0930 T5 논문발표

+논문 (Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer)

Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer

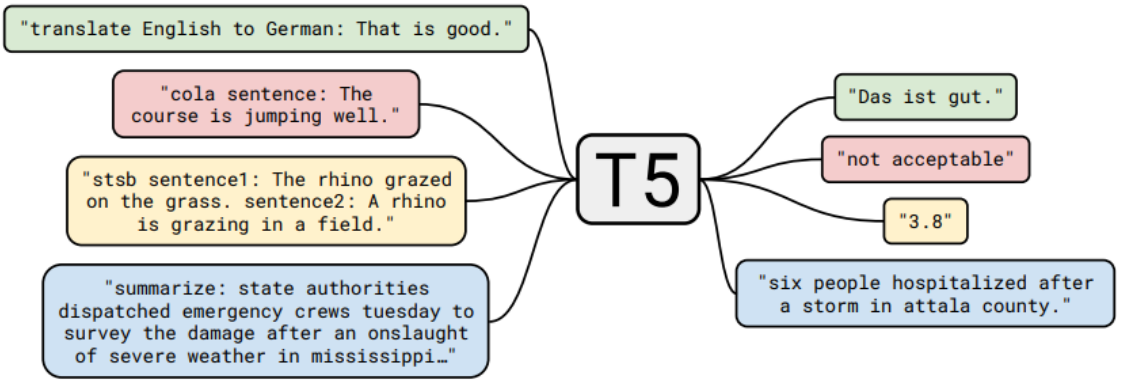
https://arxiv.org/abs/1910.10683

0. Abstract

unsupervised learning을 통해 pretrain을 하고(upstream task), 그 다음 supervised learning을 통해 finetuning 하는 (downstream task) 전이학습의(transfer learning) 방식이 제일 좋은 성능을 보였음

T5방식으로 모든 언어문제의 접근방식을 달리함.

1. Introduction



text를 input으로 받아 text를 output으로 받는 것이 본 논문의 핵심

generation task(번역, 요약 등)뿐만 아니라 classification, regression 문제도 text-to-text로 접근

→ 모든 task를 하나의 방식으로 풀게 되면, 다양한 downstream task에 동일한 model, loss function, hyperparameters 적용 가능

2. Setup_배경지식

2.1 model

RNN을 활용한 전이학습에서 Transformer을 활용한 전이학습이 더 일반적이게 변함.

Transformer의 주요 구성 요소는 self-attention(시퀀스를 처리할 때 나머지 요소들의 가중 평균으로 각 요소를 대체하는 방식)

Encoder만 있는 BERT, Decoder만 있는 GPT와는 다르게 T5는 Encoder와 Decoder가 함께 있음

- 변경점
  - Transformer의 Layer Normalization에 사용되는 bias를 제거하고 rescale만 수행하는 simplified layer normalization 사용
  - 이후로는 residual skip connection이 적용되어 block의 input을 output에 더해줌
  - feed-forward network, skip connection, attention weight, input, output에 dropout이 적용됨
  - Absolute positional embedding 대신 Relative positional embedding 사용
  - (위치정보) model layer 전체에서 positional embedding parameter sharing

2.2 The Colossal Clean Crawled Corpus

모델을 훈련하기 위해 사용된 대규모의 데이터 세트로 다음과 같은 전처리 과정을 시행함

- .,?,! 등과 같이 종결형 문장 부호로 끝나는 문장들만 남긴 후 나머지 문장 삭제
- 5 문장 미만으로 구성된 page는 폐기하고, 최소 3 단어 이상으로 구성된 문장만 남김
- "List of Dirty, Naughty, Obscene or Otherwise Bad Words" 등 비속어 및 은어와 같은 단어 삭제
- Javascript 단어가 있는 문장 삭제
- "lorem ipsum" 구문이 보이는 페이지 삭제
- {} 중괄호를 포함하고 있는 페이지 삭제
- 3 문장 이상으로 구성된 span이 2번 이상 반복되면 해당 페이지 삭제

2.3 Downstream Tasks

모델의 성능을 실험하기 위한 downstream task에 대한 설명

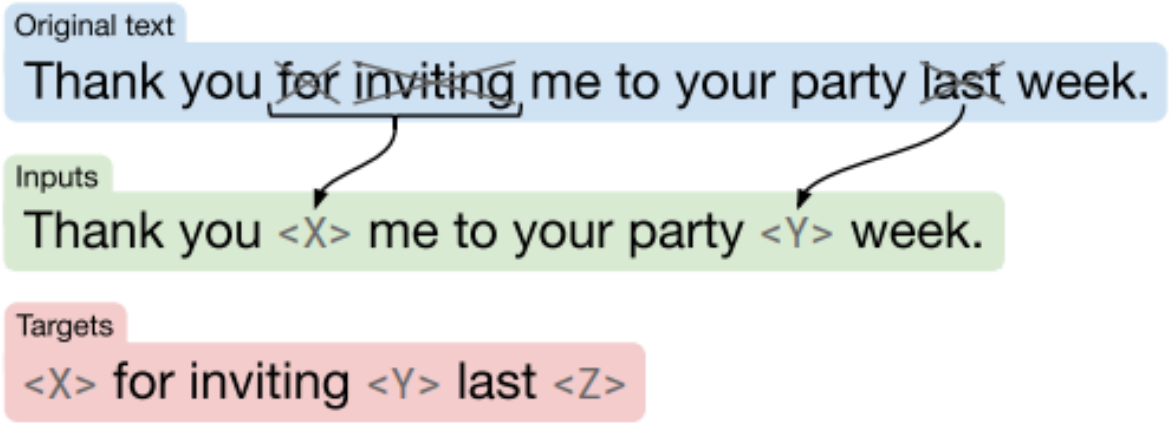
2.4 Input and Output Format

T5는 모든 task를 text-to-text format으로 다룸.

pre-training 과 fine-tuning에서 일관된 training objective를 제공한다는 장점

task에 관련없이 maximum likelihood objective로 학습됨(가장 가능성이 높은 출력을 생성함)

[Denoising Corrupted Span]



pre-training objective : SpanBERT논문에서 제안된 기법 사용

각 token을 마스킹하는 것이 아니라 Span을 하나의 token으로 마스킹함(→ 성능 향상+계산효율 향상)

Objective	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
Prefix language modeling	80.69	18.94	77.99	65.27	**26.86**	39.73	**27.49**
BERT-style (Devlin et al., 2018)	**82.96**	**19.17**	**80.65**	**69.85**	**26.78**	**40.03**	**27.41**
Deshuffling	73.17	18.59	67.61	58.47	26.11	39.30	25.62

masked language modeling 방식을 사용한 BERT-style방식이 가장 좋은 성능을 나타냄

Objective	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
BERT-style (Devlin et al., 2018)	82.96	19.17	**80.65**	69.85	26.78	**40.03**	27.41
MASS-style (Song et al., 2019)	82.32	19.16	80.10	69.28	26.79	**39.89**	27.55
★ Replace corrupted spans	83.28	**19.24**	**80.88**	**71.36**	**26.98**	39.82	**27.65**
Drop corrupted tokens	**84.44**	**19.31**	**80.52**	68.67	**27.07**	39.76	**27.82**

BERT-style : 15%의 token 마스킹

MASS-style : span 내 token들을 마스킹하고 해당 token들을 예측

Replace corrupted spans : T5에서 사용한 일정 span을 하나의 masking token으로 대체하는 방식(→최고)

Drop corrupted tokens : input sequence의 tokens를 제거하고 다시 복원하는 방식

[C4 Dataset]

Colossal Clean Crawled Corpus로, gpt3에 사용되었던 데이터에 일부 클리닝 작업한 데이터

- .,?,! 등과 같이 종결형 문장 부호로 끝나는 문장들만 남긴 후 나머지 문장 삭제
- 5 문장 미만으로 구성된 page는 폐기하고, 최소 3 단어 이상으로 구성된 문장만 남김
- "List of Dirty, Naughty, Obscene or Otherwise Bad Words" 등 비속어 및 은어와 같은 단어 삭제
- Javascript 단어가 있는 문장 삭제
- "lorem ipsum" 구문이 보이는 페이지 삭제
- {} 중괄호를 포함하고 있는 페이지 삭제
- 3 문장 이상으로 구성된 span이 2번 이상 반복되면 해당 페이지 삭제

Data set	Size	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ C4	745GB	83.28	**19.24**	80.88	71.36	**26.98**	**39.82**	**27.65**
C4, unfiltered	6.1TB	81.46	19.14	78.78	68.04	26.55	39.34	27.21
RealNews-like	35GB	**83.83**	**19.23**	80.39	72.38	**26.75**	**39.90**	**27.48**
WebText-like	17GB	**84.03**	**19.31**	**81.42**	71.40	**26.80**	**39.74**	**27.59**
Wikipedia	16GB	81.85	**19.31**	81.29	68.01	**26.94**	39.69	**27.67**
Wikipedia + TBC	20GB	83.65	**19.28**	**82.08**	**73.24**	**26.77**	39.63	**27.57**

C4 데이터셋의 성능이 뛰어나지 않은 이유는, 다른 데이터셋들이 더 도메인에 국한되어 있기 때문

[Multi-task pre-training]

여러 종류의 task에 대해서 한번에 training을 진행

Mixing strategy	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Baseline (pre-train/fine-tune)	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
Equal	76.13	19.02	76.51	63.37	23.89	34.31	26.78
Examples-proportional, $K = 2^{16}$	80.45	19.04	77.25	69.95	24.35	34.99	27.10
Examples-proportional, $K = 2^{17}$	81.56	19.12	77.00	67.91	24.36	35.00	27.25
Examples-proportional, $K = 2^{18}$	81.67	19.07	78.17	67.94	24.57	35.19	27.39
Examples-proportional, $K = 2^{19}$	81.42	**19.24**	79.78	67.30	25.21	36.30	**27.76**
Examples-proportional, $K = 2^{20}$	80.80	**19.24**	**80.36**	67.38	25.66	36.93	**27.68**
Examples-proportional, $K = 2^{21}$	79.83	18.79	79.50	65.10	25.82	37.22	27.13
Temperature-scaled, $T = 2$	81.90	**19.28**	79.42	69.92	25.42	36.72	27.20
Temperature-scaled, $T = 4$	80.56	**19.22**	77.99	69.54	25.04	35.82	27.45
Temperature-scaled, $T = 8$	77.21	19.10	77.14	66.07	24.55	35.35	27.17

Training strategy	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Unsupervised pre-training + fine-tuning	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	39.82	27.65
Multi-task training	81.42	**19.24**	79.78	67.30	25.21	36.30	27.76
Multi-task pre-training + fine-tuning	**83.11**	**19.12**	**80.26**	**71.03**	**27.08**	39.80	**28.07**
Leave-one-out multi-task training	81.98	19.05	79.97	**71.68**	**26.93**	39.79	**27.87**
Supervised multi-task pre-training	79.93	18.96	77.38	65.36	26.81	**40.13**	**28.04**

multi-task pre-training + fine_tuning이 기존의 unsupervised pre-training + fine-tuning과 비슷한 성능

[Scaling Model Size]

Scaling strategy	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Baseline	83.28	19.24	80.88	71.36	26.98	39.82	27.65
1× size, 4× training steps	85.33	19.33	82.45	74.72	27.08	40.66	27.93
1× size, 4× batch size	84.60	19.42	82.52	74.64	27.07	40.60	27.84
2× size, 2× training steps	**86.18**	19.66	**84.18**	77.18	27.52	**41.03**	28.19
4× size, 1× training steps	**85.91**	19.73	**83.86**	**78.04**	27.47	40.71	28.10
4× ensembled	84.77	**20.10**	83.09	71.74	**28.05**	40.53	**28.57**
4× ensembled, fine-tune only	84.05	19.57	82.36	71.55	27.55	40.22	28.09

당연하게도, 모델의 크기가 증가하고 학습의 시간이 더 길수록 성능이 좋아짐.

!! 24개의 task중 18개의 task에서 SOTA달성

- Text-to-text
  - text-to text framework 사용하여 하나의 모델로 다양한 태스크 학습 가능
- Architectures
  - Encoder-Decoder 구조 사용
- Unsupervised objectives
  - Denoising objectives 사용
- Data sets
  - Colossal Clean Crawled Corpus (C4) 데이터셋
- Scalling
  - 모델의 크기를 키울 때 어떻게 해야 효과적인지 실험함.
- Pushing the limits
  - 위의 방법 모두 사용, 11B 파라미터의 큰 모델을 학습시켜 다양한 task에서 SOTA 달성.

+논문리뷰 참고

https://sunho99.tistory.com/entry/exploring-the-limits-of-transfer-learning-with-a-unified-text-to-text-transformer-논문-리뷰-분석

https://gbdai.tistory.com/62