AI Programming

Lab 06: KOGPT2\_dist

소속: 컴퓨터정보공학부

학번: 2019202103

이름: 이은비

텍스트, 로고, 폰트, 등록 상표이(가) 표시된 사진

자동 생성된 설명

**<Lab Objective and Whole simulation program flow>**

언어 모델이란 문장 혹은 단어에 확률을 할당하여 컴퓨터가 처리할 수 있도록 하는 모델입니다. 또한 언어 모델링 (Language Modeling)은 기존의 데이터셋을 바탕으로 주어진 task 안에서의 단어 혹은 문장을 예측하는 작업을 의미합니다. 자연어 처리 (Natural Language Processing), 짧게 말해서 NLP에서 자연어는 말그대로 인간이 사용하는 언어를 뜻하며 자연어를 컴퓨터가 처리하는 과정을 프로세싱 (Processing)이라는 표현을 붙여 자연어 처리 (Natural Language Processing)라고 일컫습니다. 그리고 유사한 데이터로 미리 학습된 모델을 Pretrained Model이라고 합니다. Pretrained Model을 잘 선택해서 활용하면 더 적은 데이터로도 같거나 뛰어난 성능을 이루어 낼 수 있습니다. 그렇기 때문에 최근 머신 러닝 모델들은 백지부터 시작하는 경우가 거의 없습니다. 그 분야에서의 pretrained model이 무엇이 있는지 잘 파악하고,

BERT, GPT 같은 모델이 주목을 받게 된 이유는 성능 때문인데 이 모델들을 사용하면 문서 분류, 개체명 인식 등 어떤 태스크든 점수가 이전 대비 큰 폭으로 오르기 때문이며 BERT, GPT 따위의 부류는 미리 학습된 언어 모델(pretrained language model)이라는 공통점이 있습니다.

BERT (Bidirectional Encoder Representations from Transformers)는 2019년 10월 25일 구글 리서치 팀에 의해 공개된 자연어처리 사전 훈련 모델입니다. BERT는 양방향 교육을 사용합니다. 즉, 문장의 맥락을 이해하기 위해 양방향에서 문장을 읽습니다. BERT는 인코더만 있고 디코더가 없습니다.

BERT 모델은 100여개가 넘는 언어 학습을 지원하며, BERT-Base, BERT-Large, BERT-Base, Multilingual, 그리고 BERT-Base, Chinese 모델이 있습니다. 각각의 모델 뒤에 Cased와 Uncased가 붙혀져 있는데, Uncased의 경우 대소문자 구분을 하지 않는 모델입니다.

BERT를 이용하여 특정 과제를 수행할 수 있는데요, 이를 위해서는 세부적인 과제를 수행하도록 파인튜닝(fine-tuning) 작업이 필요합니다. BERT은 사전 학습된 언어모델로서 파인튜닝을 통해, 원하는 작업을 수행할 수 있습니다.

GPT 모델은openAI에서 개발한 자연어 처리 모델입니다. GPT-1, GPT-2, GPT-3 순으로 공개되었는데, 2019년 GPT-2의 공개 후 엄청난 성능으로 많은 사람들을 놀라게 하였고, 2020년 6월에는 GPT-3로 전세계 인공지능계에 또 한 번 돌풍을 몰고 왔습니다.

GPT-2는 주어진 텍스트의 다음 단어를 잘 예측할 수 있도록 학습된 언어모델이며 문장 생성에 최적화되어 있습니다. KoGPT2는 영어와는 어순이며 여러가지로 다르고 해석의 성능이 부족한 한국어 성능을 극복하기 위해 40GB 이상의 텍스트로 학습된 한국어 디코더(decoder) 언어모델 입니다.

| **Column 1** | **Column 2** | **Column 3** | **Column 4** | **Column 5** | **Column 6** | **Column 7** |
| --- | --- | --- | --- | --- | --- | --- |
| Model | # of params | Type | # of layers | # of heads | ffn\_dim | hidden\_dims |
| kogpt2-base-v2 | 125M | Decoder | 12 | 12 | 3072 | 768 |

GPT-2는 주어진 텍스트의 다음 단어를 잘 예측할 수 있도록 학습된 언어모델이며 문장 생성에 최적화되어 있습니다. KoGPT2는 부족한 한국어 성능을 극복하기 위해 40GB 이상의 텍스트로 학습된 한국어 디코더(decoder) 언어모델 입니다. KoGPT2는 한국어 위키 백과 이외, 뉴스, 모두의 말뭉치 v1.0, 청와대 국민청원 등의 다양한 데이터를 학습시켜 만든 언어모델 입니다. 이 모델은 GPT-2와 마찬가지로 논문 Attention Is All You Need에서 제시한 인코더+디코더 구조에서 인코더 블록을 제거하고 디코더 블록만 사용한 transformer모델입니다. 그렇지만 Transformer 은 그 목적이 번역을 잘하는 것이었기에 영어를 Encode 하고, 다시 프랑스어로 Decode 하는 과정이 필요했기 때문에 Encoder, Decoder가 필요 했던 것입니다. 하지만, BERT 와 GPT는 그 목적이 언어 모델을 사전 학습시키는 것이기 때문에 Encoding 과 Decoding의 과정이 필요하지 않습니다. GPT-2는 Transformer 디코더 블록을 사용하여 제작되었습니다. 반면 BERT는 Transformer 인코더 블록을 사용합니다. 둘 사이의 한 가지 주요 차이점은 GPT2가 기존 언어 모델과 마찬가지로 한 번에 하나의 토큰을 출력한다는 것입니다. 이 모델이 실제로 작동하는 방식은 각 토큰이 생성된 후 해당 토큰이 입력 시퀀스에 추가된다는 것입니다. 그리고 그 새로운 시퀀스는 다음 단계에서 모델에 대한 입력이 됩니다. 실습에서 simulation하는 내용 또한 위의 설명한 GPT2의 모델을 이용하여서 training및 임의의 문장을 통해 그에 관한 예측 되는 답변을 출력하는 과정을 거칩니다. 전체적인 실습과정 flow를 간단하게 설명하면 packages를 import하고 tokenizer와 model에 대해서 'skt/kogpt2-base-v2'라는 pre-trained model을 옵션값으로 두어서 할당하는 과정을 거친 후, model의 summary된 내용을 print합니다. 그리고 dataset이 올바르게 load되었는지에 대해서 몇 개의 값을print하는 과정을 통해 그 값이 옳은지를 확인합니다. 그리고 csv파일을 다운로드 한 후 그 것을 다시 load하고 train\_data에 할당하는 작업을 진행합니다.그리고 할당된 변수에 대해서 display함으로써 그 값이 맞는지 또한 확인합니다. tokenizer를 사용하여 대화 context를 encoding합니다.  'train\_data' DataFrame에서 질문과 답변을 반복한 결과를 시작 및 종료 시퀀스에 대한 특수 토큰을 포함하여 대화를 나타내는 토큰을 산출합니다 이후,생성기 함수로 'get\_chat\_data'를 지정하여 'from\_generator' 메서드를 사용하여 TensorFlow 데이터 집합을 만듭니다. tokenizer를 사용하여 배치를 해독하고 결과를 출력합니다. 또한 배치를 인코딩하고 결과를 출력합니다. 지정된 수의 에포크(epoch)를 반복하고 신경망 모델에 대한 기울기 하강을 사용하여 모델을 훈련하는 훈련 루프를 돕니다. 또한 대화 시뮬레이션을 위해 입력 텍스트 형식 지정 후 tokenizer로 encoding하고 사전 교육된 모델을 사용하여 텍스트를 생성합니다. 그리고 생성된 출력을 decoding합니다. 그에 대한 응답 추출 및 정리 후 모델을 사용하여 텍스트 생성, 그리고 새로 생성된 출력 디코딩하고, 함수를 정의하면 사용자의 텍스트를 입력으로 받아 포맷하고 언어 모델을 사용하여 응답을 생성합니다. 생성된 응답을 반환하여 챗봇 모델에 의한 응답 또한 생성합니다  
몇 개의 임의의 사용자의 입력에 따른 챗봇의 응답을 회신하는 과정을 거칩니다.

**<Describe the feature of the dataset>**

Pre-trained되는 koGPT2모델의 한국어 위키 백과 이외, 뉴스, 모두의 말뭉치 v1.0, 청와대 국민청원 등의 다양한 데이터가 모델 학습에 사용되었습니다

챗봇에 대한 dataset은 인공데이터 입니다. 일부 이별과 관련된 질문에서 다음카페 "사랑보다 아름다운 실연( http://cafe116.daum.net/\_c21\_/home?grpid=1bld )"에서 자주 나오는 이야기들을 참고하여 제작하였습니다. 가령 "이별한 지 열흘(또는 100일) 되었어요"라는 질문에 챗봇이 위로한다는 취지로 답변을 작성하였습니다. 챗봇 트레이닝용 문답 페어 11,876개

일상다반사 0, 이별(부정) 1, 사랑(긍정) 2로 레이블링 된 데이터입니다.

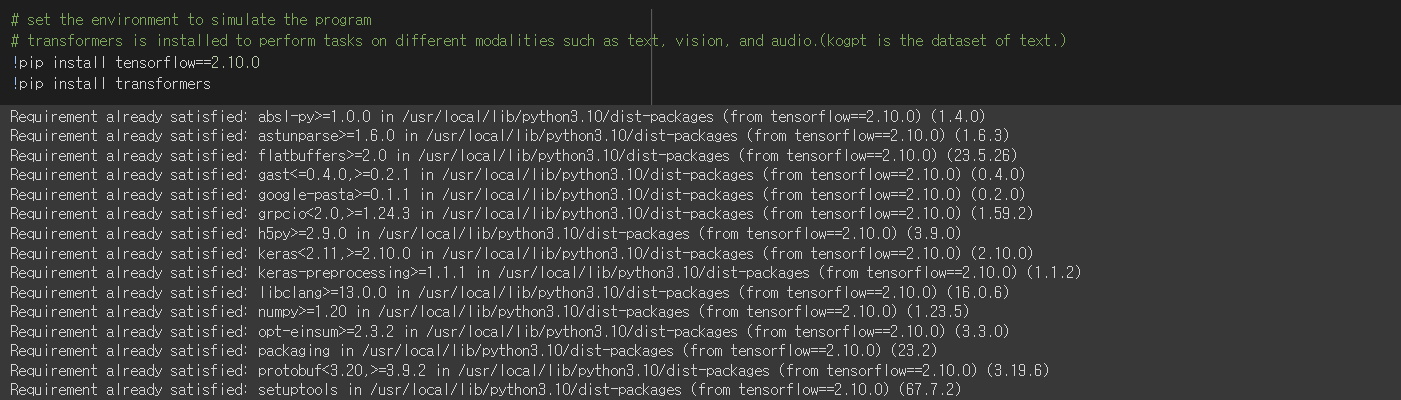
**<Describe and Compare the input and output data sizes of the model>**

Input data size를 형식과 크기로 보면 다음과 같습니다. 입력 형식은 한국어로 된 텍스트 시퀀스입니다. 또한 크기는 토큰화 및 시퀀스 길이에 따라 가변. KoGPT-2는 텍스트를 토큰화하여 어휘의 토큰을 나타내는 일련의 정수로 변환합니다.

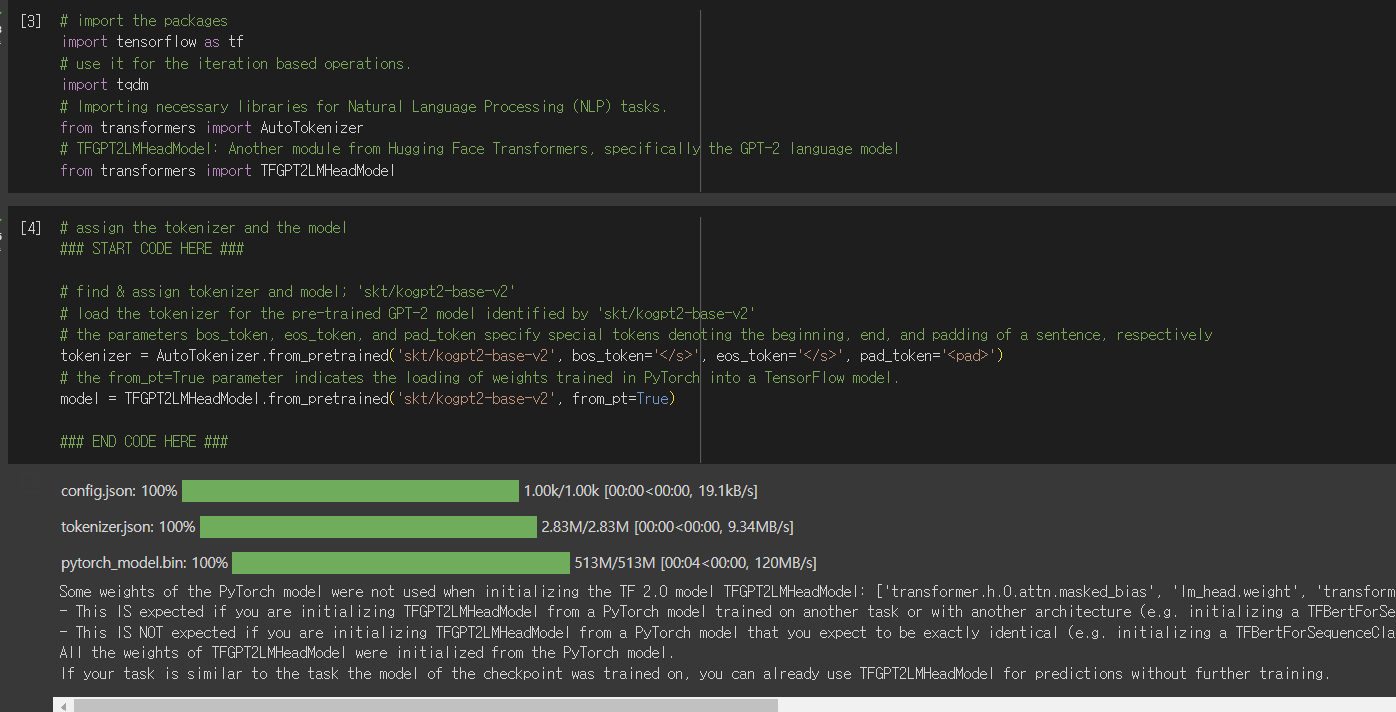
Output data size를 형식과 크기로 보면 다음과 같습니다. 출력 형식은 한국어로 된 텍스트 시퀀스를 생성합니다. 또한 크기는 생성된 응답 길이를 기준으로 가변하며, 생성 시 설정된 최대 길이에 의해 제한됩니다. 텍스트 생성 상황 및 작업에 따라 크기가 달라집니다.

**< Simulation results >**

2.10.0 버전의 tensorflow와 transformers를 install한 결과입니다.

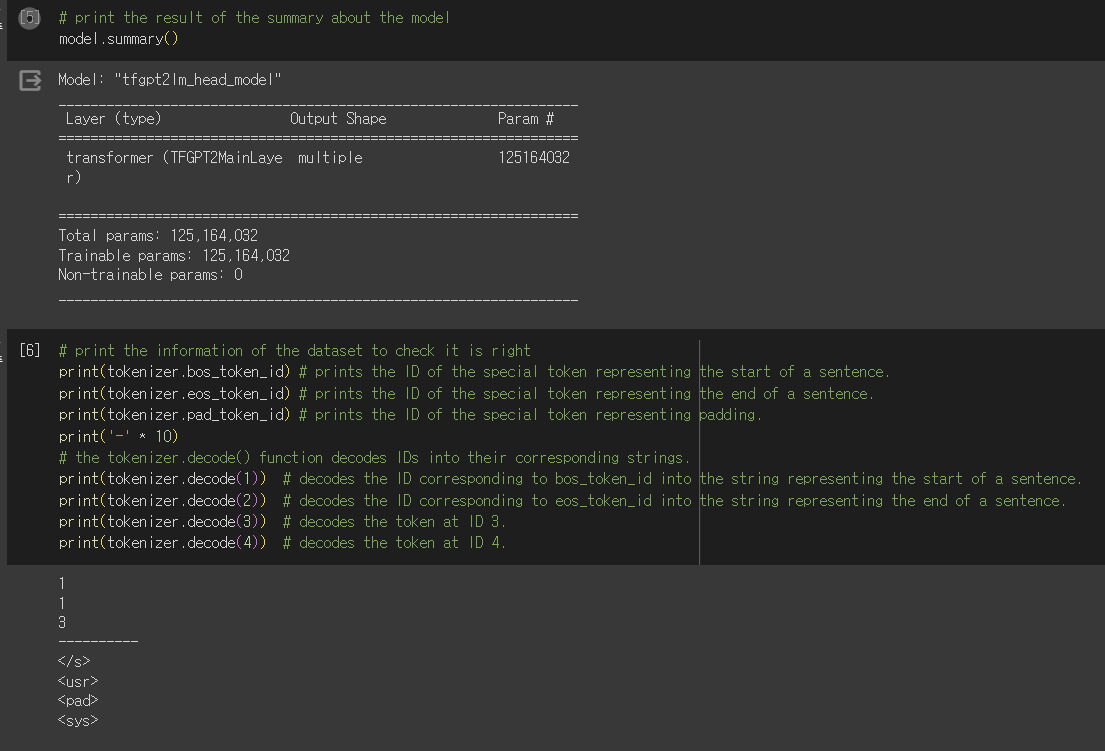


Package를 import하고, tokenizer를 알맞은 dataset과 parameter에 맞게 model 과 tokenizer를 assign합니다.



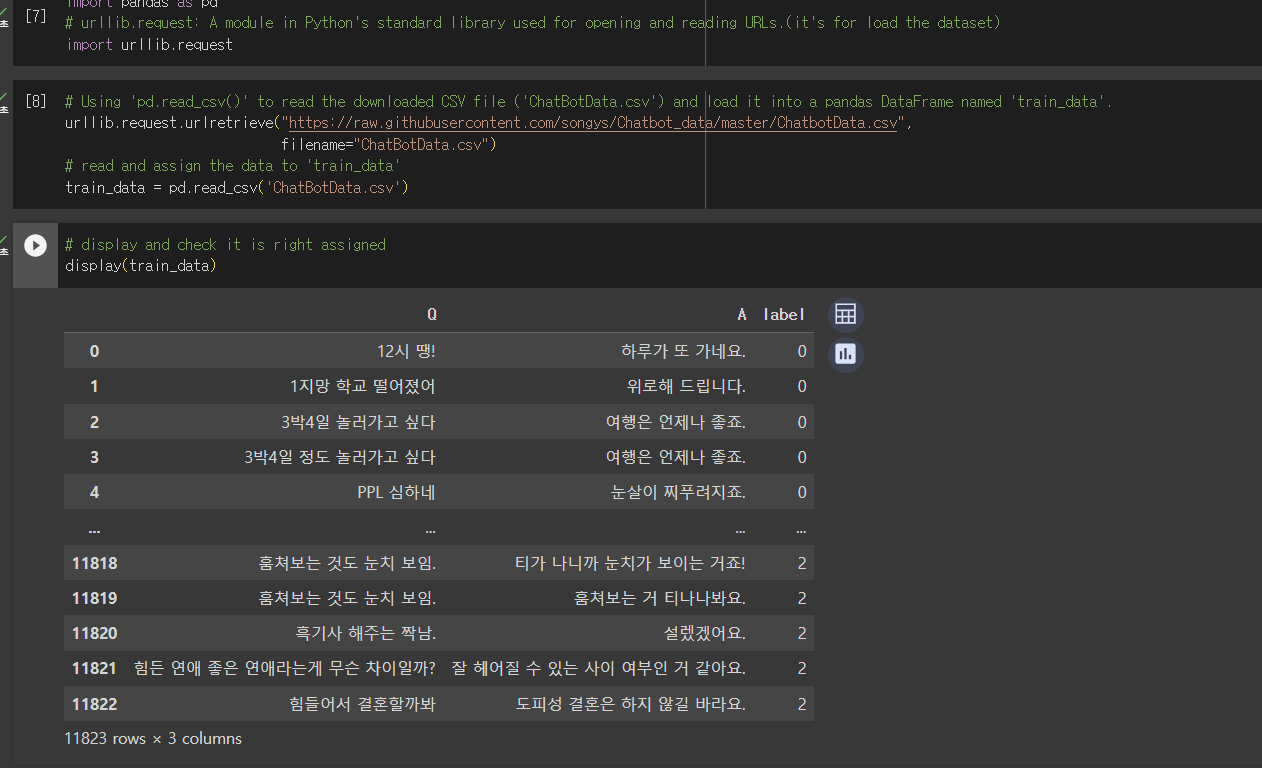
TFGPT2MainLayer 라는 transformer model을 summary 한 결과를 출력합니다.

Tokenizer에 할당된 정보를 print out하여 확인하는 절차를 거칩니다.

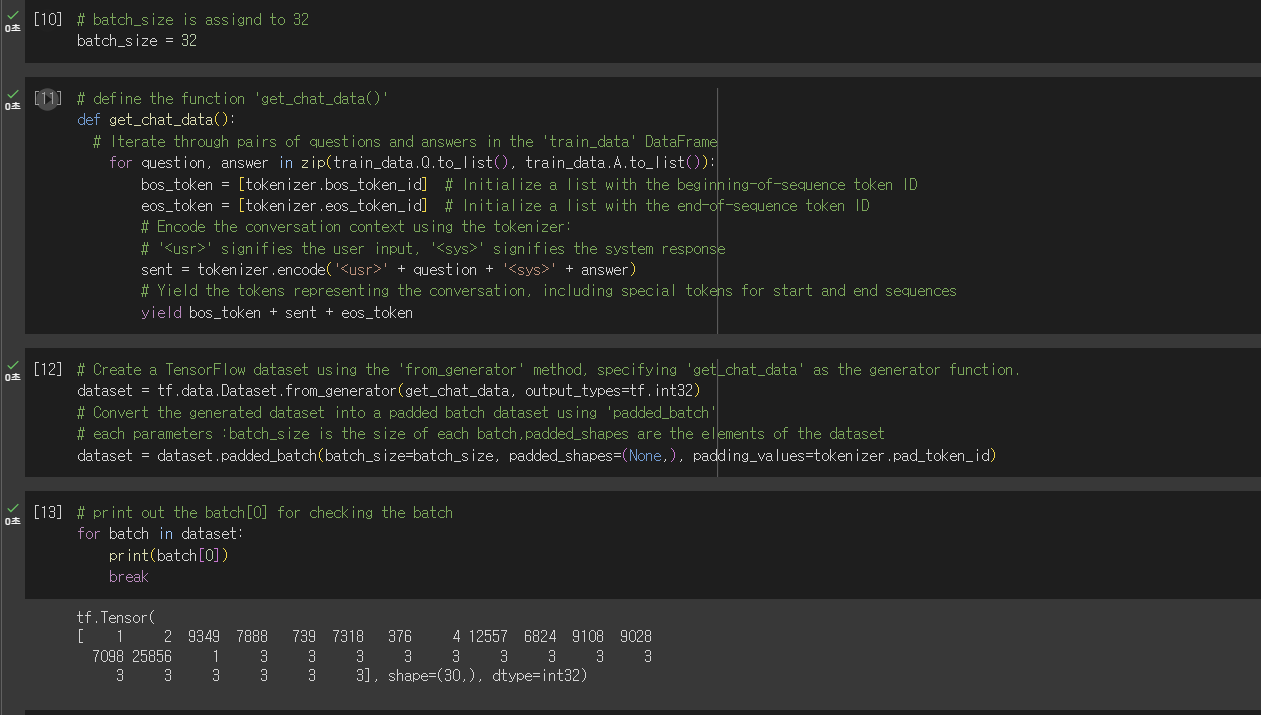


url을 통해 ChatBotData.csv파일을 읽고 다운로드하는 절차를 거칩니다. 그리고 train\_data에 할당합니다.

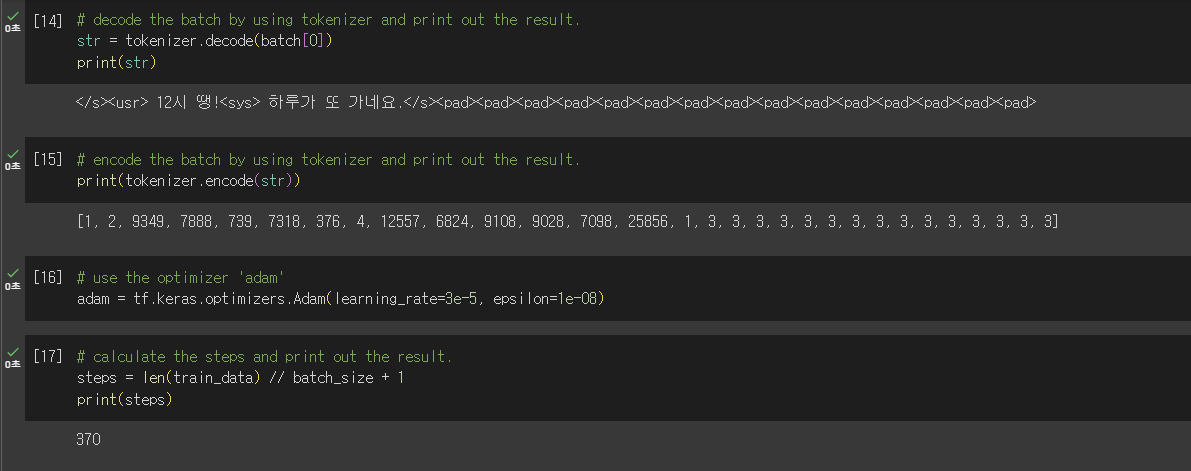
train\_data를 display합니다.



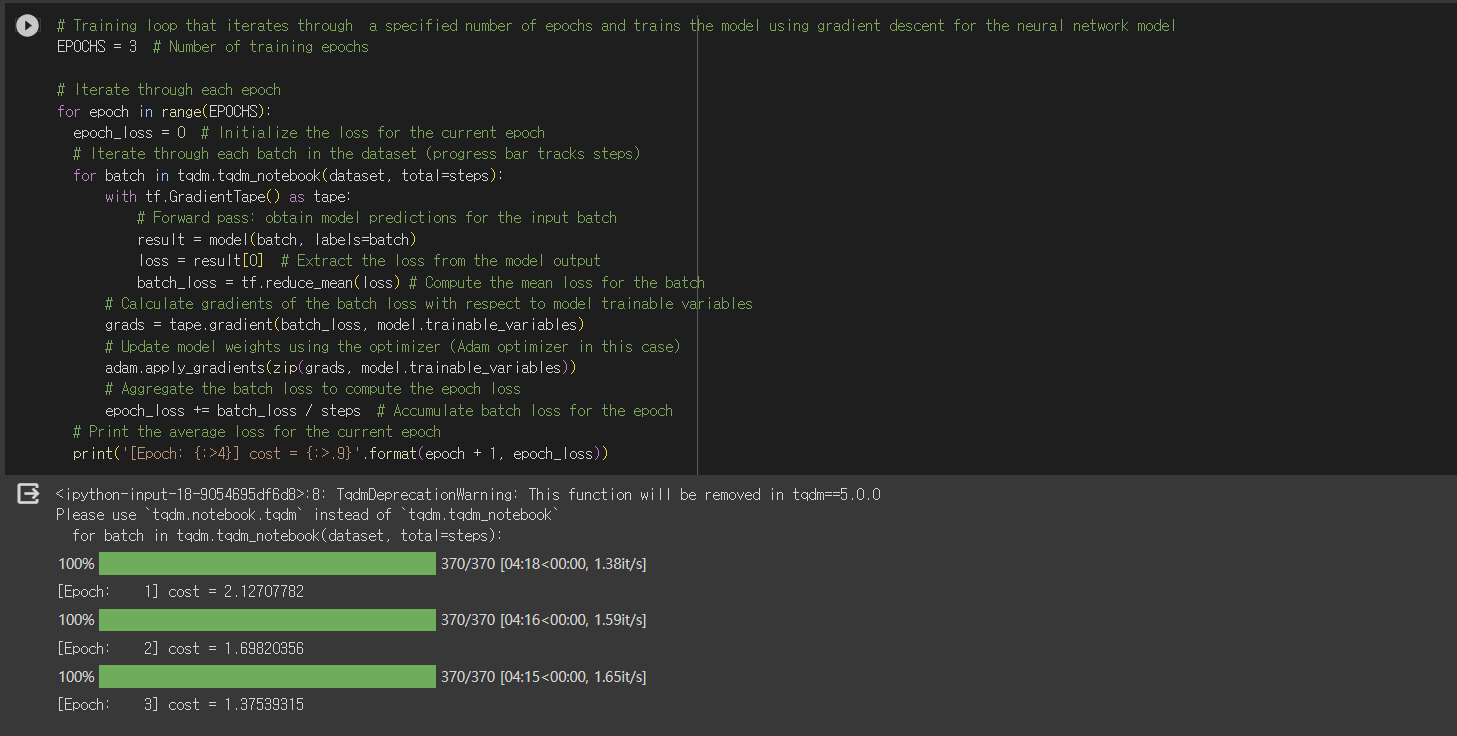
Batch\_size를 32로 할당합니다. 그리고 get\_chat\_data()함수를 통해 tokenizer를 거친 결과가 할당될 수 있도록 합니다. 그리고 dataset변수에 generate function을,그리고 padded\_batch를 할당한 후,batch[0] 를 출력함으로써test결과를 출력합니다.



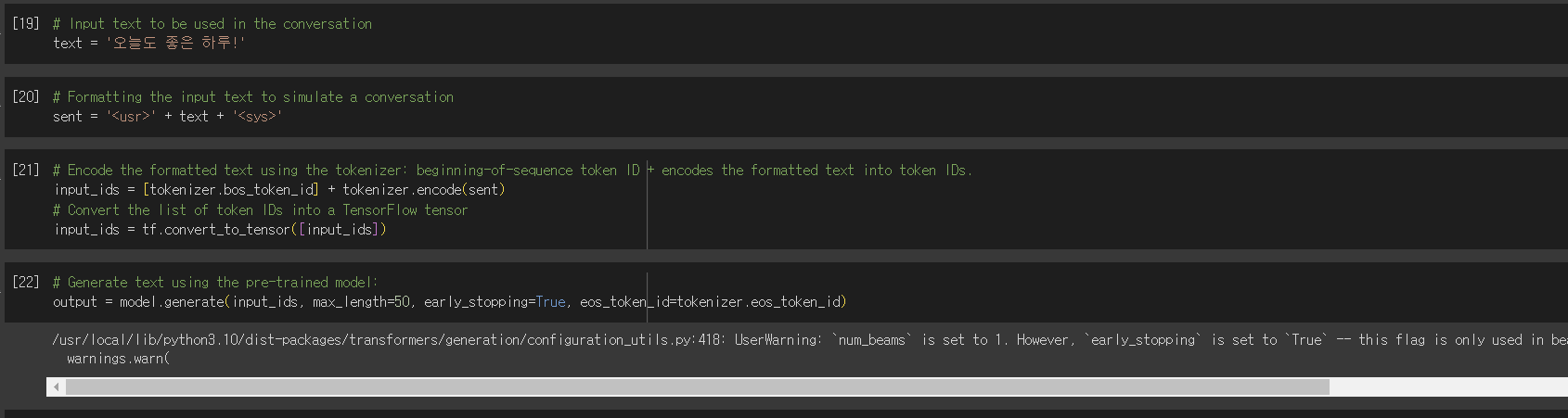
batch[0]를 decode한 결과가 출력될 수 있도록 한 후, encode 한 결과 또한 출력하도록 합니다. 그리고 ‘adam’ optimizer을 설정해서 caculate합니다.



Network model을 사용하고 gradient descent를 사용한 모델을 iterate를 통해 training합니다.

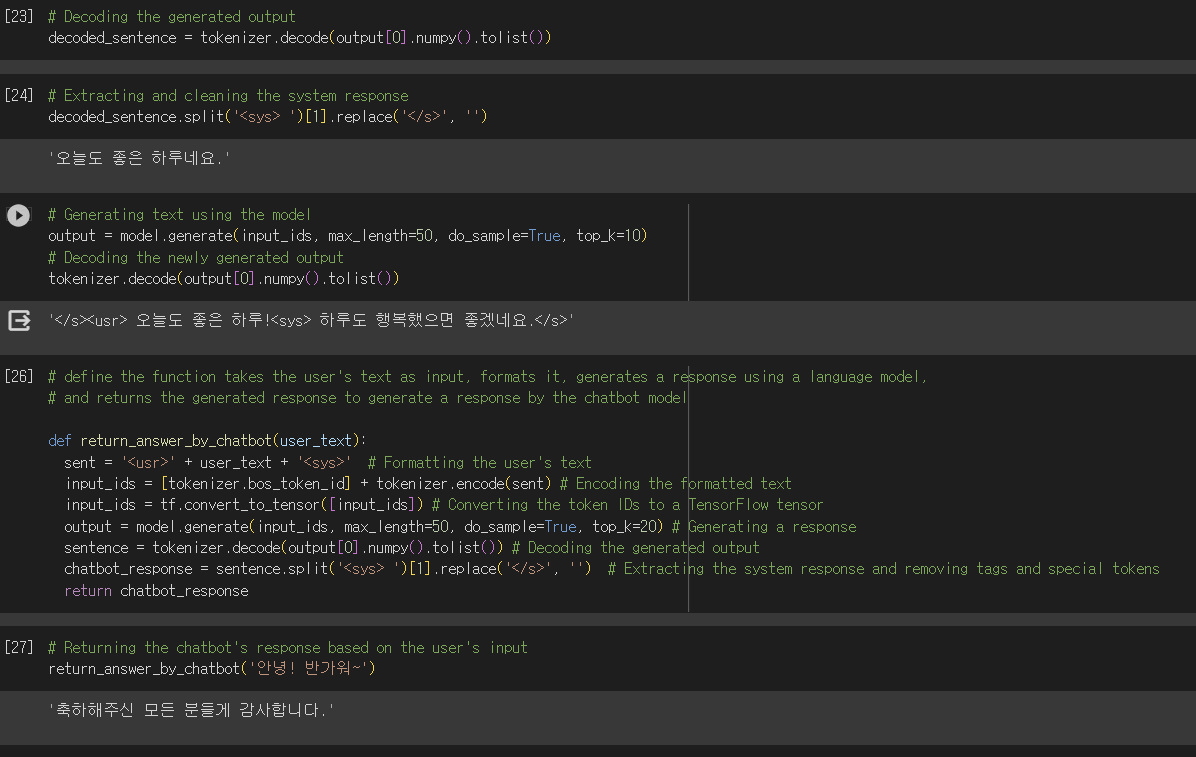


Test data를 assign한 후, formatting하고, encode합니다. 그리고 convert to tensor하고 pre-trained model을 사용한 text를 생성합니다.



Generated 된 output을 decoding합니다.

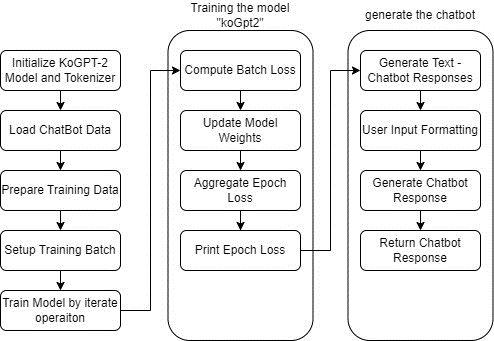
모델을 사용하는 text를 generating하고, chatbot을 통한 답을 return하도록 하는 함수를 통해 임의의 text에 대한 입력 및 반환을 하도록 합니다.





<Discussion (Inference routine with flow chart)>

Flowchart를 통해 전체적인 프로그램의 흐름을 보면 다음과 같습니다.



또한 처음에 tokenizer = AutoTokenizer.from\_pretrained('skt/kogpt2-base-v2', bos\_token='</s>', eos\_token='</s>', pad\_token='<pad>')

model = TFGPT2LMHeadModel.from\_pretrained('skt/kogpt2-base-v2', from\_pt=True)

부분에서 tokenizer의 parameter를 할당하는 부분에서 prarameter를 별도로 지정안했을 때와 참고자료만 보고 다음과 같이 진행했을 때tokenizer = AutoTokenizer.from\_pretrained('skt/kogpt2-base-v2', bos\_token='', eos\_token='', pad\_token='') 와 같이 작성했는데 따라서 그에 따른

print(tokenizer.bos\_token\_id)

print(tokenizer.eos\_token\_id)

print(tokenizer.pad\_token\_id)

의 결과가 각각 51200 51200 0 , 51200 51200 51200 과 같이 나오고 따라서 그 뒤의 결과들도 마찬가지로 고쳐지지 않았습니다. 따라서 expected 결과 중에서 output = model.generate(input\_ids, max\_length=50, do\_sample=True, top\_k=10)

tokenizer.decode(output[0].numpy().tolist()) 의 결과가 '</s><usr> 오늘도 좋은 하루!<sys> 오늘은 좋은 하루로 기억 남을거라 믿어요.</s>인 것을 보고 parameter를 고치니까 print(tokenizer.bos\_token\_id)

print(tokenizer.eos\_token\_id)

print(tokenizer.pad\_token\_id)

원래의 expected result와 그 이후의 것들도 맞게 출력되었습니다. 다만 test text에 대한 결과가 기존의 expected result에 비해서 덜 정확한 것 같기는 하나 데training의 epoch에 대한 cost값이 거의 비슷하였기 때문에 맞게 했을 거라고 생각하고 위 실습과정을 진행하였습니다.