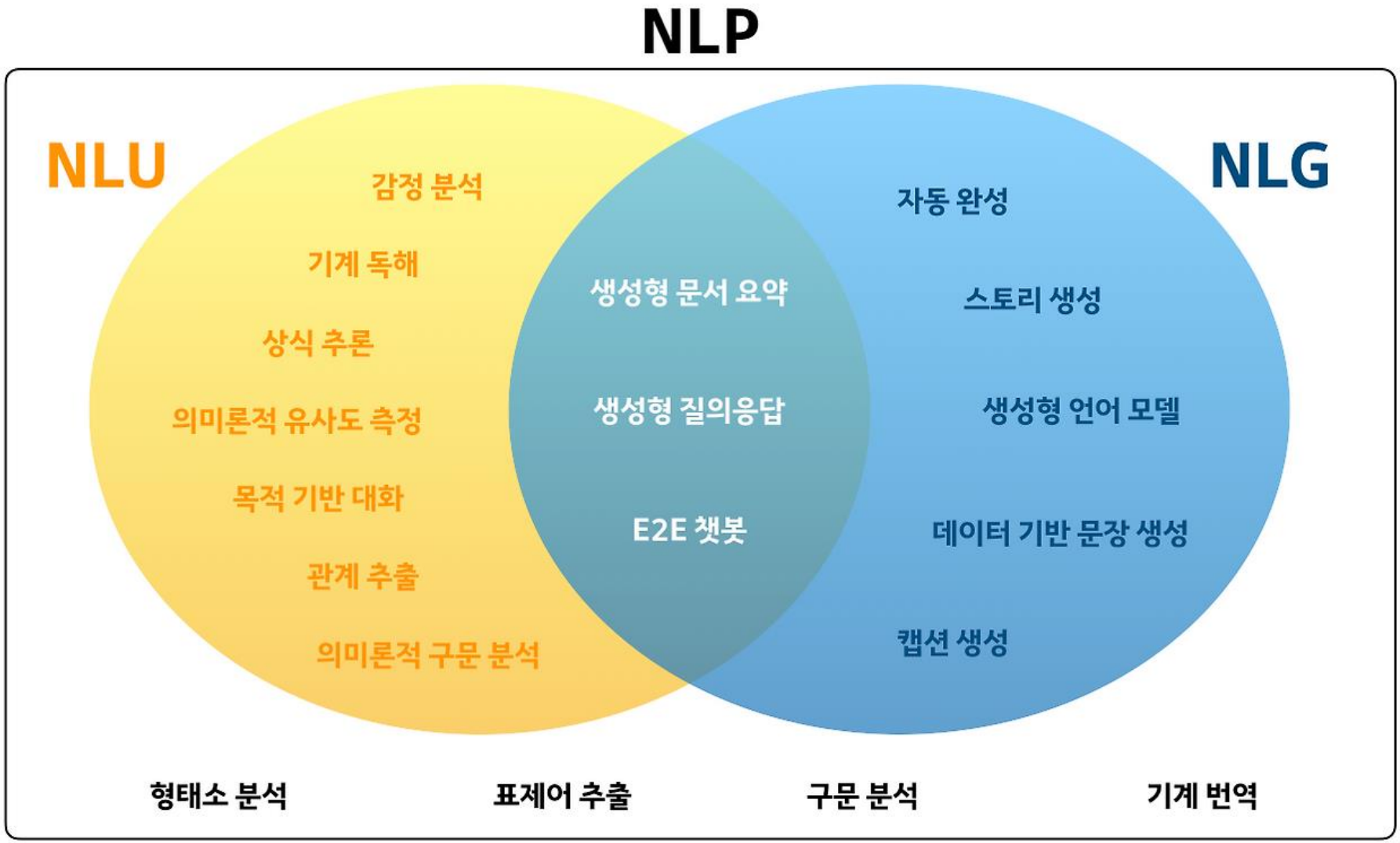
NLU, BERT, and Tokenizer

Finda
전희국

NLP = NLU + NLG

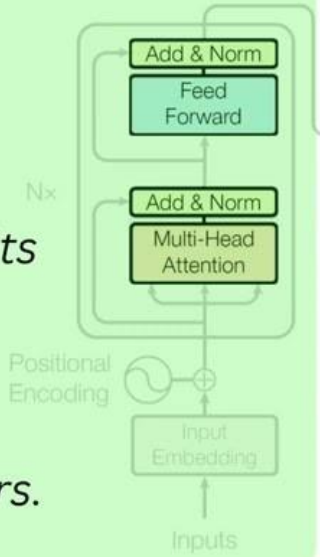

NLP
NLU
감정 분석
기계 독해
상식 추론
의미론적 유사도 측정
목적 기반 대화
관계 추출
의미론적 구문 분석
생성형 문서 요약
생성형 질의응답
E2E 챗봇
NLG
자동 완성
스토리 생성
생성형 언어 모델
데이터 기반 문장 생성
캡션 생성
형태소 분석
표제어 추출
구문 분석
기계 번역

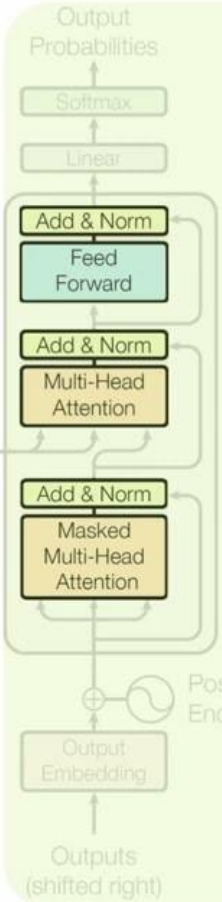
Transformer, BERT, GPT

BERT

Google

use transfer learning to ***continue learning*** *from its existing data when adding user-specific tasks and layers.*





GPT

OpenAI

decodes from its massive pre-learned embeddings to present output that matches user prompts. It

Figure 1: The Transformer - model architecture.

BERT Embeddings

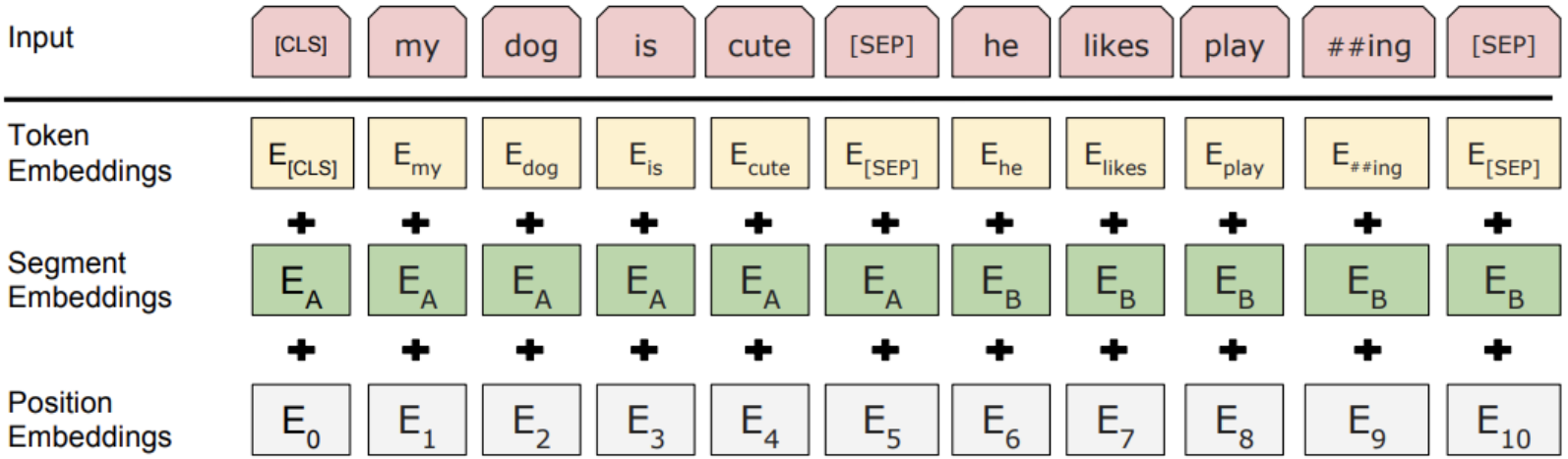
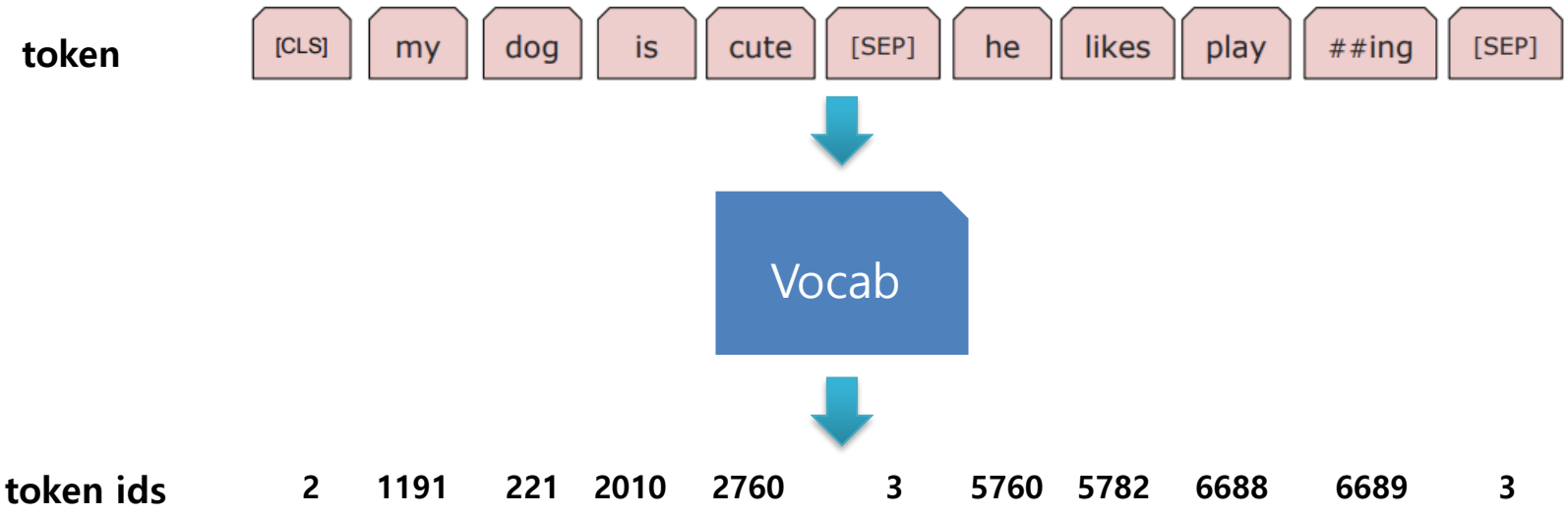


Figure 2: BERT input representation. The input embeddings is the sum of the token embeddings, the segmentation embeddings and the position embeddings.

BERT Input1: Token IDs

- integer encoded using vocabulary



Tokenizer: Encode

```
from transformers import AutoTokenizer

kobert_tokenizer = AutoTokenizer.from_pretrained("monologg/kobert")

token_ids = kobert_tokenizer.encode("날은  아주  좋았어.")
```

```
[2, 1407, 7086, 3128, 4208, 6855, 54, 3]
```

Tokenizer: Decode

```
decoded_tokens = kobert_tokenizer.encode(
   [2, 1407, 7086, 3128, 4208, 6855, 54, 3])
```

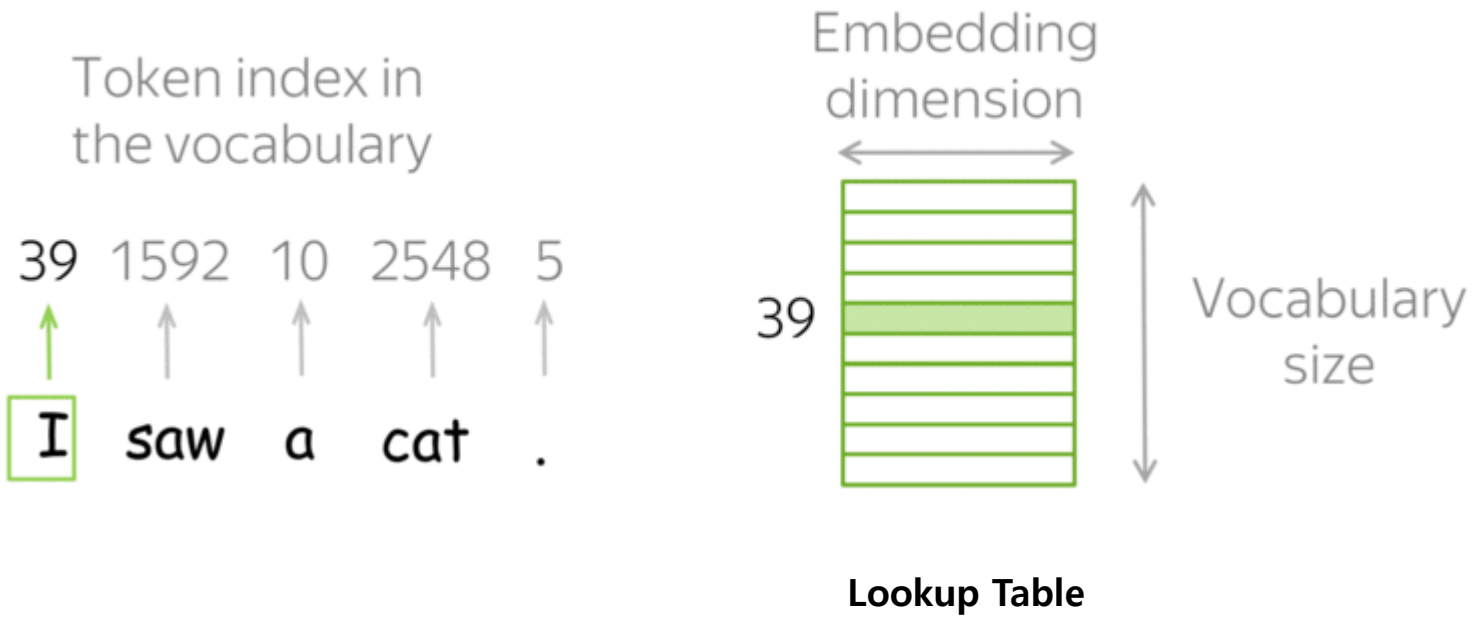
날은 아주 좋았어.

Tokenizer: Vocab Size

```
from transformers import AutoTokenizer

kobert_tokenizer = AutoTokenizer.from_pretrained("monologg/kobert")

kobert_tokenizer.vocab_size
```

```
8002
```

Embedding Layer

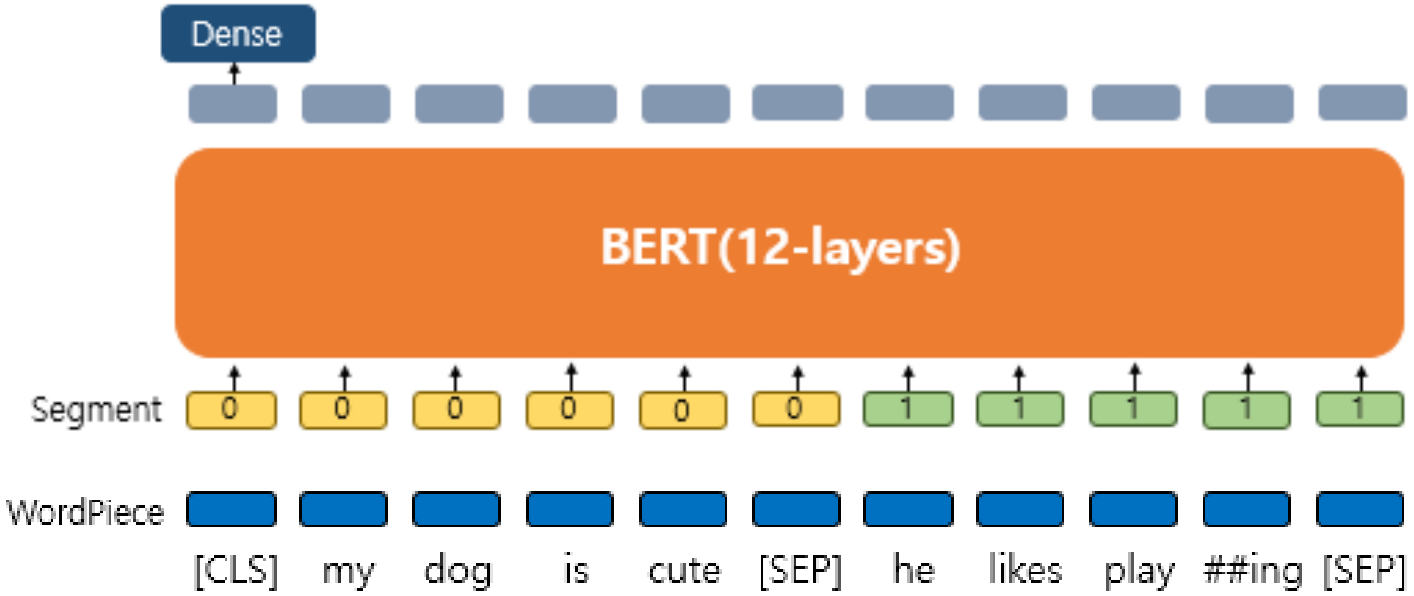


Lookup Table

https://lena-voita.github.io/nlp_course/word_embeddings.html

BERT Input2: Segment IDs

- segment_ids = token_type_ids

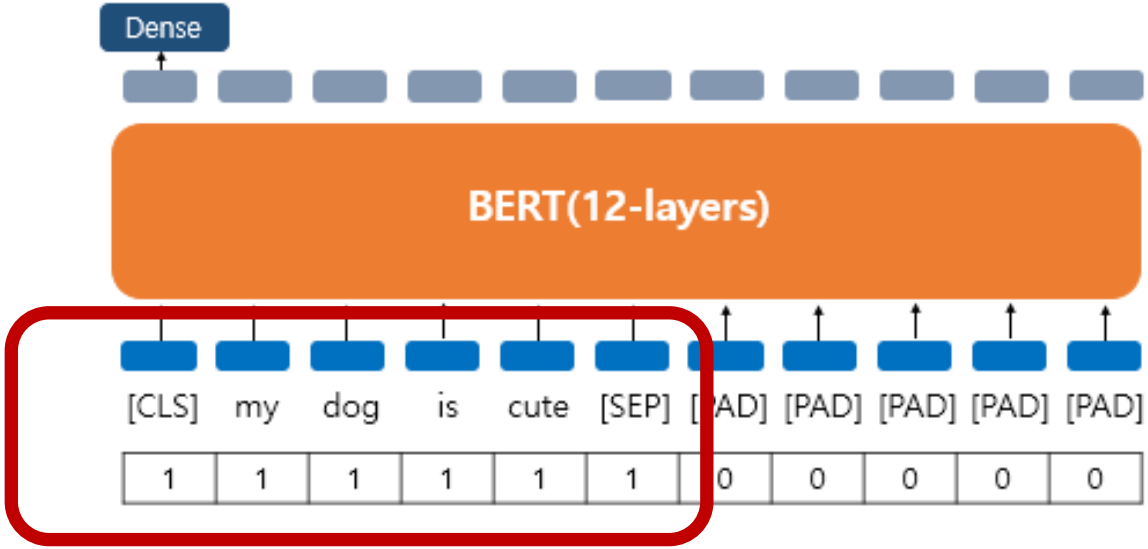


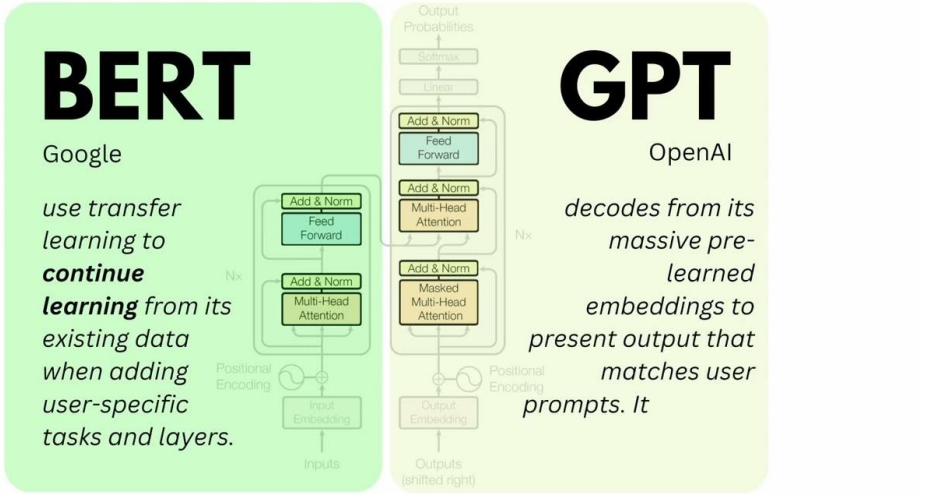
Tokenizer: __call__

```
from transformers import AutoTokenizer

kobert_tokenizer = AutoTokenizer.from_pretrained("monologg/kobert")

inputs = kobert_tokenizer(
        "그는 밥을 먹는다",
        padding="max_length",
        truncation=True,
        return_tensors="pt",
        max_length=20
    )
```

```
{
'input_ids': tensor([[2, 1191, 2266, 7088, 2010, 5760, 5782, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]]),

'token_type_ids': tensor([[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]),

'attention_mask': tensor([[1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]])
}
```

BERT Input3: Attention Mask





Tokenizer: __call__

```
from transformers import AutoTokenizer

kobert_tokenizer = AutoTokenizer.from_pretrained("monologg/kobert")

inputs = kobert_tokenizer(
        "그는 밥을 먹는다",
        padding="max_length",
        truncation=True,
        return_tensors="pt",
        max_length=20
    )
```

```
{
'input_ids': tensor([[2, 1191, 2266, 7088, 2010, 5760, 5782, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]]),

'token_type_ids': tensor([[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]),

'attention_mask': tensor([[1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]])
}
```

Tokenizer: all_special_tokens

```
from transformers import AutoTokenizer

kobert_tokenizer = AutoTokenizer.from_pretrained("monologg/kobert")

print(kobert_tokenizer.all_special_tokens)

print(kobert_tokenizer.all_special_ids)
```

```
['[UNK]', '[SEP]', '[PAD]', '[CLS]', '[MASK]']

[0, 3, 1, 2, 4]
```