

BERT 모델

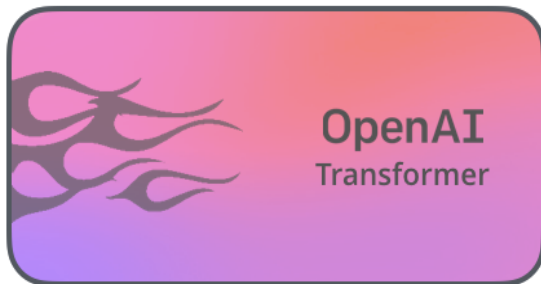
BERT의 탄생에 많은 사전연구들이 공헌했다.

*Dr. Rhee
Feb 2020*

BERT(Bidirectional Encoder Representation from Transformer)의 탄생

BERT의 탄생

- NLP의 새로운 시대의 서막: NLP의 ImageNet moment
 - BERT의 탄생에 많은 사전 연구들에 기여했다.
 - Open Source



BERT의 성과

- BERT의 Leaderboard

SQuAD v1.1 dataset leaderboard

Rank	Model	EM	F1
	Human Performance <i>Stanford University</i> (Rajpurkar et al. '16)	82.304	91.221
1 Oct 05, 2018	BERT (ensemble) <i>Google AI Language</i> https://arxiv.org/abs/1810.04805	87.433	93.160
2 Oct 05, 2018	BERT (single model) <i>Google AI Language</i> https://arxiv.org/abs/1810.04805	85.083	91.835

BERT의 탄생

- BERT의 2스텝 절차
스텝1: 다운로드 사전학습 모델 (trained on un-annotated data)
스텝2; 모델 Fine-Tuning

1 - **Semi-supervised** training on large amounts of text (books, wikipedia..etc).

The model is trained on a certain task that enables it to grasp patterns in language. By the end of the training process, BERT has language-processing abilities capable of empowering many models we later need to build and train in a supervised way.

Semi-supervised Learning Step

Model:



Dataset:



Objective:

Predict the masked word
(language modeling)

2 - **Supervised** training on a specific task with a labeled dataset.

Supervised Learning Step

Classifier

75% Spam
25% Not Spam

Model:
(pre-trained
in step #1)

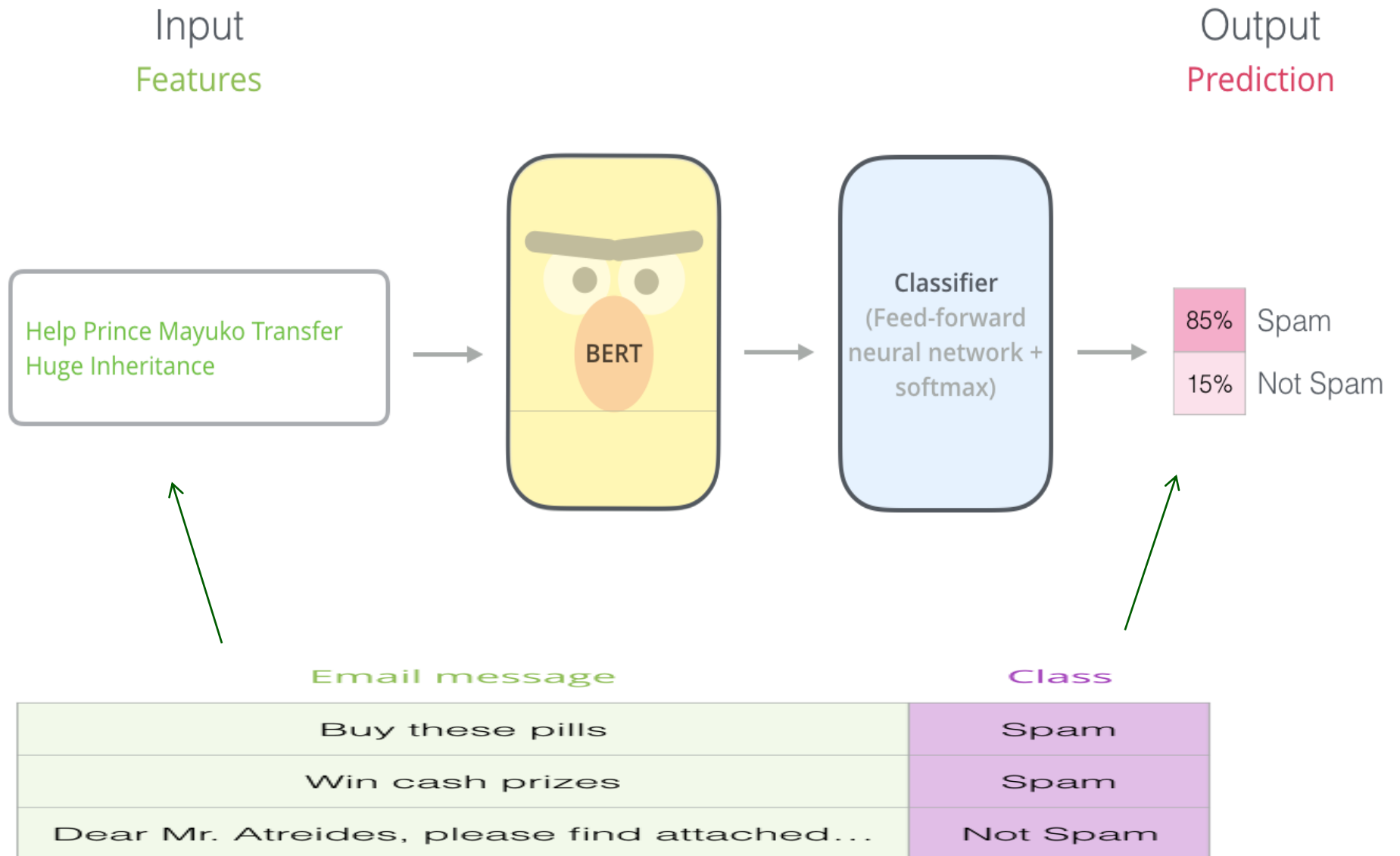


Dataset:

Email message	Class
Buy these pills	Spam
Win cash prizes	Spam
Dear Mr. Atreides, please find attached...	Not Spam

BERT의 응용

- 문장 분류예



BERT의 응용 분야

Sentiment analysis

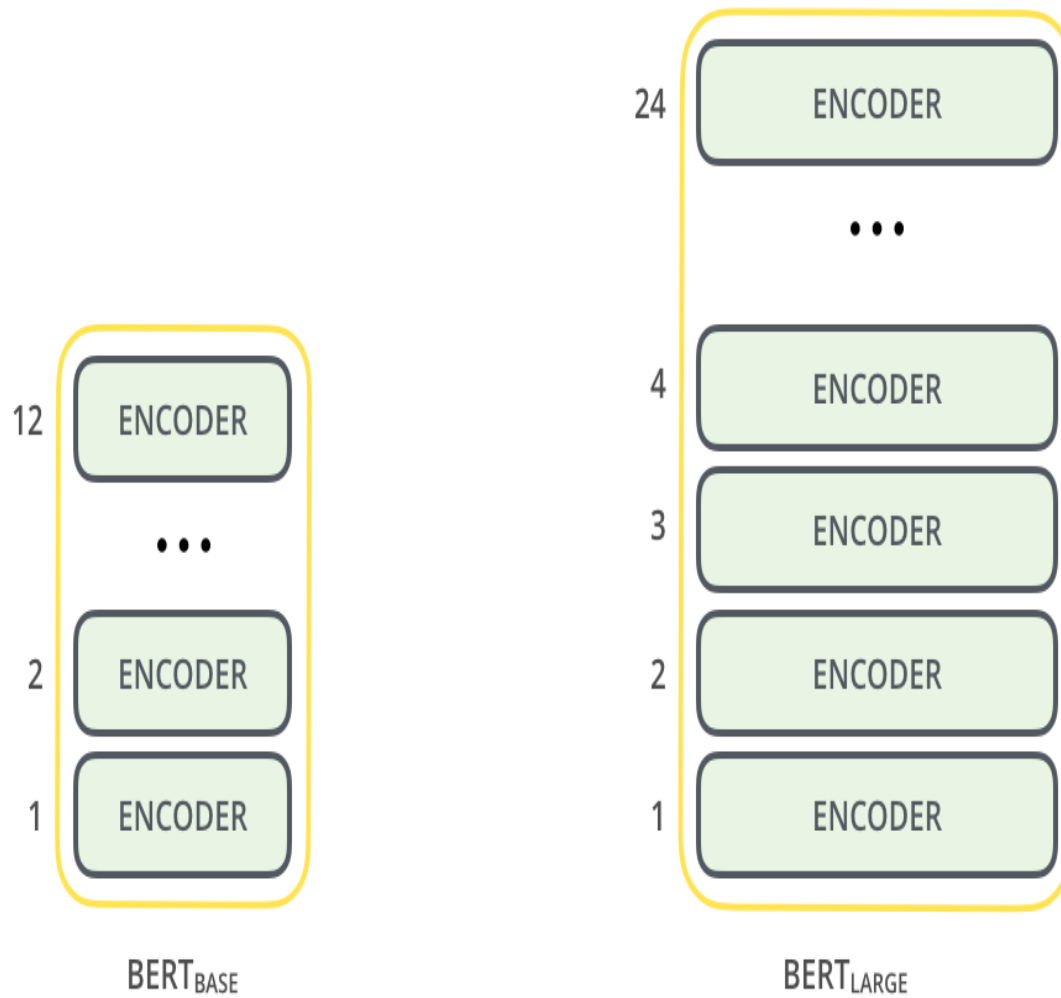
- Input: Movie/Product review. Output: is the review positive or negative?
- Example dataset: [SST](#)

Fact-checking

- Input: sentence. Output: “Claim” or “Not Claim”
- More ambitious/futuristic example:
 - Input: Claim sentence. Output: “True” or “False”
- [Full Fact](#) is an organization building automatic fact-checking tools for the benefit of the public. Part of their pipeline is a classifier that reads news articles and detects claims (classifies text as either “claim” or “not claim”) which can later be fact-checked (by humans now, by with ML later, hopefully).
- Video: [Sentence embeddings for automated factchecking - Lev Konstantinovskiy](#)

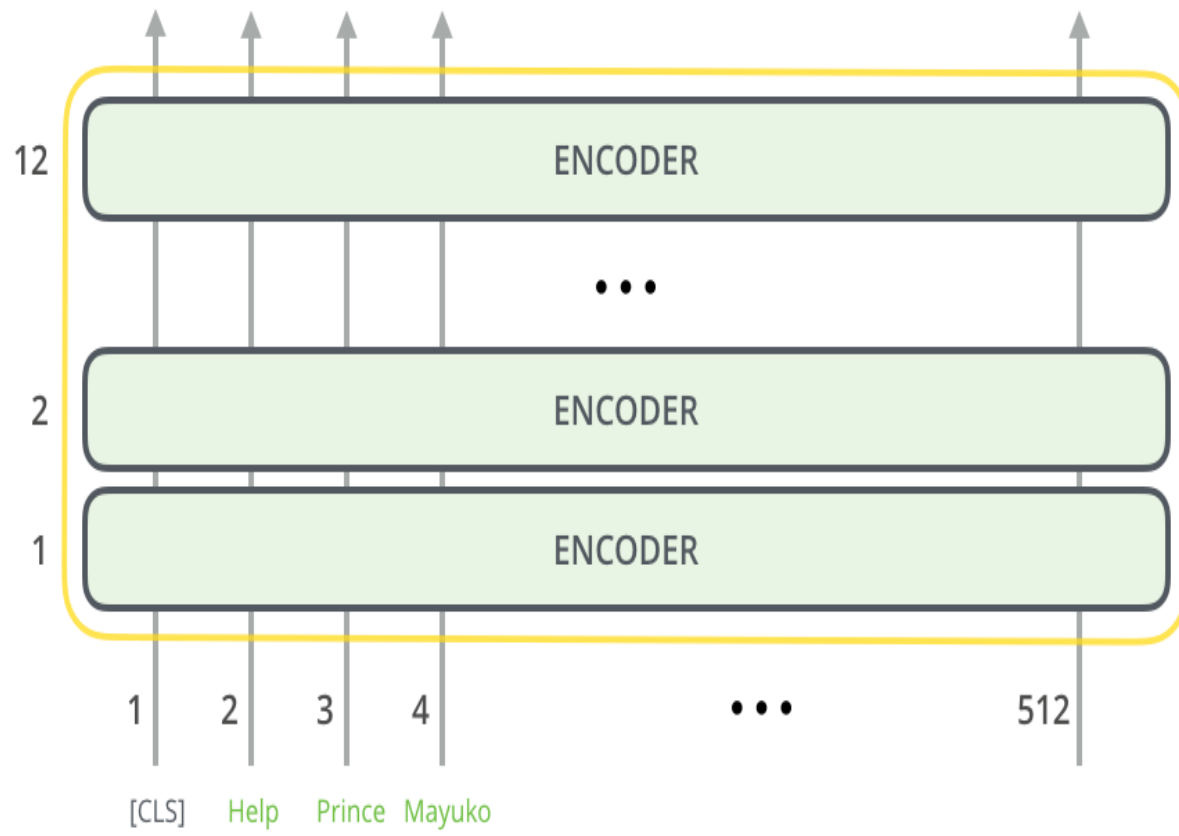
BERT 구조

- BERT의 종류.



BERT 구조

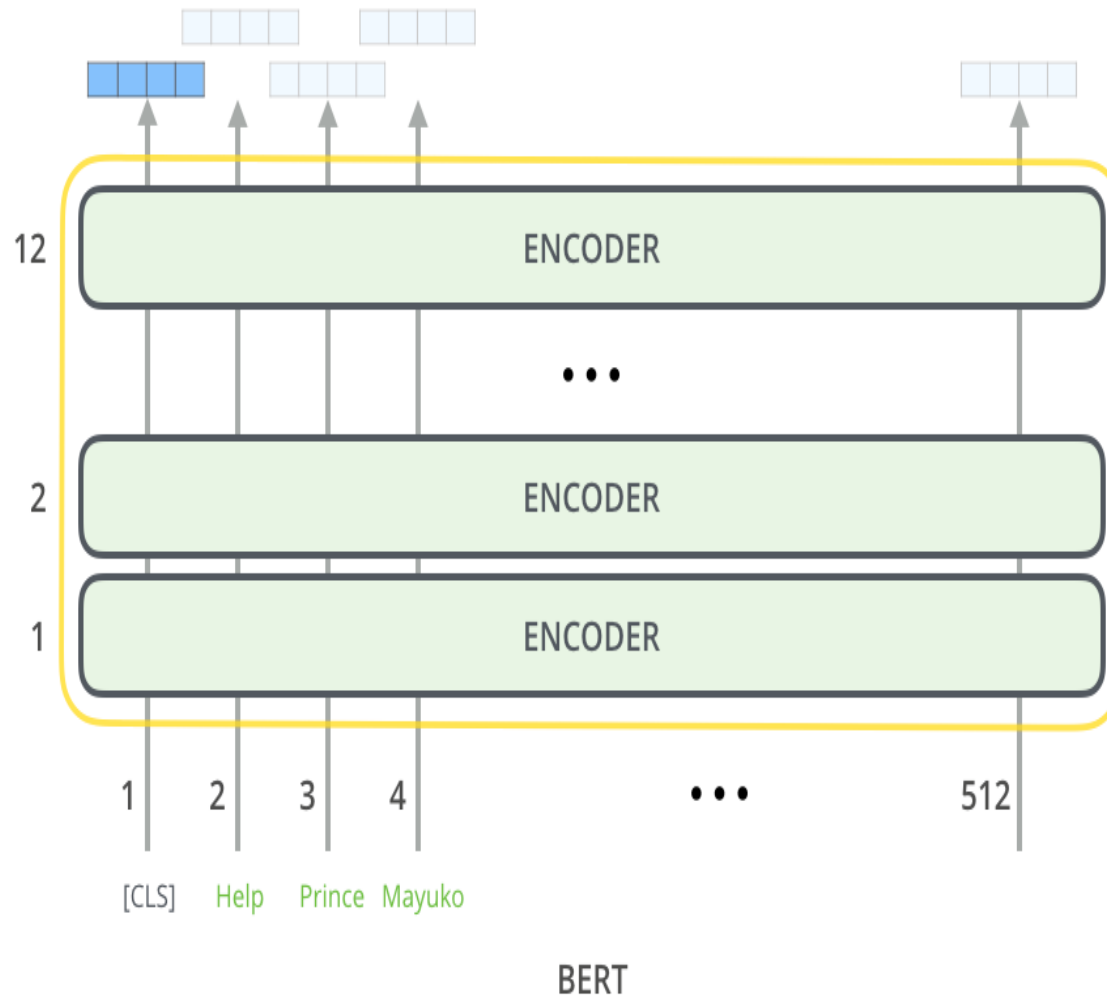
- BERT 모델 입력
 - 첫째 입력 토큰은 특수 토큰 [CLS]로 공급된다. CLS는 Classification의 약자이다.



BERT

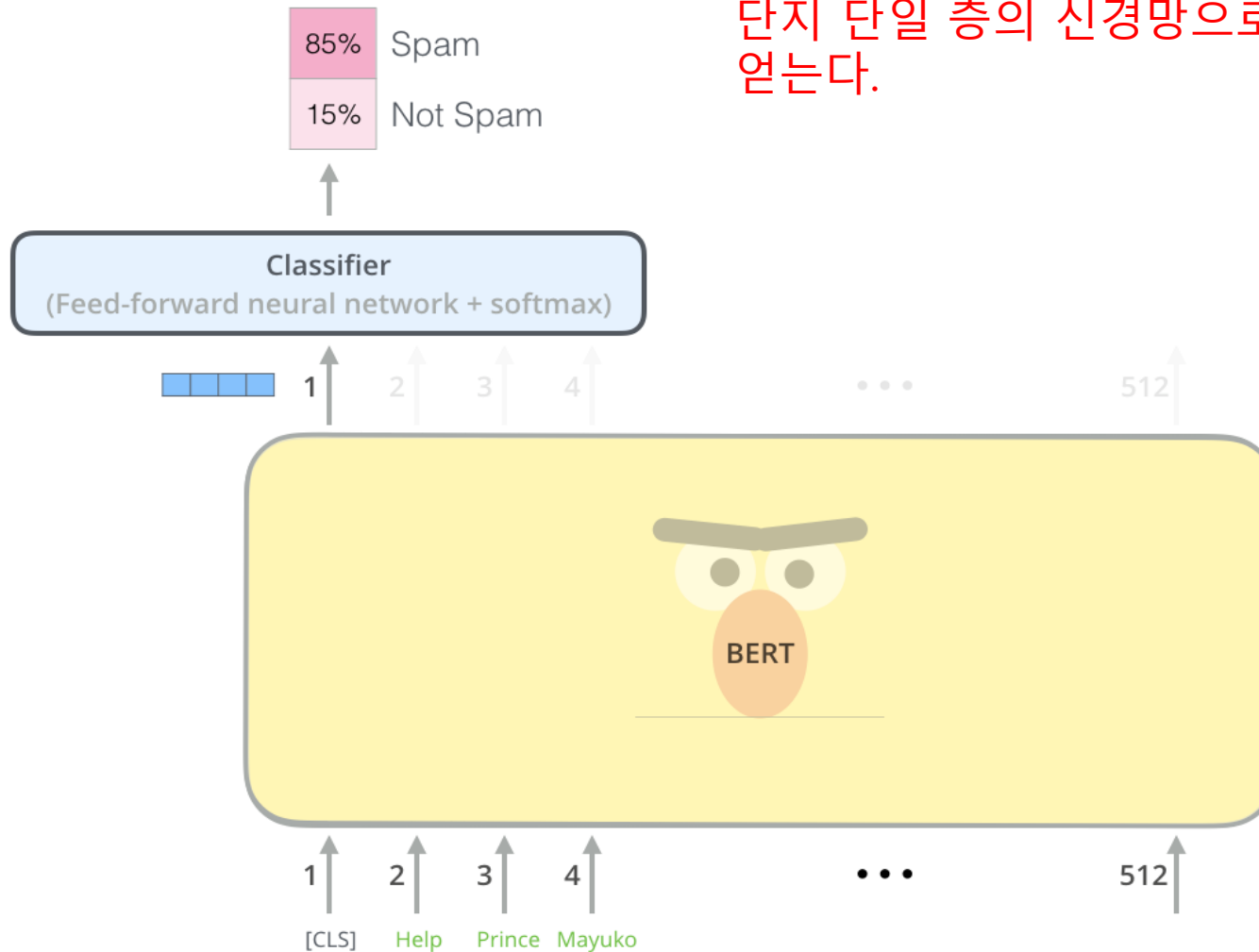
BERT 구조

- BERT 출력
 - 각 포지션은 hidden_size (BERT의 경우, 768) 크기의 벡터를 산출한다. 분류문제에서는 첫째 포지션([CLS]에 대한) 출력에만 초점을 맞춘다.



BERT 구조

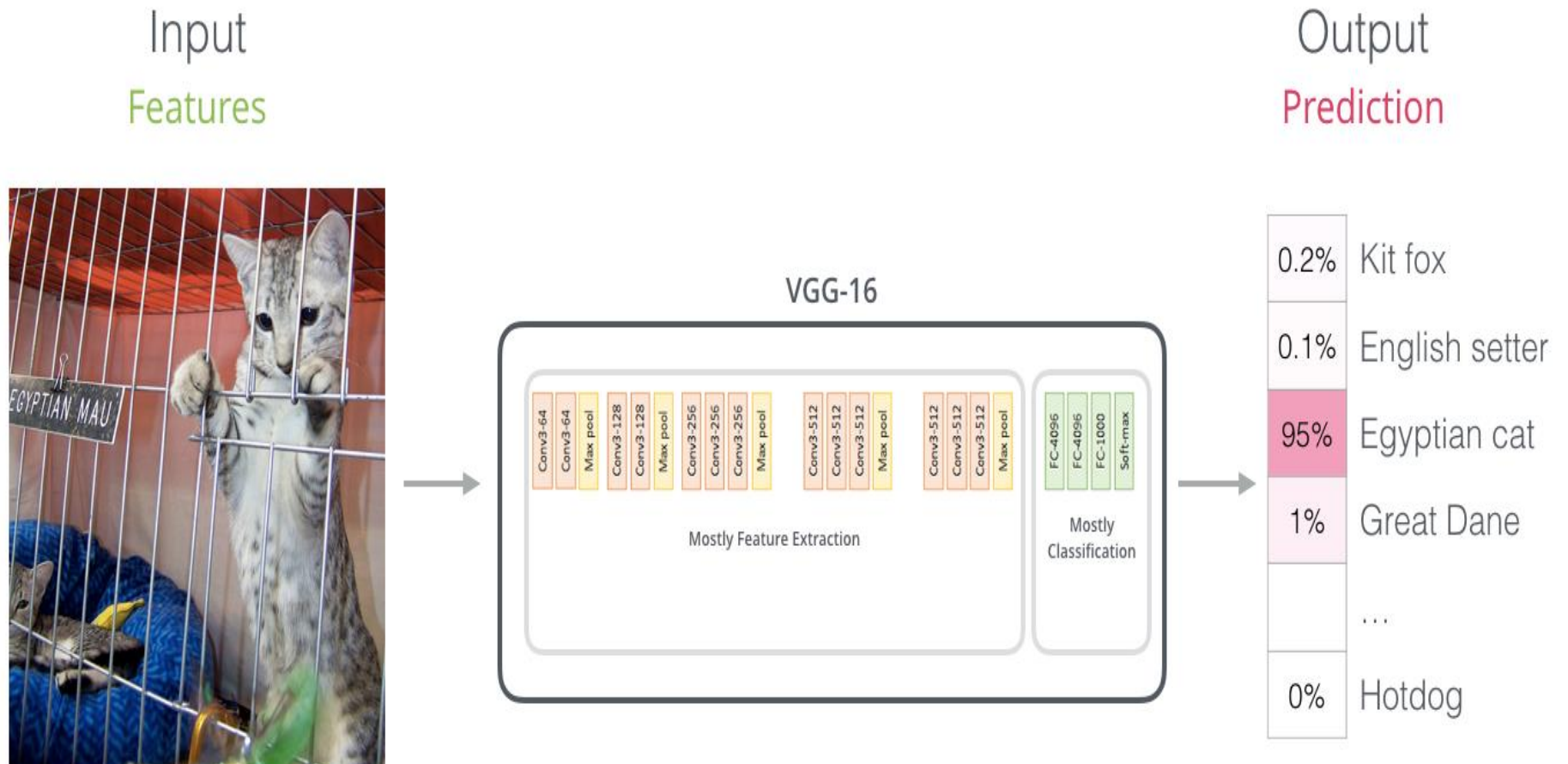
- BERT의 출력



- 그 벡터가 분류기의 입력으로 사용된다. 단지 단일 층의 신경망으로 좋은 효과를 얻는다.

BERT 구조 – 컴퓨터 비전과의 비교

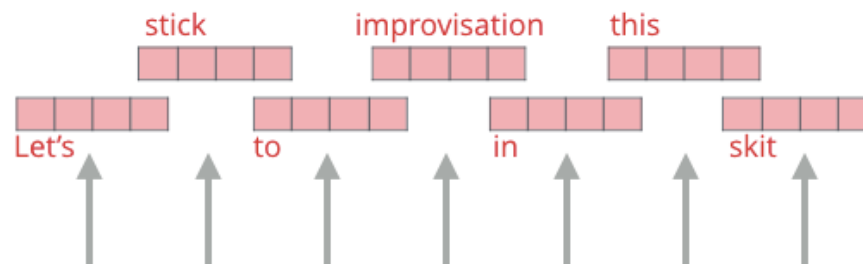
- Convolution Network와 유사한 용도



BERT의 탄생 공헌 - ELMo

- Elmo 임베딩 – context에 따른 한 단어에 여러 의미 부여와 그에 따른 임베딩

ELMo
Embeddings



- 각 단어에 대해 고정 임베딩을 사용하지 않고, 각 단어에 임베딩을 부여하기 전에 전체 문장을 살펴본다. 양방향 LSTM을 사용해 임베딩을 산출한다.

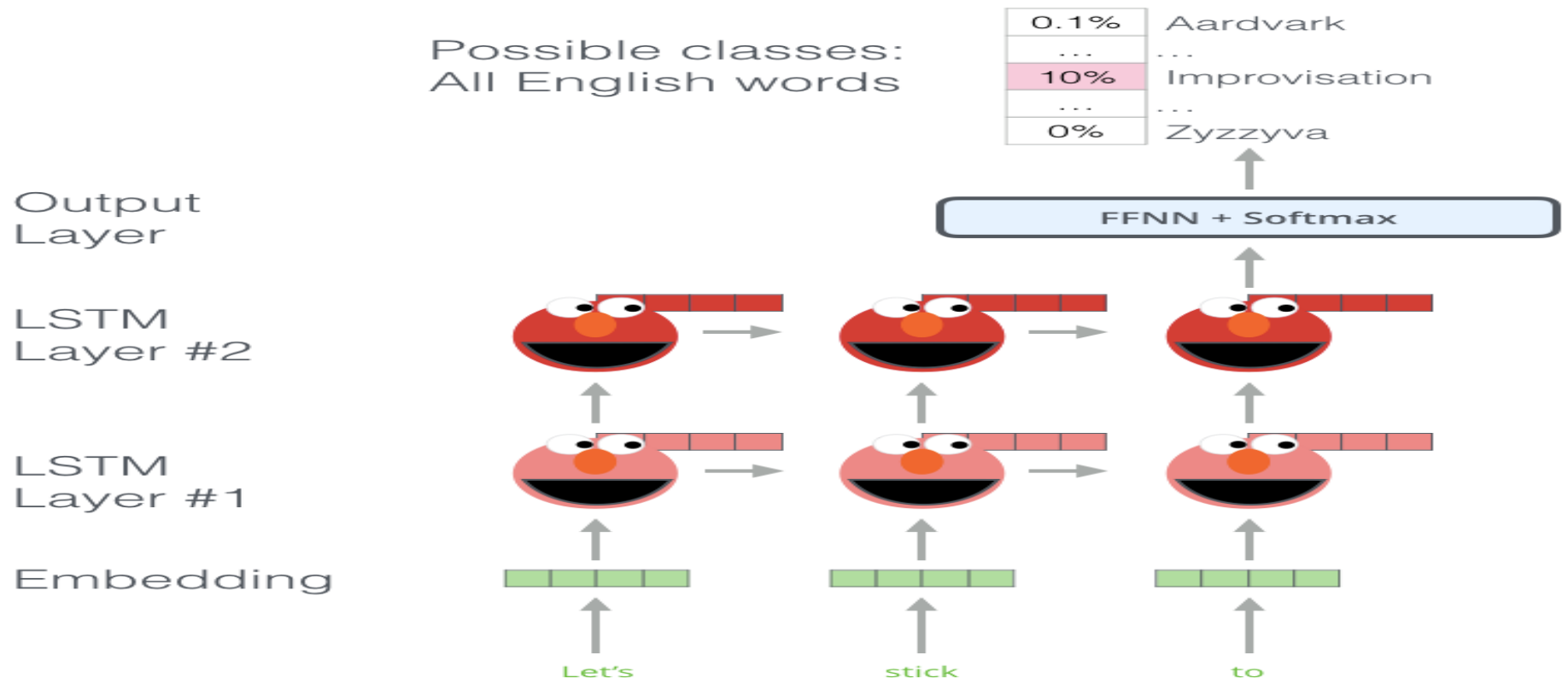


Words to embed



BERT의 탄생 공헌 - ELMO

- ELO 학습 (LSTM)



- ELMo의 사전학습과정: “Let’s stick to” 를 입력으로 하고, 다음에 나올 가능성이 가장 높은 단어를 예측하는 언어모델로 언어 패턴을 학습하기 시작한다. 예를 들면, “hang” 다음에 “out”에 나올 확률이 “camera”보다는 크게 할당 할 것이다.

BERT의 탄생 공헌 - ELMO

- ELMO 학습 (양방향 학습)

Embedding of “stick” in “Let’s stick to” - Step #2

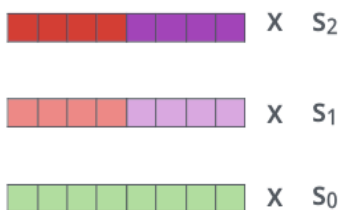
● 순방향

● 역방향

1- Concatenate hidden layers



2- Multiply each vector by a weight based on the task

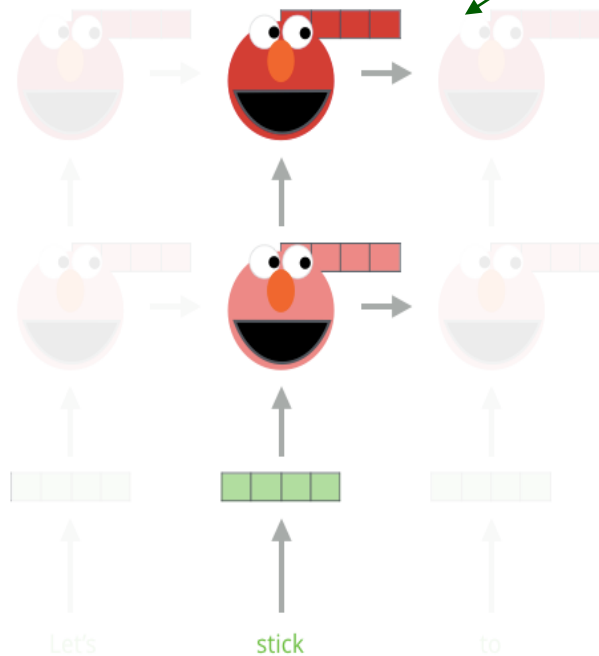


3- Sum the (now weighted) vectors

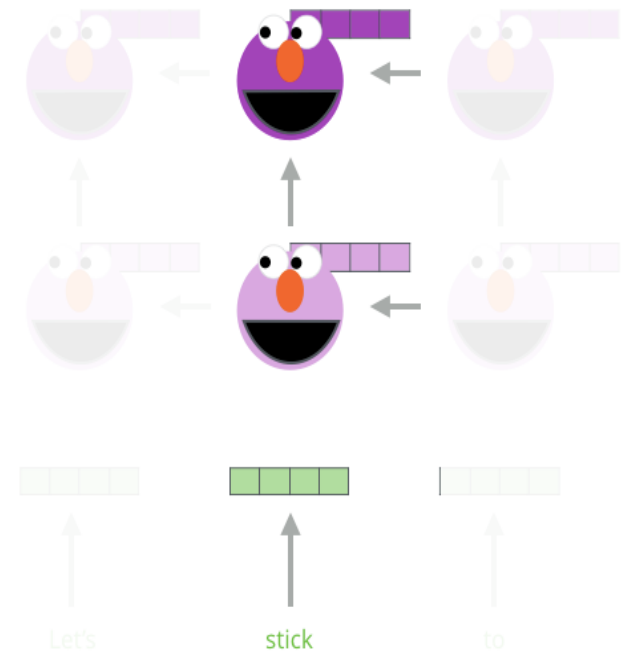


ELMo embedding of “stick” for this task in this context

Forward Language Model

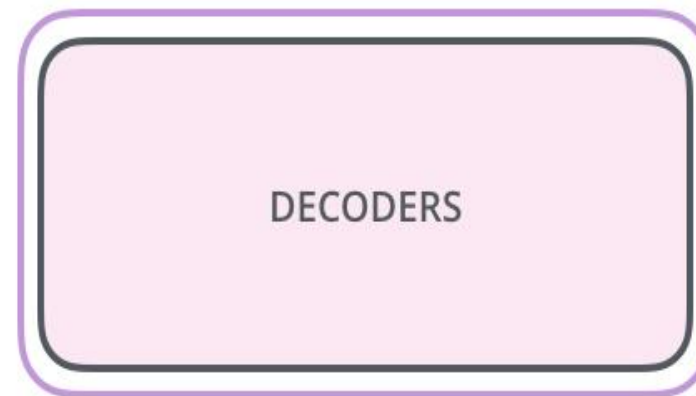
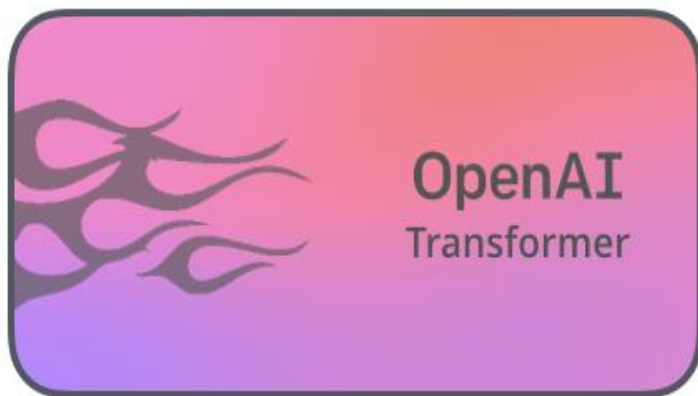


Backward Language Model



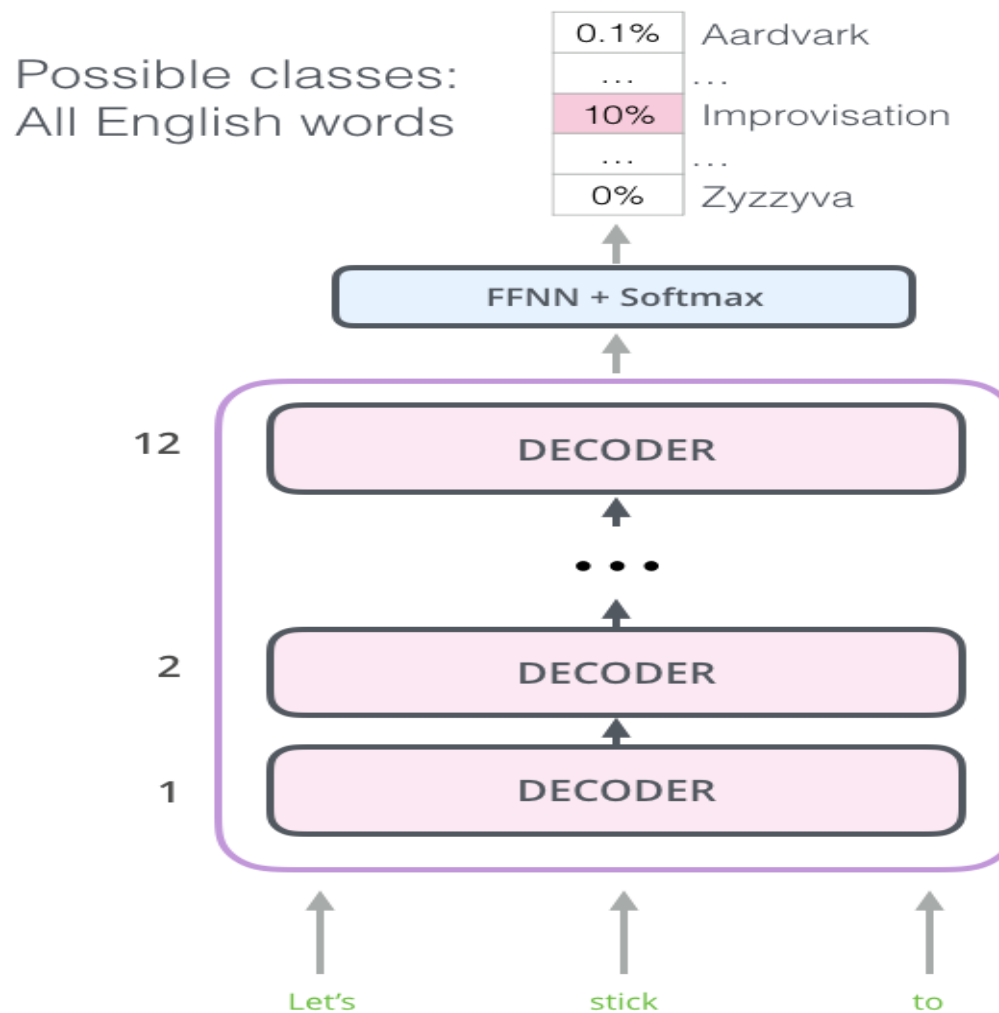
BERT의 탄생 공헌 – ULM-FiT와 OpenAI Transformer

- BERT의 탄생에 많은 사전 연구들에 기여했다.
- ULM-FiT: Transfer Learning
- OpenAI Transformer: 언어모델용으로 트랜스포머 디코더를 사전학습



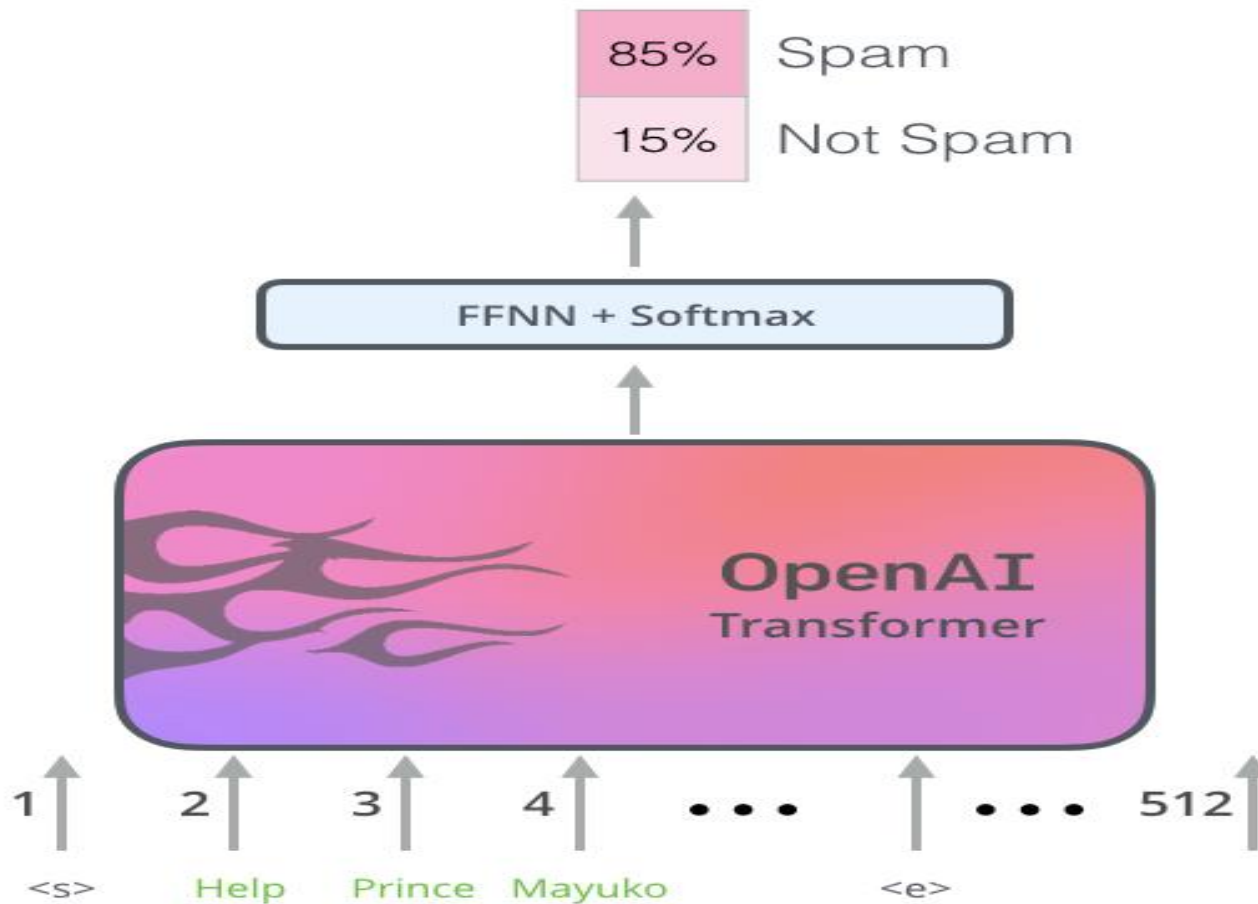
BERT의 탄생 공헌 – OpenAI Transformer

- 광대한 데이터셋을 사용해 다음 단어를 예측: 7,000권의 책, 필요시 twitter와 기사들을 사용해 학습.



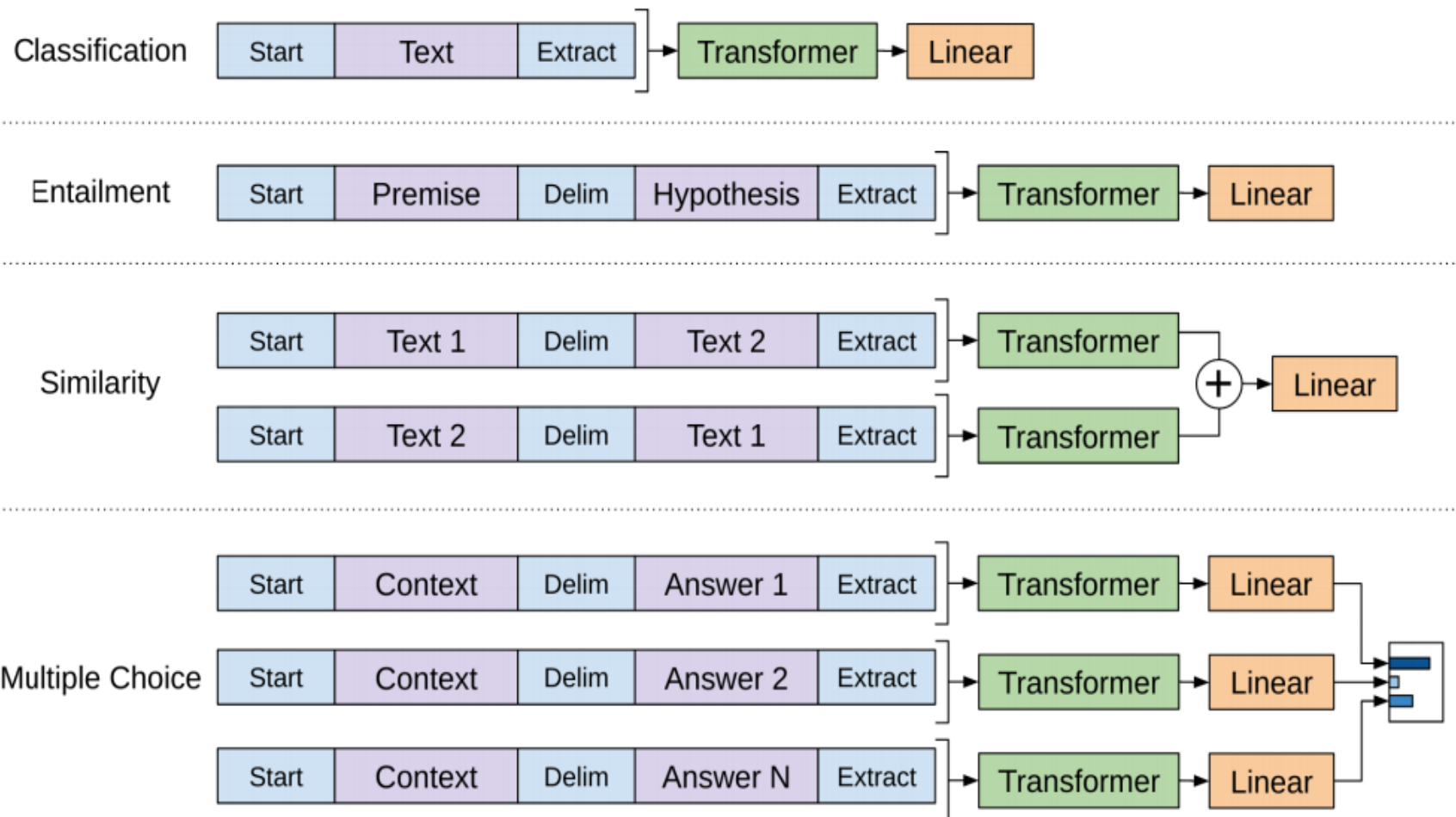
BERT의 탄생 공헌 – OpenAI Transformer

- 이렇게 사전학습된 트랜스포머를 이용해 다른 태스크를 수행 (예를 들면 스팸문장 분류)



BERT의 탄생 공헌 – OpenAI Transformer

- OpenAI는 다양한 과제를 수행하기 위해 다양한 입력 transforme를 도입했다.



- ELMo의 언어모델은 양방향이었으나, OpenAI 트랜스포머는 순방향 언어모델만을 학습한다.

BERT의 탄생

- BERT의 탄생: 디코더에서 (양방향인) 인코더로 사용한다.

Use the output of the masked word's position to predict the masked word

Possible classes:
All English words

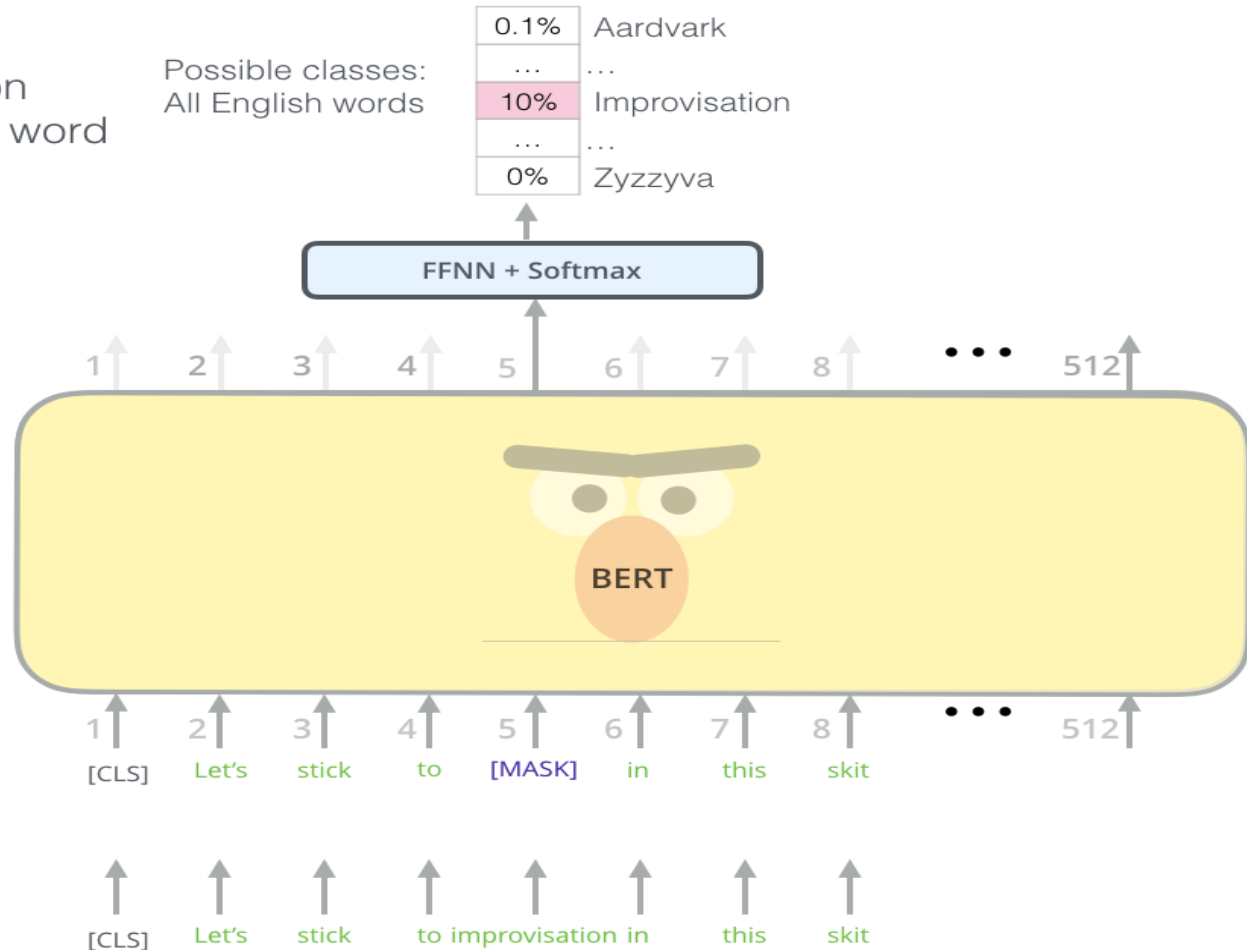
0.1%	Aardvark
...	...
10%	Improvisation
...	...
0%	Zyzzzyva

FFNN + Softmax

- 마스크된 단어의 위치의 출력을 이용해 마스크된 단어를 예측한다.

Randomly mask 15% of tokens

Input

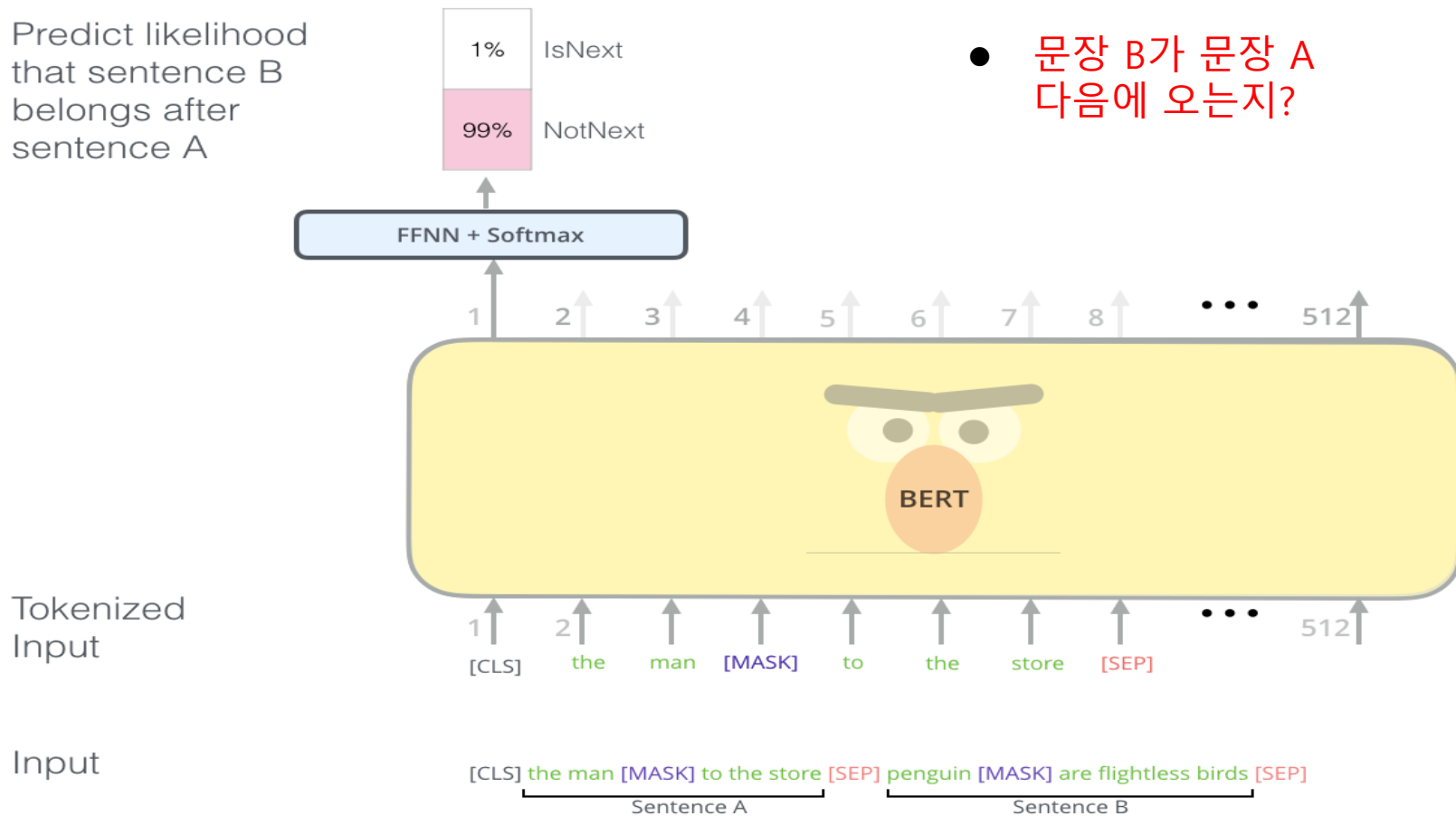


- BERT의 첫번째 과제인 언어모델로 입력 단어의 15%를 마스크를 사용해 모델이 숨겨진 단어를 예측하도록 한다.

BERT의 탄생

- BERT의 탄생에 많은 사전 연구들에 기여했다.

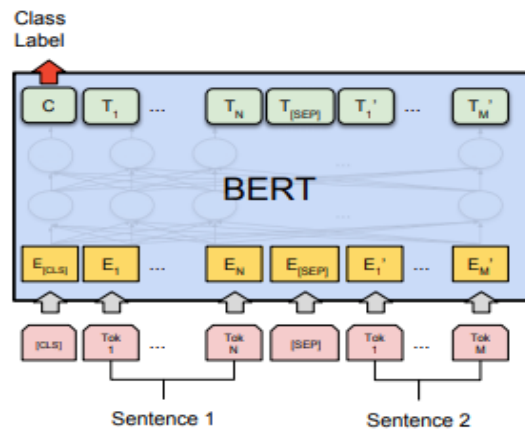
Predict likelihood
that sentence B
belongs after
sentence A



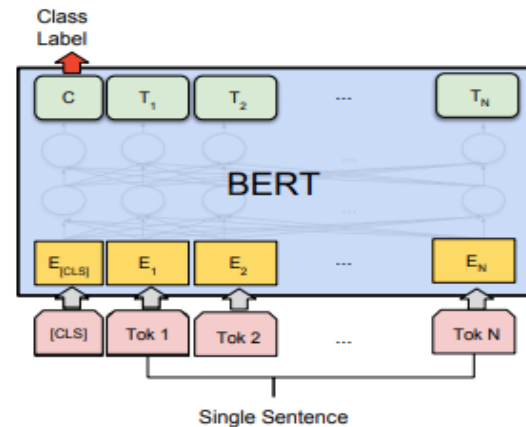
- BERT의 두번째 과제는 두 문장의 분류이다. 위 그림에서 B문장이 A문장 다음에 나올 확률을 예측한다. 그림에서 단순화를 위해 단어를 토큰으로 사용했지만, BERT는 WordPieces를 토큰으로 사용한다.

BERT의 탄생

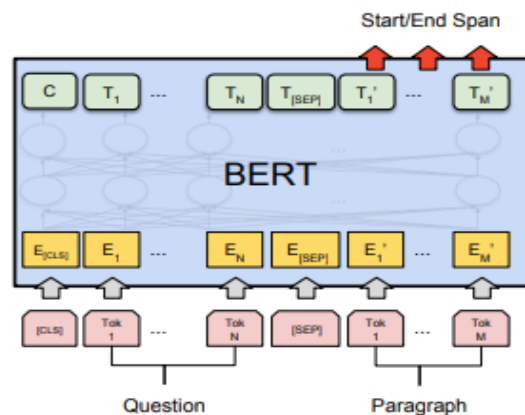
- BERT의 작업특화 모델



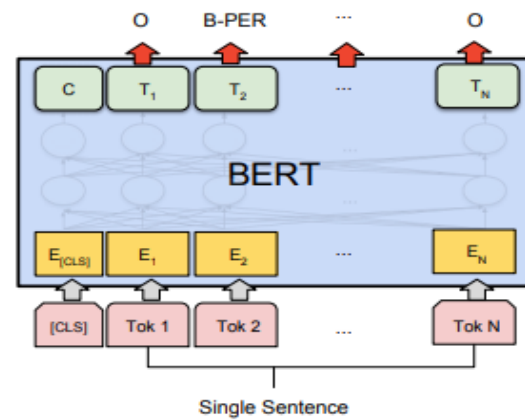
(a) Sentence Pair Classification Tasks:
MNLI, QQP, QNLI, STS-B, MRPC,
RTE, SWAG



(b) Single Sentence Classification Tasks:
SST-2, CoLA



(c) Question Answering Tasks:
SQuAD v1.1



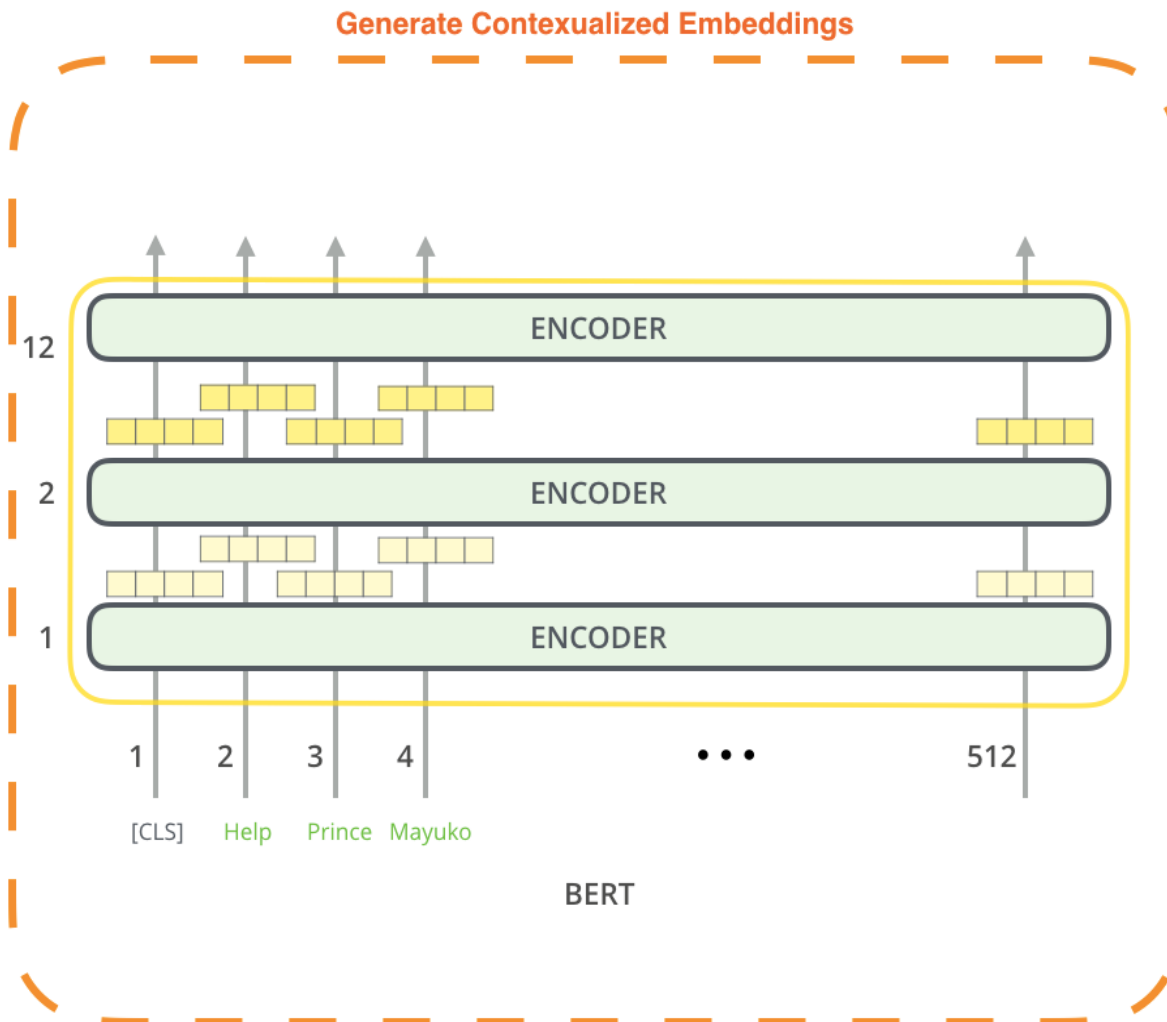
(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER

- BERT는 다양하게 사용될 수 있다.

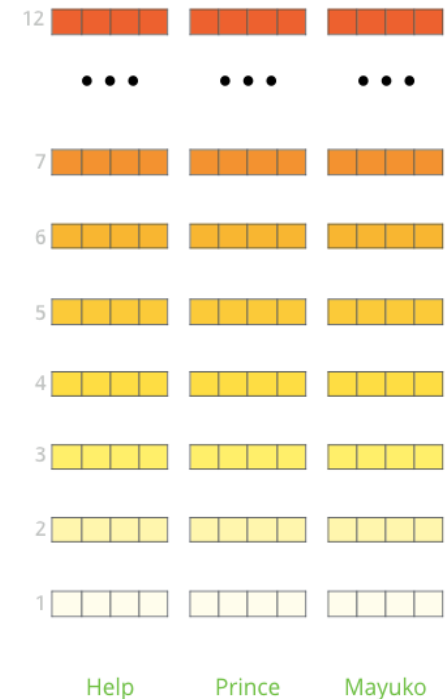
BERT의 탄생

- 각 토큰의 경로를 따르는 각 인코더층의 출력은 해당 토큰을 대표하는 특성으로 사용할 수 있다.

- BERT의 특성 추출: 사전학습된 모델의 성과가 뒤지지 않는다.



The output of each encoder layer along each token's path can be used as a feature representing that token.



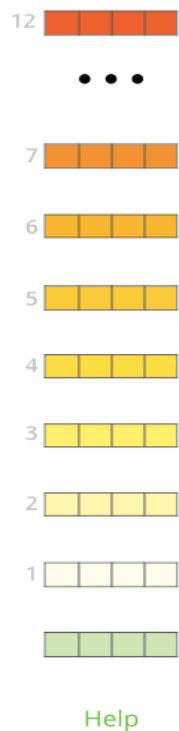
But which one should we use?

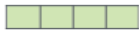


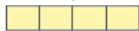
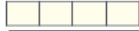




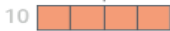
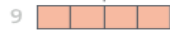



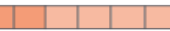

BERT의 탄생

- BERT의 특성 추출

- 최적의 컨텍스트를 반영하는 임베딩은 무엇인가? (NER 테스트)-> 작업에 맞는 선택

What is the best contextualized embedding for “**Help**” in that context?
For named-entity recognition task CoNLL-2003 NER



		Dev F1 Score
First Layer	Embedding 	91.0
Last Hidden Layer	12 	94.9
Sum All 12 Layers	 + ... + 2  + 1  = 	95.5
Second-to-Last Hidden Layer	11 	95.6
Sum Last Four Hidden	 +  +  +  = 	95.9
Concat Last Four Hidden	   	96.1