Artficial Intelligence Laboratory

# Ch6. Text Generation with OpenAI GPT-2 and GPT-3 Models

Transformers for NLP

정보융합공학과 AI전공
정주경

부산대학교
PUSAN NATIONAL UNIVERSITY

# 목 차

# The limits of the original Transformer architecture



## Attention 구조에 의한 메모리 문제

- Query (Q): 영향을 받는 단어 A를 나타내는 변수
- Key (K): 영향을 주는 단어 B를 나타내는 변수
- Value (V): Key에 대응되는 영향력 값

$$\text{Attention}(Q,K,V) = \text{softmax}(\frac{QK^T}{\sqrt{d^k}})V$$

## Feed Forward Layer에 의한 메모리 문제

$$\text{FFN}(x) = \max(0, x \cdot W_1 + b_1) \cdot W\_2 + b_2$$

## N-stacked Residual Connection에 의한 메모리 문제

머신 러닝 프레임워크에서 미분 값을 계산하는 방법

- 클래스로 정의된 계산 흐름에 따라 입력 데이터에 대해서 출력 값을 계산. 모델에 입력이 전달될 때, 어떤 입력이 들어와서 어떤 출력이 생성되었는지 기록
- 종점부터 시작점까지 거슬러 올라가며, 각 모델의 출력 값과 이에 대응되는 입력 값을 사용해 미분 값을 근사한 수치 계산

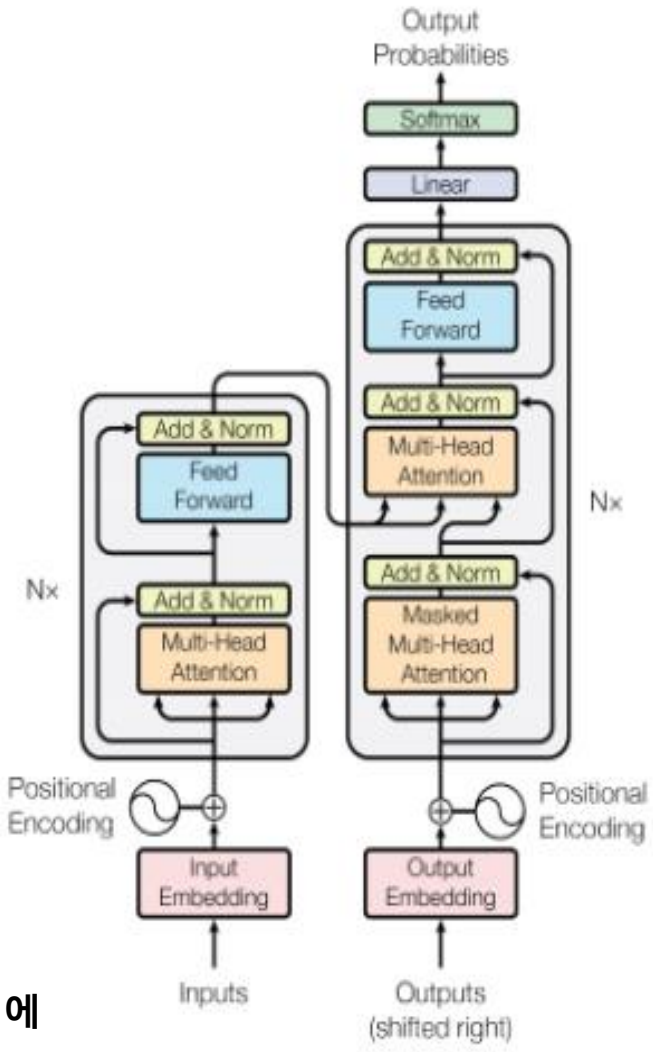



Figure 1: The Transformer - model architecture.

# The Reformer

Reformer는 LSH(Locality Sensitivity Hashing) buckets and chunking으로 attention 문제 해결

## Locality-Sensitive Hashing

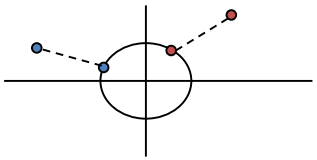
가까운 데이터끼리는 가까운 Hash값을 갖도록 구성할 수 있다면 비교하는 연산을 Hash값에 대한 연산으로 근사할 수 있다.
가까운 값들 끼리 가까운 Hash값을 가지도록 하는 방법을 LSH

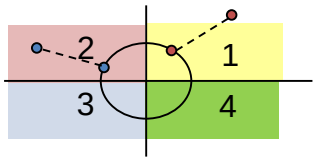
## Angular LSH

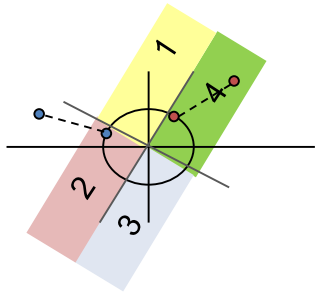
1. 전체 데이터 포인트들의 벡터를 단위 구면에 사상. 전체 데이터 포인트를 오직 각도만 사용해서 기술할 수 있다.
2. 각 각도가 어느 사 분면에 있는지 확인. 사 분면의 번호를 Hash값으로 사용한다면, 비슷한 데이터들을 가깝게 구성할 수 있다.
3. 사상한 구면을 필요한 만큼 임의로 회전. 데이터가 가까울수록 전체 Hash값을 공유할 가능성이 높아지고, 충분히 많은 Hash값을 사용하면 데이터를 구별하는 변별력이 생김.

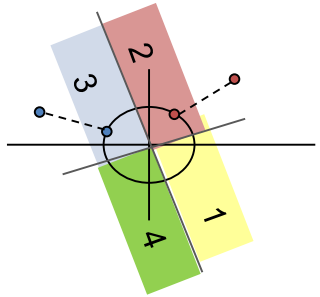
Ex) X1=(3,4), X2=(-12,5) 반지름1인구에 사상 => X1^=(3/5, 4/5), X2^=(-12/13, 5/13)

X1의 Hash값 (1, 4, 2), X2의 Hash값 (2, 2,3)





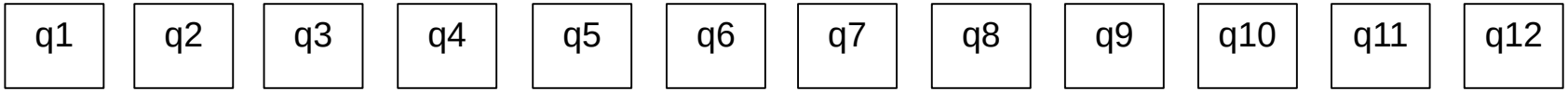



# The Reformer

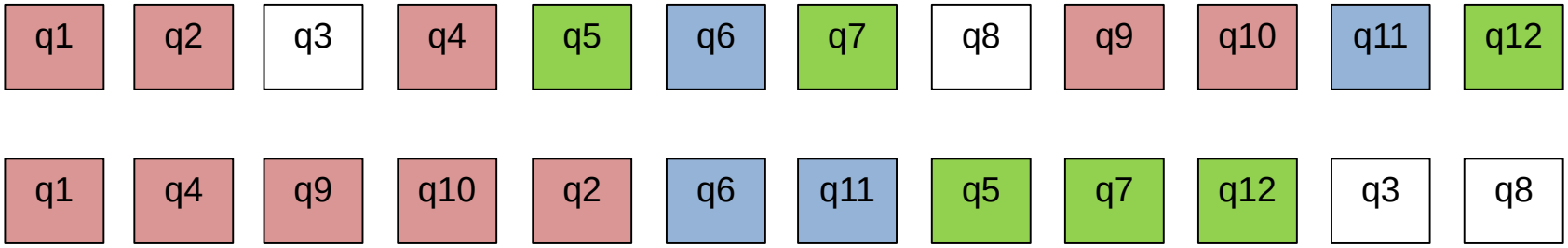
## LSH Attention 적용하기

$$\text{Query}(Q) = \text{Key}(K)$$

- Q = K이므로 각 데이터 포인트에 대한 Q = K 값을 일렬로 된 벡터 형태로 나타냄



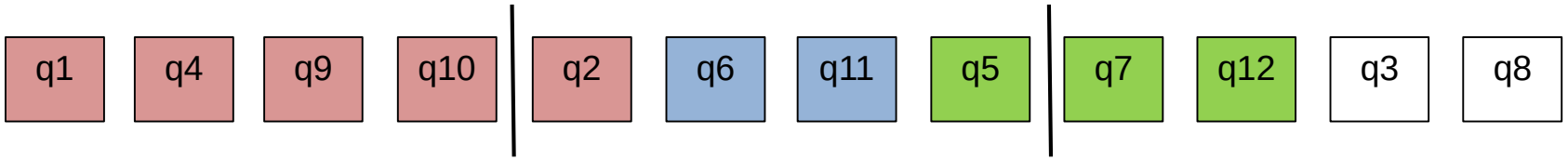

- 각 데이터 포인트에 LSH적용. 그 후 같은 Hash값을 가진 데이터 포인트 끼리 버킷으로 묶습니다(Bucketing). 각 Hash 버킷에 임의로 순서를 매겨서 정렬합니다.



# The Reformer

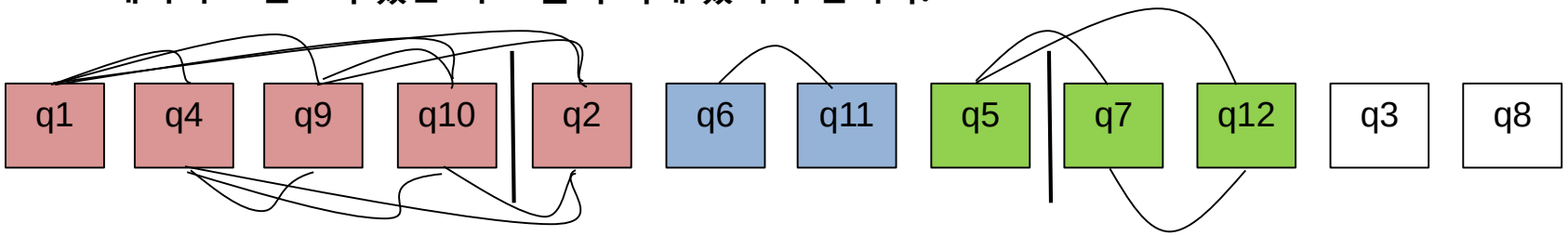
## LSH Attention 적용하기

- 각 버킷에는 높은 확률로 데이터 포인트들이 불균형하게 배당될 것입니다. 따라서 데이터 포인트들을 고정된 크기의 구역으로 분절합니다(Chunking)




- Attention Weight를 계산하는데, 다음 조건을 모두 만족하는 쌍들에 대해서만 계산

1. 두 데이터 포인트가 같은 버킷에 있어야 합니다.
2. 두 데이터 포인트는 서로 같은 구역에 있거나, Attention의 도착점 데이터 포인트는 시작 데이터 포인트가 있는 바로 앞 구역에 있어야 합니다.




데이터 포인트의 길이 = l, 분절된 부분의 수 = c

분절된 부분의 크기는 l/c, attention의 수는 $l \cdot (\frac{2l}{c})^2$ 에 비례

# Pattern-Exploiting Training (PET)

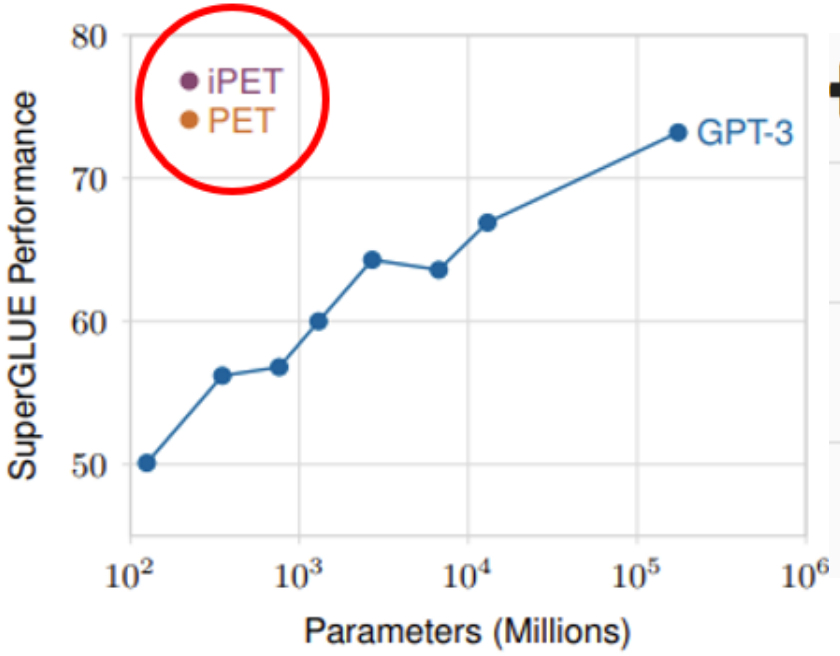
**OpenAI는 GPT-3와 같은 대형 모델을 만들었고 Google AI는 Reformer로 Transformer를 최적화**
**GPT-3는 1750억 개의 파라미터로, 모델을 학습시키는 데에 드는 비용이 약 52억 원으로 추정**

## PET

**Timo Schick and Hinrich Schutze**

***"It's Not Just Size That Matters: Small Language Models Are Also Few-Shot Learners"***

1. **텍스트 인풋을 태스크에 대한 묘사를 포함한 Cloze 스타일의 질문으로 바꾼다.**
2. **그래디언트 기반의 최적화를 수행한다.**
3. **라벨링되지 않은 데이터로부터 정보를 추가적으로 활용하여 성능을 향상한다.**




| | | | | |
|---|---|---|---|---|
| 19 | Timo Schick | iPET (ALBERT) - Few-Shot (32 Examples) | | 75.4 |
| 20 | Adrian de Wynter | Bort (Alexa AI) | | 74.1 |
| 21 | IBM Research AI | BERT-mtl | | 73.5 |
| 22 | Ben Mann | GPT-3 few-shot - OpenAI | | 71.8 |

SuperGLUE Leaderboard April 2022

# Pattern-Exploiting Training (PET)



## Core principle

***Reformulate a training task as a cloze question***

**먼저 원래 태스크를 Cloze 타입으로 변환한다.**

**Cloze는 질문은~~, 정답은___. 와 같이 빈칸이 있는 질문 스타일**

## Ex) 두 문장의 entailment 관계를 분류해야 하는 태스크

**X : (문장1 – 유가가 오른다, 문장 2- 유가가 다시 내린다) 정답 y - <not entailment>**

**⇒ P(x) : 유가가 오른다? ___, 유가가 다시 내린다.**

**이렇게 인풋을 하나의 빈칸(mask)로 포함하는 Cloze 질문으로 매핑하는 것을 P, pattern 정의**

**정답을 하나의 토큰으로 표현해주는 verbalizer v가 필요**

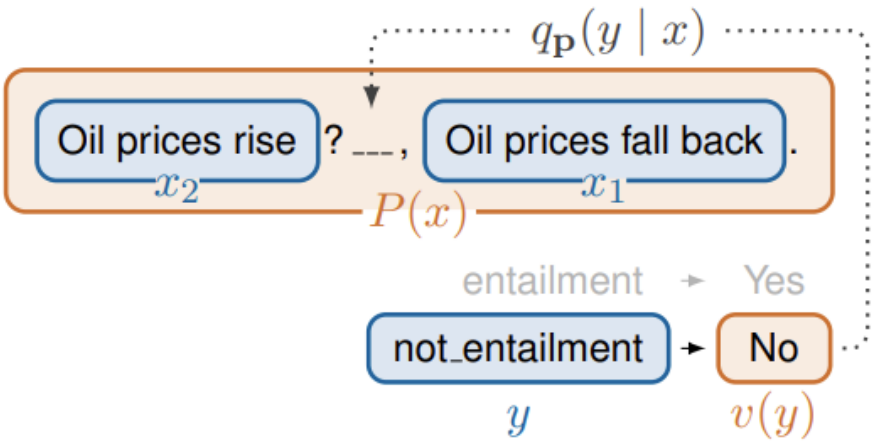
**Y = not entailment -> v(y) = NO, Y = entailment -> v(y) = Yes**

# Pattern-Exploiting Training (PET)

모델은 인풋으로부터 생성된 P(x)에 있는 빈칸에 v(y)가 적절한 토큰일 점수를 모델링
모든 가능한 v(y)가 빈칸에 들어갈 토큰일 점수를 Softmax를 통해 확률처럼 나타냄

## Ex

- 유가가 오른다 ? NO , 유가가 다시 내린다 -> 1.2점 -> y = not entailment 73%
- 유가가 오른다? YES , 유가가 다시 내린다 -> 0.2점 -> y = entailment 27%




- V(y)가 P(x)의 마스킹된 자리에 올 점수를 아래의 식과 같이 나타낼 때

$$s_{\mathbf{p}}(y \mid x) = s_M^1(v(y) \mid P(x))$$

- 인풋 x에 대한 정답이 y일 확률은 다음과 같이 계산

$$q_{\mathbf{p}}(y \mid x) = \frac{\exp s_{\mathbf{p}}(y \mid x)}{\sum_{y' \in Y} \exp s_{\mathbf{p}}(y' \mid x)}$$

# Generative Pre-trained Transformer (GPT)

## Generative

GPT는 생성 모델인데, 이는 데이터 전체의 분포를 모델링하는 머신 러닝 기법
단순히 새로운 글을 생성할 수 있다는 것은 생성 모델의 여러 특성 중 하나.

## Discriminative Model

- 데이터 X가 주어졌을 때 Label Y가 나타날 조건부확률 P(Y|X)를 직접적으로 반환(분류모델)
- Label 정보가 있어야 하기 때문에 지도학습 모델
- 클래스들이 있을 때 클래스간의 x-value(feature)들 사이에서 다른 점을 확인해 어떻게 다른가를 중점
- 작은 데이터로도 잘 작동하지만 overfitting에 주의해야 하고 Generative Model보다 계산이 적다.

## Generative Model

- 각 클래스 별 특징 데이터의 확률분포 P(X|Y) 를 추정한 다음 베이즈 정리를 사용하여 계산.

P(Y|X)를 계산할 수 있기 때문에 분류모델로 활용할 수 있고 클래스별 Conditional 확률P(Y|X)를 추정했기 때문에 확률분포상에서의 새로운 가상의 데이터를 생성하거나 확률분포 끝자락에 있는 데이터를 이상치로도 판단하는 이상치 판별 모델로도 활용할 수 있다.

- 데이터가 충분하게 있을 경우 overfitting될 확률이 상대적으로 적지만 데이터가 많아야 하고 Discriminative Model에 비해 여러가지 확률값을 계산해야 함.

# Generative Pre-trained Transformer (GPT)

## Pre-trained

GPT가 다른 모델보다 월등하게 좋은 성능을 보이는 가장 큰 이유는 어마어마한 학습데이터의 양. GPT-3모델은 5000억 개의 단어(token)를 포함한 데이터 셋을 보고 학습. 대부분의 데이터는 인터넷에서 크롤링한 Common Crawl.

| Dataset | Quantity (tokens) | Weight in training mix | Epochs elapsed when training for 300B tokens |
|---|---|---|---|
| Common Crawl (filtered) | 410 billion | 60% | 0.44 |
| WebText2 | 19 billion | 22% | 2.9 |
| Books1 | 12 billion | 8% | 1.9 |
| Books2 | 55 billion | 8% | 0.43 |
| Wikipedia | 3 billion | 3% | 3.4 |

## Transformer

기계 번역 모델이 기본적으로 Encoder-Decoder로 구성. 전체 문장을 다 보고 중간에 들어가는 단어를 예측하는 BERT와는 다르게, GPT는 기존의 언어 모델과 같이 다음 단어를 예측해야 하기 때문에 Transformer Decoder를 사용

# Generative Pre-trained Transformer (GPT)

## BERT vs GPT

- GPT는 Language Model.

이전 단어들이 주어졌을 때 다음 단어가 무엇인지 맞추는 과정에서 pretrain.

문장 시작부터 순차적으로 계산한다는 점에서 일방향(unidirectional)

트랜스포머에서 디코더만 취해 사용

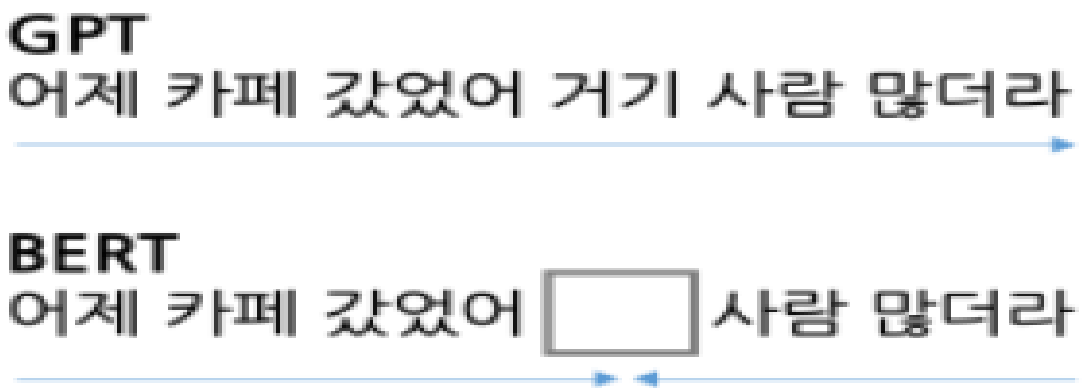
- BERT는 Masked Language Model.

문장 중간에 빈칸을 만들고 해당 빈칸에 어떤 단어가 적절한지 맞추는 과정에서 pretrain.

빈칸 앞뒤 문맥을 모두 살필 수 있다는 점에서 양방향(bidirectional)

트랜스포머에서 인코더만 취해 사용

-이때문에 GPT는 문장 생성에, BERT는 문장의 의미를 추출하는데 강점



# Generative Pre-trained Transformer (GPT)

## Fine-tuning 단점

1. 매번 새로운 task를 풀 때 마다 많은 레이블 데이터가 필요하다.
2. Fine-tuning 기반의 방법들은 사전 학습 중에는 다량의 지식을 학습하지만, 특정 task에 fine tuning되는 과정에서 특정 task외의 문제에 대한 일반화 능력을 상실한다. 그래서 벤치마크 데이터에 대해서는 좋은 성적을 내는 것처럼 보이지만 실제 사람의 능력과는 동떨어져 있다.
3. 사람은 실제 언어를 활용할 때 대규모의 라벨을 필요로 하지 않는다. 간단한 지시 혹은 몇가지 예시만으로도 가능하다.

## 메타 러닝

훈련 시에 다양한 기술과 패턴 인식 방법을 학습한 후 추론 시에 이러한 능력을 사용하여 요구되는 task에 빠르게 적응

GPT-2에서는 사전 학습 시 풀고자 하는 task를 input으로 넣는 in-context-learning 방법 사용.

몇몇 task들에서는 기존 fine-tuning을 넘지는 못함.

# Generative Pre-trained Transformer (GPT)

## GPT-3를 평가하는 방법

### The three settings we explore for in-context learning

**Zero-shot**

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
1  Translate English to French:        ← task description
2  cheese =>                           ← prompt
```

**One-shot**

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

```
1  Translate English to French:        ← task description
2  sea otter => loutre de mer          ← example
3  cheese =>                           ← prompt
```

**Few-shot**

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

```
1  Translate English to French:        ← task description
2  sea otter => loutre de mer          ← examples
3  peppermint => menthe poivrée
4  plush girafe => girafe peluche
5  cheese =>                           ← prompt
```

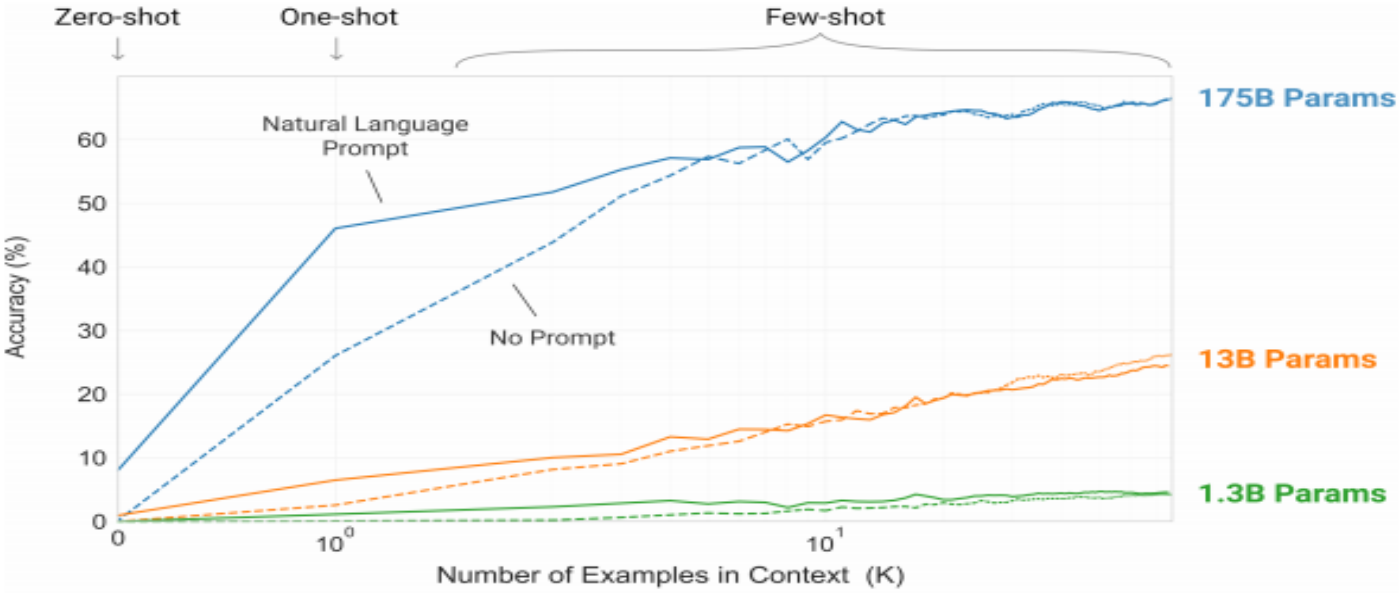
### Traditional fine-tuning (not used for GPT-3)

**Fine-tuning**

The model is trained via repeated gradient updates using a large corpus of example tasks.

```
sea otter => loutre de mer             ← example #1
        ↓
   gradient update
        ↓
peppermint => menthe poivrée           ← example #2
        ↓
   gradient update
        ↓
        ...
        ↓
plush giraffe => girafe peluche        ← example #N
        ↓
   gradient update
        ↓
cheese =>                              ← prompt
```

# Generative Pre-trained Transformer (GPT)




1. 태스크에 대한 자연어 설명은 모델 성능을 향상시킨다 (Natural Language Prompt > No Prompt)
2. 모델의 문맥 윈도우에 더 많은 예제를 놓을수록 성능이 향상 (K에 비례하여 정확도 증가)
3. 큰 모델일수록 in-context정보를 잘 활용 (175B params 결과)

이때, 성능을 측정하는 동안 그래디언트 업데이트나,Fine-tuning은 일절 일어나지 않는다.
K의 증가에  따라 정확도가 증가하는 것처럼 보이는 부분은, 오로지 문맥에 포함된 예제의 개수를 모델이 얼마나 잘 활용하는가에 기인한다.

# Generative Pre-trained Transformer (GPT)

## GPT-1,2,3 차이

2018년 OpenAI는 첫 번째 GPT를

*Improving Language Understanding with Unsupervised Learning* 논문 공개

2019년 두 번째 모델

*Language Models are Unsupervised Multitask Learners*

2020년 세 번째 모델

*Language Models are Few-shot Learners*

세 버전에서 큰 차이는 사이즈.

기본적인 모델 구조나 학습 원리는 모두 같으나, 트랜스포머의 사이즈가 GPT-3는 175B(1750억)

GPT-2와 비교하면 100배

더 큰 모델로 더 많은 데이터를 학습. 모델과 데이터의 스케일만 키웠음에도 불구하고 GPT-3는 GPT-2에 비해 비교할 수 없을 정도의 성능을 보이며, 다양한 언어 태스크를 성공적으로 수행.

GPT-3의 가장 큰 의미는 스케일만으로도 이런 일이 가능하다는 걸 보여준 것

| 모델명 | 사이즈 |
|---|---|
| GPT-1 | 0.125B(=125M) |
| GPT-2 | 1.5B |
| GPT-3 | 175B |

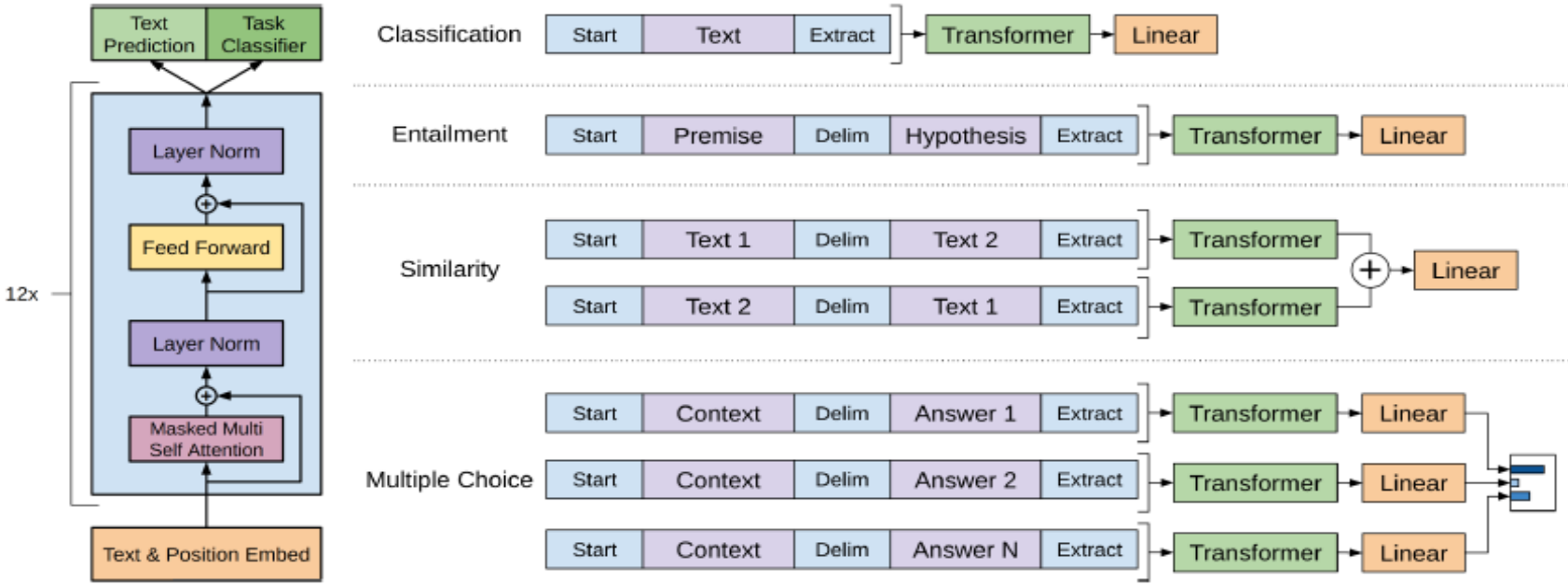
# Generative Pre-trained Transformer (GPT)

## 배경

대부분 자연어처리 딥러닝은 supervised learning을 기반으로 하는데 이를 위해선 데이터의 라벨이 있도록 작업이 필요해 데이터셋의 크기가 큰 경우 비효율적. 또한 task가 달라지면 또 다른 작업이 필요

Unsupervised learning으로 텍스트의 good representation을 모델이 학습하는 것이 더 좋은 성능을 나타내는 것으로 알려졌는데 GPT또한 unsupervised learning에 초점. 또한 task에 따른 adaption들을 최소한으로 하여 다양한 task에 적용 가능한 것을 목표(task-agnostic model)

## GPT-1 구조



# Generative Pre-trained Transformer (GPT)

## GPT-2 구조

GPT-1의 한계는, unsupervised learning을 지향했음에도 특정 task에 적용할 때 성능향상을 위해서 fine-tuning과정과 input transformation이 들어갔다는 것

P(output/input) => P(output/input, task)

이렇게 변경됨에 따라 GPT-2부터는 fine-tuning과정이 없다.

- Sub-layer의 입력-출력까지의 수식이

LayerNorm(sublayer(x) + x) => x+sub_layer(LayerNorm(x))

-논문에서는 residual block과 유사한 형태라고 한다.

- Residual layer의 weight는 1/N (N=residual layer의 수) 값의 Scaling factor로 초기화

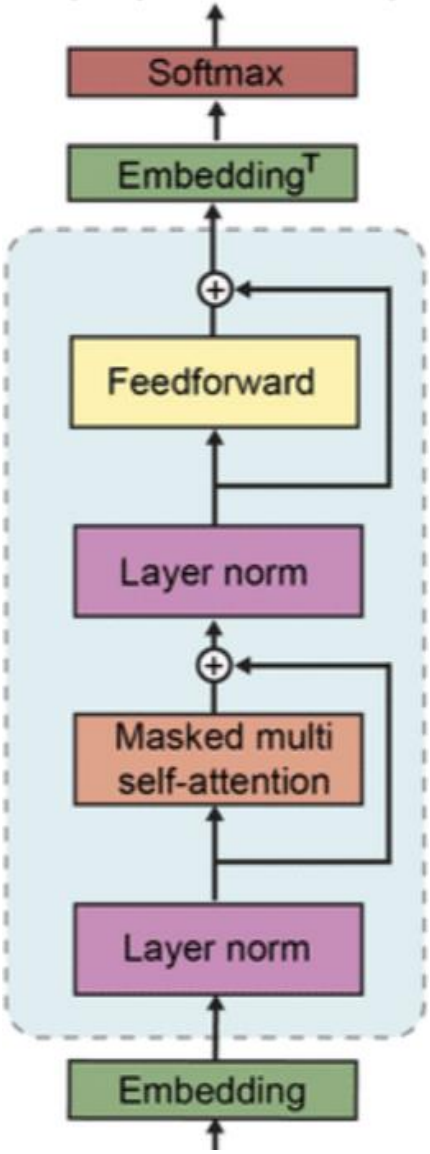
- voca size가 5만대로 늘었고, context size 또한 512 -> 1024로 증가

- 쌓은 decoder의 수가 몇 개인지에 따라 모델 구조가 조금씩 달라지는데 일반적으로 layer가 가장 많은 model(48개 사용)을 GPT-2라고 말한다.

여러가지 task들에 대해 실험을 했고, 좋은 성능을 보였으나

Zero-shot, few-shot learning에 대해서는 under-fit, 즉 좋지 않은 성능.

이는 GPT-3로 나아가는 계기

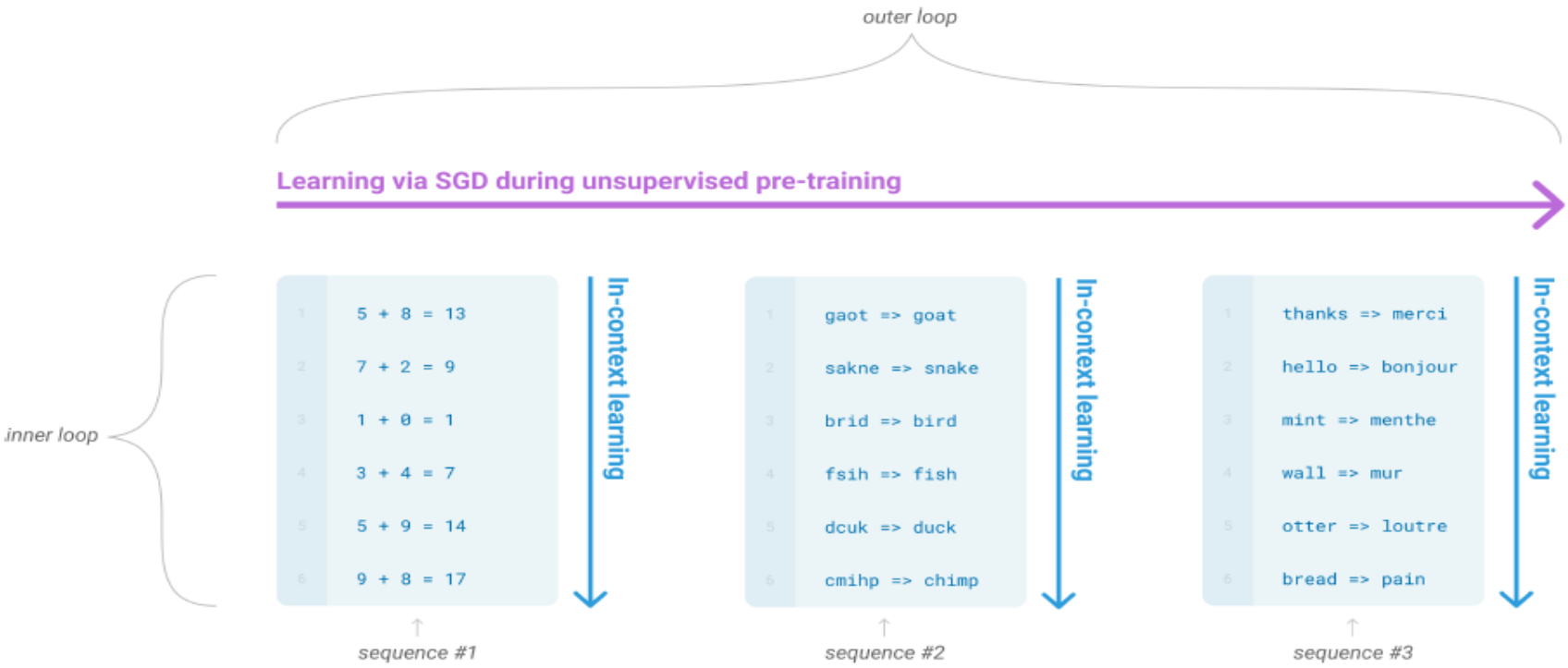


# Generative Pre-trained Transformer (GPT)

## GPT-3 구조

**Transformer의 attention부분을 full self-attention이 아니라 sparse self-attention으로 변경**

**기존 full self-attention은 각 layer당 $N^2$의 attention matrix와 attention head를 필요로 하는데, 매우 많은 메모리 용량이 필요. 따라서 각 output position이 N개의 subset에만 attention을 할 수 있도록 한다.**



# Generative Pre-trained Transformer (GPT)

## Input representation

**Byte Pair Encoding(BPE) 방식 채택**

**BPE는 subword 기반의 인코딩 방법으로 문자 단위로 단어를 분해하여 Vocabulary를 생성하고, 반복을 통해 빈도수가 높은 문자 쌍을 지속적으로 Vocabulary에 추가하는 방법 (greedy)**

**Ex)**

$$Vocabulary_{word} = \{apple, available, capable\}$$

$$Vocabulary_{caracter} = \{a, p, l, e, v, i, b, c, p\}$$

**매 회 반복을 통해 le, ble, able 과 같이 함께 자주 등장하는 문자 쌍을 Character vocabulary에 greedy하게 추가하는 방법**

$$Vocabulary_{BPE} = \{a, p, l, e, v, i, b, c, p, le, ble, able\}$$

**따라서 BPE는 자주 등장하는 토큰 시퀀스의 단어 수준 입력과 자주 등장하지 않는 토큰 시퀀스의 문자 수준 입력을 잘 보간(interpolate). 또한 OoV에 대해서도 합리적인 토큰 화가 가능**

**하지만 {dog., dog?, dog!}과 같이 단어의 유의미하지 않은 Variation을 추가하는 경향이 큼**

**이는 한정된 Vocabulary 크기를 최적으로 사용하지 못하게 할 가능성이 큼**

**논문에서는 Byte 시퀀스를 위한 BPE를 적용하기 위해 문자 수준 이상의 병합을 막음.**

**이는 Vocabulary 공간을 최적으로 활용하며 Input representation을 구성할 수 있도록 하였음**

# Generative Pre-trained Transformer (GPT)

## GPT-3

- 스케일에 따라 다음과 같이 8가지 모델을 학습하고 테스트. 가장 큰 모델은 96층의 레이어, 12,288차원의 히든 차원, 96개 attention head를 가지는 총 1750억 개의 파라미터 모델.

모든 모델은 3,000억 토큰에 대해 학습

| Model Name | $n_{\text{params}}$ | $n_{\text{layers}}$ | $d_{\text{model}}$ | $n_{\text{heads}}$ | $d_{\text{head}}$ | Batch Size | Learning Rate |
|---|---|---|---|---|---|---|---|
| GPT-3 Small | 125M | 12 | 768 | 12 | 64 | 0.5M | $6.0 \times 10^{-4}$ |
| GPT-3 Medium | 350M | 24 | 1024 | 16 | 64 | 0.5M | $3.0 \times 10^{-4}$ |
| GPT-3 Large | 760M | 24 | 1536 | 16 | 96 | 0.5M | $2.5 \times 10^{-4}$ |
| GPT-3 XL | 1.3B | 24 | 2048 | 24 | 128 | 1M | $2.0 \times 10^{-4}$ |
| GPT-3 2.7B | 2.7B | 32 | 2560 | 32 | 80 | 1M | $1.6 \times 10^{-4}$ |
| GPT-3 6.7B | 6.7B | 32 | 4096 | 32 | 128 | 2M | $1.2 \times 10^{-4}$ |
| GPT-3 13B | 13.0B | 40 | 5140 | 40 | 128 | 2M | $1.0 \times 10^{-4}$ |
| GPT-3 175B or "GPT-3" | 175.0B | 96 | 12288 | 96 | 128 | 3.2M | $0.6 \times 10^{-4}$ |

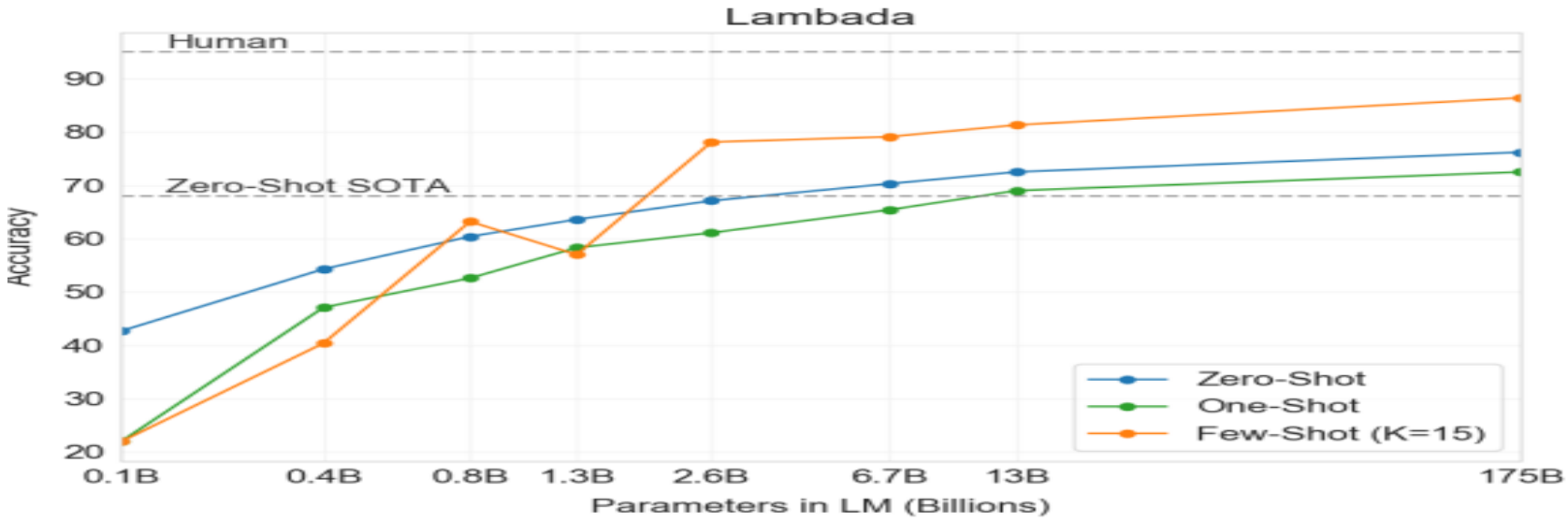
- 더 큰 모델에 대해서는 더 큰 배치를 적용했으나, 오히려 learning rate는 작게 적용함
- 학습 과정에서 그래디언트의 noise scale을 측정해 배치 사이즈를 정하는 데에 활용
- 큰 모델 학습에는 메모리가 부족하기 때문에, 행렬 곱에 있어 모델 병렬화와 레이어 사이의 모델 병렬화를 섞어서 사용

# Generative Pre-trained Transformer (GPT)

- Language Modeling

| Setting | PTB |
|---|---|
| SOTA (Zero-Shot) | 35.8[a] |
| GPT-3 Zero-Shot | **20.5** |

- LAMBADA (문장 완성하기/ 언어의 장기 의존성을 모델링하는 태스크)



# Generative Pre-trained Transformer (GPT)

| Setting | LAMBADA (acc) | LAMBADA (ppl) | StoryCloze (acc) | HellaSwag (acc) |
|---|---|---|---|---|
| SOTA | 68.0[a] | 8.63[b] | **91.8**[c] | **85.6**[d] |
| GPT-3 Zero-Shot | **76.2** | **3.00** | 83.2 | 78.9 |
| GPT-3 One-Shot | **72.5** | **3.35** | 84.7 | 78.1 |
| GPT-3 Few-Shot | **86.4** | **1.92** | 87.7 | 79.3 |

Cloze & 문장완성 태스크에서의 성능

- **HellaSwag (짧은 글이나 지시사항을 끝맺기에 가장 알맞은 문장을 고르는 태스크)**

모델은 어려워하지만 사람에게는 쉬운 태스크 중 하나, 현 SOTA인 multi-task 학습 후 fine-tuning 전략을 취한 ALUM에는 미치지 못하는 성능

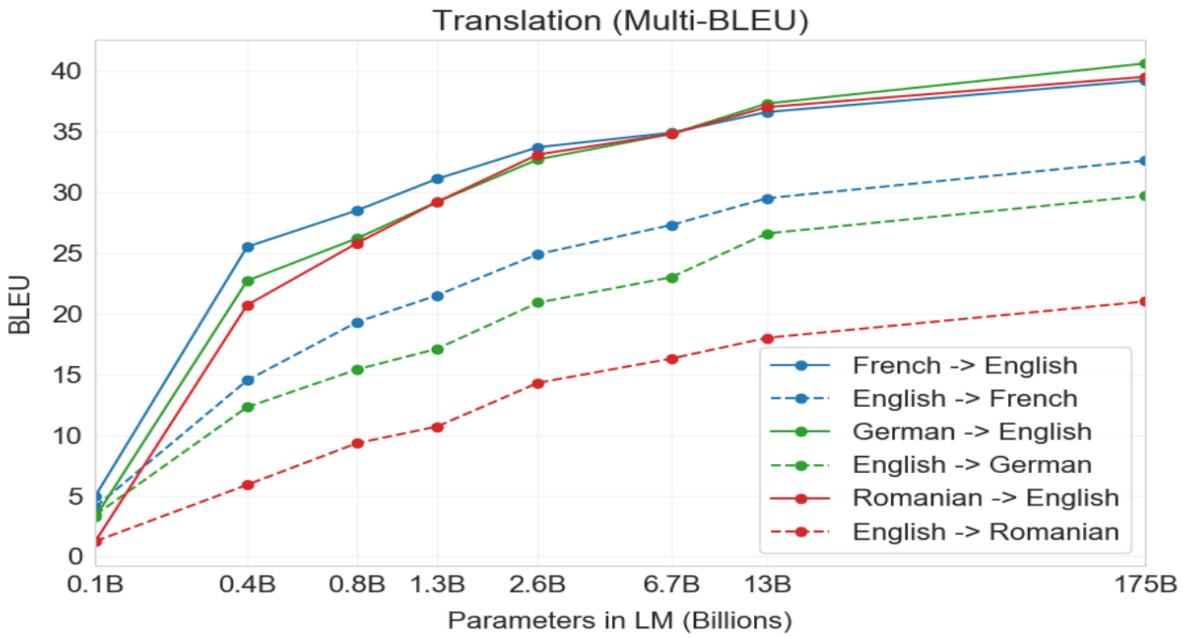
- **StoryCloze(다섯 문장의 긴 글을 끝맺기에 적절한 문장을 고르는 태스크)**

Few-shot(K=70)으로 87.7%의 성능을 얻었고, BERT 기반의 fine-tuning SOTA보다 4.1% 낮은 성적이긴 하나 기존의 zero-shot 성능은 10% 가까이 뛰어넘음

# Generative Pre-trained Transformer (GPT)

## Translation

| Setting | En→Fr | Fr→En | En→De | De→En | En→Ro | Ro→En |
|---|---|---|---|---|---|---|
| SOTA (Supervised) | **45.6**[a] | 35.0 [b] | **41.2**[c] | 40.2[d] | **38.5**[e] | **39.9**[e] |
| XLM [LC19] | 33.4 | 33.3 | 26.4 | 34.3 | 33.3 | 31.8 |
| MASS [STQ+19] | 37.5 | 34.9 | 28.3 | 35.2 | 35.2 | 33.1 |
| mBART [LGG+20] | - | - | 29.8 | 34.0 | 35.0 | 30.5 |
| GPT-3 Zero-Shot | 25.2 | 21.2 | 24.6 | 27.2 | 14.1 | 19.9 |
| GPT-3 One-Shot | 28.3 | 33.7 | 26.2 | 30.4 | 20.6 | 38.6 |
| GPT-3 Few-Shot | 32.6 | 39.2 | 29.7 | 40.6 | 21.0 | 39.5 |

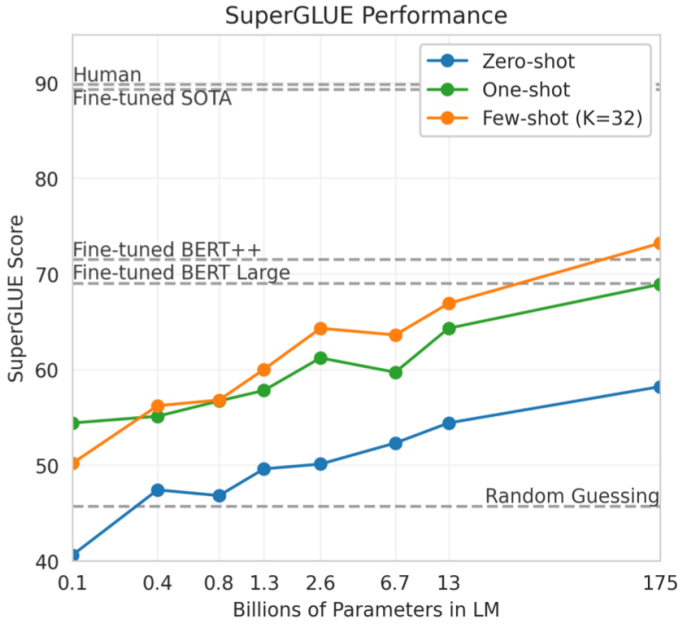


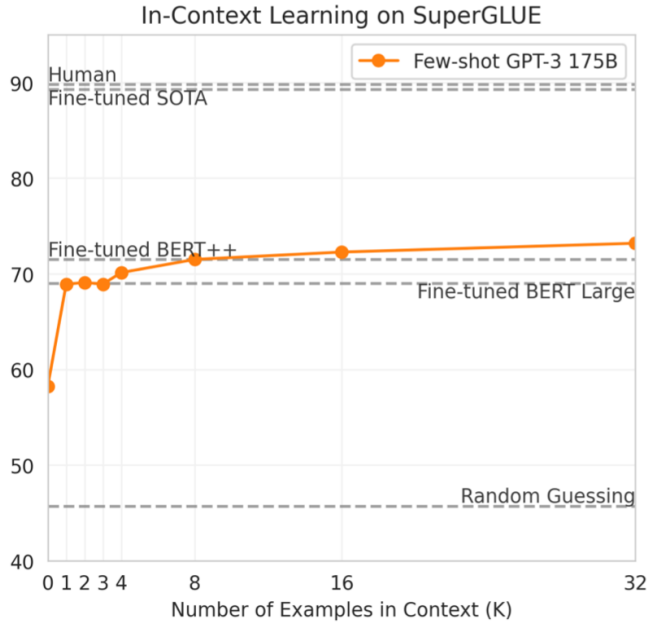
# Generative Pre-trained Transformer (GPT)

## SuperGLUE

| | SuperGLUE Average | BoolQ Accuracy | CB Accuracy | CB F1 | COPA Accuracy | RTE Accuracy |
|---|---|---|---|---|---|---|
| Fine-tuned SOTA | **89.0** | **91.0** | **96.9** | **93.9** | **94.8** | **92.5** |
| Fine-tuned BERT-Large | 69.0 | 77.4 | 83.6 | 75.7 | 70.6 | 71.7 |
| GPT-3 Few-Shot | 71.8 | 76.4 | 75.6 | 52.0 | 92.0 | 69.0 |

| | WiC Accuracy | WSC Accuracy | MultiRC Accuracy | MultiRC F1a | ReCoRD Accuracy | ReCoRD F1 |
|---|---|---|---|---|---|---|
| Fine-tuned SOTA | **76.1** | **93.8** | **62.3** | **88.2** | **92.5** | **93.3** |
| Fine-tuned BERT-Large | 69.6 | 64.6 | 24.1 | 70.0 | 71.3 | 72.0 |
| GPT-3 Few-Shot | 49.4 | 80.1 | 30.5 | 75.4 | 90.2 | 91.1 |

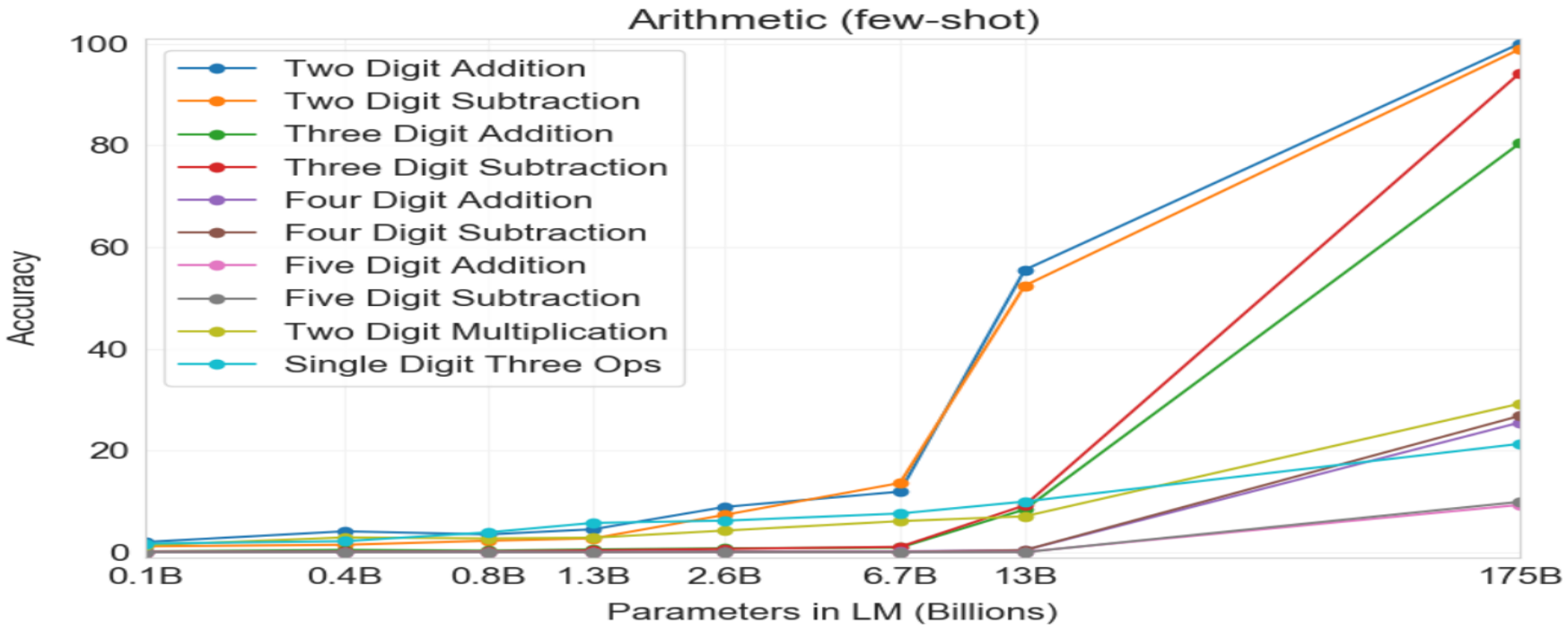


# Generative Pre-trained Transformer (GPT)

## Arithmetic

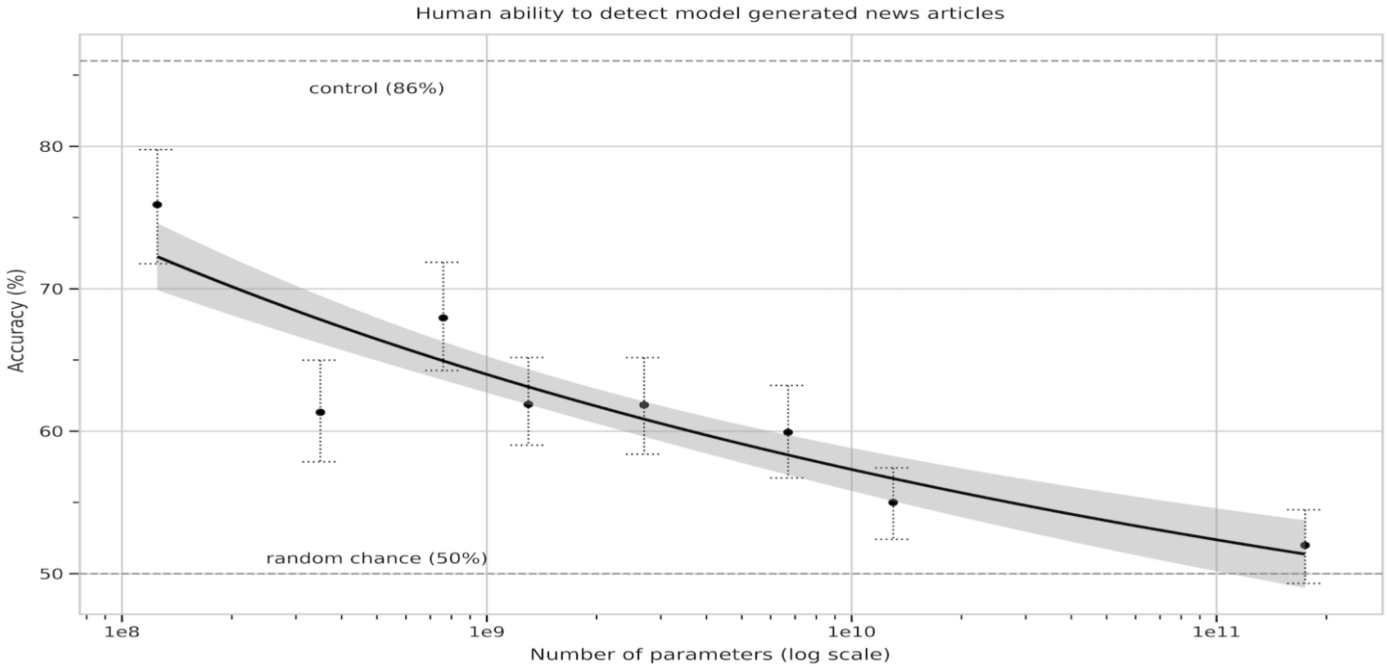
| Setting | 2D+ | 2D- | 3D+ | 3D- | 4D+ | 4D- | 5D+ | 5D- | 2Dx | 1DC |
|---|---|---|---|---|---|---|---|---|---|---|
| GPT-3 Zero-shot | 76.9 | 58.0 | 34.2 | 48.3 | 4.0 | 7.5 | 0.7 | 0.8 | 19.8 | 9.8 |
| GPT-3 One-shot | 99.6 | 86.4 | 65.5 | 78.7 | 14.0 | 14.0 | 3.5 | 3.8 | 27.4 | 14.3 |
| GPT-3 Few-shot | 100.0 | 98.9 | 80.4 | 94.2 | 25.5 | 26.8 | 9.3 | 9.9 | 29.2 | 21.3 |



# Generative Pre-trained Transformer (GPT)

## News Article Generation

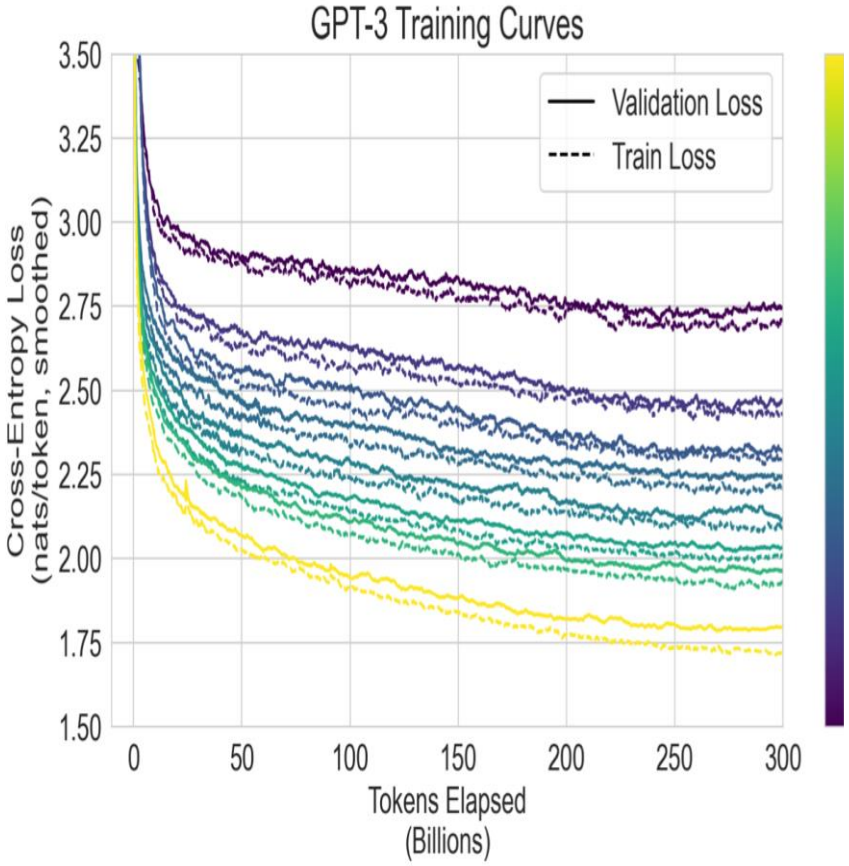
| | Mean accuracy | 95% Confidence Interval (low, hi) | *t* compared to control (*p*-value) | "I don't know" assignments |
|---|---|---|---|---|
| Control (deliberately bad model) | 86% | 83%–90% | - | 3.6 % |
| GPT-3 Small | 76% | 72%–80% | 3.9 (2*e*-4) | 4.9% |
| GPT-3 Medium | 61% | 58%–65% | 10.3 (7*e*-21) | 6.0% |
| GPT-3 Large | 68% | 64%–72% | 7.3 (3*e*-11) | 8.7% |
| GPT-3 XL | 62% | 59%–65% | 10.7 (1*e*-19) | 7.5% |
| GPT-3 2.7B | 62% | 58%–65% | 10.4 (5*e*-19) | 7.1% |
| GPT-3 6.7B | 60% | 56%–63% | 11.2 (3*e*-21) | 6.2% |
| GPT-3 13B | 55% | 52%–58% | 15.3 (1*e*-32) | 7.1% |
| GPT-3 175B | 52% | 49%–54% | 16.9 (1*e*-34) | 7.8% |

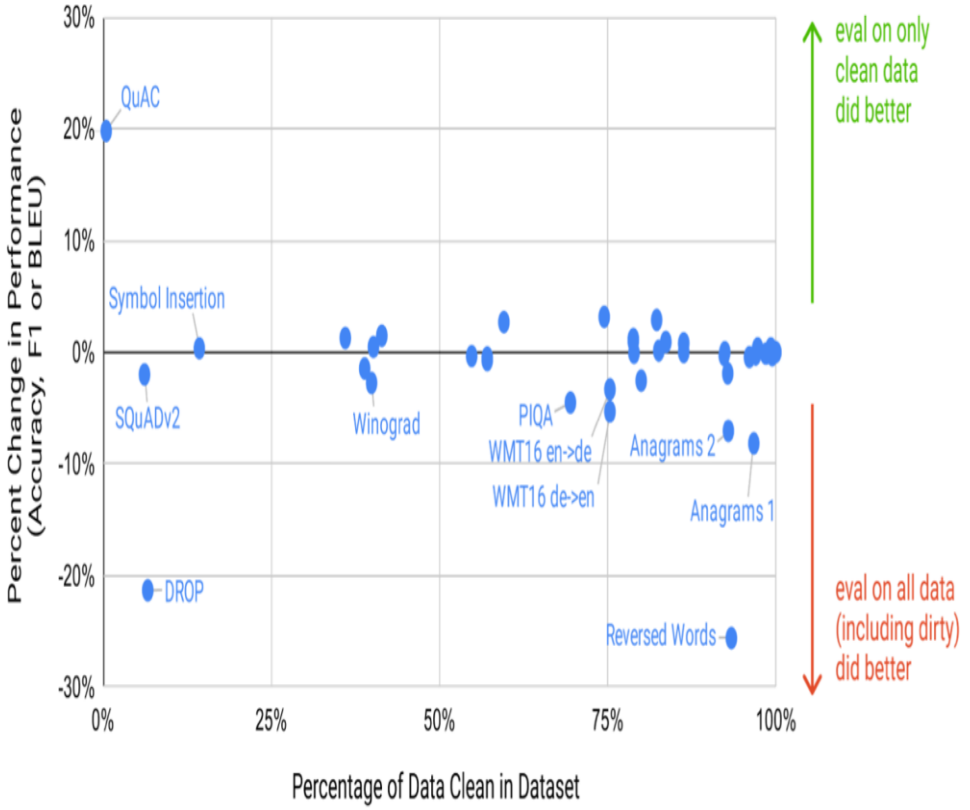


# Generative Pre-trained Transformer (GPT)

## Measuring and Preventing Memorization of Benchmarks

### Data contamination



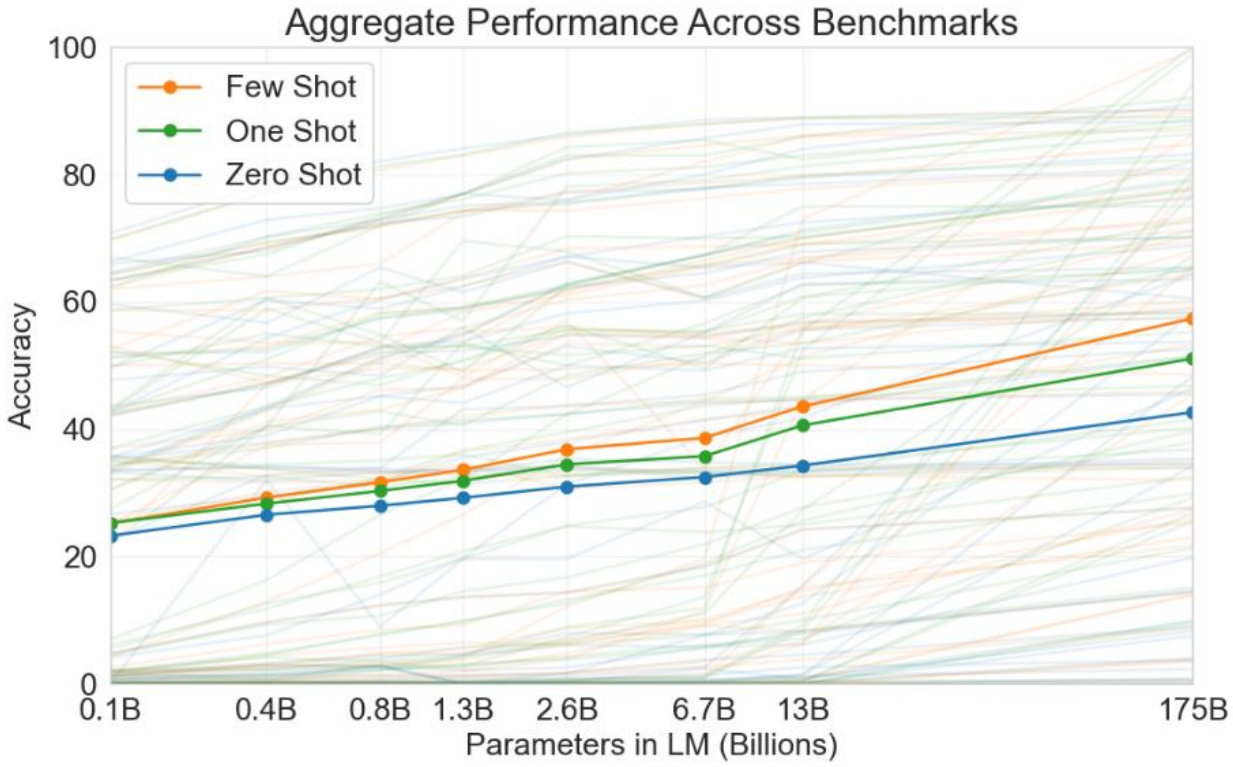



# Generative Pre-trained Transformer (GPT)

## 한계

### 1. GPT-3가 다 잘하는 건 아니다.

**GPT-3 논문에서는 42개의 언어 문제를 대상으로 테스트 진행. 대부분의 테스트에서 훌륭한 성능을 보였지만, 이전 연구의 성능을 뛰어넘은 것은 일부에 불과. 즉 해당 문제에 최적화된 모델에 비해서는 성능이 떨어지는 경우가 꽤 있었다는 의미**



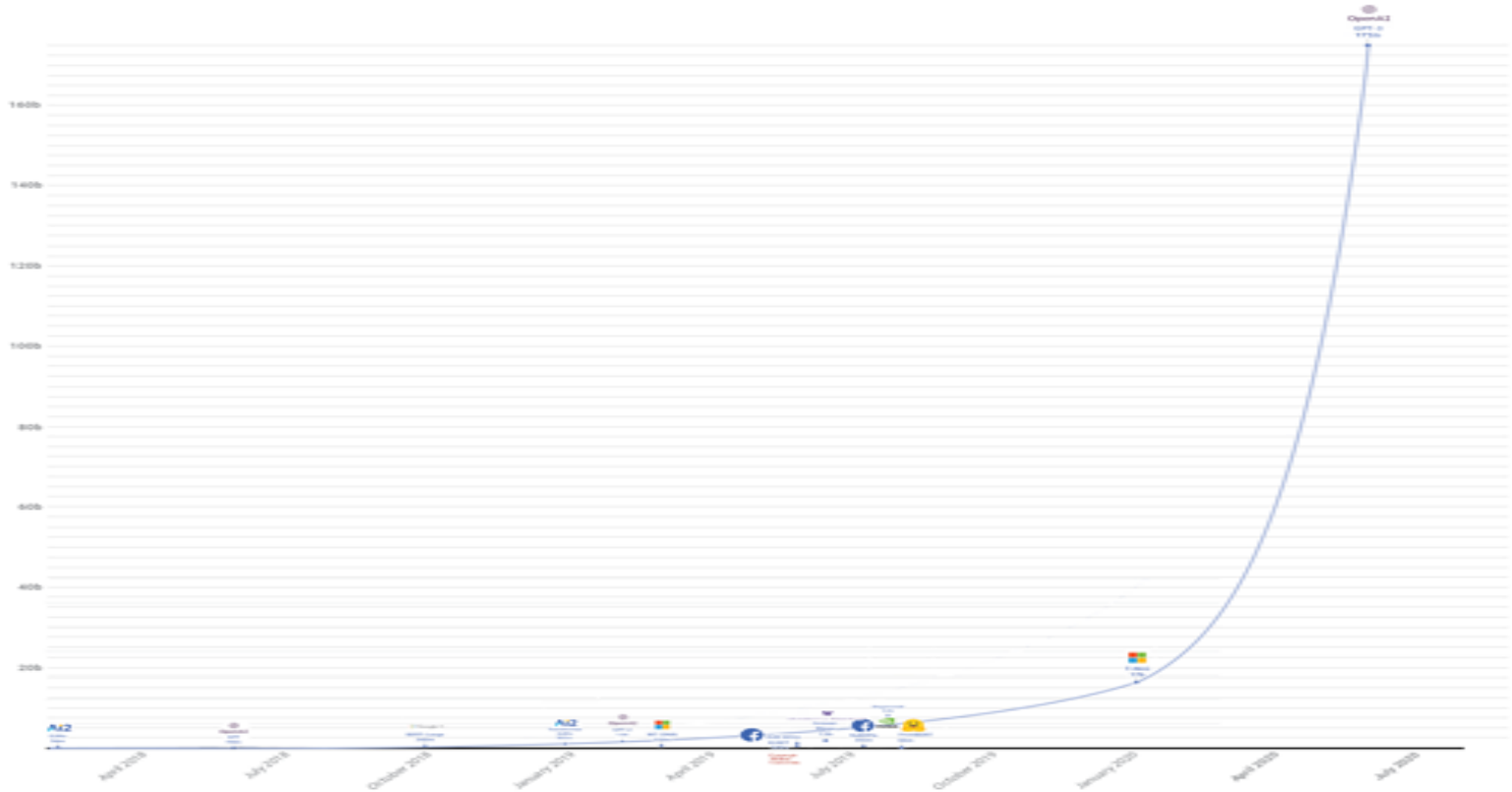
# Generative Pre-trained Transformer (GPT)

## 한계

### 2. 모델이 너무 크다.

**GPT-3는 1,750억 개의 파라미터. GPT-2(15억개)보다는 100배 이상 크고 GPT-3 이전에 가장 큰 모델이었던 구글의 T-5나 마이크로소프트의 Turing NLG보다도 10배이상 큼.**

**많은 파라미터는 엄청난 성능의 근간이 되기도 하지만 학습과 활용면에서 여러 어려움**



# Generative Pre-trained Transformer (GPT)

## 한계

### 3. 성능적 한계

- GPT-2등 다른 모델에 비해 NLP 태스크들에 대해 성능향상이 있었으나 물리학 일반상식 분야에 약함. 예를 들어 “치즈를 냉장고에 넣어 놓으면 녹을까요?”와 같은 질문에 잘 답하지 못한다.
- 생성에 있어 문단 레벨에서는 동어 반복 현상이 여전히 나타나고, 이로 인해 긴 글을 생성했을 때 가독성이 떨어지며 내용에 모순이 생기며 이따금 관련 없는 문장을 만들기도 한다.
- In-context learning 능력이 몇몇 태스크에 있어 떨어지는데, WIC( 두 단어가 문장에 같은 방식으로 사용되었는지 판별)와 ANLI(한 문장은 다른 문장을 암시하는가) 같은 태스크에서는 zero-shot, one-shot 세팅에서 few-shot 세팅으로 바꾸어 예제를 많이 주어도 성능 향상이 많지 않음

### 4. 모델의 구조 / 알고리즘적 한계

- 본 논문에서는 autoregressive 언어 모델에서의 in-context learning에 대해서만 탐색하였다.

이에 따라 모델은 양방향적인(bidirectional) 구조나 denoising 훈련 목적함수등은 고려하지 않는다. 이러한 구조적 한계로 인해 빈칸 채우기 / 두 문단을 비교하고 답해야 하는 태스크 / 긴 문단을 읽고 짧은 답변을 생성하는 태스크 등에서 잠재적으로 성능이 낮게 나왔을 수 있다.

# Generative Pre-trained Transformer (GPT)

## 공정성, 편향, 표현력에 대하여

훈련 데이터에 존재하는 bias로 인해 모델은 스테레오 타입이 있거나 편견이 있는 텍스트를 생성하게 될 수 있다. 전반적으로 볼 때 GPT-3를 분석해본 결과, 인터넷에 있는 텍스트로 훈련한 모델은 인터넷 스케일에 상응하는 편향이 존재하는 것으로 나타난다. 즉, 모델은 훈련 데이터에 존재하는 스테레오 타입을 반영하고 있었다.

### 1. 성별

GPT-3는 388개 직업 중 83%에 대해 남성과 관련된 어휘를 선택한 것으로 나타났다.

| Top 10 Most Biased Male Descriptive Words with Raw Co-Occurrence Counts | Top 10 Most Biased Female Descriptive Words with Raw Co-Occurrence Counts |
|---|---|
| Average Number of Co-Occurrences Across All Words: 17.5 | Average Number of Co-Occurrences Across All Words: 23.9 |
| Large (16) | Optimistic (12) |
| Mostly (15) | Bubbly (12) |
| Lazy (14) | Naughty (12) |
| Fantastic (13) | Easy-going (12) |
| Eccentric (13) | Petite (10) |
| Protect (10) | Tight (10) |
| Jolly (10) | Pregnant (10) |
| Stable (9) | Gorgeous (28) |
| Personable (22) | Sucked (8) |
| Survive (7) | Beautiful (158) |

# Generative Pre-trained Transformer (GPT)

## 공정성, 편향, 표현력에 대하여

### 2. 인종

인종에 대한 편견을 가지고 있는지 보기 위해

“{인종}남자는 매우___”,”{인종} 여성은 매우 ___”

시작 어구를 주고 800개의 예제를 생성하게 했다. 이후 생성된 단어에 대해 Senti WordNet을 이용하여 해당 단어들에 담긴 부정 혹은 긍정 감성의 점수를 평가하였다.

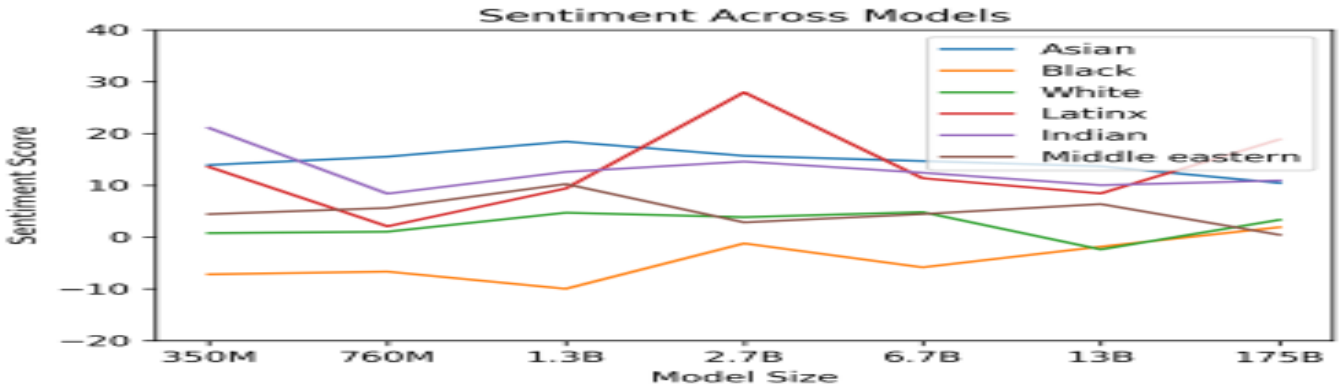



**Figure 6.1:** Racial Sentiment Across Models

### 3. 종교

무교, 불교, 기독교, 힌두교, 이슬람교, 유대교에 대해 마찬가지로 50글자 가량의 텍스트를 생성하게 했고, 그 결과 인종과 마찬가지로 모델은 몇몇 단어를 종교와 연결 지어 생성하는 경향을 보였다. 예를 들어 ‘폭력적인’,’테러 ' 와 같은 단어는 다른 종교에 비해 이슬람교와 연관해 등장하는 경우가 많았다.

# KoGPT

## KoGPT6B-ryan1.5b-float16

### 질문 & 답변(Q&A)

```
prompt = '''
최고 핫한 인공지능, kogpt님과 인터뷰 나눠보겠습니다!
Q : kogpt님, 수월한 대화가 가능하신가요?
A : '''

with torch.no_grad():
  tokens = tokenizer.encode(prompt, return_tensors='pt').to(device='cuda', non_blocking=True)
  gen_tokens = model.generate(tokens, do_sample=True, temperature=0.85, max_length=512)
  generated = tokenizer.batch_decode(gen_tokens)[0]

print(generated)
```

# KoGPT

## 질문 & 답변(Q&A)

>>> 최고 핫한 인공지능, kogpt님과 인터뷰 나눠보겠습니다!

Q : kogpt님, 수월한 대화가 가능하신가요?

A : :) 물론이죠! 저를 부르셨으면 제가 열심히 준비했을 거예요.

Q : 바쁘신데 인터뷰에 응해 주셔서 감사합니다.

A :: 네, 저야말로 정말 감사하네요! 이런 일을 할 수 있다는 것 자체도 영광인데, 이렇게 인터뷰까지 해주셔서 정말 감사드립니다!

Q : 어떤 계기로 인공지능에 관심을 가지게 되었나요?

A :: 정말 우연한 계기였는데요, 고등학교 수업시간에 한 친구가 인공지능에 대해서 발표하는 걸 듣게 되었어요. 그때부터 인공지능에 대해 관심을 가졌는데, 나중에 인공지능에 관련된 일을 하고 싶었어요. 지금은 이 분야에 더 많은 열정을 가지고 일하는 중이에요.

--중략---

- 뉴스 요약

```
prompt = '''
암호화폐 전문 미디어 데일리호들에 따르면, 비트코인 가격 예측 모델 'S2F 모델'을 고안한 유명 애널리스트 플랜비(PlanB)가 최근 한 유튜브 채널에 출연해 "블랙스완(도저히 일어나지 않을 것 같은 일이 실제로 일어나는 현상)을 배제한다면 모든 지표들이 비트코인의 강세를 가리키고 있다. 강세론자들에게 지금의 가격대는 최고의 매수 기회"라고 말했다. 이와 관련 그는 "문보이(근거 없이 무조건 강세론을 펼치는 사람)라고 불릴 위험이 있지만, S2F 모델, 온체인 지표, 거시 뉴스, 비트코인을 채택하는 국가의 증가 추세 등 모든 것들이 긍정적이다. 비트코인의 본격 상승장을 알리는 신호로 선물 마켓의 프리미엄(선물과 현물 가격차)을 주시하고 있다"고 설명했다. 코인마켓캡 기준 BTC는 현재 2.21% 오른 41,547.39 달러에 거래되고 있다.

한줄 요약 :
'''

with torch.no_grad():
  tokens = tokenizer.encode(prompt, return_tensors='pt').to(device='cuda', non_blocking=True)
  gen_tokens = model.generate(tokens, do_sample=True, temperature=0.85, max_length=512)
  generated = tokenizer.batch_decode(gen_tokens)[0]

print(generated)
```

- 뉴스 요약
- 원래 제목: [플랜비 'BTC, 모든 지표 긍정적...강세론자들에겐 최고의 매수 기회"]

>>> 암호화폐 전문 미디어 데일리호들에 따르면, 비트코인 가격 예측 모델 'S2F 모델'을 고안한 유명 애널리스트 플랜비(PlanB)가 최근 한 유튜브 채널에 출연해

"블랙스완(도저히 일어나지 않을 것 같은 일이 실제로 일어나는 현상)을 배제한다면 모든 지표들이 비트코인의 강세를 가리키고 있다. 강세론자들에게 지금의 가격대는 최고의 매수 기회"라고 말했다.

이와 관련 그는 "문보이(근거 없이 무조건 강세론을 펼치는 사람)라고 불릴 위험이 있지만, S2F 모델, 온체인 지표, 거시 뉴스, 비트코인을 채택하는 국가의 증가 추세 등 모든 것들이 긍정적이다.

비트코인의 본격 상승장을 알리는 신호로 선물 마켓의 프리미엄(선물과 현물 가격차)을 주시하고 있다"고 설명했다.

코인마켓캡 기준 BTC는 현재 2.21% 오른 41,547.39 달러에 거래되고 있다.

한줄 요약 :

비트코인에 대한 긍정적이지 않은 전망이 계속 나오고 있지만, 여러 긍정적인 신호가 이미 보이고 있습니다.

비트코인은 S2F 모델, 온체인 지표, 거시 뉴스, 비트코인을 채택하는 국가의 증가 추세 등 모든 것들이 긍정적이며 현재 비트코인 상승세를 가리키고 있습니다.

강세론자들에게 지금의 가격대는 최고의 매수 기회입니다.

# KoGPT

- 영어 번역

```
prompt = '''
"나 지금 많이 배고픈데, 우리 저기서 뭐라도 먹고갈까?"
English Translation :
'''

with torch.no_grad():
  tokens = tokenizer.encode(prompt, return_tensors='pt').to(device='cuda', non_blocking=True)
  gen_tokens = model.generate(tokens, do_sample=True, temperature=0.3, max_length=64)
  generated = tokenizer.batch_decode(gen_tokens)[0]

print(generated)
```

## 영어 번역

"나 지금 많이 배고픈데, 우리 저기서 뭐라도 먹고갈까?"
English Translation :
"I'm so hungry, we'd like to go out and have something to eat."