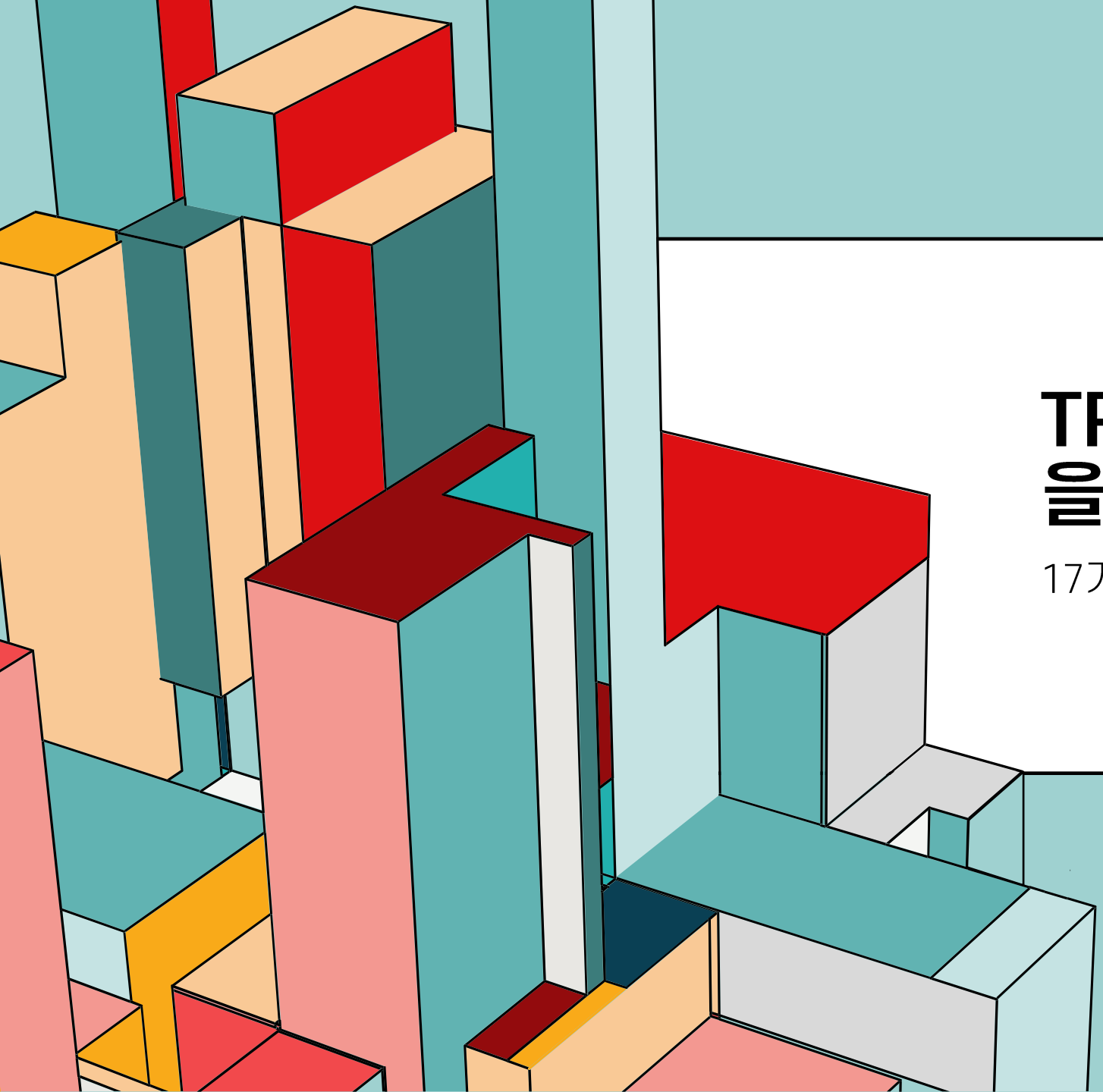
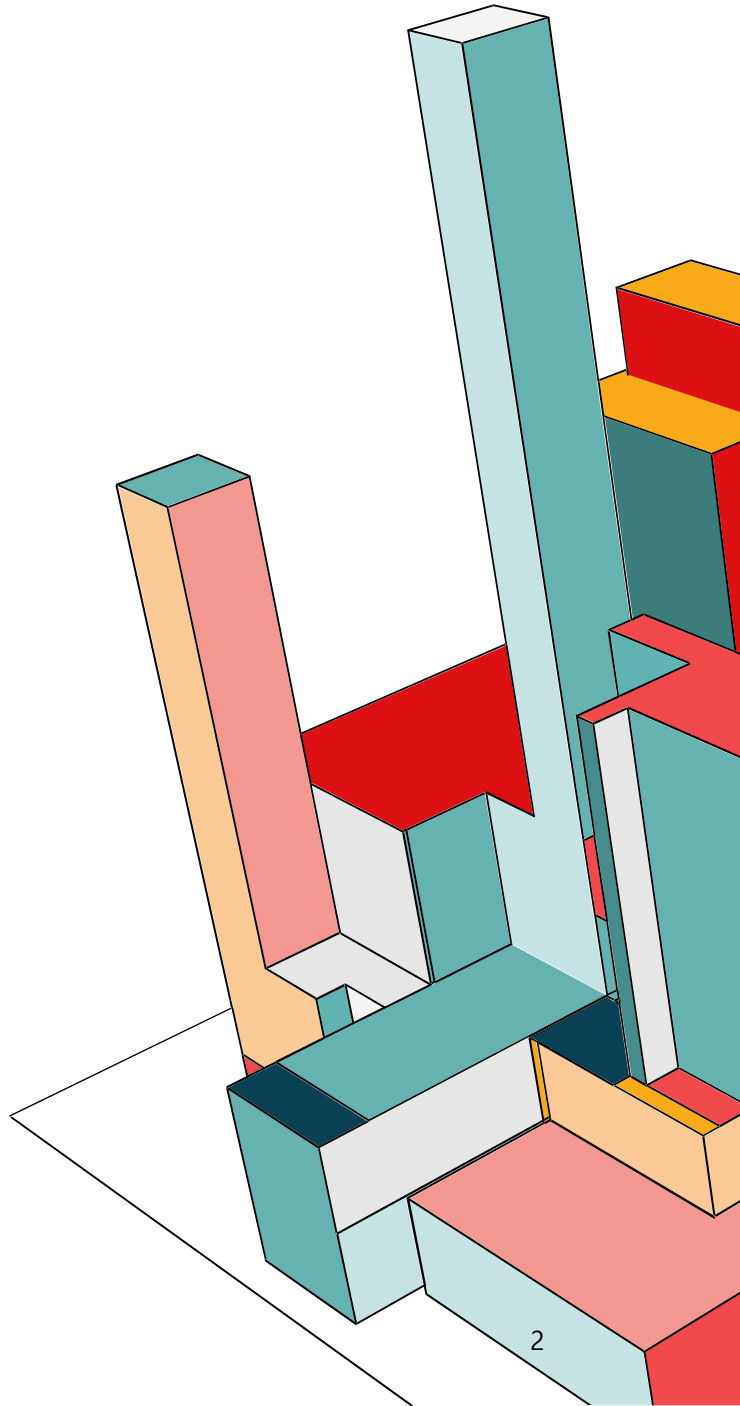


TRANSFORMER 모델을 이용한 표절 탐지

17기 송지훈, 우윤규, 황우현

목차



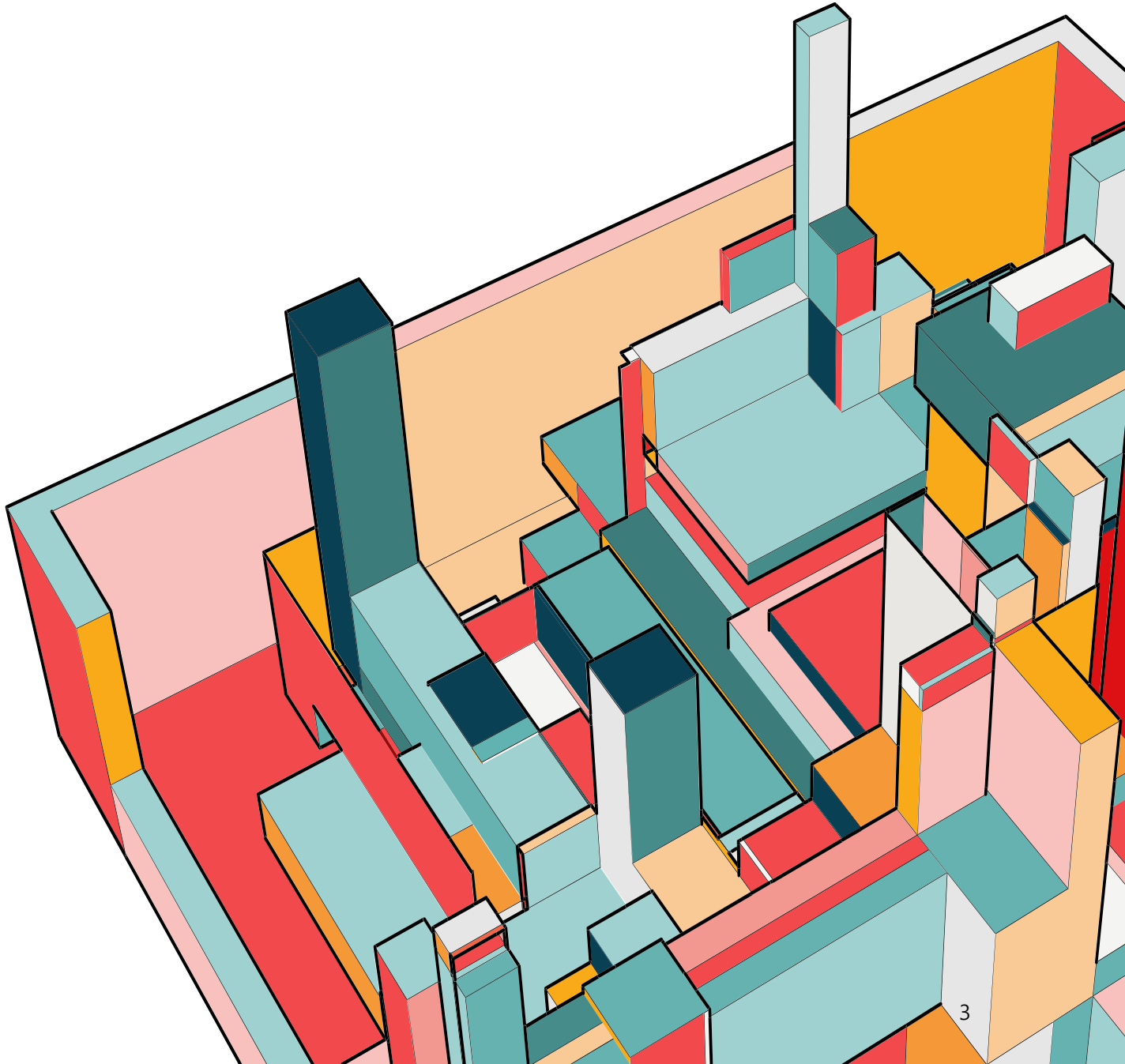
1. 표절 탐지

표절 탐지에는 2가지 주요한 제한사항이 있습니다.

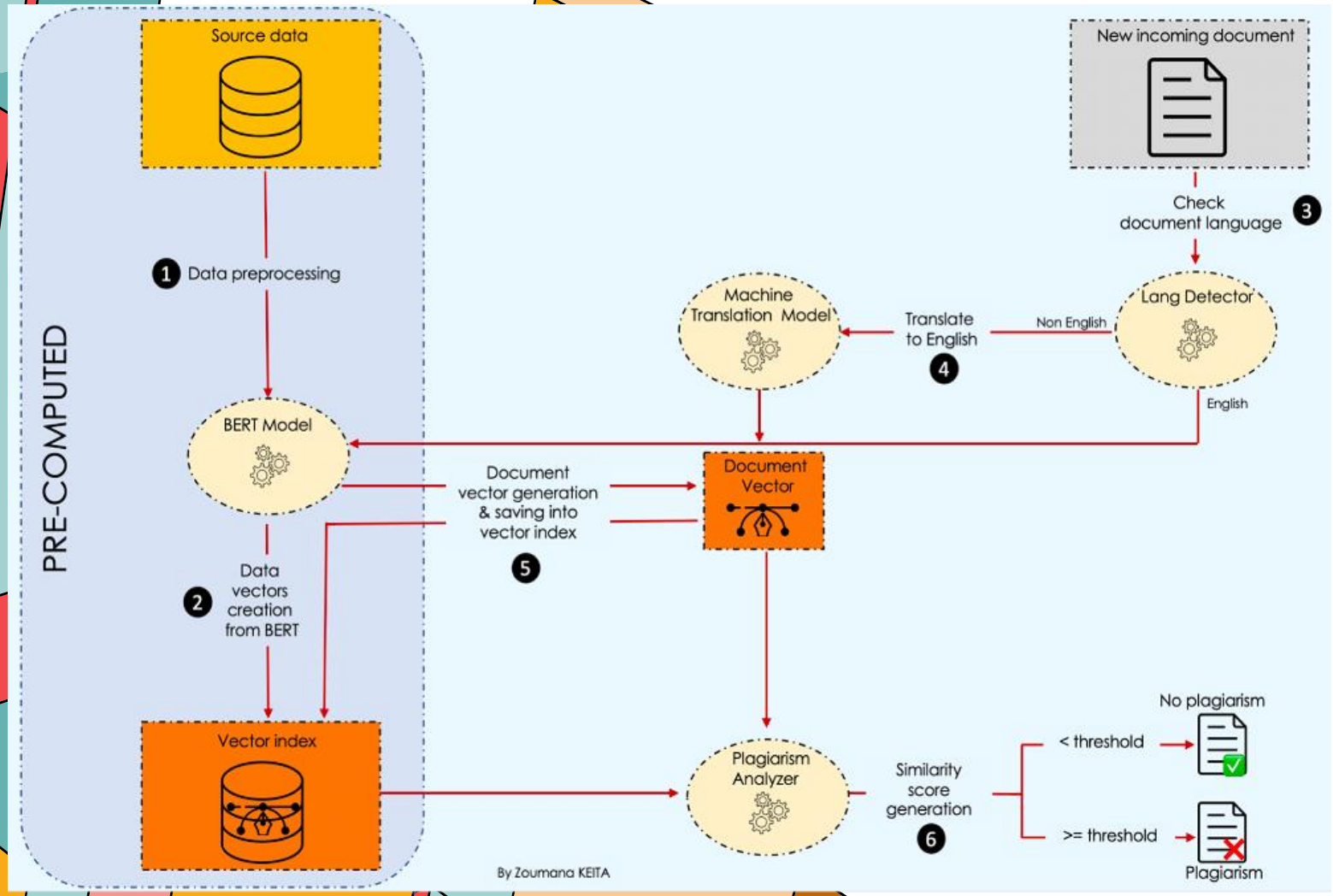
첫 번째로는 전체 문맥을 보면서 동의어와 반의어를 고려하기가 어렵습니다.

두 번째로는 맥락과 아예 다른 내용이 나온다면 탐지하기가 어려울 수 있습니다.

본 프로젝트에서는 이를 극복하기 위해서 Transformer 기반 Pre-trained모델을 활용하였습니다.



WORK FLOW



데이터 베이스 구축

나무위키의 데이터(86만 7천건)을 미리 토큰화 시킨 Database를 미리 구축

Pre-trained 모델 사용

KoBERT 모델을 활용하여 토큰화 된 데이터를 벡터화

유사도 계산

벡터화된 데이터들을 기반으로 유사도 계산(코사인 유사도)

표절 유무 판단

사용자가 미리 설정한 threshold값 기준으로 유사도가 그 값 이상이면 표절, 아니면 정상으로 판단

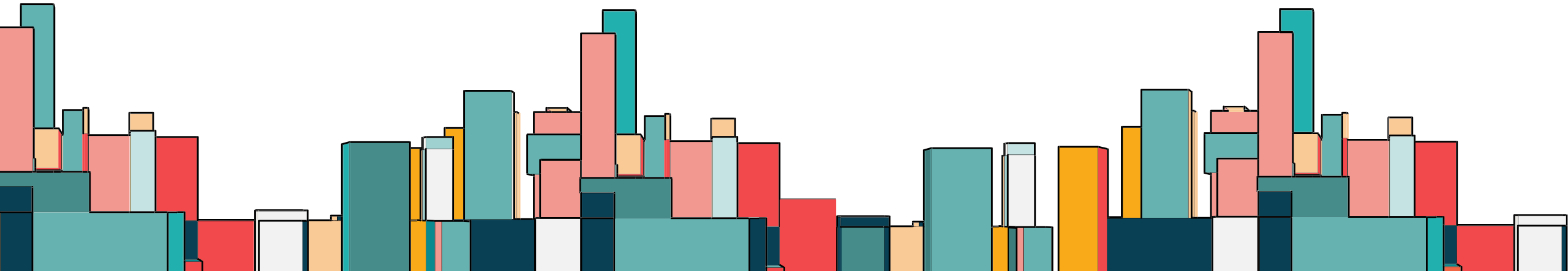
2. 데이터 전처리

1) 단일 문서의 index 생성

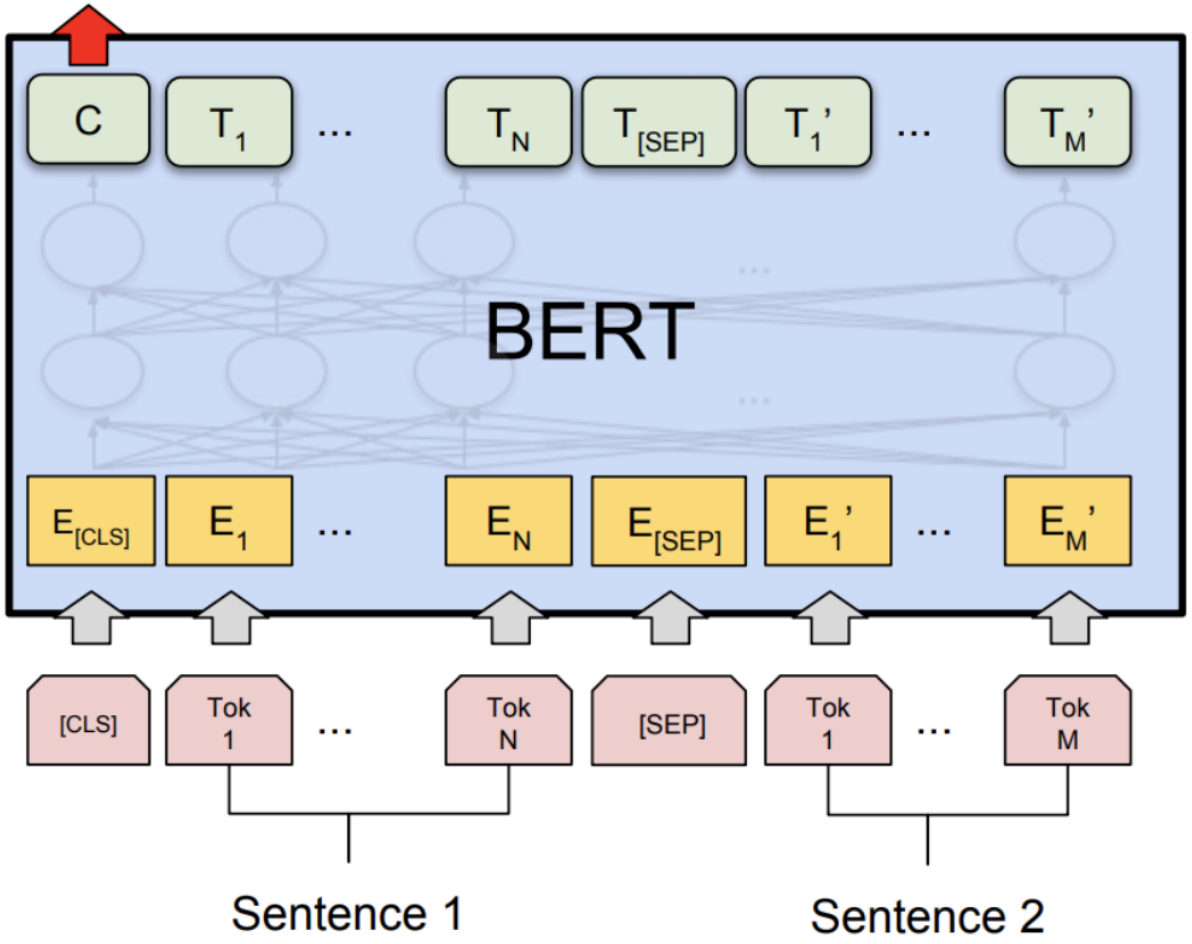
나무위키 데이터들을 Pre-trained된 KoBERT Model의 tokenizer을 활용하여 input_ids, attention_mask를 반환. 또한, 길이를 모두 동일하게 맞춰주기 위해서 Post-padding 수행.

2) 단일 문서의 벡터 표현 생성

1)에서 수행한 input_ids, attention_mask을 기반으로 모델에 넣음으로써 해당 문서를 나타내는 벡터 계산



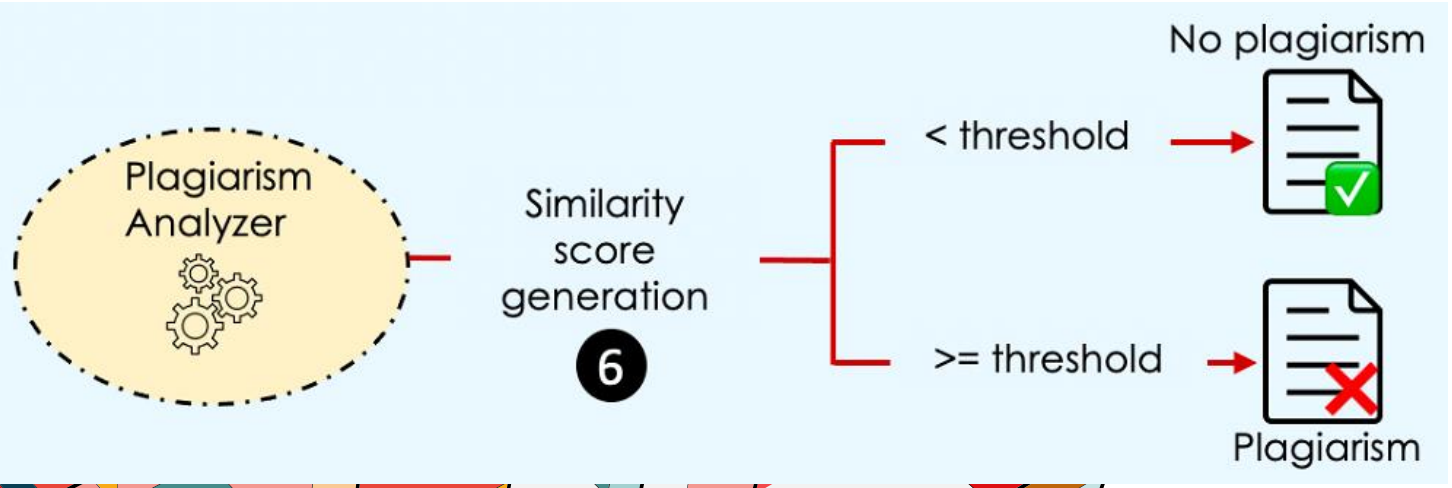
3. KoBERT, 유사도 계산



- ✓ 한국어 위키의 데이터를 기반으로 만든 BERT모델
- ✓ 본 프로젝트에서는 해당 모델을 활용해서 문서 간의 유사도를 계산
- ✓ 이때, BERT모델의 마지막 결과값이 Class Label이 아닌, 그 전의 Vector값을 기반으로 코사인 유사도를 계산하여 문서 간의 유사도를 계산하여 새로운 문서가 나무위키로부터의 표절 유무를 가릴 수 있습니다.

3. KoBERT, 유사도 계산

$$similarity = cos(\Theta) = \frac{A \cdot B}{||A||\ ||B||} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B_i)^2}}$$



좌측의 코사인 유사도를 계산하는 식을 활용하여 새로운 문서가 들어왔을 때, 원래 Database에 있던 문서들과 얼마만큼 유사한지를 나타내는 코사인 유사도를 계산합니다.

또한, 이에 대해서 사용자가 미리 지정해둔 특정 기준(Threshold)를 넘으면 나무위키에서 표절을 해왔다고 볼 수 있습니다.

이렇게 사전 학습된 모델을 통해서 전체적인 문맥을 고려하면서 벡터화 시킨 후에 표절 유무를 가림으로서 처음에 표절 탐지에 있어 제기하였던 문제점을 어느정도 극복할 수 있습니다.

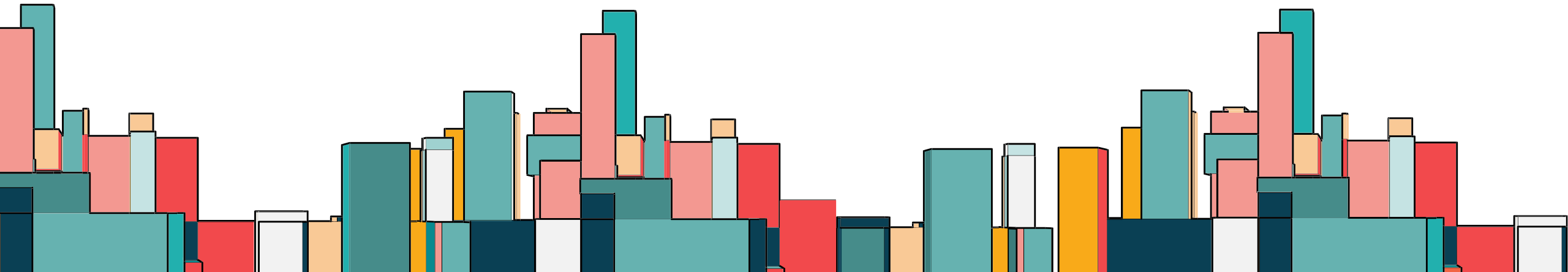
4. 결론

전체적인 Workflow 구축

실제로 대량의 데이터를 데이터베이스화 시킴으로써 쉽게 데이터를 불러들일 수 있게 하였습니다. 또한, 전체적으로 데이터들을 목적에 맞게 전처리시켜 향후에 다른 task가 생겼을 때에도 그에 맞게 잘 대응할 수 있습니다.

사전 학습된 모델 사용

Hugging face에서 사전 학습된 모델 (KoBERT)를 사용해보았고, 이를 통해서 모델을 불러들일 수 있고 목적에 맞게 사용할 수 있다는 점을 알게 되었습니다.
특히, 이번 프로젝트에서는 텍스트 데이터들을 정수 인코딩, 벡터화 시킨 후에 코사인 유사도를 계산한다는 점에서 흥미로웠습니다.



감사합니다