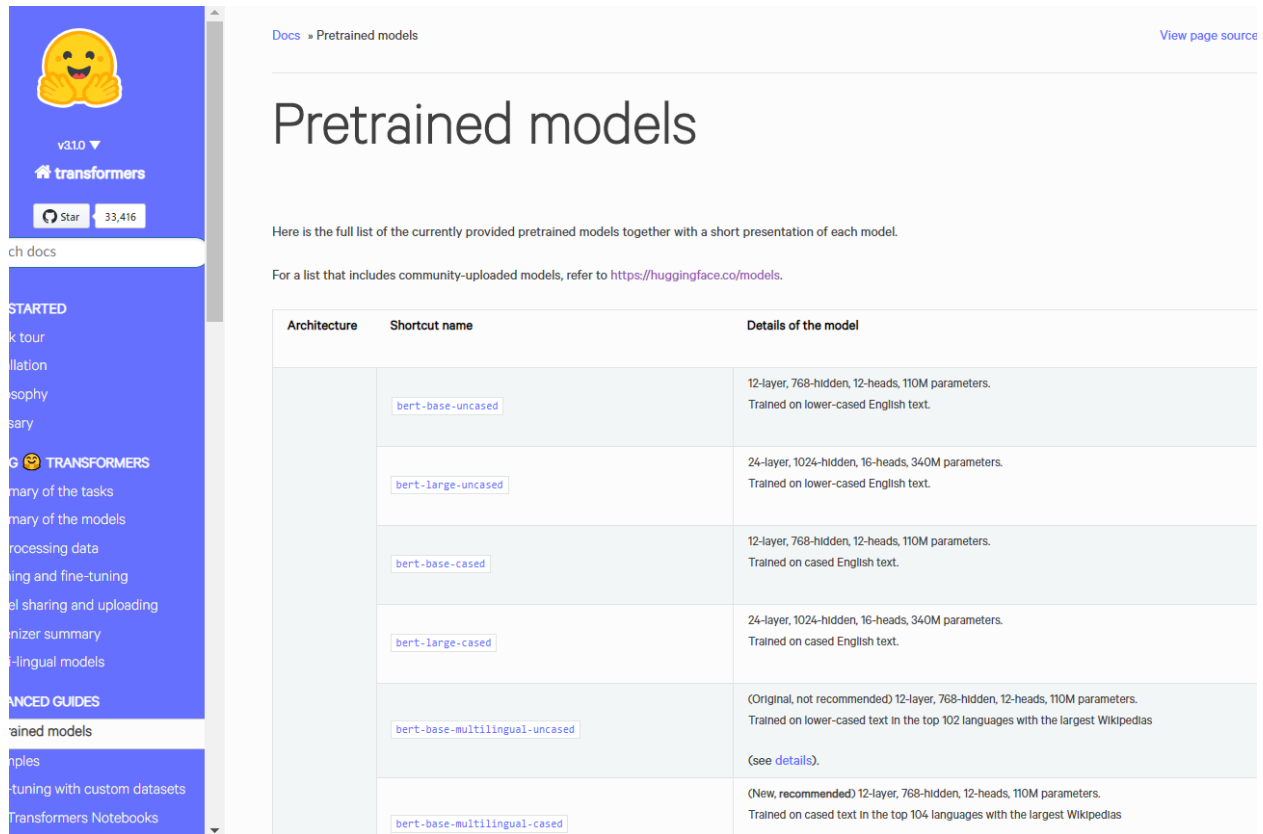


# 20200909\_다섯번째미팅



Docs » Pretrained models [View page source](#)

## Pretrained models

Here is the full list of the currently provided pretrained models together with a short presentation of each model.

For a list that includes community-uploaded models, refer to <https://huggingface.co/models>.

Architecture	Shortcut name	Details of the model
	<code>bert-base-uncased</code>	12-layer, 768-hidden, 12-heads, 110M parameters. Trained on lower-cased English text.
	<code>bert-large-uncased</code>	24-layer, 1024-hidden, 16-heads, 340M parameters. Trained on lower-cased English text.
	<code>bert-base-cased</code>	12-layer, 768-hidden, 12-heads, 110M parameters. Trained on cased English text.
	<code>bert-large-cased</code>	24-layer, 1024-hidden, 16-heads, 340M parameters. Trained on cased English text.
	<code>bert-base-multilingual-uncased</code>	(Original, not recommended) 12-layer, 768-hidden, 12-heads, 110M parameters. Trained on lower-cased text in the top 102 languages with the largest Wikipedias (see <a href="#">details</a> ).
	<code>bert-base-multilingual-cased</code>	(New, recommended) 12-layer, 768-hidden, 12-heads, 110M parameters. Trained on cased text in the top 104 languages with the largest Wikipedias

허깅 페이스 DOC 사이트를 들여다 보니 여러가지 Pre-Train 모델이 있었다 .



HUGGING FACE

[⬆ Back to home](#)

## All Models and checkpoints

Also check out our list of [Community contributors](#) 🏆 and [Organizations](#) 🌐.

Tags: All ▾

Sort: Most downloads ▾

[monologg/kobert](#)

[monologg/koelectra-small-v2-discriminator](#)

[monologg/koelectra-base-discriminator](#) ★

[ktrapeznikov/albert-xlarge-v2-squad-v2](#) ★

[monologg/koelectra-base-v2-discriminator](#)

[ktrapeznikov/biobert\\_v1.1\\_pubmed\\_squad\\_v2](#) ★

[monologg/koelectra-small-discriminator](#) ★

[monologg/distilkobert](#)

[taeminlee/kogpt2](#)

[monologg/koelectra-small-v2-distilled-korquad-384](#)

[monologg/koelectra-base-v2-finetuned-korquad](#)

[monologg/koelectra-base-v2-finetuned-korquad-384](#)

이런식의 검색으로 모델을 가지고 이미 학습된 모델을 가지고와 나의 데이터 셋에 적용 시키는 구조.

이중 KoELECTRA 에 대해 코드를 통해 좀더 알아보아야한다.

<https://github.com/monologg/KoELECTRA>

**Bert???? >> 자연어 처리 ..??**

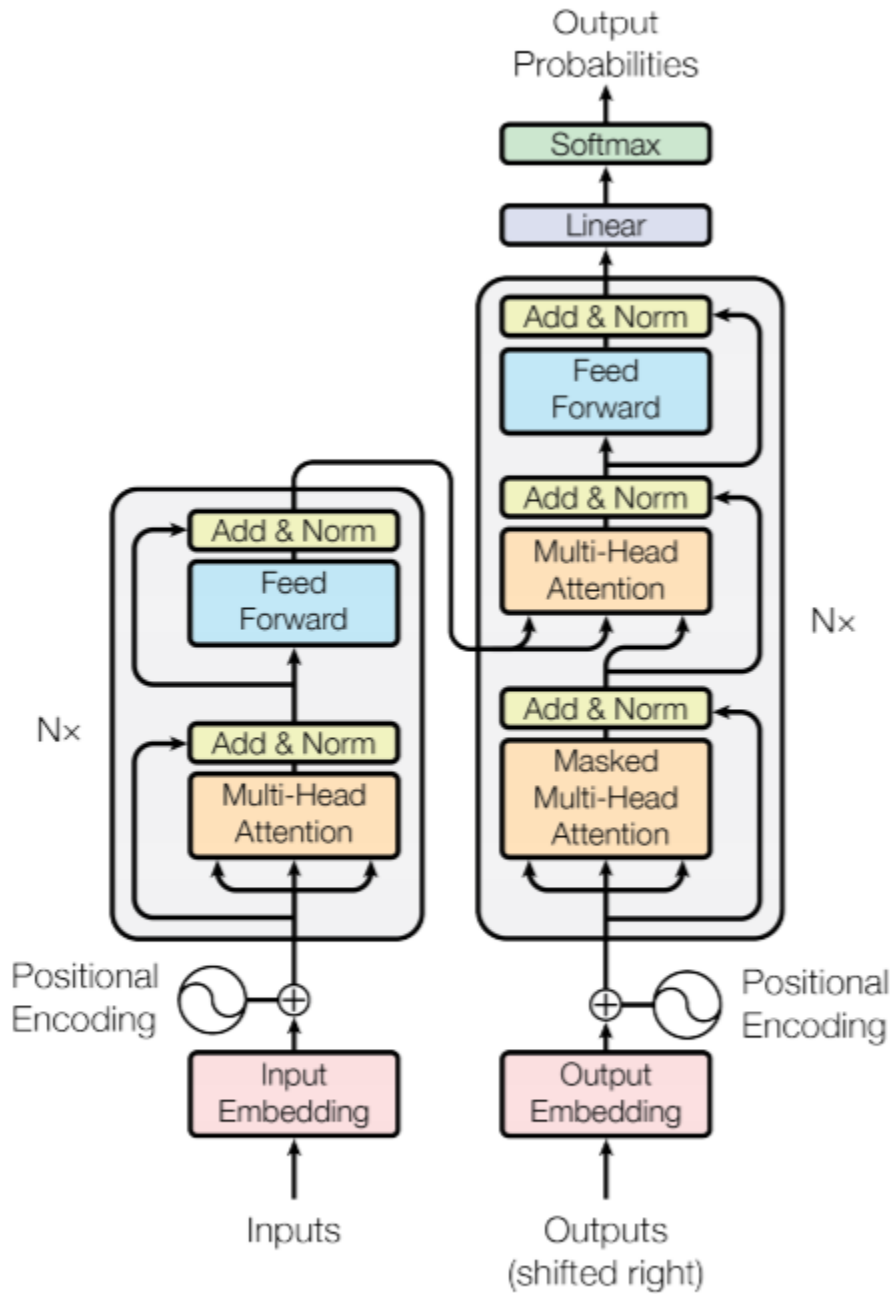


Figure 1: The Transformer - model architecture.

특징 분류

원핫인코딩 > Word2Vec > FastText > Seq2Seq > Bert

## one-hot encoding

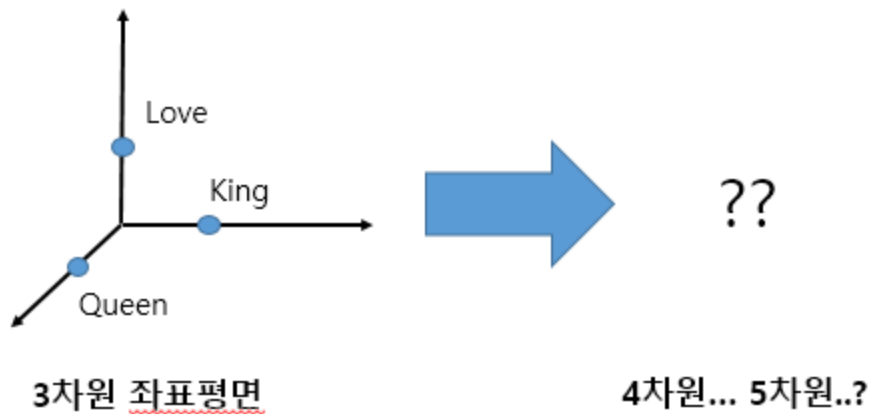
일단 언어의 feature extraction을 위해 단어를 벡터화.

King Love Queen

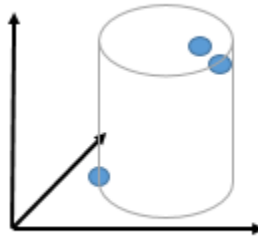


word	one-hot
King	[1,0,0]
Love	[0,1,0]
Queen	[0,0,1]

이는 단어 개수가 늘어나면 늘어 날수록 차원이 많아지는 문제점이 생김...



## Word2Vec



원통은 이해를 돕기위함임 아무런 의미는 없습니다.

이를 해결하기 위해 단어 자체를 다차원 좌표평면에 '벡터화' > 중심단어의 주변단어를 이용해 중심단어를 추론

W2V 는 사전에 없는 단어 즉, Out of Vocabulary에는 적용이 불가능한 문제가 발생.

## FastText

이를 해결하기 위해 주변단어와 원래 단어를 n-gram으로 자르는 FastText가 나옴.

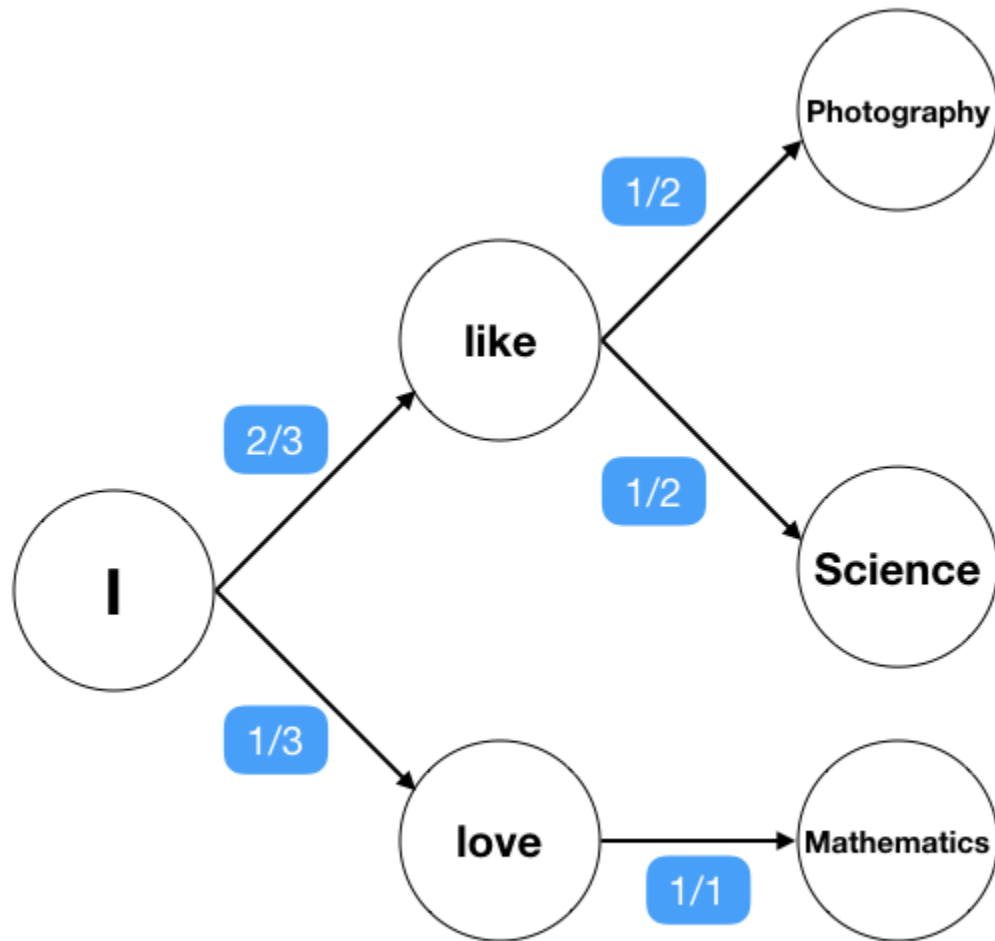
2-5 Orange단어를 예로 들면 or , ra , an , ng , ge 다시 ora, ran, ang, nge ... orang..  
range 로 나누어 정보를 저장

이 정보를 이용해 Oranges 라는 Orange와 비슷한 OOV를 입력받았을때 Orange와 비슷하게 벡터화가 진행됨.

하지만 이 경우 동형어, 다의어 등에서는 임베딩 성능이 좋지않은 문제가 발생

## 언어모델(LM)

위의 해결책으로 문맥을 보는 언어 모델이 등장 대표적으로 Seq2Seq 모델임. 마코프 체인 (마코프 테이블) 의 개념으로 다음 단어가 어떤 단어가 나올지를 확률로 계산



State transition diagram for our sample data

딥러닝기반 언어 모델은 해당 확률을 예상

여기서 RNN 모델을 이용하여 결과 단어를 벡터화 할때 이전의 모든 단어를 이용 할 수 있음.

그러나 이 또한 히든레이어가 많아 질 수록 앞쪽의 가중치는 희미해지는 문제가 발생. 이는 중요하지 않은 단어가 중요한 역할을 할 수 있는 문제를 야기함.

## Attention 모델

이를 해결하기 위해 인간의 정보처리와 마찬가지로 중요한 feature는 더욱 중요하게 고려한 는 Attention 모델을 만들어냄.

(정확한 구조는 좀더 공부하여 설명..)

Attention 모델은 RNN을 이용하기 때문에 시간이 오래걸린다는 단점.