chatbotDataset.py

```python
import numpy as np
from torch.utils.data import Dataset
from transformers import  PreTrainedTokenizerFast
import re

# 챗봇 데이터를 처리하는 클래스를 만든다.
class ChatbotDataset(Dataset):
    def __init__(self, chats, max_len=40):  # 데이터셋의 전처리를 해주는 부분
        self._data = chats
        self.max_len = max_len
        self.q_token = "<usr>"
        self.a_token = "<sys>"
        self.sent_token = '<unused1>'
        self.EOS = '</s>'
        self.MASK = '<unused0>'
        self.BOS = '</s>'
        self.PAD = '<pad>'
        self.UNK = '<unk>'
        self.tokenizer = PreTrainedTokenizerFast.from_pretrained("skt/kogpt2-base-v2",
                                                            bos_token=self.BOS, eos_token=self.EOS, unk_token=self.UNK,
                                                            pad_token=self.PAD, mask_token=self.MASK)

    def __len__(self):  # chatbotdata 의 길이를 리턴한다.
        return len(self._data)

    def __getitem__(self, idx):  # 로드한 챗봇 데이터를 차례차례 DataLoader로 넘겨주는 메서드
        turn = self._data.iloc[idx]
        q = turn["Q"]  # 질문을 가져온다.
        q = re.sub(r"([?.!,])", r" ", q)  # 구둣점들을 제거한다.

        a = turn["A"]  # 답변을 가져온다.
        a = re.sub(r"([?.!,])", r" ", a)  # 구둣점들을 제거한다.

        q_toked = self.tokenizer.tokenize(self.q_token + q + self.sent_token)
        q_len = len(q_toked)

        a_toked = self.tokenizer.tokenize(self.a_token + a + self.EOS)
        a_len = len(a_toked)

        #질문의 길이가 최대길이보다 크면
        if q_len > self.max_len:
            a_len = self.max_len - q_len        #답변의 길이를 최대길이 - 질문길이
            if a_len <= 0:       #질문의 길이가 너무 길어 질문만으로 최대 길이를 초과 한다면
                q_toked = q_toked[-(int(self.max_len / 2)) :]   #질문길이를 최대길이의 반으로
                q_len = len(q_toked)
                a_len = self.max_len - q_len              #답변의 길이를 최대길이 - 질문길이
            a_toked = a_toked[:a_len]
            a_len = len(a_toked)

        #질문의 길이 + 답변의 길이가 최대길이보다 크면
        if q_len + a_len > self.max_len:
            a_len = self.max_len - q_len        #답변의 길이를 최대길이 - 질문길이
            if a_len <= 0:       #질문의 길이가 너무 길어 질문만으로 최대 길이를 초과 한다면
                q_toked = q_toked[-(int(self.max_len / 2)) :]   #질문길이를 최대길이의 반으로
                q_len = len(q_toked)
                a_len = self.max_len - q_len              #답변의 길이를 최대길이 - 질문길이
            a_toked = a_toked[:a_len]
            a_len = len(a_toked)

        # 답변 labels = [mask, mask, ...., mask, ..., <bos>,..답변.. <eos>, <pad>....]
        labels = [self.MASK,] * q_len + a_toked[1:]

        # mask = 질문길이 0 + 답변길이 1 + 나머지 0
        mask = [0] * q_len + [1] * a_len + [0] * (self.max_len - q_len - a_len)
        # 답변 labels을 index 로 만든다.
        labels_ids = self.tokenizer.convert_tokens_to_ids(labels)
        # 최대길이만큼 PADDING
        while len(labels_ids) < self.max_len:
            labels_ids += [self.tokenizer.pad_token_id]

        # 질문 + 답변을 index 로 만든다.
        token_ids = self.tokenizer.convert_tokens_to_ids(q_toked + a_toked)
        # 최대길이만큼 PADDING
        while len(token_ids) < self.max_len:
```

```
        token_ids += [self.tokenizer.pad_token_id]

    #질문+답변, 마스크, 답변
    return (token_ids, np.array(mask), labels_ids)
```