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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee,

Sharan Narang, Michael Matena, Yanqi zhou, Wei Li, Peter J. Liu

JMLR 2020

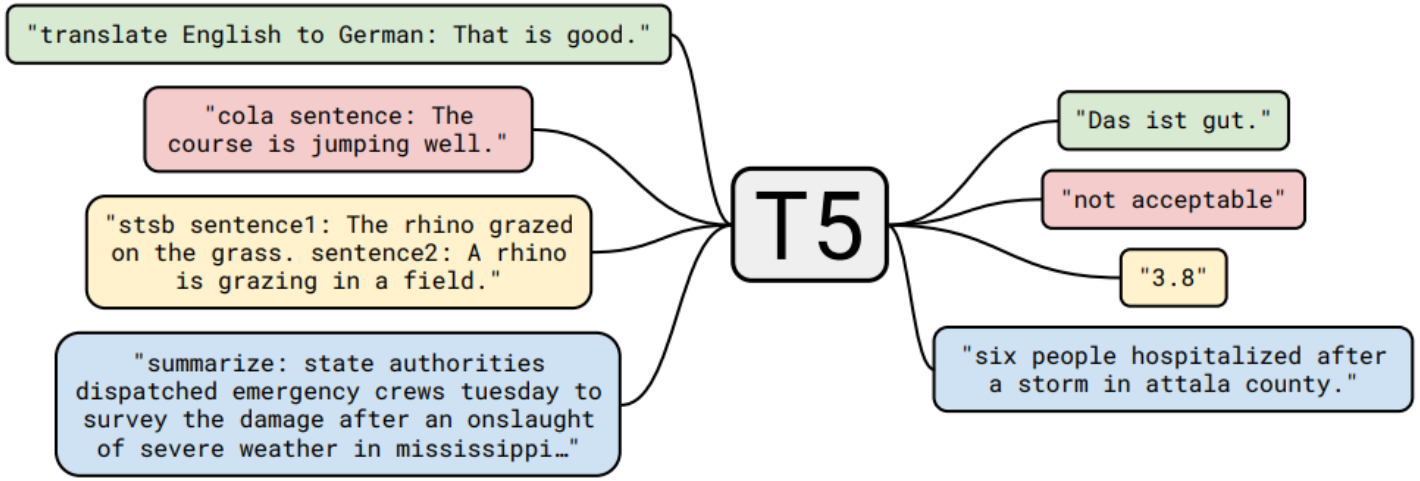
지능융합학과 2022109953 정민지

목차

- Summary
- Setup
- Experiments
- Reflection

Summary

- T5: **T**ext-**t**o-**T**ext **T**ransfer **T**ransformer



- Text-to-Text: 단일 모델로 여러가지 task를 수행하게끔 함, input – output format이 모두 text
- Transfer learning: 특정 task를 학습한 모델을 다른 task 수행에 재사용
- Transformer: Transformer 의 구조를 사용

Setup – Dataset

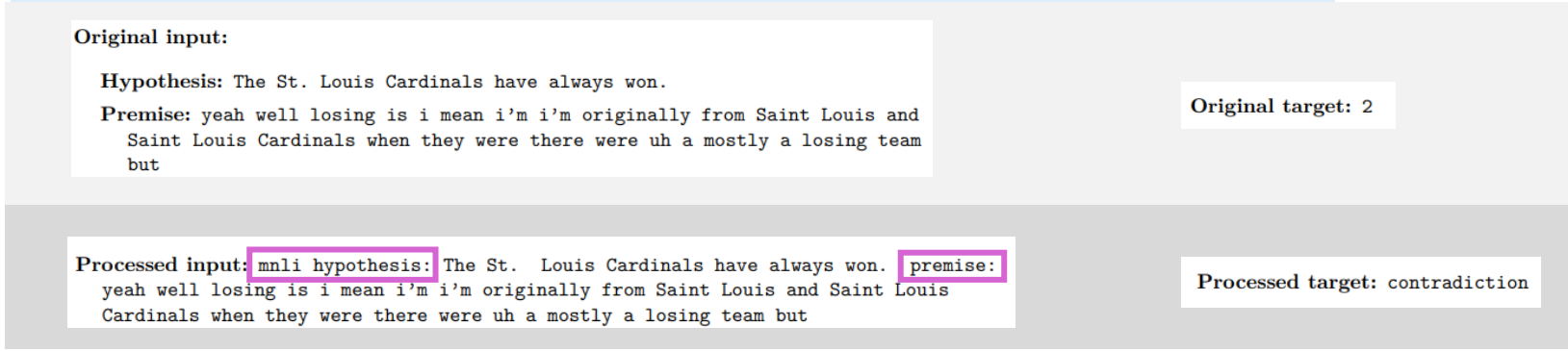
- Pre-training : C4 dataset
  - The **C**olossal **C**lean **C**rawled **C**orpus
  - 영어로 된 Data
  - Common Crawl data 수집 및 전처리
    - 맨 마지막 줄이 . ! ? ." 로 끝나는 line만
    - 5문장 이하인 페이지 삭제, line에 3단어 이상인 것만 놔둠
    - 안 좋은 단어가 포함된 페이지 삭제 (List of Dirty, Naughty, Obscene or Otherwise Bad Words)
    - Javascript, placeholder로 시작되는 문장, 페이지 삭제
    - {} 들어간 페이지 삭제 (코드로 간주)
    - 중복 문장 삭제
    - langdetect 활용하여 페이지가 영어일 확률 99% 미만인 페이지 삭제
  - 750GB 정도 수집

Setup – Dataset

- Fine-Tuning
  - GLUE / SuperGLUE text classification
  - CNN / Daily Mail abstract summarization
  - SQuAD question answering
  - WMT English to German, French and Romanian translation

Setup – Input & Output format

- 모든 문제를 Text-to-Text로 바꾸어서 처리
- 몇가지 Input / Output format은 Text-to-Text 형식에 적용하기 위해 수정해야함
  - MNLI (Classification – entailment, neutral, contradiction 3가지 중 하나로 분류)
    - 하지만 위 3가지 라벨 이외의 라벨이 생성될 수 있음 → 틀린 것으로 처리



  - STS-B (유사도를 1~5사이의 값으로 나타냄, Regression)
    - Annotation이 보통 0.2 단위로 되어있는 것을 확인
    - 모든 Data 반올림, 21개의 classification 문제로 변경 (“1.0”, “1.2”, “1.4” … “4.8”, “5.0”)

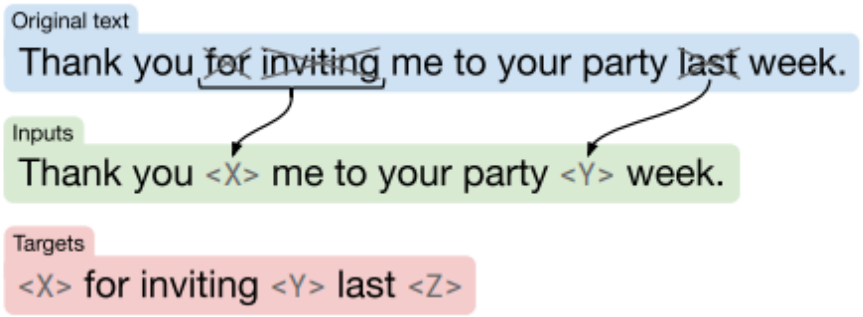
Experiments – Training

- Pre-training
  - C4 dataset으로 Pre-training수행
  - 2^{19} = 524288 step
  - Maximum sequence length: 512
  - Batch size: 128

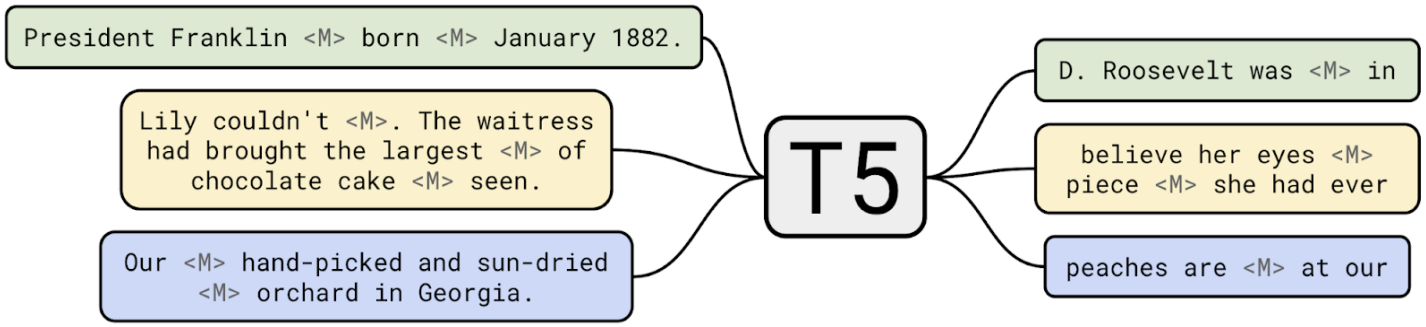
  - Fine-tuning시 English to German/French/Romanian으로 Translation 하는 Task 존재
    → 본 언어도 학습하기 위하여 langdetect에서 German/French/Romanian으로 나온 페이지를 추가
  - C4 dataset 언어 비율
    - English : German : French : Romanian = 10 : 1 : 1 : 1 로 구성

Experiments – Training

- Pre-training



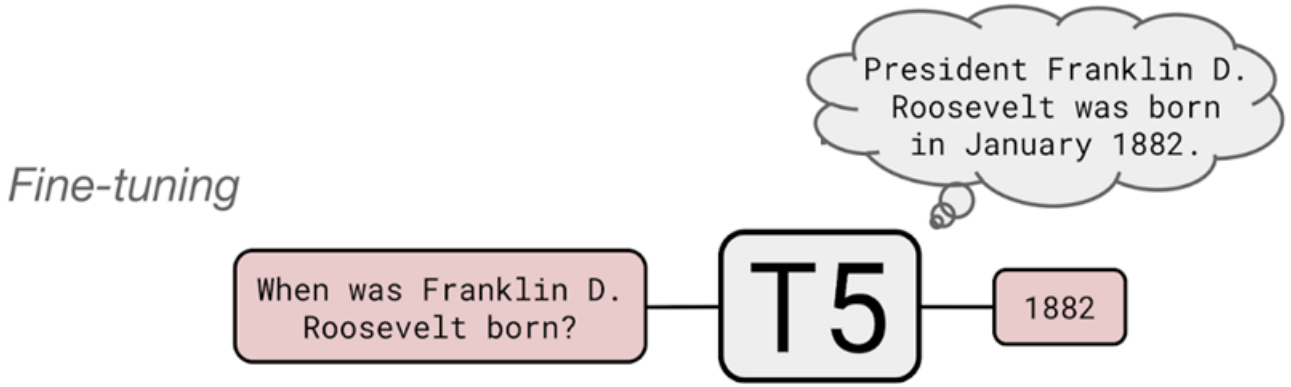
- Input의 15%의 토큰에 Masking
- 연속되어 Masking 될 경우, 하나의 sentinel token으로 처리



Pre-training

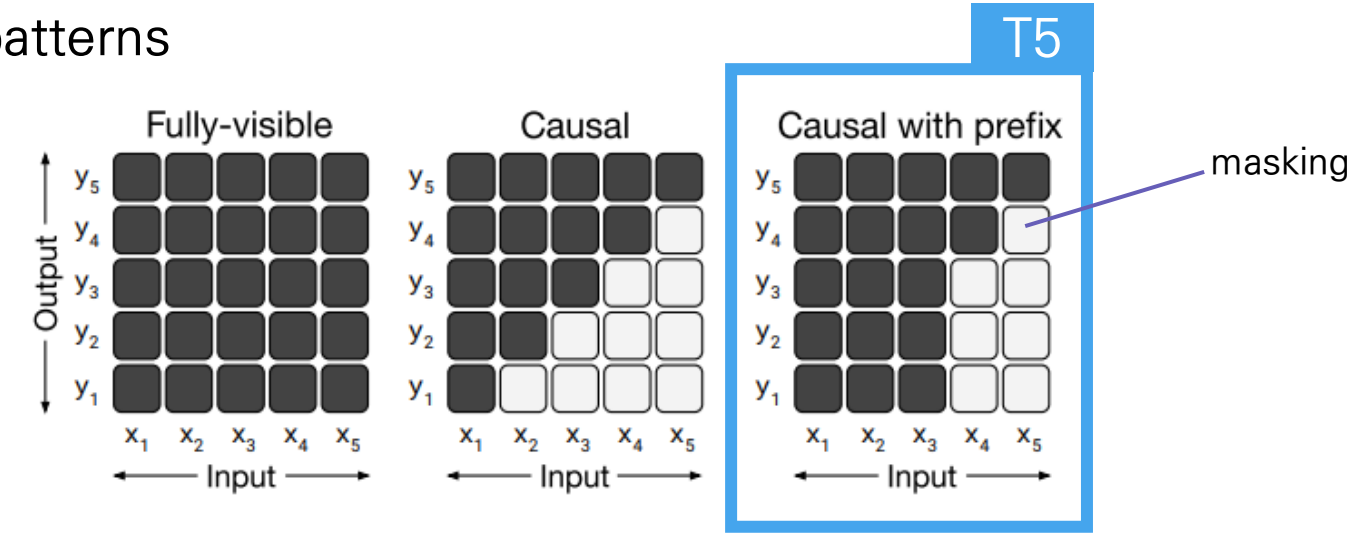
Experiments – Training

- Fine-Tuning
  - 모든 작업에 대하여 2^{18} = 262144 step
  - Maximum sequence length: 512
  - Batch size: 128

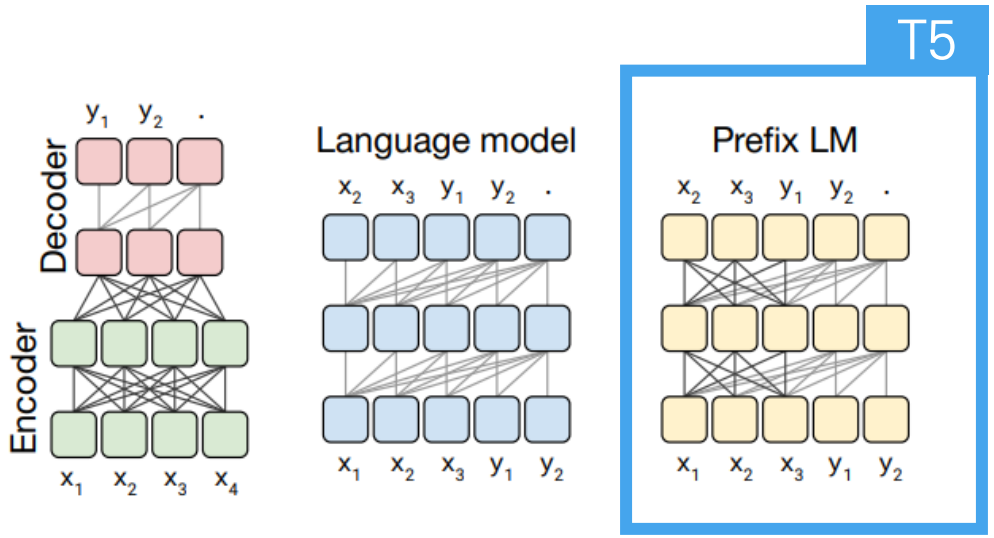


Experiments – Architecture

- Attention mask patterns

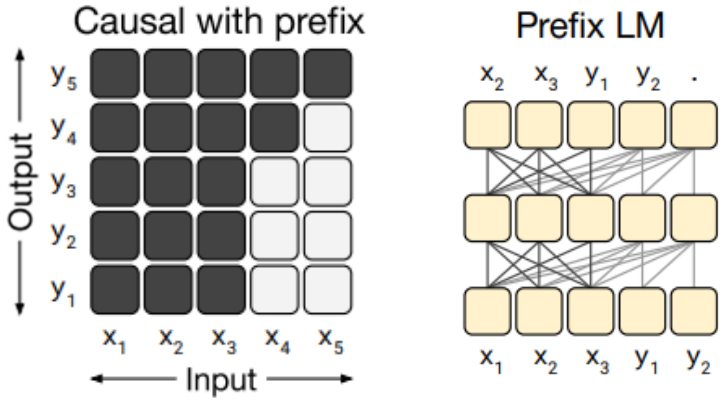


- Transformer architecture variants



Experiments – Architecture

- Example



  - X: **translate English to German:** That is good. **target:**
  - Y: Das ist gut

  - X: **mnli premise:** I hate pigeons. **hypothesis:** My feelings towards pigeons are filled with animosity. **target:**
  - Y: entailment

→ Model은 전체 input에 대해 계속 볼 수 있으며, 접두사를 학습

→ “target: ”가 BERT에서 classification token과 비슷한 역할을 함

Experiments – Architecture

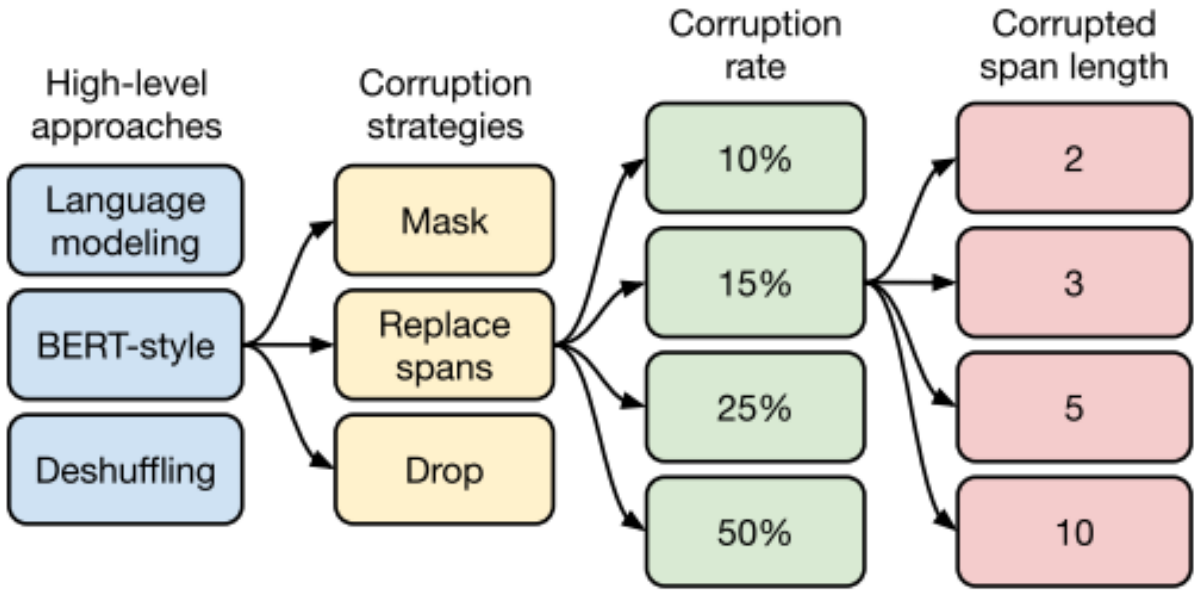
- Performance of the different architectural variants

Architecture	Objective	Params	Cost	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Encoder-decoder	Denoising	$2P$	M	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
Enc-dec, shared	Denoising	P	M	82.81	18.78	**80.63**	**70.73**	26.72	39.03	**27.46**
Enc-dec, 6 layers	Denoising	P	$M/2$	80.88	18.97	77.59	68.42	26.38	38.40	26.95
Language model	Denoising	P	M	74.70	17.93	61.14	55.02	25.09	35.28	25.86
Prefix LM	Denoising	P	M	81.82	18.61	78.94	68.11	26.43	37.98	27.39

- Encoder-Decoder + Denoising이 성능이 가장 좋음
- Encoder-Decoder shared는 Parameter가 반으로 줄었지만 성능 하락이 적음
- Layer 수를 반으로 줄이는 것은 성능 하락 많이 발생함

Experiments – Unsupervised Objectives

- 실험해봐야 할 Objective의 종류가 매우 많음
  - 우선순위를 정하여 실험하되, 성능이 좋게 나오는 것에 대해서만 하위 실험 진행

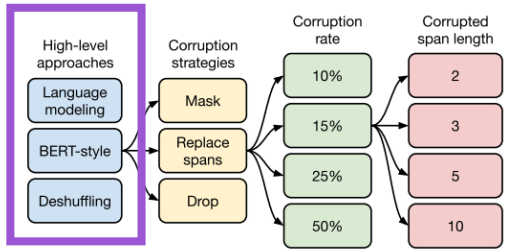


Experiments – Unsupervised Objectives

- Pretraining data format
  - 이하 실험에서 언급하는 data format은 다음과 같은 형식을 가짐

Objective	Inputs	Targets
Prefix language modeling	Thank you for inviting	me to your party last week .
BERT-style Devlin et al. (2018)	Thank you <M> <M> me to your party apple week .	*(original text)*
Deshuffling	party me for your to . last fun you inviting week Thank	*(original text)*
MASS-style Song et al. (2019)	Thank you <M> <M> me to your party <M> week .	*(original text)*
I.i.d. noise, replace spans	Thank you <X> me to your party <Y> week .	<X> for inviting <Y> last <Z>
I.i.d. noise, drop tokens	Thank you me to your party week .	for inviting last
Random spans	Thank you <X> to <Y> week .	<X> for inviting me <Y> your party last <Z>

Experiments – Unsupervised Objectives

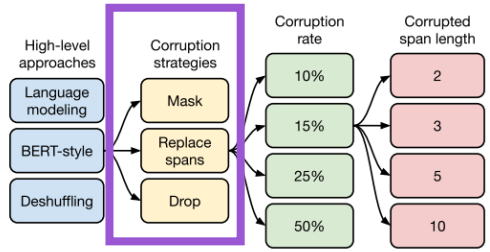


- High-level approaches

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

- BERT-style이 가장 좋은 성능을 보임
- Objective 설명 (예시 – 이전페이지(p.14)에 있음)
  - Prefix language modeling: 문장 앞부분을 제공, 뒷부분 맞춤
  - BERT-style: MLM
  - Deshuffling: token을 섞고 원문장을 맞추는 방식

Experiments – Unsupervised Objectives



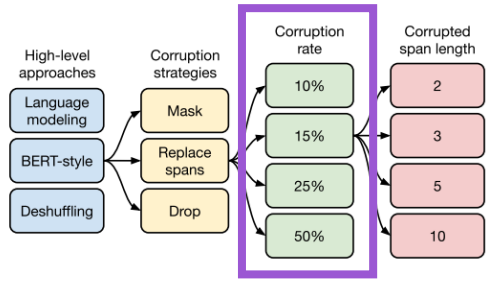
- Corruption strategies

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**

- 이전 단계에서 BERT-style approach가 가장 좋은 성능을 냈기 때문에 이 방법이 가장 좋다고 가정하고 하위 실험 진행
- BERT-style의 Objective를 사용한 대표 방법들을 실험 진행

Objective	Inputs	Targets
BERT-style Devlin et al. (2018)	Thank you <M> <M> me to your party apple week .	*(original text)*
MASS-style Song et al. (2019)	Thank you <M> <M> me to your party <M> week .	*(original text)*
I.i.d. noise, replace spans	Thank you <X> me to your party <Y> week .	<X> for inviting <Y> last <Z>
I.i.d. noise, drop tokens	Thank you me to your party week .	for inviting last

Experiments – Unsupervised Objectives

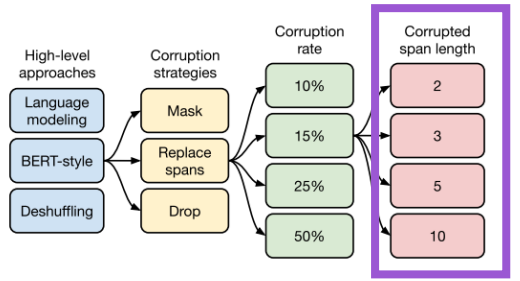


- Corruption rate

Corruption rate	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
10%	**82.82**	19.00	**80.38**	69.55	**26.87**	39.28	**27.44**
★ 15%	**83.28**	19.24	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
25%	**83.00**	**19.54**	**80.96**	70.48	**27.04**	**39.83**	**27.47**
50%	81.27	19.32	79.80	70.33	**27.01**	**39.90**	**27.49**

  - Masking rate를 얼마로 했는지에 따른 성능 비교
  - 이전 실험과 동일하게 15%로 했을 때가 성능 가장 좋았음

Experiments – Unsupervised Objectives



- Corrupted span length

Span length	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Baseline (i.i.d.)	**83.28**	19.24	80.88	71.36	**26.98**	**39.82**	**27.65**
2	**83.54**	19.39	**82.09**	**72.20**	**26.76**	**39.99**	**27.63**
3	**83.49**	**19.62**	**81.84**	**72.53**	**26.86**	39.65	**27.62**
5	**83.40**	19.24	**82.05**	**72.23**	**26.88**	39.40	**27.53**
10	82.85	19.33	**81.84**	70.44	**26.79**	39.49	**27.69**

  - Corrupt 할 때 적당한 span 길이를 찾음
  - Translation task를 제외하고 Span length = 3일 때가 성능이 Baseline보다 좋다.
  - 하지만 속도를 고려했을 때 Baseline을 사용하는 것이 나음
    - 예측해야 할 것이 적기 때문

Experiments – Pretraining Dataset

- Unlabeled dataset

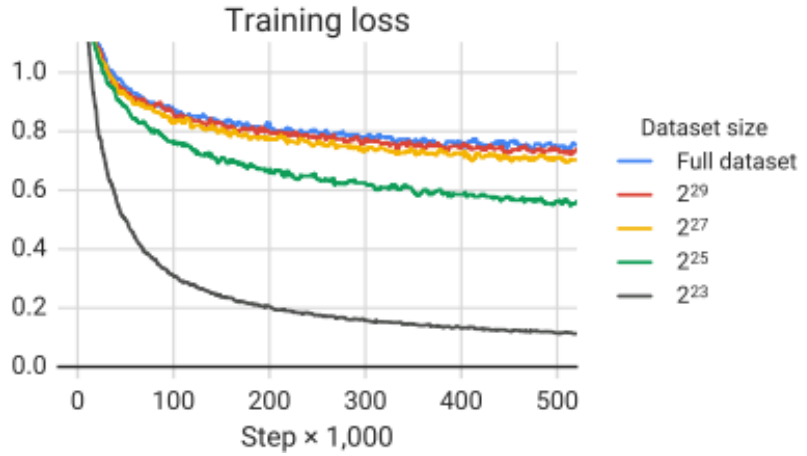
Data set	Size	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ C4	745GB	83.28	**19.24**	80.88	71.36	**26.98**	**39.82**	**27.65**
C4, unfiltered	6.1TB	81.46	19.14	78.78	68.04	26.55	39.34	27.21
RealNews-like	35GB	**83.83**	**19.23**	80.39	72.38	**26.75**	**39.90**	**27.48**
WebText-like	17GB	**84.03**	**19.31**	**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**

- C4의 성능이 가장 좋음
- 다른 Data가 더 성능이 높기도 했지만, 데이터셋이 적음 (문제점 – 뒷장(p.20) 참조)
  - C4, unfiltered: langdetect로 영어만 추린 것
  - RealNews-like: 뉴스데이터
  - WebText-like: Reddit과 같은 인터넷 데이터
  - Wikipedia: 위키피디아
  - Wikipedia + TBC: 위키피디아 + Toronto book corpus

Experiments – Pretraining Dataset

- Pretraining Dataset size

Number of tokens	Repeats	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Full data set	0	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
2^{29}	64	**82.87**	**19.19**	**80.97**	**72.03**	**26.83**	**39.74**	**27.63**
2^{27}	256	82.62	**19.20**	79.78	69.97	**27.02**	**39.71**	27.33
2^{25}	1,024	79.55	18.57	76.27	64.76	26.38	39.56	26.80
2^{23}	4,096	76.34	18.33	70.92	59.29	26.37	38.84	25.81



- Full dataset : 2^{35}개, 점차 줄여가며 실험 진행
- 토큰의 수가 줄어든 만큼 반복
- 데이터의 크기가 작은 경우 Overfitting이 발생함

Experiments – Training Strategy (Fine-tuning)

- Fine-tuning methods

Fine-tuning method	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ All parameters	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
Adapter layers, $d = 32$	80.52	15.08	79.32	60.40	13.84	17.88	15.54
Adapter layers, $d = 128$	81.51	16.62	79.47	63.03	19.83	27.50	22.63
Adapter layers, $d = 512$	81.54	17.78	79.18	64.30	23.45	33.98	25.81
Adapter layers, $d = 2048$	81.51	16.62	79.47	63.03	19.83	27.50	22.63
Gradual unfreezing	82.50	18.95	79.17	**70.79**	26.71	39.02	26.93

- 전체를 사용했을 때가 가장 성능이 좋음

- Adapter layers: Transformer에서 Feed-forward layer 뒤에 Adapter layer를 추가하여 그 부분만 학습
- Gradual unfreezing: 모든 layer를 한번에 finetuning 하지 않고, iteration마다 top layer부터 unfreeze하며 top → bottom순으로 학습시키는 방법

Experiments – Training Strategy (Fine-tuning)

- Multi-task training

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

- 어떤 Task의 Data를 얼마만큼의 비율로 학습할 것인지? – Sampling 안 하는게 가장 성능이 좋았음
- Equal: Data의 size에 상관 없이 같은 수의 문장을 sampling
- K threshold 사용: data가 K개보다 클 때는 K개만 사용
- Temperature-scaled: 각 data를 T로 나누고 하나씩 합쳐서 사용 ($K=2^{21}$)
  - T가 무한히 커지면 Equal과 같고 T=1이면 Examples-proportional과 같음

Experiments – Scaling

- NLP에서 모델의 크기가 클 수록 더 좋은 모델이 나오는 추세
- 4배의 computing power를 가진다면, 모델의 크기를 어떻게 늘리는 것이 효과적인가?

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

- Model size, training step을 늘린 것이 가장 효과 있음

- 4x ensembled: pre-training, fine-tuning 4번 따로 진행 후 앙상블
- 4x ensembled, fine-tune only: fine-tune만 따로 진행 후 앙상블

Experiments – Putting It All together

- 실험 정리

Objective	• i.i.e Denoising objective • Span length 3 • Corrupt 15%
Longer Training	• Data를 반복해서 학습 할 필요 없이 C4 dataset는 충분히 큼 • Pre-training은 효과적임 • Batch size, training step 늘리는 것 효과적 • C4 말고 다른 Dataset으로 Pretraining하는 것이 좋은 성능을 보였으나, Data 양이 적어서 Overfitting이 될 우려 존재
Model Size	• Base: 220 million parameters • Small: d_{model} = 512, d_{ff} = 2048, 8-headed attention, 6 layers, 60 million parameters • Large: d_{model} = 1024, d_{ff} = 4096, 16-headed attention, 12 layers, 770 million parameters • 3B: d_{model} = 1024, d_{ff} = 16384, 32-headed attention, 24 layers, 2.8 billion parameters • 11B: d_{model} = 1024, d_{ff} = 65536, 128-headed attention, 24 layers, 11 billion parameters
Multi-task pretraining	• Mixture of Unsupervised and supervised task → Unsupervised만으로 pretraining 하는 것보다 성능이 좋음
Fine-tuning on individual GLUE and SuperGLUE	• GLUE/SuperGLUE 벤치마크 안에 있는 것을 한번에 fine-tuning • 성능의 저하 약간 발생
Beam Search	• WMT translation과 CNN/DM summarization의 성능 향상을 위하여 beam-search 사용 (beam width = 4)

Experiments – Putting It All together

- 실험 결과

Model	GLUE Average	CoLA Matthew's	SST-2 Accuracy	MRPC F1	MRPC Accuracy	STS-B Pearson	STS-B Spearman
Previous best	89.4[a]	69.2[b]	97.1[a]	**93.6**[b]	**91.5**[b]	92.7[b]	92.3[b]
T5-Small	77.4	41.0	91.8	89.7	86.6	85.6	85.0
T5-Base	82.7	51.1	95.2	90.7	87.5	89.4	88.6
T5-Large	86.4	61.2	96.3	92.4	89.9	89.9	89.2
T5-3B	88.5	67.1	97.4	92.5	90.0	90.6	89.8
T5-11B	**90.3**	**71.6**	**97.5**	92.8	90.4	**93.1**	**92.8**

Model	QQP F1	QQP Accuracy	MNLI-m Accuracy	MNLI-mm Accuracy	QNLI Accuracy	RTE Accuracy	WNLI Accuracy
Previous best	74.8[c]	**90.7**[b]	91.3[a]	91.0[a]	**99.2**[a]	89.2[a]	91.8[a]
T5-Small	70.0	88.0	82.4	82.3	90.3	69.9	69.2
T5-Base	72.6	89.4	87.1	86.2	93.7	80.1	78.8
T5-Large	73.9	89.9	89.9	89.6	94.8	87.2	85.6
T5-3B	74.4	89.7	91.4	91.2	96.3	91.1	89.7
T5-11B	**75.1**	90.6	**92.2**	**91.9**	96.9	**92.8**	**94.5**

Model	SQuAD EM	SQuAD F1	SuperGLUE Average	BoolQ Accuracy	CB F1	CB Accuracy	COPA Accuracy
Previous best	90.1[a]	95.5[a]	84.6[d]	87.1[d]	90.5[d]	95.2[d]	90.6[d]
T5-Small	79.10	87.24	63.3	76.4	56.9	81.6	46.0
T5-Base	85.44	92.08	76.2	81.4	86.2	94.0	71.2
T5-Large	86.66	93.79	82.3	85.4	91.6	94.8	83.4
T5-3B	88.53	94.95	86.4	89.9	90.3	94.4	92.0
T5-11B	**91.26**	**96.22**	**88.9**	**91.2**	**93.9**	**96.8**	**94.8**

Model	MultiRC F1a	MultiRC EM	ReCoRD F1	ReCoRD Accuracy	RTE Accuracy	WiC Accuracy	WSC Accuracy
Previous best	84.4[d]	52.5[d]	90.6[d]	90.0[d]	88.2[d]	69.9[d]	89.0[d]
T5-Small	69.3	26.3	56.3	55.4	73.3	66.9	70.5
T5-Base	79.7	43.1	75.0	74.2	81.5	68.3	80.8
T5-Large	83.3	50.7	86.8	85.9	87.8	69.3	86.3
T5-3B	86.8	58.3	91.2	90.4	90.7	72.1	90.4
T5-11B	**88.1**	**63.3**	**94.1**	**93.4**	**92.5**	**76.9**	**93.8**

Model	WMT EnDe BLEU	WMT EnFr BLEU	WMT EnRo BLEU	CNN/DM ROUGE-1	CNN/DM ROUGE-2	CNN/DM ROUGE-L
Previous best	**33.8**[e]	**43.8**[e]	**38.5**[f]	43.47[g]	20.30[g]	40.63[g]
T5-Small	26.7	36.0	26.8	41.12	19.56	38.35
T5-Base	30.9	41.2	28.0	42.05	20.34	39.40
T5-Large	32.0	41.5	28.1	42.50	20.68	39.75
T5-3B	31.8	42.6	28.2	42.72	21.02	39.94
T5-11B	32.1	43.4	28.1	**43.52**	**21.55**	**40.69**

Reflection – Takeaways

Text-to-Text	•	같은 loss function과 decoding 과정으로 사용할 수 있는 single model
Architectures	•	Encoder + Decoder, sharing 구조도 성능 저하가 많지 않음을 보임
Unsupervised objectives	•	짧은 target sequence를 생성하는 objective 제안. 이는 Text-to-text에서도 성능이 비슷하게 나옴.
Data sets	•	C4 dataset 제작
Training strategies	•	기존 방법들을 mixing하여 pre-training & fine-tuning 진행
Scaling	•	다양한 방법 실험
Pushing the limits	•	위의 모든 방법 적용, 가장 큰 11B모델(11 billion parameter)은 sota 달성

Reflection – Outlook

The inconvenience of large models	• 큰 모델이 성능이 잘 나오는 경향이 있지만, transfer learning을 사용하는 것의 장점 중 하나는 low-resource task에서 좋은 성능을 얻어내는 것임 • 저렴한 모델로 stronger performance 낼 수 있는 연구 필요
More efficient knowledge extraction	• Pre-training은 downstream model에 성능 향상을 위한 지식을 제공하는 것 • 하지만 현재 denoising을 하도록 훈련시킴 • 이 기법이 general-purpose knowledge를 학습시기에 효율적이지 않을 수 있음
Formalizing the similarity between tasks	• Pre-training을 Wikipedia dataset으로 시켰을 경우, 출처가 같은 Data의 성능이 더 높음
Language-agnostic models	• 영어로 학습한 모델 – 언어 의존적

QnA

Appendix

Adapter layers