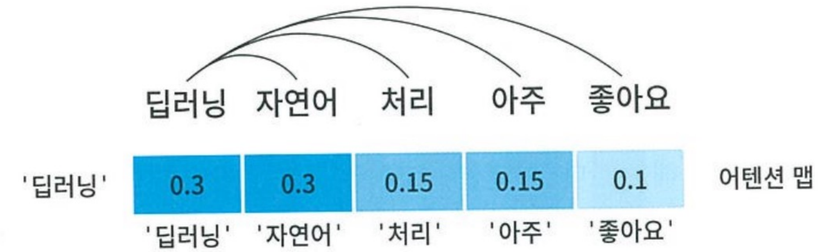
자기 자신을 잘 표현하는 attention을 구하자!



Self-attention 모델

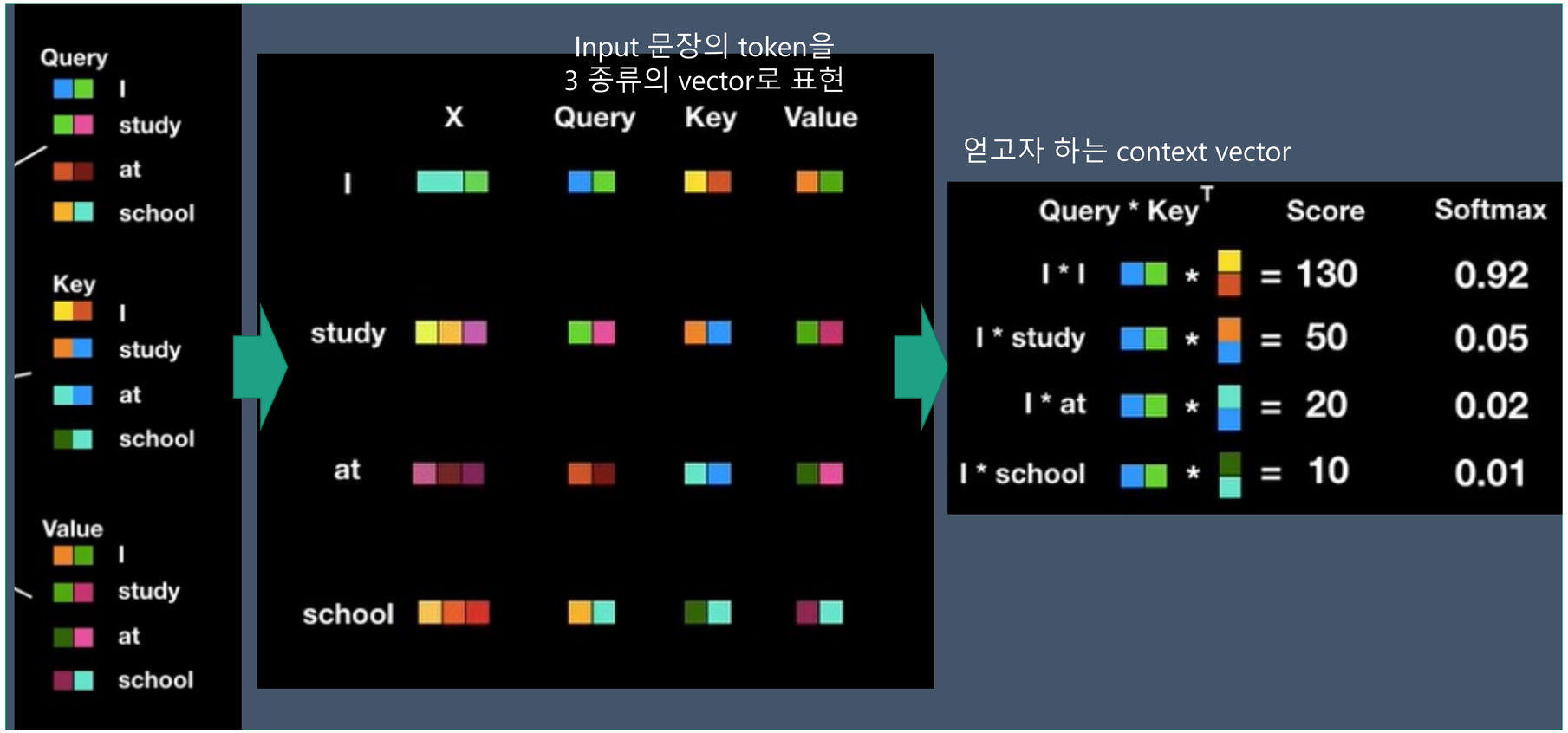
언어 모델 (Language Model, LM)

목표



어텐션 맵

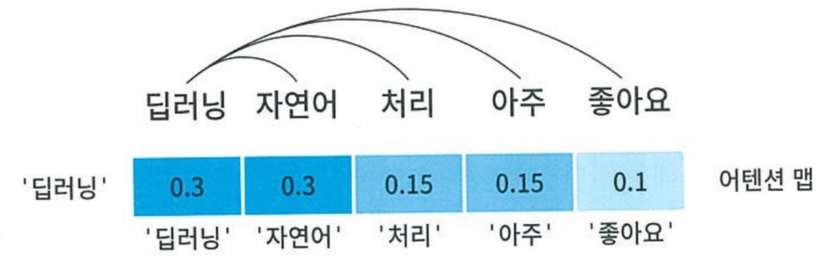
자기 자신을 잘 표현하는 attention을 구하자!



Self-attention 모델

언어 모델 (Language Model, LM)

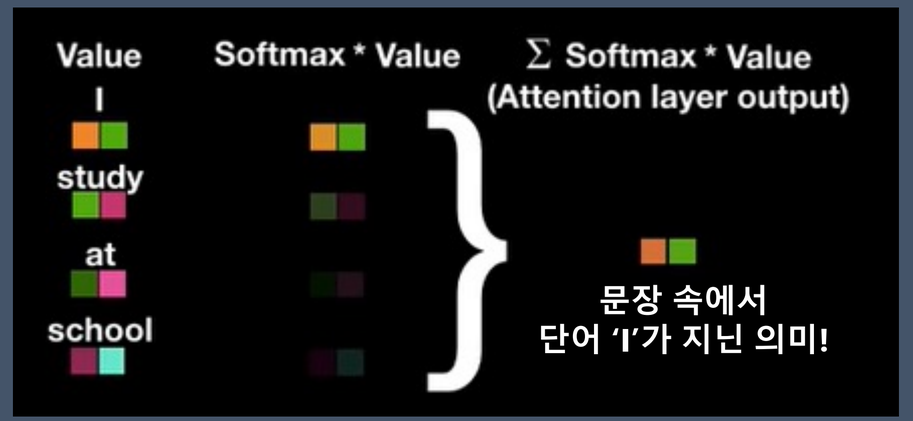
목표



어텐션 맵

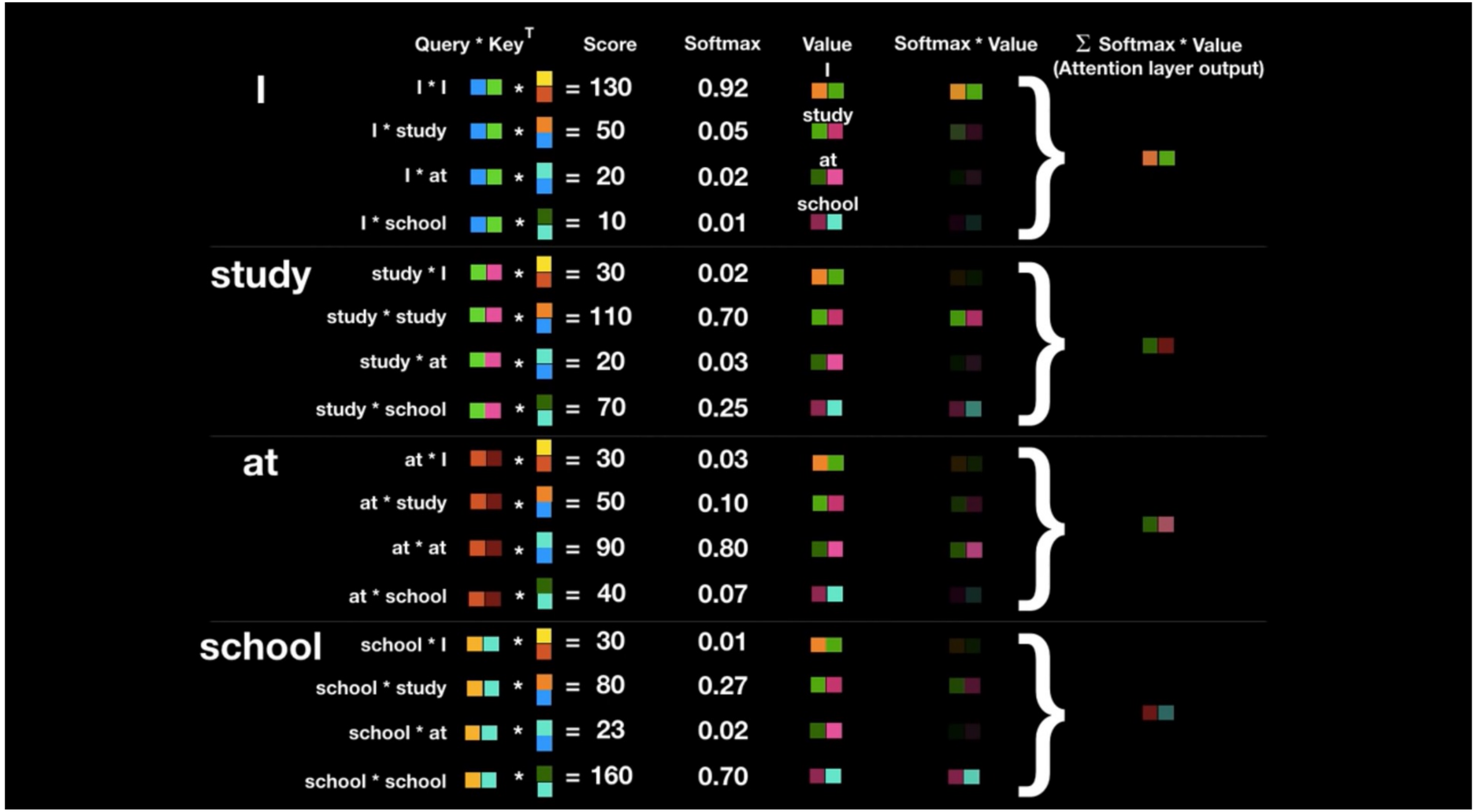
자기 자신을 잘 표현하는 attention을 구하자!

얻고자 하는 context vector



Self-attention 모델

언어 모델 (Language Model, LM)

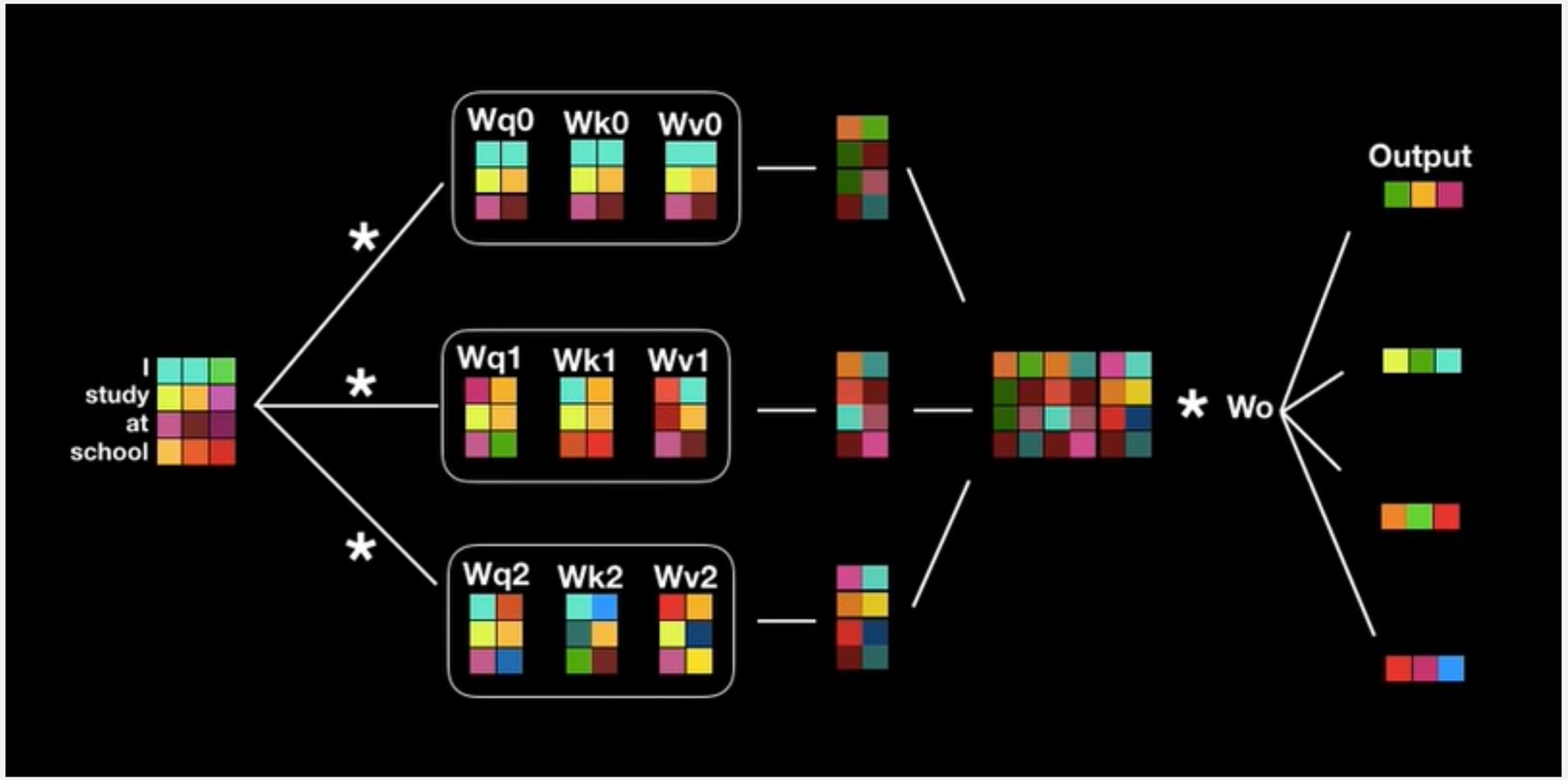


* 트랜스포머 (어텐션 이즈 올 유 니드)
(https://www.youtube.com/watch?v=mxGCEWOxfe8)

Multi-head Self Attention 모델

언어 모델 (Language Model, LM)

- Query, Key, Value로 구성된 attention layer를 동시에 여러 개를 수행

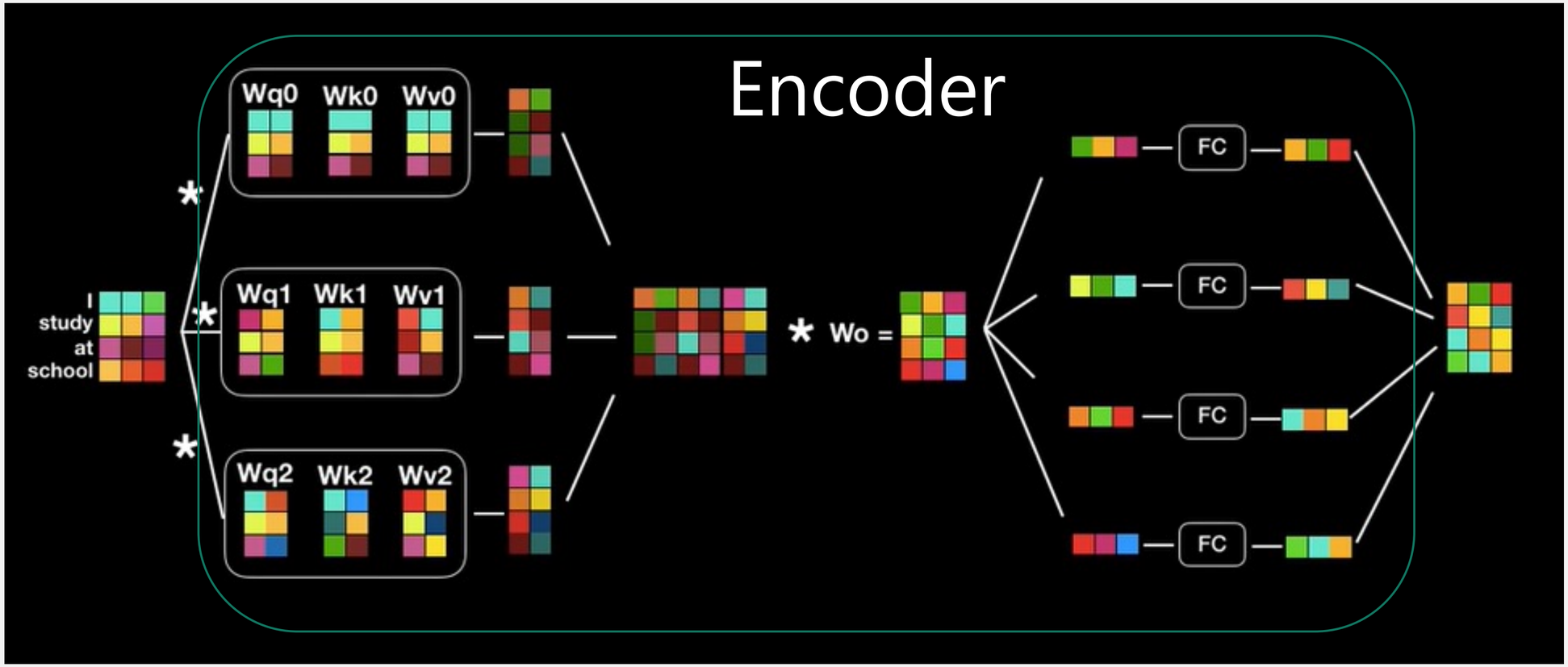


* 트랜스포머 (어텐션 이즈 올 유 니드)
(https://www.youtube.com/watch?v=mxGCEWOxfe8)

Multi-head Self Attention 모델

언어 모델 (Language Model, LM)

- Query, Key, Value로 구성된 attention layer를 동시에 여러 개를 수행
- 최종적으로 자기 자신을 표현하는 vector 획득

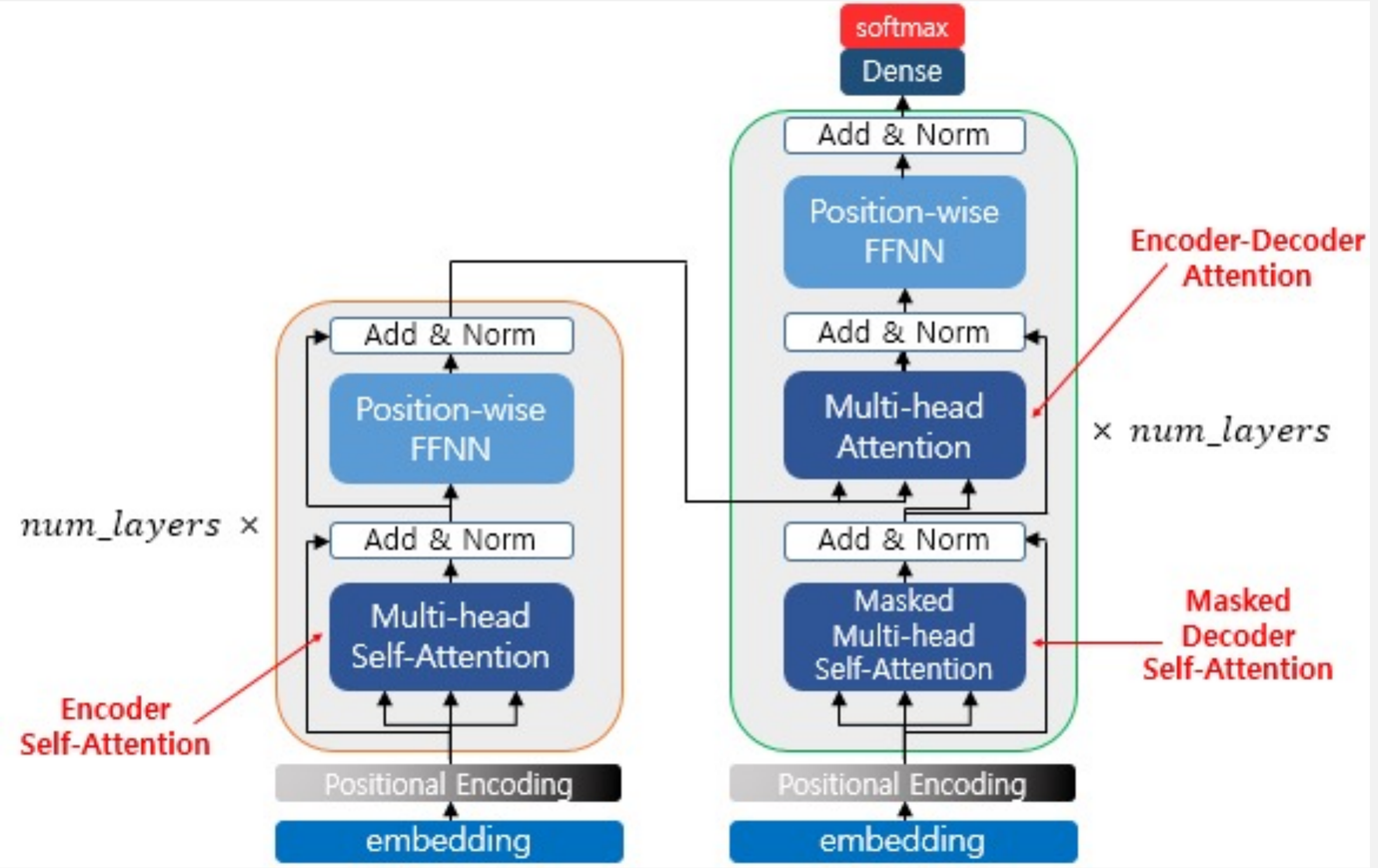


* 트랜스포머 (어텐션 이즈 올 유 니드)
(https://www.youtube.com/watch?v=mxGCEWOxfe8)

Transformer 모델

언어 모델 (Language Model, LM)

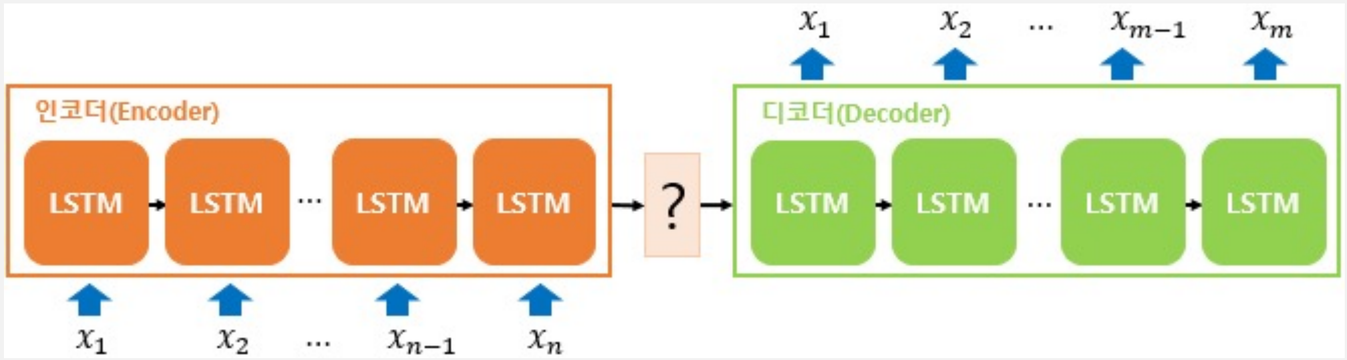
- 구글 연구 팀이 공개한 딥러닝 아키텍쳐로 뛰어난 성능으로 주목 받았음
- GPT, BERT 등의 모델의 기본 모델로 활용되고 있음



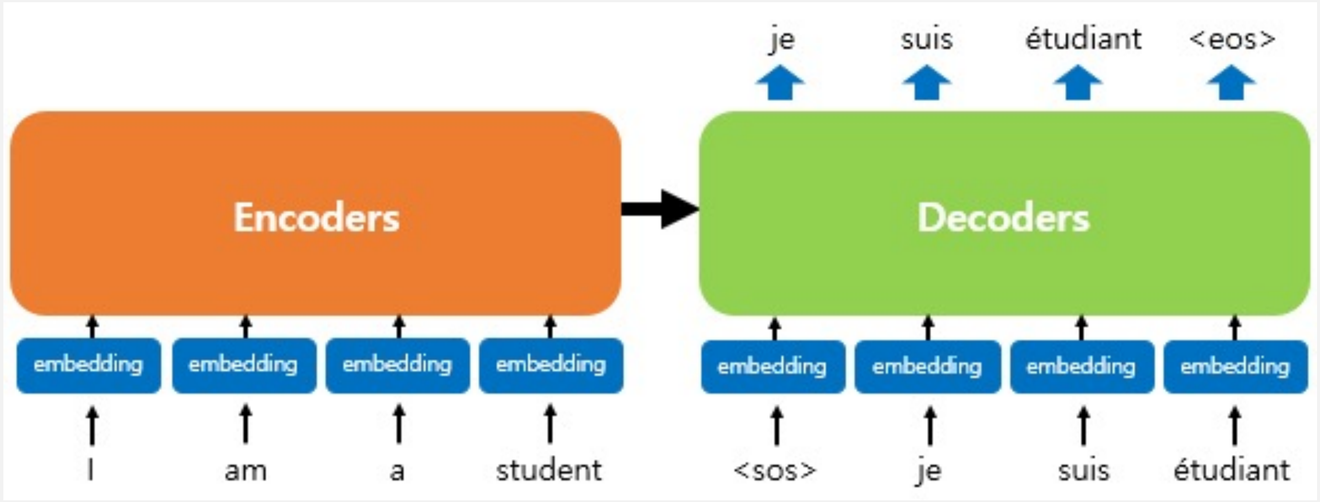
Transformer 모델

언어 모델 (Language Model, LM)

Seq2Seq model



Transformer network

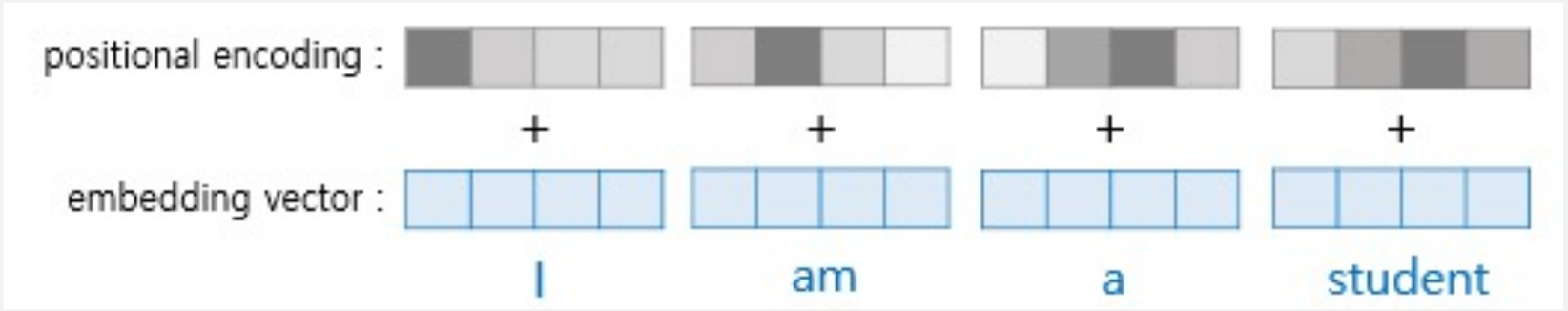


Transformer 모델

언어 모델 (Language Model, LM)

Positional Encoding

- 트랜스포머는 단어 입력을 순차적으로 받는 방식이 아니므로,
  단어의 위치 정보를 다른 방식으로 알려줄 필요가 있음
- 각 단어의 임베딩 벡터에 위치 정보들을 더하여 모델 입력으로 사용하며, 이를 포지셔널 인코딩이라 함
- 포지셔널 인코딩에는 아래와 같은 함수를 활용하여 위치 정보를 생성



Transformer 모델

언어 모델 (Language Model, LM)

셀프 어텐션(Self Attention)

- 셀프 어텐션은 쿼리, 키, 벨류가 모두 같은 경우

멀티헤드 어텐션(Multi-head Attention)

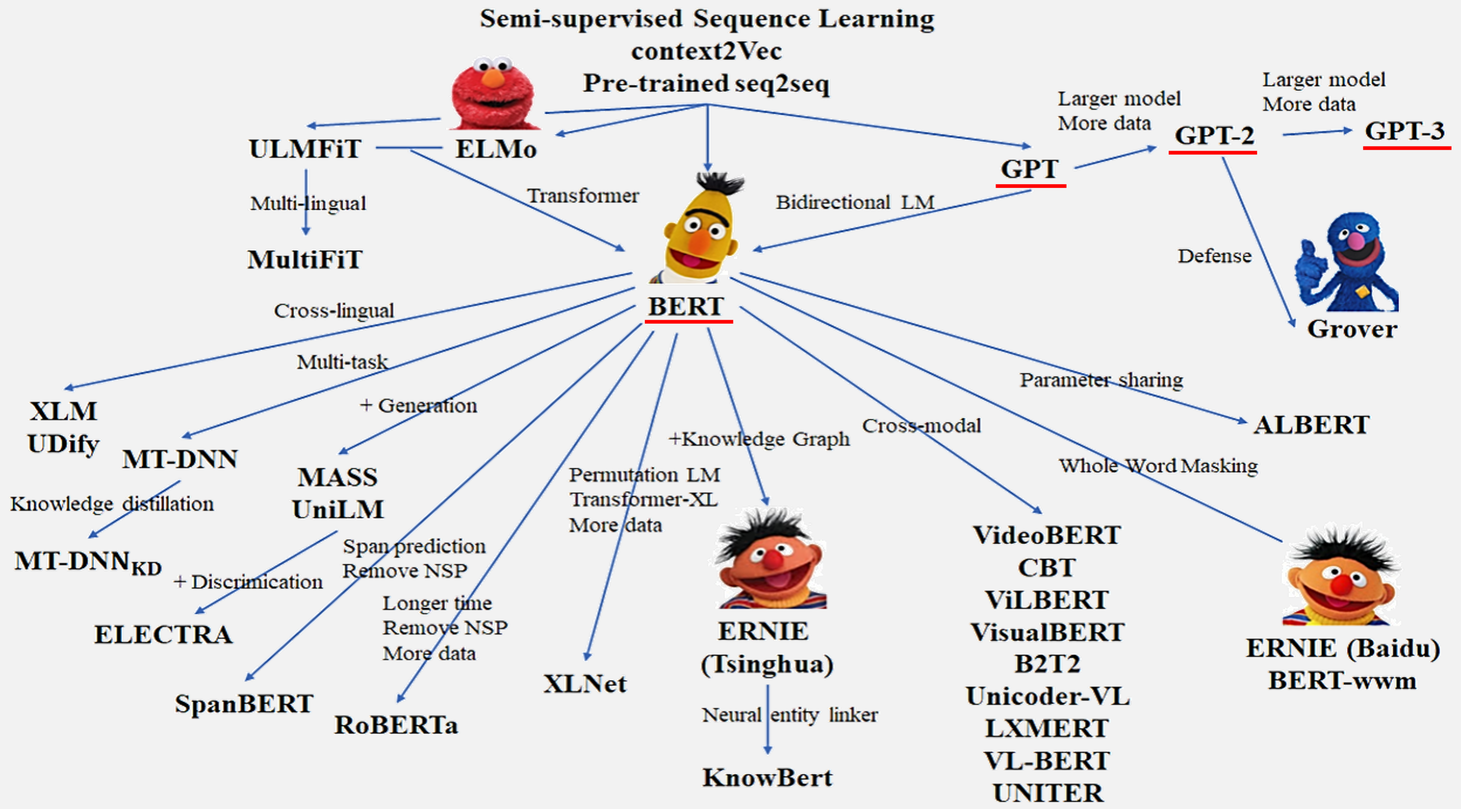
- multi-head는 어텐션을 head 개수만큼 병렬로 수행하는 방법

트랜스포머의 어텐션

- 트랜스포머에는 3가지의 어텐션이 활용됨
- Encoder Self-Attention: Query = Key = Value
- Masked Decoder Self-Attention: Query = Key = Value
- Encoder-Decoder Attention: Query=디코더 벡터, Key=Value=인코더 벡터

Transformer Revolution!

언어 모델 (Language Model, LM)



Transformer Revolution!

언어 모델 (Language Model, LM)

Dual-Transformer

Length-Adaptive Transformer

Markov Transformer

Linformer

Transformer

Sparse Transformer

Reformer

Highway Transformer

Cascade Transformer

Longformer

Turbo Transformer

CrossTransformer

Fast Transformer

Attention Free-Transformer

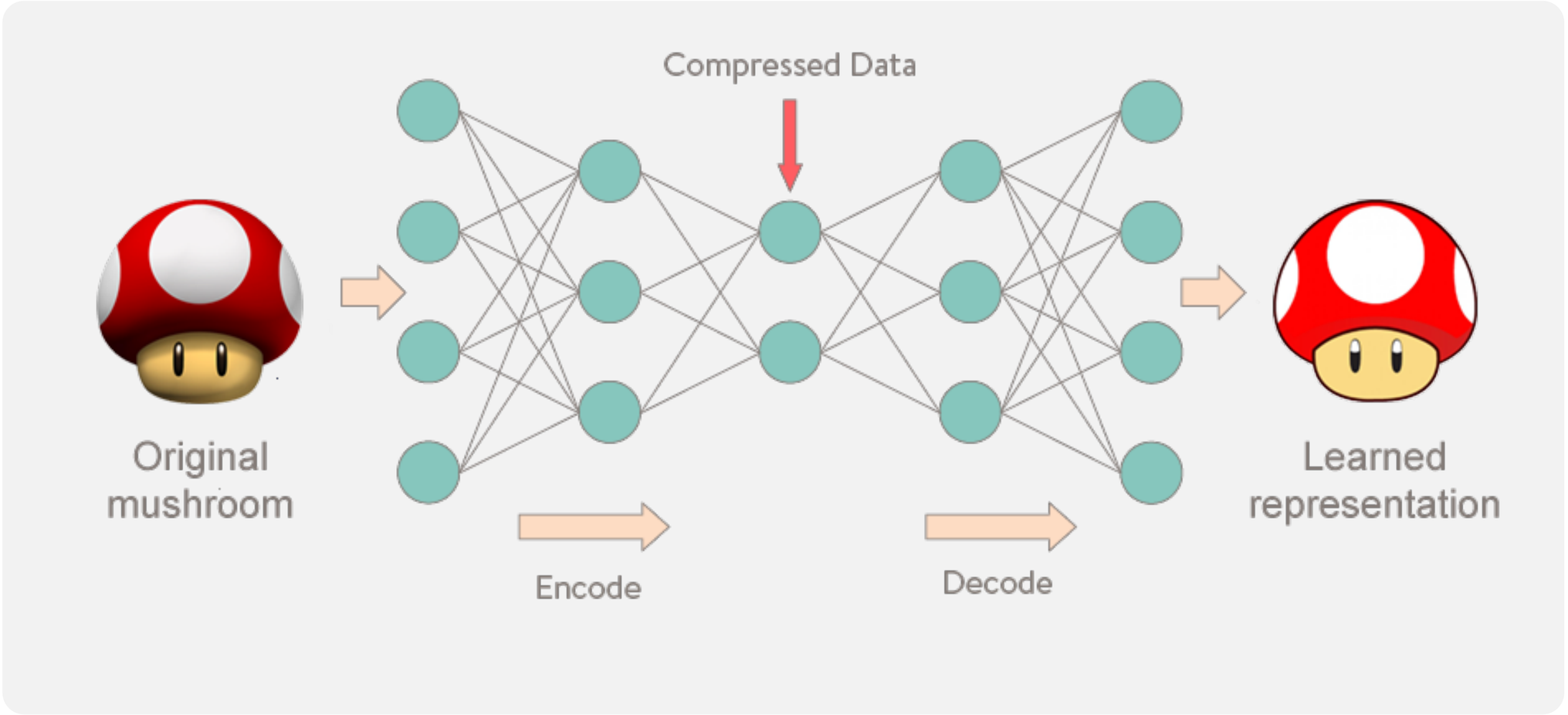
Transformer-xl

Markov Transformer

Funnel-Transformer

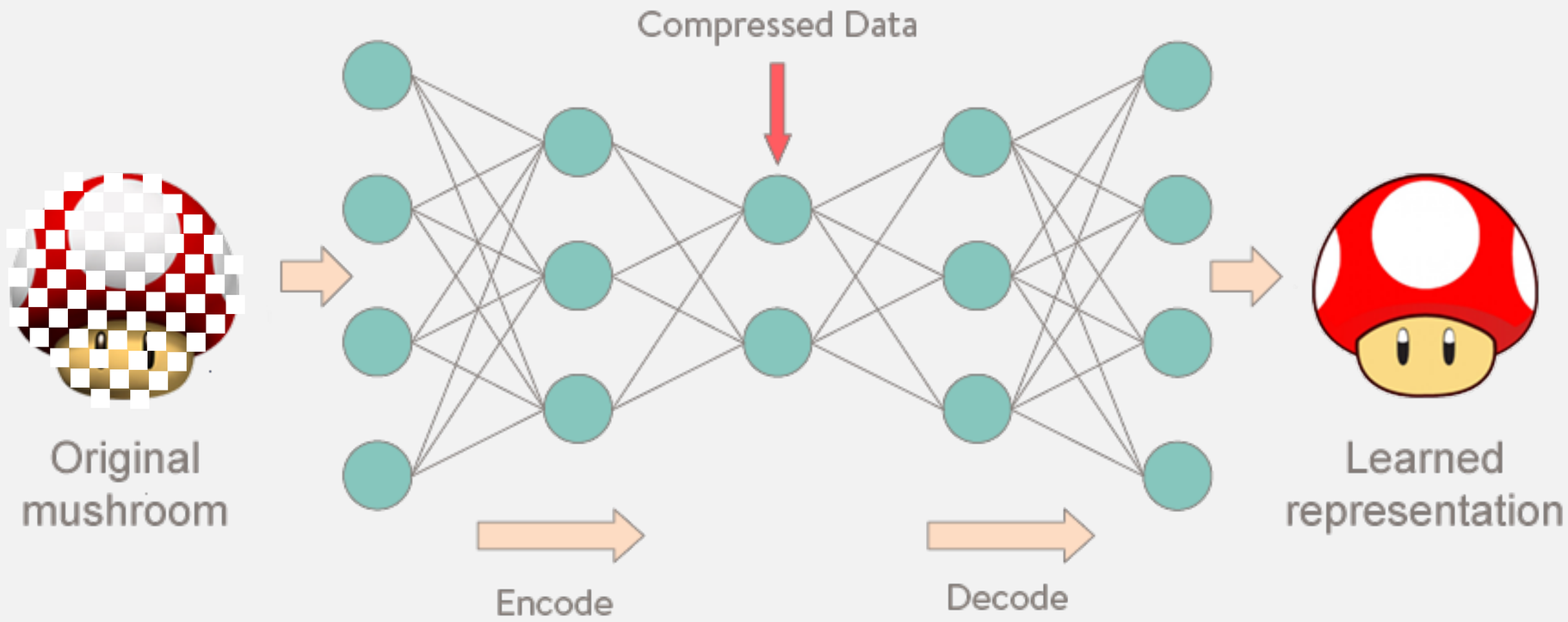
Autoencoder

언어 모델 (Language Model, LM)



Denoising Autoencoder (BERT 계열)

언어 모델 (Language Model, LM)



Autoregressive (GPT 계열)

언어 모델 (Language Model, LM)