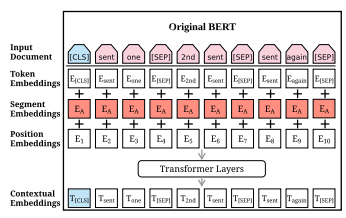
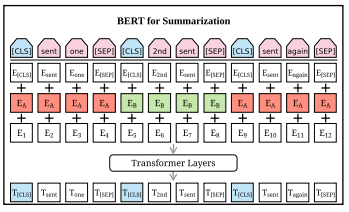
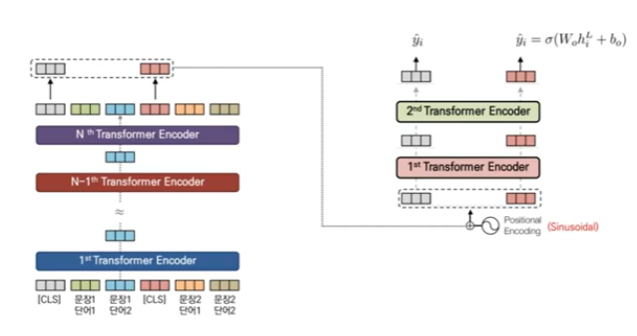
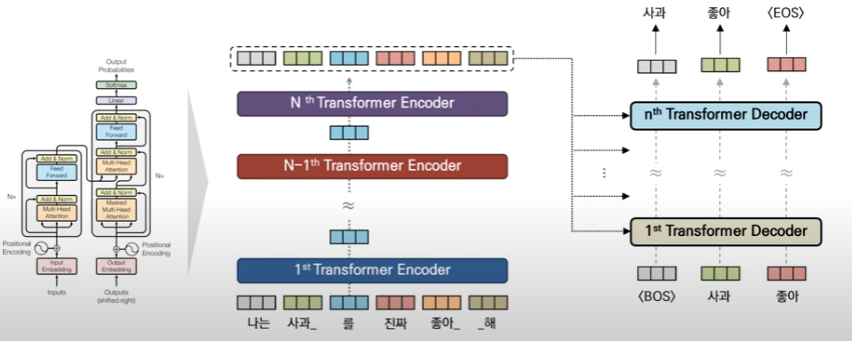
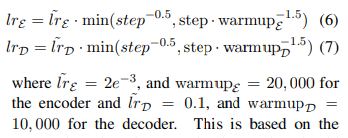
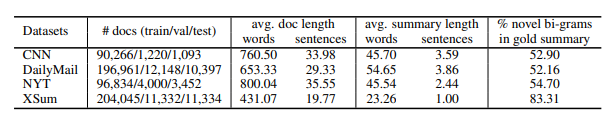
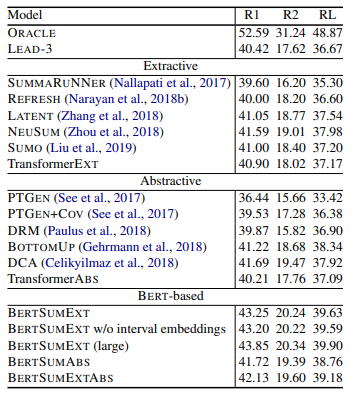
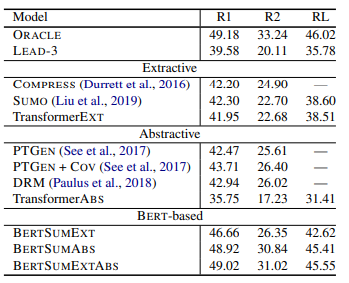
Text Summarization with Pretrained Encoder(2019) 리뷰

1. Introduction

* ELMo, GPT, BERT와 같은 Pretrained Language Model은 감성분석, 질의응답, 개체명 인식 등 다양한 자연어 처리 작업에서 뛰어난 성능을 보이고 있으며 다양한 downstream task를 위해 fine-tuning이 가능함.
* 요약은 개별 단어 및 문장의 의미를 넘어서는 광범위한 자연어 이해를 필요로 하며 abstract 방식은 원본 텍스트에 포함되지 않은 새로운 단어와 구문을 생성하는 능력이 필요하며 extract 방식은 문장을 포함할지 여부를 나타내는 label을 사용하여 binary classification 작업으로 정의됨.
* 본 논문에서는 pretrained BERT를 기반으로 source text의 representation을 추출한 뒤, 이를 사용하는 Transformer 기반 abstract summary model, extract summary model 2가지를 제안
* Background
  + BERT에 대한 설명(wiki 링크 추가하기)  
    

2. Fine-tuning BERT for Summarization

* Summarization Encoder
  + - 기존 BERT에서 사용한 Input 방식  
      
      * 각 문장은 [SEP] 토큰으로 구분
      * Input의 시작에만 [CLS] 토큰이 들어감
        + 이는 Input에 대한 classification 문제 학습 시 사용되며 classification은 개별 단어나 input 전체에 대해서만 수행 가능(ex, 감성분석)
      * Segment Embedding은 두 종류의 텍스트(문장, 문단, 문서 등)를 구분하기 위한 embedding으로, QA와 같은 task 수행 시, 질문과 본문을 입력 받고, 둘을 구분하기 위해 질문과 본문의 Segment embedding을 각각 다르게 사용하지만 각 질문과 본문은 다수의 문장으로 구분될 수 있음.  
        (ex, 감성분석의 경우 텍스트 구분이 필요 없으므로 Segment embedding은 모두 동일한 값)
    - 논문에서 사용한 Input 방식  
      
      * 각 문장의 시작위치마다 [CLS] 토큰을 삽입하여 input을 이루는 개별 문장마다 classification 수행 가능
      * Interval segment embedding : Segment embedding은 동일하게 두가지 값만 갖되, 홀수/짝수번째 문장끼리만 동일한 embedding 값을 갖도록 설정
        + 초반 transformer layer에서는 인접한 문장간 관계를 학습 가능
        + 후반 tansformer layer에서는 self-attention을 통해 여러 문장간 관계를 학습 가능
  + Extractive Model(BERTSumExt)  
    
    - (사진 왼쪽)BERT를 거쳐 나온 output 중 문장별 [CLS] 토큰 벡터만 선택
    - (사진 오른쪽)선택한 [CLS] 토큰들을 두 개의 layer로 이루어진 Transformer Encoder의 input으로 사용(논문에서 1,2,3개 실험 결과 2개의 layer가 가장 좋은 성능을 보였다고 함)
    - 이후, binary classification을 진행하여 해당 문장을 요약에 포함할지 안할지를 분류하게 됨
  + Abstract Model(BERTSumAbs)  
    
    - Pretrained BERTSum encoder(사진 왼쪽) 위에 Transformer Decoder(사진 오른쪽)를 추가해 abstract 요약 수행 -> 논문에서는 6개의 Decoder layer를 사용했다고 함
    - Decoder에서 [BOS] 토큰을 입력 받아 요약문의 첫 단어를 예측하며 이후 예측된 단어를 다시 입력으로 받아 다음 단어를 예측하는 방식으로 [EOS] 토큰이 출력될 때 까지 요약문을 생성하게 됨(auto-regressive)
    - Encoder는 pretraining이 이미 진행된 BERT(BERTSum)이고, Decoder는 처음부터 학습을 해야하므로 Fine-tuning 과정에서 불균형이 있을 수 있음. 이를 해소하고자 Encoder와 Decoder의 learning rate, warm-up step을 다르게 사용  
      
  + BERTSumExtAbs
    - 추가적으로, Two-stage fine-tuning 방식(BERTSumExt fine-tuning 된 Encoder -> BERTSumAbs 과정 fine-tuning)을 사용해 학습한 BERTSumExtAbs 모델이 단순 BertSum Encoder로 fine-tuning한 BERTSumAbs 보다 좋은 성능을 보였다고 함
      * (directed, unweighted)
        + 중요도 산출 공식  
          
        + i : 현재 기준 vertex indx, j : 현재 기준 vertex와 연결된 vertex index  
          d : randomness를 부여하는 damping factor(0~1 사이, 논문은 0.85)  
          초기 score: 1 or 랜덤 지정  
          모든 vertex의 점수가 수렴할 때까지 반복  
          수렴조건 : (현재 iter 중요도) – (이전 iter 중요도) < threshold
        + 예를들어 ‘criteria’의 중요도는 ‘criteria’ vertex로 화살표 가 향하는 ‘natural’, ‘numbers’ vertex 중요도 점수로 계산이 됨  
          즉, 위 시그마에서 In(V ‘criteria’) = {V ‘natural’, V ‘numbers’} 가 됨  
          따라서, natural의 중요도에 natural의 out집합 원소 개수만큼 나누고, numbers의 중요도에 numbers의 out 집합 원소 개수만큼 나눈 후, 더하여 damping factor 적용하면 criteria 중요도가 산출됨
  + Experiment
    - Datasets  
      
      * 존재하는 대부분 요약 데이터셋 정답의 경우 abstract 기준으로 만들어졌으므로, extract 요약문은 abstract 요약문과 가장 유사한(Rouge-2 f1 score 기준) 원본 문서 내 문장들을 greedy 하게 선택하여 자체 생성(ORACLE)
      * 특히, CNN/DailyMail dataset은 extract 요약문과 유사하고, XSum은 대부분 새로운 단어로만 구성되어 abtractive함
    - Results
      * 공통 baseline모델
        + ORACLE : greedy 하게 추출된 extract summary
        + LEAD-3 : 첫 세 문장을 추출한 extract summary
      * CNN/DailyMail  
        
      * NYT  
        
      * XSum  
        