투빅스 19기 김윤서

CS24N Lecture 15: T5 and large language model

INSTRUCTIONS

1. why and what is T5
2. T5 base structure
   - Data source
   - Pretraining objective
   - Baseline experimental procedure
3. T5를 위한 design dicision
   - Model structure
   - pre-training objective
   - variants of pre-training dataset
   - multi-tasking pretraining
   - scaling strategy
4. MT5
5. closed-book question-answering
6. Do model memorize training data?

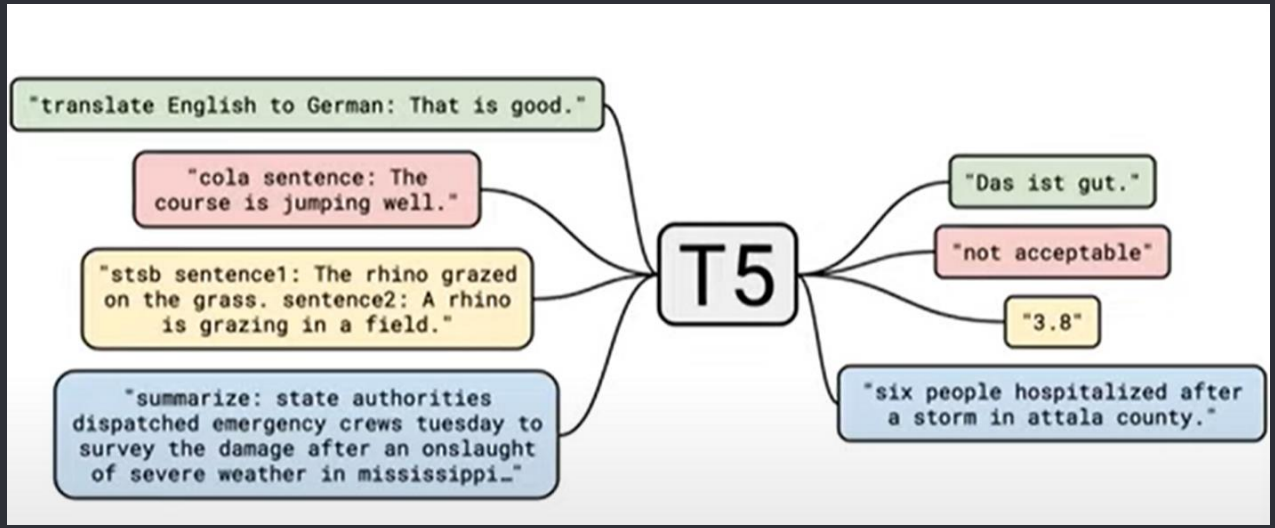
1 Why and What is T5

Why and what is T5

- 특정 task를 해결하기 위해 transformer의 구조를 수정하거나 파라미터를 조정하는 시도
- 모든 것을 같게 세팅하는 최고의 모델을 만들 순 없는가?
- “Treating all text problems in the same format “(모든 문제를 같은 포맷으로 해결한다)

T5, text-to-text transfer transformer

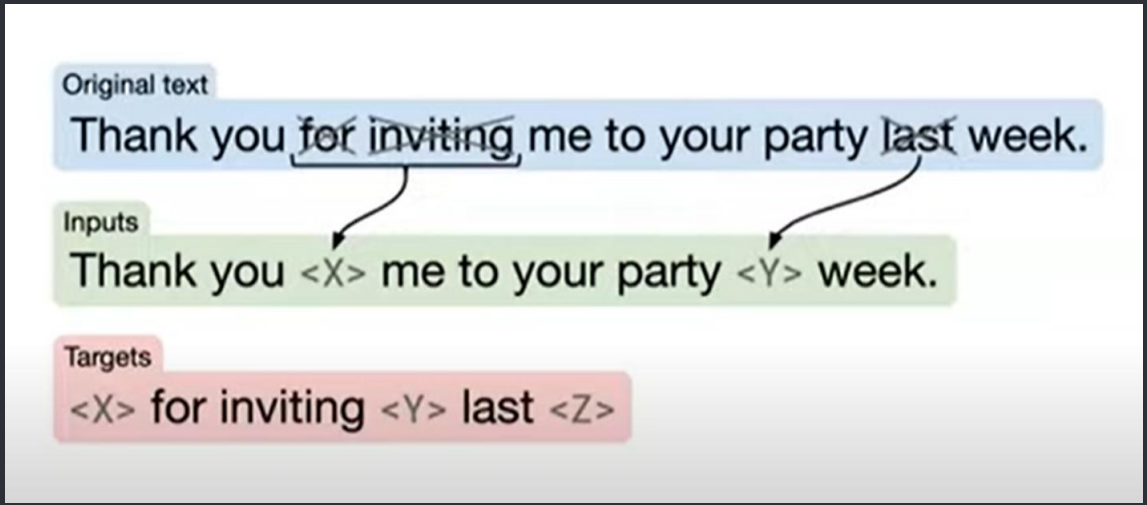
Why and what is T5



- 모든 것을 text로서 학습함
- "Accept", "3.8" 도 분류, 회귀모델이 아닌 text로 학습
- Vanilla transformer를 제안된 그대로 사용할 수 있다는 장점

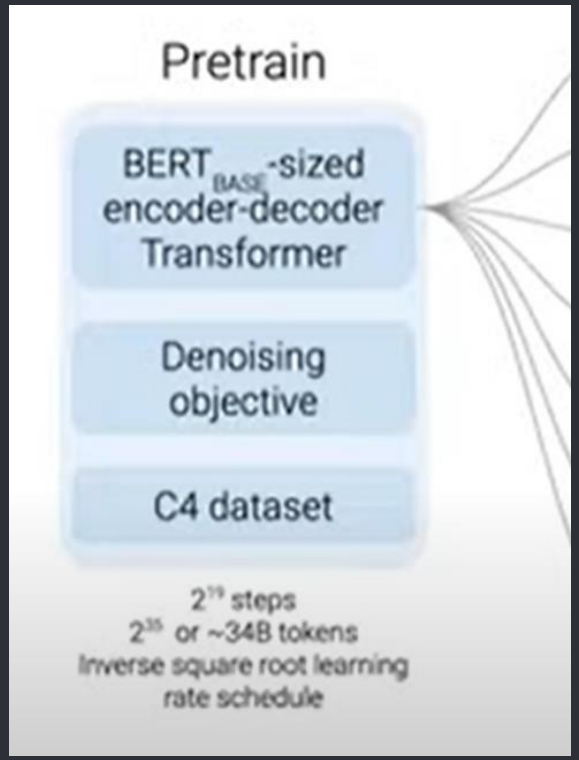
2 T5 base structure

T5 base structure – Pretraining objective



- 랜덤하게 token 선택하고, drop 함
- Drop된 token은 sentinal token으로 대체
- Model의 goal은 빈칸을 채우는 것
- Missing word를 재건축한다는 점에서 BERT모델과 유사

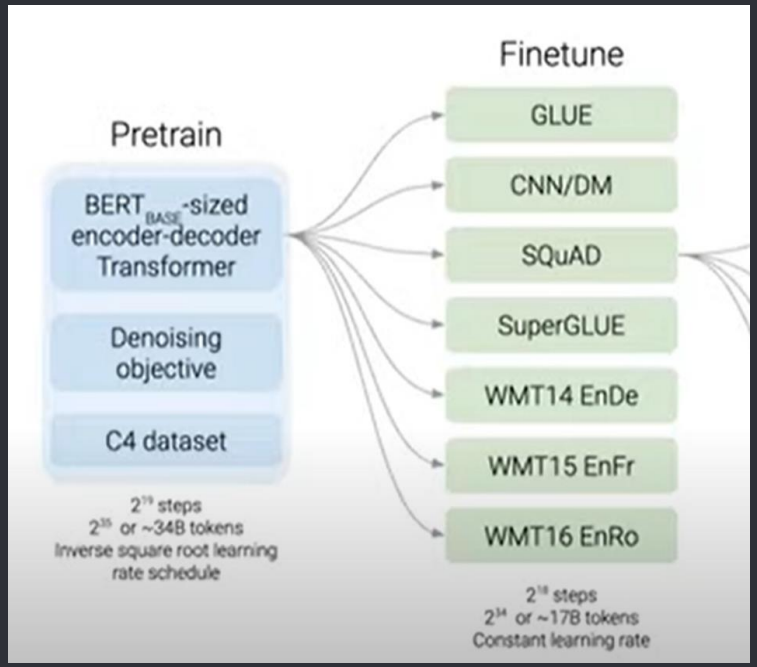
T5 base structure – Baselline experimental procedure



Pretrain Model:

- BERT base size의 인코더와 디코더, BERT보다 파라미터가 두배정도 많음
- Denoising objective : masked language modeling
- C4 dataset 사용
- Pre-train할때 34billion 개의 token사용, BERT의 ¼정도

T5 base structure – Baselline experimental procedure



여러 task를 진행하기 위한 finetuning

- GLUE : sentence classification, sentence-pair classification, regression task
- CNN/Daliy Mail : summarization
- SQuAD : question answering, reading comprehensive benchmark
- SuperGLUE: GLUE의 difficult version
- WMT14EnDe/WMT15EnFr/WMT16EnRo : translation
- Pre-train모델을 가지고 각각의 task에 따라 개별적으로 fine tuning
- 약 17 billion 개 token 으로 finetuning

T5 base structure – Baseline experimental procedure

	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Baseline average	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
Baseline standard deviation	0.235	0.065	0.343	0.416	0.112	0.090	0.108
No pre-training	66.22	17.60	50.31	53.04	25.86	**39.77**	24.04

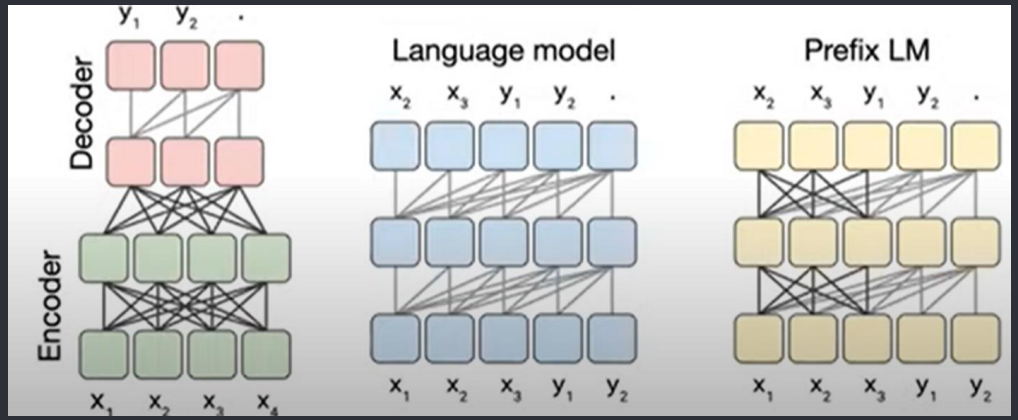
- Pre-training하지 않은 모델은 성과가 매우 안좋음
- GLUE,CNNDM, SQuAD는 BERT와 비교해도 낮은 성능이 아님

3 T5를 위한 design decision

- 많은 실험들로 성능비교
- 하이퍼 파라미터를 조정하는 실험은 하지 않음
- 왜냐하면 우리는 모든 문제들을 같은 framework로 다루기 원하기 때문에 하이퍼 파라미터를 조정할 필요가 없다
- Text-to-text maximum likelihood training

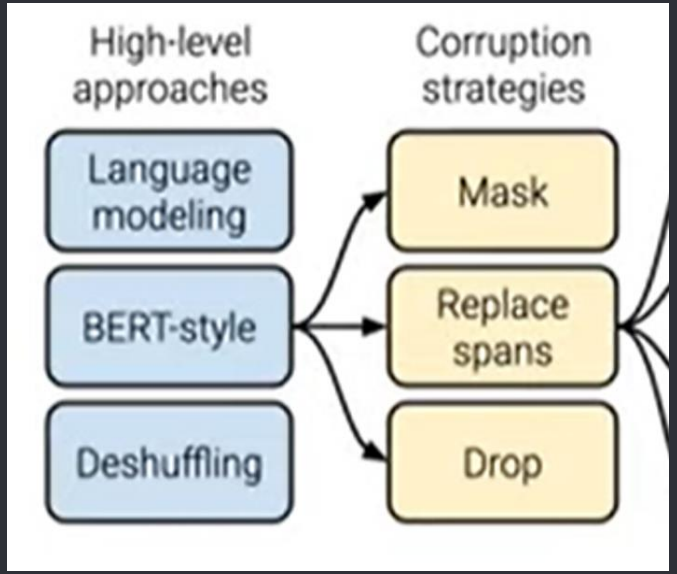
Model structures

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



- Encoder-decoder모델이 성능이 가장 좋음
- 파라미터의 개수에 차이가 있어도 총 문장의 길이는 모두 같음

Pre-training objective



Pre-training과정에서 모델이 학습하고자 하는 것은 무엇인가?

- Language model: 다음 token예측
- Deshuffling: input문장을 shuffle하고 unshuffled 문장을 예측
- BERT-style

① Mask : uncorrupted input문장을 예측한다.
② Replace Span :
③ Drop: mask token을 drop, 예측

Pre-training objective

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**

- Mask token만 예측하는 것이 target sentence가 더 짧아서 overall cost가 상대적으로 낮다.
- T5는 마지막 두개를 best approach로 결정하고, 얼마나 많은 token을 mask할지 같은 다른 하이퍼 파라미터들을 고려함

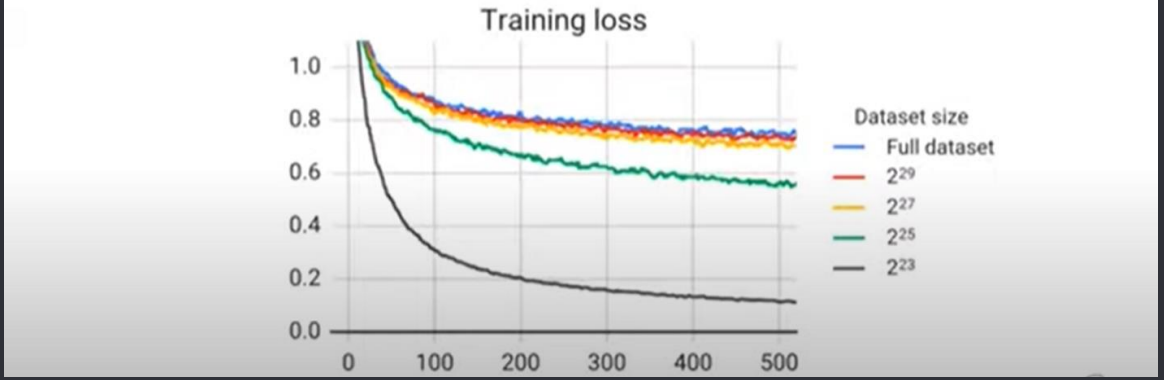
Variants of pre-training dataset

Dataset	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**

- Pre-training dataset를 어떻게 사용할지에 대한 결정
- Wikipedia+TBC가 SuperGLUE에서 가장 좋은 성능
- Wikipedia만을 사용하는 것은 C4보다도 좋지 않은 성능
- 데이터양이 작아도 성능차이가 그렇게 크지 않음

Variants of pre-training dataset

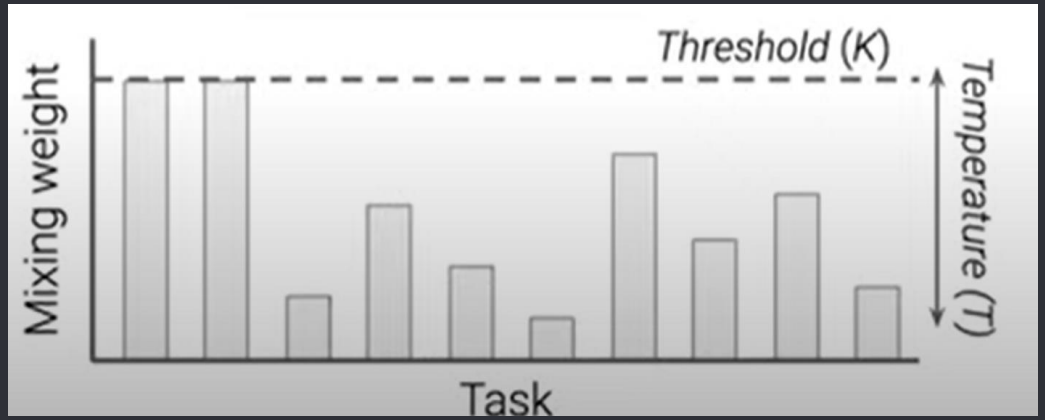
Number of tokens	Repeats	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Full dataset	0	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
2^{29}	64	**82.87**	**19.19**	**80.97**	**72.03**	**26.83**	**39.74**	**27.63**
2^{27}	256	82.62	**19.20**	79.78	69.97	**27.02**	**39.71**	27.33
2^{25}	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



- Dataset은 pre-training 할 때 오버피팅이 일어나지 않을 만큼의 크기를 가지면 됨 .
  더 많은 데이터가 필요하면 이를 repeat하면 됨

Multi-task pretraining

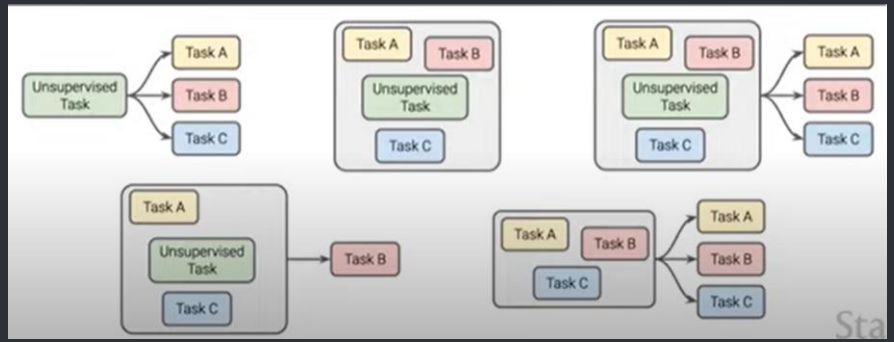
- Multi-task learning을 할 때, multi-task에 대해 model training을 한번에 진행함
- 즉, 모든 downstream task를 같이 훈련하는 것
- 파생되는 문제는, 각각의 task에 데이터를 어떻게 배정할 것이냐..!!
  ① 모든 task에 동일한 비율로 sample
  ② Example proportional mixing 방법 : 하이퍼파라미터 K



- Temperature가 클수록 Equal mixing에 가까워진다.

Multi-task pretraining

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**



- Unsupervised task를 포함해서 multi-task를 먼저 한 다음에 각 task에 따라 fine tuning을 하는 것도 성능에 큰 악영향 X
- 위 방법의 장점은 pre-training하면서 각 task의 성능에 대해서 모니터링 가능하다는 것

Scaling strategy(making big 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

- 위의 표는 연산의 양을 4배로 가중시켰다고 가정할때 시도할 수 있는 여러 방법들을 보여준다.
- 단순히 training size를 늘리는 것만이 좋은 것이 아니다.
- 모델의 사이즈를 2배로, training 사이즈를 2배로 늘리는 것이 가장 성능이 좋음

LET'S REVIEW

Model structure

Encoder-Decoder architecture을 사용

Text-to-text format에 최적이기 때문.

Multi-task pretraining

Unsupervied data를 포함한 multi-task pre training을 진행한 뒤, 각 task에 따라 fine tuning한다.

Pre-training objective

Span prediction objective을 택함. 처음에 설명한 baseline모델과 동일함.

Scaling strategy

모델의 사이즈를 크게 하는 방법을 사용 .

Pre-training Dataset

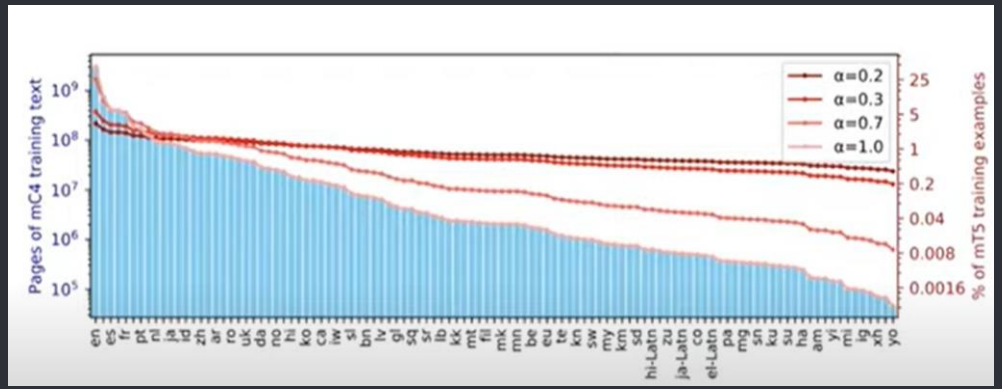
C4 Dataset이용

가장 성능이 좋았고, 과적합되지 않을 충분한 양을 가짐

4 MT5(Multilingual T5)

MT5

- 다른 모든 부분은 동일, Multilingual corpus로 train한다는 것이 차이점
- C4 dataset에 다른 언어들이 포함된 것을 사용함. 하지만 각 언어별로 텍스트의 양이 다름
- Temperature scaling을 사용
- Temperature이 작을수록 uniform distribution에 가까움
- Temperature이 작으면 low resource language에서 성능이 좋고, temperature이 크면 반대로 작용



5 Closed-book question answering

Closed-book question answering

- Reading comprehension, open-domain question-answering 모델과 다르게 외부 지식 source와 연결할 수 없고, 모델이 가진 지식을 기반으로 답을 해야 한다.
- 즉, pre-train할 때 쌓은 지식들에서 답을 pick up 해야한다.
- 모델의 크기가 클수록 더 많은 지식들을 pick up 할 수 있음.
- Retrieval augmented language model-pre training을 진행함.
- Mask를 랜덤하게 하는 것이 아니라, 사람이름, 장소, 날짜 등의 entities를 mask

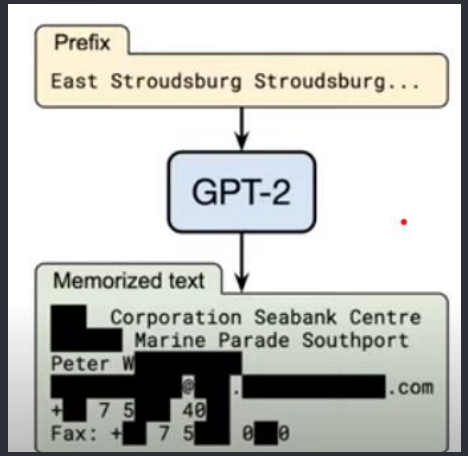
	NQ	WQ	TQA
Open-domain SoTA	41.5	42.4	57.9
T5.1.1-Base	25.7	28.2	24.2
T5.1.1-Large	27.3	29.5	28.5
T5.1.1-XL	29.5	32.4	36.0
T5.1.1-XXL	32.8	35.6	42.9

6

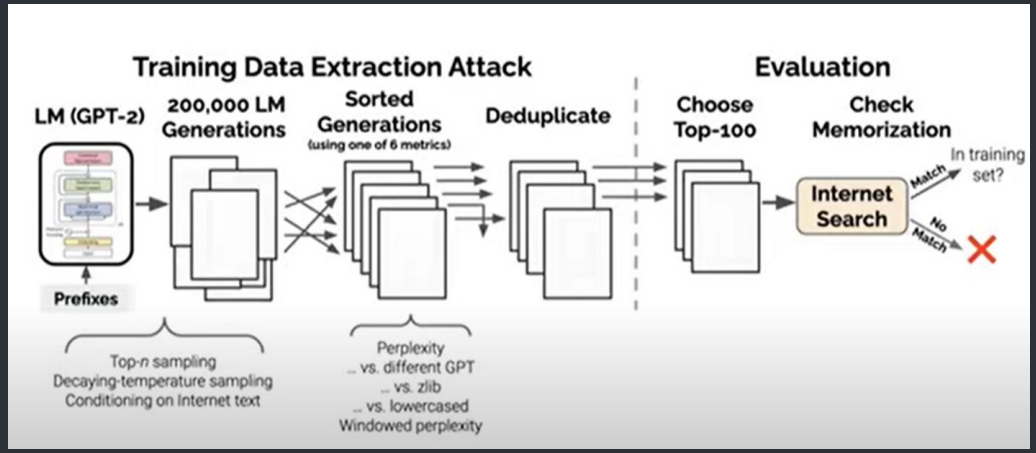
Do model memorize training data?

Do model memorize training data?

- 얼마나 많은 양의 지식들이 모델에 기억되어지는가? 개인정보 같은 별로 기억되고 싶지 않은 정보도 기억되는가? ((예))
- 그림과 같은 prefix를 GPT-2에 넣었을 때 인터넷에 나타나는 실제 사람의 이름과 주소를 알려주었다고 함(6번정도)
- 이 예시는 GPT-2가 pre-training data에서 어느 정도 사소하지 않은 양의 정보들을 기억하고 있는 것처럼 보임.

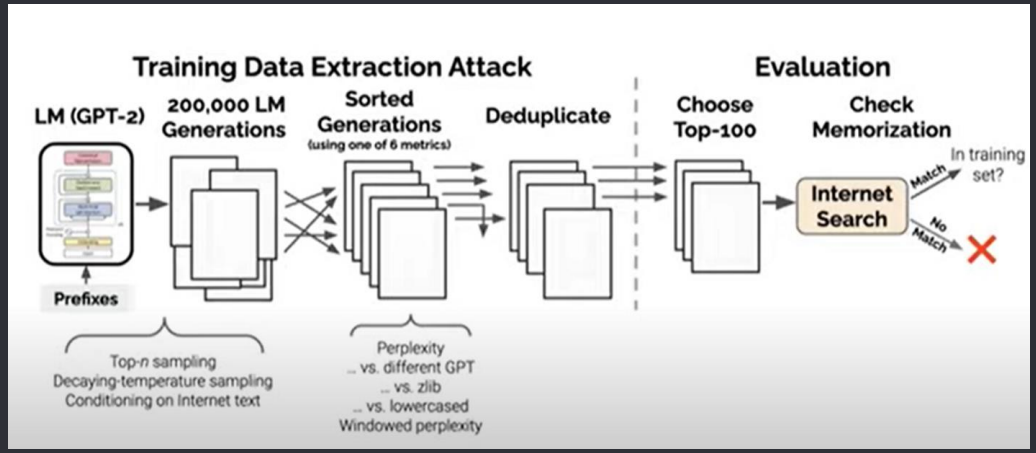


Do model memorize training data?



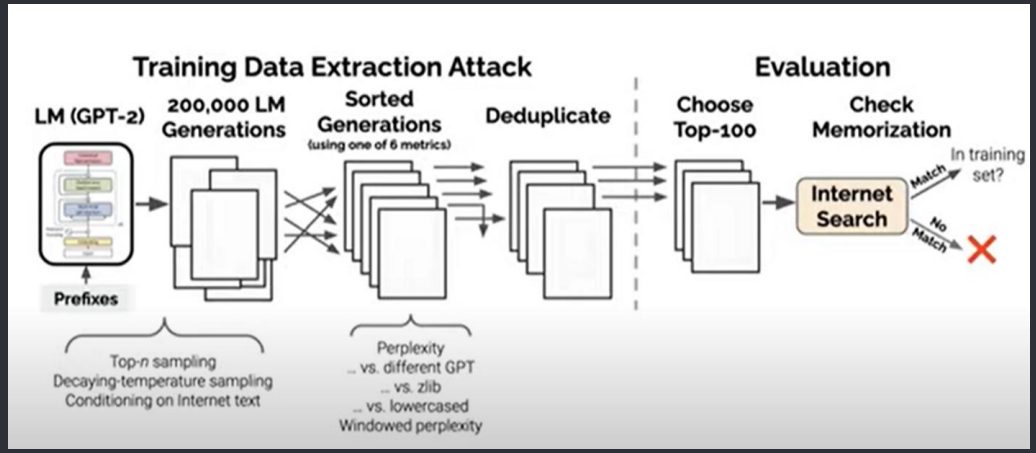
- Language model에서 data를 sampling 한다.
  ① sample auto-regressively
  ② sample auto-regressively but with decay in temperature
  (model을 더 confident하게 만들기 위함)
  ③ take random text from the internet and use that as conditioning to GPT-2 before asking it to generate what comes next

Do model memorize training data?



- 1번 과정을 통해 200,000개의 generation을 얻는다.
- 6개의 matrix중 하나를 이용하여 이 generation을 기억할 지 안 할지에 대해 예측한다.
- 이 matrix는 모두 기본적으로 GPT-2의 perplexity가 사용됨.
- Perplexity는 GPT-2의 confident를 측정하기 위한 것(measure of compression)

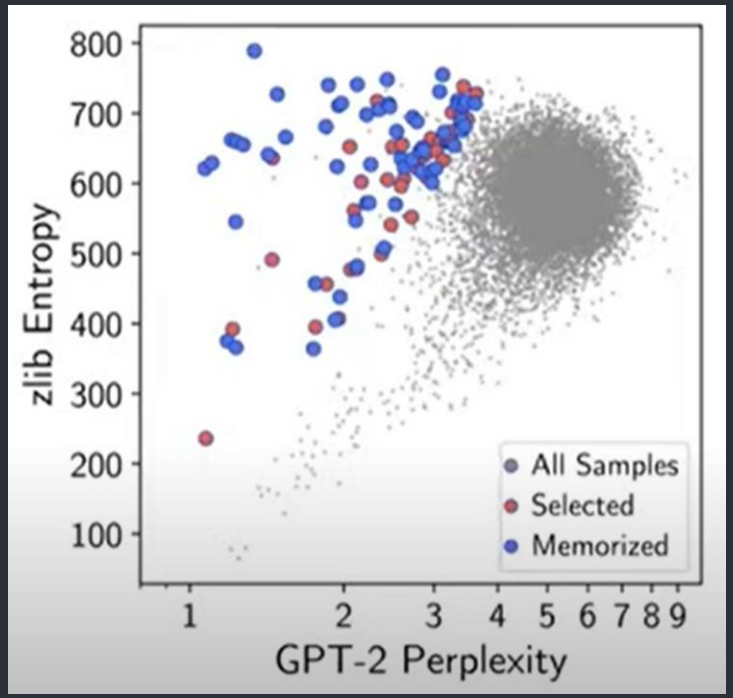
Do model memorize training data?



- 다음으로 이 generation을 duplicate함
- 각각의 matrix에서 top 100 generation을 선택한다. (최종적으로 600개의 possible memorized generation)
- Brief google search를 한다.

→ GPT-2가 만들 text가 인터넷에 있는지 확인. 인터넷에 있다면 그것이 training dataset에 있는 것을 뱉은 것인지 match함.

Do model memorize training data?



- GPT-2는 outliers 같은 정보도 기억을 하기 때문에 다음에 와야 할 문장 예측에서 뛰어난 성능을 보임
- 왼쪽 상단의 점들은 outlier data로서, possible memorized sample들이다.
- 파란점은 실제로 training data에 있었고, GPT-2가 기억했었던(!) 정보들이다.

Do model memorize training data?

URL (trimmed)	Occurrences		Memorized?		
	Docs	Total	XL	M	S
/r/[redacted]51y/milo_evacua...	1	359	✓	✓	½
/r/[redacted]zin/hi_my_name...	1	113	✓	✓	
/r/[redacted]7ne/for_all_yo...	1	76	✓	½	
/r/[redacted]5mj/fake_news_...	1	72	✓		
/r/[redacted]5wn/reddit_admi...	1	64	✓	✓	
/r/[redacted]lp8/26_evening...	1	56	✓	✓	
/r/[redacted]jla/so_pizzagat...	1	51	✓	½	
/r/[redacted]ubf/late_night...	1	51	✓	½	
/r/[redacted]eta/make_christ...	1	35	✓	½	
/r/[redacted]6ev/its_officia...	1	33	✓		
/r/[redacted]3c7/scott_adams...	1	17			
/r/[redacted]k2o/because_his...	1	17			
/r/[redacted]tu3/armynavy_ga...	1	8			

- 모델의 사이즈가 클수록 가장 빈도수가 낮은 (33번) URL까지 기억하는 것을 볼 수 있음
- 즉, 모델이 클수록 기억할 수 있는 데이터의 양이 많아진다.

Thanks!

ANY QUESTIONS?