# Text Summarization (NLP 과제)

| | |
|---|---|
| 종류 | 기타 |
| 학습 일자 | @April 30, 2022 |
| 참고자료 | 문서 요약 참고 링크 |

# 문서 요약 (Text Summarization)

## 요약 Methods

### 1. Extractive Method

- 전통적인 문서 요약 방식
- 중요한 문장을 구분하여 요약본에 포함하는 방식 (즉, 원본 문서의 문장 전체가 그대로 요약본에 포함됨)

### 2. Abstractive Method

- 중요한 부분을 구분하여 문맥을 파악하고 문장을 새롭게 재구성하는 방식
- 중요한 문장이 가능한 가장 짧은 문장으로 표현

## Extractive Methods

### 1) Gensim의 Text Rank

자주 나타나는 단어들을 중요하다고 판단. 이에 근거하여 문서의 각 문장에 점수를 부여해 top-rank 문장들을 요약본에 포함함.

### 2) Sumy 활용 문서 요약

파이썬에서 제공하는 문서 요약 라이브러리로 아래 알고리즘들을 포함함.

1. **LexRank**

a. 다른 문장들과 유사한 의미를 가진 문장이 중요할 가능성이 높다고 판단. rank가 높을 수록 요약본에 포함될 확률이 높음.

2. **LSA (Latent Semantic Analysis)**
   a. 비지도학습 알고리즘으로, 특이값 분해(SVD) 방식 활용
   b. DTM 이나 DTM에 단어의 중요도에 따른 가중치를 주는 TF-IDF 행렬은 단어의 의미를 고려하지 못한다는 단점이 있음. LSA는 DTM이나 TF-IDF에서 절단된 SVD를 사용하여 차원을 축소시키고 단어들의 잠재적인 의미를 끌어냄.
   c. LSA는 쉽고 빠르게 구현 가능. 단어의 잠재적인 의미를 이끌어낼 수 있어 문서의 유사도 계산 등에서 좋은 성능을 보여줌.
   d. 이미 계산된 LSA에 새로운 데이터를 추가하여 계산하려고 하면 처음부터 다시 계산해야 한다는 단점. (즉, 새로운 정보에 대해 업데이트 어렵)

   참고: https://wikidocs.net/24949

3. **Luhn**
   a. TF-IDF 기반 문서 요약 방식

4. **KL-Sum**
   a. 원본 문서와의 divergence를 계산하는 KL Divergence값이 최소화 될 때까지 요약문에 문장을 더하는 방식

   참고: https://iq.opengenus.org/k-l-sum-algorithm-for-text-summarization/

# Abstractive Methods

## 1) T5 Transformers

encoder-decoder 모델로 각 문장마다 학습 과정을 거쳐 target text가 생성됨

참고: https://paperswithcode.com/method/t5

## 2) BART transformers

sequence-to-sequence 모델로 BERT와 GPT를 일반화한 것. Noising의 유연성이 장점.

참고: https://chloelab.tistory.com/34#:~:text=BART

### 3) GPT-2 Transformers

비지도학습 기반 모델로 flexible transfer가 가능하며, fine-tuning 없이 사용 가능.

참고: https://supkoon.tistory.com/25#:~:text=GPT-2