# 과제 1

- 서브워드 학습 데이터와 방법 그리고 하이퍼파라미터(vocab_size) 변경에 따른 차이점 보고서
- Language & AI융합전공 202402050 최재원

---

## 학습 데이터

- **dataset:** wikimedia/wikipedia
- **subset:** 20231101.ko
- **split:** train
- **used:** first 1k rows

---

## 학습 방법

- 데이터셋 로드

```
from datasets import load_dataset

dataset = load_dataset("wikimedia/wikipedia", "20231101.ko",
split="train").select(range(1000))
```

- 텍스트 데이터를 리스트 형식으로 변환

```
texts = [item["text"] for item in dataset]
```

- Tokenizer 인스턴스 변수 초기화

```
from tokenizers import Tokenizer, models, trainers, pre_tokenizers

tokenizer = Tokenizer(models.BPE(unk_token="[UNK]"))
```

- 공백을 분리하고, 단어 단위로 나눌 수 있도록 설정

```
tokenizer.pre_tokenizer = pre_tokenizers.Whitespace()
```

- Trainer 설정

```
trainer = trainers.BpeTrainer(
    vocab_size=vocab_size,
```

```
    min_frequency=2,
    special_tokens=["[UNK]", "[CLS]", "[SEP]", "[PAD]", "[MASK]"],
)
```

- 토크나이저 학습

```
tokenizer.train_from_iterator(texts, trainer=trainer)
```

---

## 하이퍼파라미터(vocab_size) 변경에 따른 차이

- 사용된 코드

```
from tokenizers import Tokenizer, models, trainers, pre_tokenizers

for vocab_size in [1600, 16_000]:
    tokenizer = Tokenizer(models.BPE(unk_token="[UNK]"))
    tokenizer.pre_tokenizer = pre_tokenizers.Whitespace()

    trainer = trainers.BpeTrainer(
        vocab_size=vocab_size,
        min_frequency=2,
        special_tokens=["[UNK]", "[CLS]", "[SEP]", "[PAD]", "[MASK]"],
    )

    tokenizer.train_from_iterator(texts, trainer=trainer)
    tokenizer.save(f"bpe_{vocab_size}.json")

    text = "bpe 토크나이저를 {}개 타입으로 학습해 테스트 중입니
다.".format(vocab_size)
    encoded_t = tokenizer.encode(text)
    print(f"===== test bpe {vocab_size} =====")
    print(encoded_t.tokens)
    print()
```

- 출력

```
===== test bpe 1600 =====
['b', 'p', 'e', '토', '크', '나', '이', '저', '를', '1', '6', '0', '0',
'개', '타', '입', '으', '로', '학', '습', '해', '테', '스', '트', '중',
'입', '니', '다', '.']

===== test bpe 16000 =====
['b', 'pe', '토', '크', '나이', '저', '를', '16', '000', '개', '타입', '으
로', '학습', '해', '테스트', '중', '입니다', '.']
```

- 설명
  - `vocab_size=1.6k`인 경우, 글자 단위로 나뉨
  - `vocab_size=16k`인 경우, 자주 등장하는 쌍이 하나의 단위로 **merge**됨
  - vocab_size를 키우면 자주 등장하는 것들이 더 많이 쌍을 짓고, 출력 결과와 같이 여러 글자가 하나의 토큰을 이루는 것을 볼 수 있음