[논문리뷰] BART : Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension

https://arxiv.org/abs/1910.13461#:~:text=We%20present%20BART%2C%20a%20denoising%20autoencoder%20for%2

250911 BOAZ WEEK9 과제

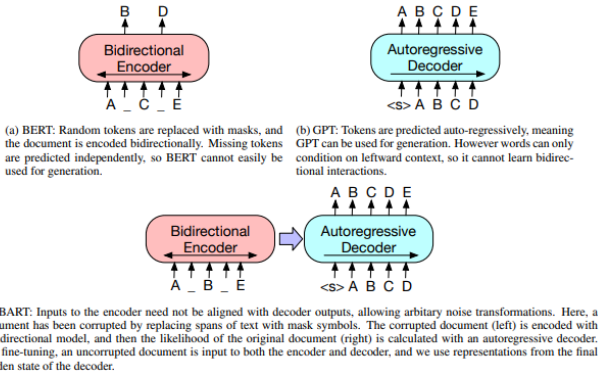
❓ 제안 배경

- 기존 Encoder-Decoder 모델의 한계
  - BERT → Encoder 기반. bidirection-attention을 통한 이해 중심. 텍스트 생성에 한계.
  - GPT → Decoder 기반. auto-regressive decoder기반으로 생성 중심. 이해에 한계

BART(Bidirectional Auto-Regressive Transformer) = BERT + GPT 의 범용적 Pre-Traning seq2seq 모델

❗ BART

Bidirectional Encoder + Auto-Regressive Decoder 를 결합한 Transformer 모델
→ pre-training 과 fine-tuning 이 가능



(a) BERT: Random tokens are replaced with masks, and the document is encoded bidirectionally. Missing tokens are predicted independently, so BERT cannot easily be used for generation.

(b) GPT: Tokens are predicted auto-regressively, meaning GPT can be used for generation. However words can only condition on leftward context, so it cannot learn bidirectional interactions.

(c) BART: Inputs to the encoder need not be aligned with decoder outputs, allowing arbitrary noise transformations. Here, a document has been corrupted by replacing spans of text with mask symbols. The corrupted document (left) is encoded with a bidirectional model, and then the likelihood of the original document (right) is calculated with an autoregressive decoder. For fine-tuning, an uncorrupted document is input to both the encoder and decoder, and we use representations from the final hidden state of the decoder.

💡 Pre-Traning 방식

(1) text를 noise-function에 의해 손상시키고 [Corruption in Encoder]

(2) Seq2Seq 모델이 원래 텍스트를 재구성 [Reconstruction in Decoder]

⇒ Bart는 **손상 입력 → 복원**이라는 denoising autoencoder 를 seq2seq로 확장한 것!

✏️ Architecture

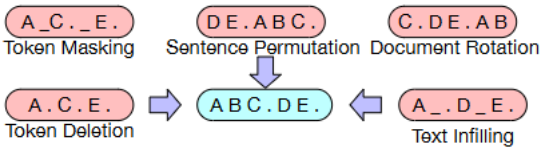
(Bert와의 차이점 위주)

- 활성화함수로 GeLU 사용
- 파라미터의 초기화: N(0, 0,02)
- (base model) = encoder 6 + decoder 6, (large model) = encoder 6 + decoder 6
- 차이점 1: Seq2Seq 구조니까 Decoder 의 각 layer는 마지막 Encoder layer랑 cross-attention 수행
  - Transformer 랑 동일하다고 보면 됨!
- 차이점 2: Bert는 Word prediction 전에 FFN 추가 사용, BART는 X
- 차이점 3: Bart가 Bert보다 파라미터 수가 10% 정도 더 크다

🌟 Pre-Training

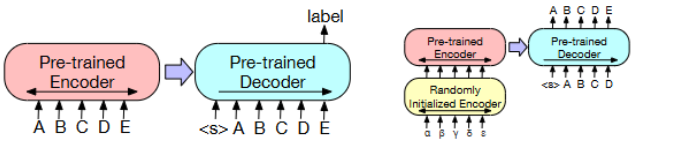
문서 손상 → 디코더의 출력과 원래 문 간의 CE를 최적화하는 과정

[Input noising 방법]



1. **Token Masking**
    a. random token을 뽑아서 [MASK] 토큰으로 대체
2. **Token Deletion**
    a. random 토큰을 뽑아서 input에서 삭제
    b. Masking은 [MASK]를 맞추는 목적, Deletion은 어느 위치에서 삭제했는지 맞추는 목적
3. **Text infilling** → 가장 성능 👍🏻
    a. 람다 = 3인 포아송 분포에서 연속된 span 단위로 샘플링 → 단일 [MASK] 토큰으로 대체
    b. 어디가 빈칸이고, 토큰이 몇개가 없는지 맞추는 목적
4. **Sentence Permutation** (`ABC.DE` → `DE.ABC.`)
    a. 마침표 기준으로 문장 split → 순서 섞기
5. **Document Rotation (** `ABC.DE` → C 선택 → `C.DE.AB`)
    a. uniform random 하게 토큰 선택 → 문장의 맨 앞에 위치하도록 회전
    b. 문장의 시작점을 예측하는 목적

🌟 Fine-Tuning



(a) To use BART for classification problems, the same input is fed into the encoder and decoder, and the representation from the final output is used.

(b) For machine translation, we learn a small additional encoder that replaces the word embeddings in BART. The new encoder can use a disjoint vocabulary.

Figure 3: Fine tuning BART for classification and translation.

[Tasks 별 구조]

1. Sequence Classification (ex. 감성분석, NLI)
    a. 최종 디코더 토큰의 최종 hidden state가 새로운 multi-class linear classifier에 입력
    b. 그림 속 (a)
2. Token Classification (ex. NER)
    a. 디코더의 상단의 hidden state가 token 각각의 representaiton
3. Sequence Generation (ex. summarization, QA, dialogue)
    a. AR 디코더가 있으므로, 바로 fine-tuning 가능
    b. 인코더에는 input sequence가 들어가고, 디코더는 AR output을 내놓도록
4. Machine Translation (ex. 다국어 변역)

a. 추가 인코더 사용 - Bart는 단일 언어에 대해서만 학습하기 때문!

b. 예를 들어 한국 → 영어 번역일 경우,
한국어 text가 additional 인코드에 들어오고 → 영어로 바꾸고 → 그것이 영어 Pre-trained 인코더에 들어가고 → 디코더에서 영어 output을 낸다.

c. 그림 속 (b)

🧪 Experiments

1️⃣ Comparing pre-training objectives

▼ 사전학습 목표

- **Language Model (LM)**
  - GPT처럼 왼쪽→오른쪽으로 단어를 예측하는 방식
- **Permuted LM (XLNet 기반)**
  - 토큰 일부(1/6)를 뽑아 순서를 섞고, 순차적으로 예측
- **Masked LM (BERT 방식)**
  - 전체의 15% 토큰을 [MASK]로 가리고 맞추기
- **Multi Masked LM (UniLM 확장)**
  - MLM에 추가적인 self-attention 제약을 둬서 더 다양한 조건을 학습
- **Masked Seq-to-Seq (MASS 방식)**
  - 입력의 절반을 가리고, seq2seq 구조로 복원 학습

▼ Tasks

- **SQuAD**: 위키피디아 기반 질의응답 (Extractive QA)
- **MNLI**: 두 문장의 관계(참, 거짓, 중립) 분류
- **ELI5**: 긴 형태의 질문-답변 데이터셋 (Abstractive QA)
- **XSum**: 뉴스 기사 요약 (짧고 핵심 요약)
- **ConvAI2**: 대화 응답 생성 (대화 맥락 + persona 기반)
- **CNN/DM**: 뉴스 기사 요약 (긴 요약, 대표적인 dataset)

Model	SQuAD 1.1 F1	MNLI Acc	ELI5 PPL	XSum PPL	ConvAI2 PPL	CNN/DM PPL
BERT Base (Devlin et al., 2019)	88.5	**84.3**	-	-	-	-
Masked Language Model	90.0	83.5	24.77	7.87	12.59	7.06
Masked Seq2seq	87.0	82.1	23.40	6.80	11.43	6.19
Language Model	76.7	80.1	**21.40**	7.00	11.51	6.56
Permuted Language Model	89.1	83.7	24.03	7.69	12.23	6.96
Multitask Masked Language Model	89.2	82.4	23.73	7.50	12.39	6.74
BART Base						
w/ Token Masking	90.4	84.1	25.05	7.08	11.73	6.10
w/ Token Deletion	90.4	84.1	24.61	6.90	11.46	5.87
w/ Text Infilling	**90.8**	84.0	24.26	**6.61**	**11.05**	5.83
w/ Document Rotation	77.2	75.3	53.69	17.14	19.87	10.59
w/ Sentence Shuffling	85.4	81.5	41.87	10.93	16.67	7.89
w/ Text Infilling + Sentence Shuffling	**90.8**	83.8	24.17	6.62	11.12	**5.41**

- **Results**
  - BART with Text Infilling이 대체적으로 성능이 좋음
  - LM은 일부 task(ELI5)만 잘하고, MLM/PLM은 생성에 약함
  - 동일 objective라도 세부 설계에 따라 성능 차이가 큼 → 항상 pre-training 방식 + 아키텍처 + 세부 기법 같이 고려 필요.

2️⃣ Large-scale Pre-training Experiment

▼ 실험 전제

- 모델구조
  - Hidden state 차원: 1024
- 학습
  - Tokenizer: BPE
  - Batch size: **8,000**
  - Training steps: **500,000**
- Corruption 방식
  - Text Infilling + Sentence Shuffling
- Regularization
  - Training step의 마지막 10% 구간에서는 dropout 제거

(1) Discriminative Tasks

	SQuAD 1.1 EM/F1	SQuAD 2.0 EM/F1	MNLI m/mm	SST Acc	QQP Acc	QNLI Acc	STS-B Acc	RTE Acc	MRPC Acc	CoLA Mcc
BERT	84.1/90.9	79.0/81.8	86.6/-	93.2	91.3	92.3	90.0	70.4	88.0	60.6
UniLM	-/-	80.5/83.4	87.0/85.9	94.5	-	92.7	-	70.9	-	61.1
XLNet	**89.0**/94.5	86.1/88.8	89.8/-	95.6	91.8	93.9	91.8	83.8	89.2	63.6
RoBERTa	88.9/**94.6**	**86.5/89.4**	**90.2/90.2**	96.4	92.2	94.7	**92.4**	86.6	**90.9**	**68.0**
BART	88.8/**94.6**	86.1/89.2	89.9/90.1	**96.6**	**92.5**	**94.9**	91.2	**87.0**	90.4	62.8

- BART는 RoBERTa 만큼의 성능을 낸다

(2) Generation Tasks

- summarization dataset에 대한 요약 생성

	CNN/DailyMail R1	CNN/DailyMail R2	CNN/DailyMail RL	XSum R1	XSum R2	XSum RL
Lead-3	40.42	17.62	36.67	16.30	1.60	11.95
PTGEN (See et al., 2017)	36.44	15.66	33.42	29.70	9.21	23.24
PTGEN+COV (See et al., 2017)	39.53	17.28	36.38	28.10	8.02	21.72
UniLM	43.33	20.21	40.51	-	-	-
BERTSUMABS (Liu & Lapata, 2019)	41.72	19.39	38.76	38.76	16.33	31.15
BERTSUMEXTABS (Liu & Lapata, 2019)	42.13	19.60	39.18	38.81	16.50	31.27
BART	**44.16**	**21.28**	**40.90**	**45.14**	**22.27**	**37.25**

- dialogue response generation에서도 높은 성능