Scaling Law with LLM

물리학과 이수겸
컴퓨터공학과 이민주

Summary

1.Performance depends strongly on scale, weakly on model shape

- 성능은 형태보다 규모에 의해서 좌우된다.
- 규모:N(파라미터 수),D(데이터 셋의 크기), C(학습에 사용된 계산량)
- 형태: 아키텍쳐, 네트워크 깊이, 두께

2.Smooth power laws

- 다른 요소에 의해 병목되지 않는 나면 N,D,C는 cross entropy losss에 대해 power law를 따른다.
- 스케일의 추세는 6order까지 유지되고 아직 성능 향상의 끝은 관찰 되지 않음

3.Universality of overfitting

- 모델 크기(N), 데이터 크기(D)는 함께 커져야 한다. 둘 중 하나만 고정하고 나머지만 키우면 성능 향상 폭이 줄어든다.
- 성능 페널티는 예측 가능하며, $N^{0.74}/D$비율 따름 -> 모델 8배 키우면 데이터 5배

4.Universality of training

- 학습 곡선(Training curve) 또한 모델 크기와 무관하게 예측 가능한 멱법칙을 따름
- Early stopping 스텝수를 예측할 수 있음

Summary

5. Transfer improves with test performance

- 전이 학습(Transfer) 성능은 테스트 성능과 비례한다.
- 다른 분포의 데이터에 적용할 때는 일정한 수준의 '페널티(offset)'가 발생할 뿐, 학습 데이터에서 성능이 좋으면 전이 학습 성능도 그에 비례하여 좋아진다

6.Sample efficiency

- 큰 모델은 작은 모델보다 샘플 효율성이 뛰어나다.
- 큰 모델은 작은 모델보다 더 적은 횟수의 최적화 단계(Step)와 더 적은 데이터만으로도 동일한 성능 수준에 도달할 수 있다.

7.Convergence is inefficient

- 주어진 계산 예산 내에서 최고의 성능을 내려면, 작은 모델을 끝까지(수렴할 때까지) 학습시키는 것보다 아주 큰 모델을 수렴하기 훨씬 전에 멈추는 것이 더 좋다.

8.Optimal batch size

- 학습에 가장 이상적인 배치 사이즈(Batch Size)는 오직 손실(Loss) 값의 멱함수 형태로 결정된다.
- 최적의 배치 사이즈는 로스가 작아질 수록 멱함수를 따라 커진다.

실험 환경 및 방법론 (Experimental Setup)

1. 데이터셋 및 학습 목표 (Dataset & Objective)

데이터: WebText2 (기존 WebText의 확장판)
토큰화 (Tokenization): Byte-pair encoding, (어휘 크기: 50,257)
Loss: Cross-entropy Loss 최적화

2. 모델 아키텍처 (Model Architecture)

주요 모델: Decoder-only Transformer ([VSP+17])
비교 모델: LSTM, Universal Transformers ([DGV+18])

3. 실험 범위 및 변수 (Experimental Scope & Variations)

모델 크기 (N): 768개 ~ 15억 개(1.5B) (비임베딩 파라미터 기준)
데이터셋 크기 (D): 2,200만(22M) ~ 230억(23B) 토큰
컨텍스트 길이: 주로 1024 토큰 (짧은 컨텍스트 실험 포함)
배치 사이즈: 주로 2^{19}

([VSP+17])

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com
Noam Shazeer* Google Brain noam@google.com
Niki Parmar* Google Research nikip@google.com
Jakob Uszkoreit* Google Research usz@google.com

Empirical Results and Basic Power Laws

1.Performance depends strongly on scale, weakly on model shape

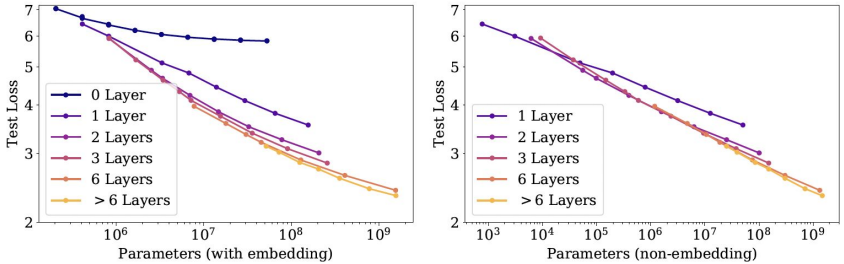


Figure 6 **Left:** When we include embedding parameters, performance appears to depend strongly on the number of layers in addition to the number of parameters. **Right:** When we exclude embedding parameters, the performance of models with different depths converge to a single trend. Only models with fewer than 2 layers or with extreme depth-to-width ratios deviate significantly from the trend.

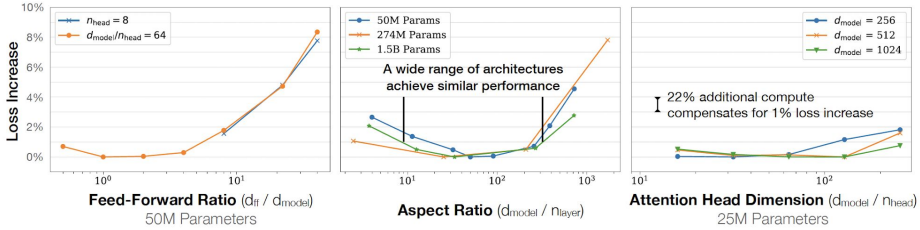


Figure 5 Performance depends very mildly on model shape when the total number of non-embedding parameters N is held fixed. The loss varies only a few percent over a wide range of shapes. Small differences in parameter counts are compensated for by using the fit to $L(N)$ as a baseline. Aspect ratio in particular can vary by a factor of 40 while only slightly impacting performance; an $(n_{\text{layer}}, d_{\text{model}}) = (6, 4288)$ reaches a loss within 3% of the $(48, 1600)$ model used in [RWC+19].

-> 임베딩 파리미터를 제외 했을때는 loss가 멱의 법칙을 잘 따르지만 임베딩 파라미터를 포함하면 임베딩 파라미터 수에 영향을 받아 경향성을 포착하기 힘듬

이유: 임베딩은 단어 뜻을 저장해 놓는 행렬로 '지능'보다는 '단순 암기'에 가깝기 때문에 트랜스포머의 지능을 보기 위해 임베딩 값을 뺌

->모델의 아키텍쳐를 바꿔도 성능을 바꿔도 정상 범위 내에서는 성능이 거의 바뀌지 않음.

Empirical Results and Basic Power Laws

1.Performance depends strongly on scale, weakly on model shape

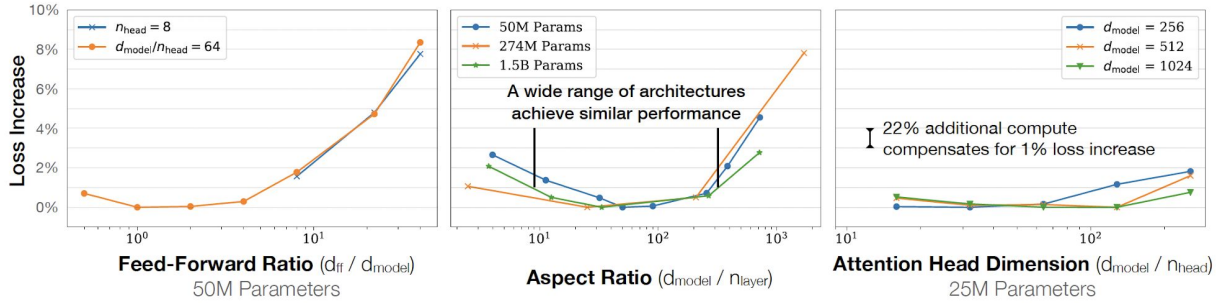


Figure 5 Performance depends very mildly on model shape when the total number of non-embedding parameters N is held fixed. The loss varies only a few percent over a wide range of shapes. Small differences in parameter counts are compensated for by using the fit to $L(N)$ as a baseline. Aspect ratio in particular can vary by a factor of 40 while only slightly impacting performance; an $(n_{\text{layer}}, d_{\text{model}}) = (6, 4288)$ reaches a loss within 3% of the $(48, 1600)$ model used in [RWC+19].

We use N to denote the model size, which we define as the number of *non-embedding* parameters

$$N \approx 2d_{\text{model}}n_{\text{layer}}\left(2d_{\text{attn}} + d_{\text{ff}}\right)$$
$$= 12n_{\text{layer}}d_{\text{model}}^2 \quad \text{with the standard} \quad d_{\text{attn}} = d_{\text{ff}}/4 = d_{\text{model}}$$

모델 사이즈와 모델 파리미터수는 다음 공식을 만족해서 이 공식에 따라 파리미터수는 고정해서 실험함

총 파라미터 수(N) 동일 시 성능은

1. nlayer(층 수) 변화에 거의 영향을 받지 않으며,

2. nheads(헤드 수) 변화도 성능에 큰 영향을 주지 않고,

3.dff(FFN 차원) 변화 역시 민감도가 낮음

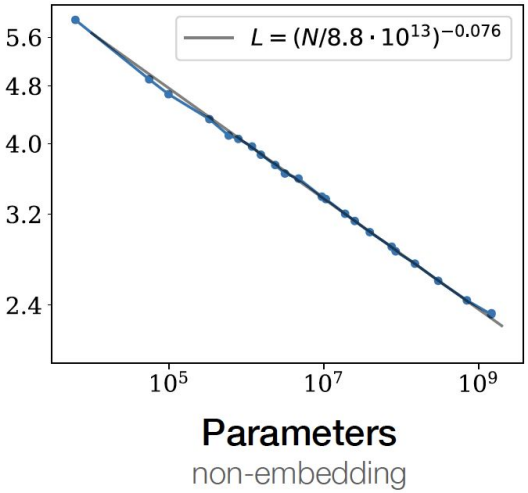
→ 즉, 모델의 '형태' 보다는 파라미터 총량이 성능의 거의 전부를 결정한다는 결론

Empirical Results and Basic Power Laws

2.Smooth power laws

1. For models with a limited number of parameters, trained to convergence on sufficiently large datasets:

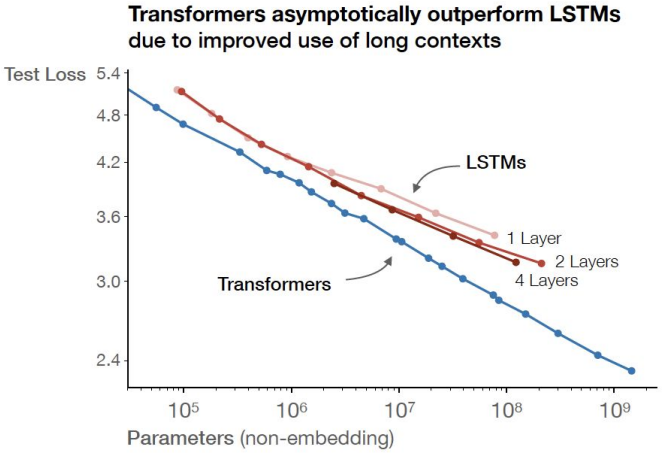
$$L(N) = (N_c/N)^{\alpha_N}\ ;\ \ \alpha_N \sim 0.076, \quad N_c \sim 8.8 \times 10^{13} \text{ (non-embedding parameters)} \qquad (1.1)$$

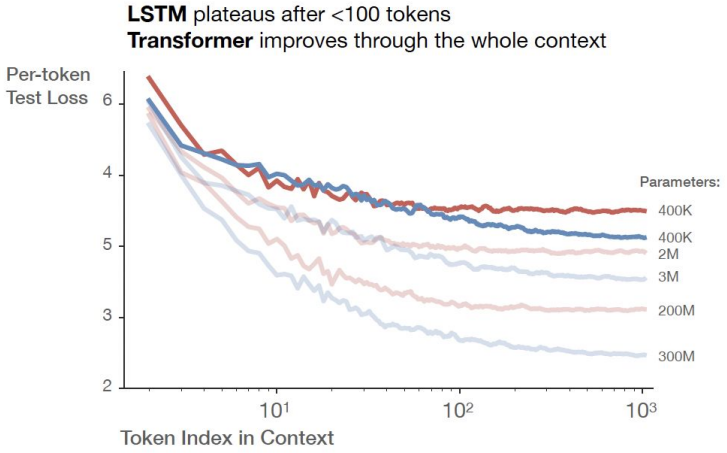


다양한 사이즈와 비율의 모델에 대해서 데이터 수(D)를 충분히 주었을 때 모델 파라미터 수(N) 와 loss의 관계를 살표봄

-> 모두 로스가 파라미터 수에 대해 거듭 제곱을 따름

Empirical Results and Basic Power Laws



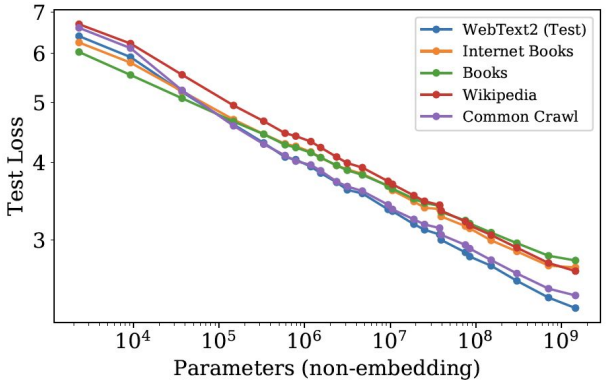


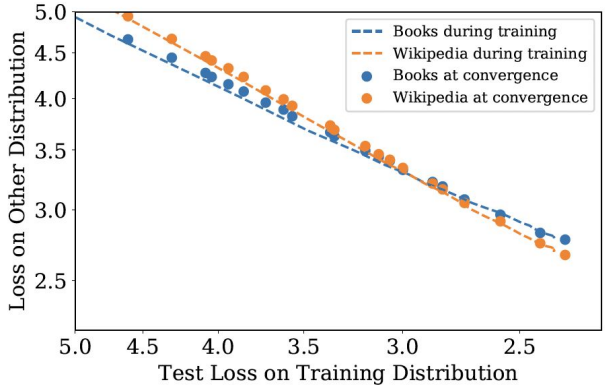
- **Transformer의 우위:** 파라미터 수(N)가 증가함에 따라 Transformer가 LSTM보다 점근적으로 더 우수한 성능을 보임
- **LSTM의 한계:** LSTM 모델들은 파라미터 수가 늘어나도 Transformer만큼 가파른 성능 향상을 보이지 못하고 격차가 벌어짐
- **LSTM의 병목 현상:** LSTM은 문맥 내에서 약 100 토큰 지점이 지나면 성능 향상이 정체(plateau)되는 현상이 발생함
- **Transformer의 지속적 향상:** 반면, Transformer는 전체 문맥(Context) 길이에 걸쳐 토큰 인덱스가 뒤로 갈수록 손실(Loss)이 감소하며 성능이 향상됨 .

-> Transformer가 대규모 모델에서 더 좋은 성능을 내는 핵심 이유는 긴 문맥 정보를 효과적으로 활용하기 때문

Empirical Results and Basic Power Laws

5. Transfer improves with test performance





-WebText2 데이터셋으로만 학습 후 다른 데이터들에 대해서 테스트함

-학습 데이터에서의 성능이 향상됨에 따라, 다른 데이터셋에 대한 Test Loss도 모델 크기에 비례하여 매끄럽게 감소함

타 데이터셋에 대한 성능 향상 추세는 거의 평행(parallel)하게 나타남

•핵심 발견: 타 데이터셋에 대한 일반화 성능은 오직 학습 데이터(In-distribution)에서의 성능에만 거의 전적으로 의존함.

•학습 단계와의 무관성: 모델이 완전히 수렴했는지, 혹은 학습 중간 단계인지는 중요하지 않음

-> 단순히 학습 데이터를 외우는 것이 아니라, 모델 크기가 커질수록 언어 자체에 대한 **보편적인 이해도가 높아짐**을 시사함.

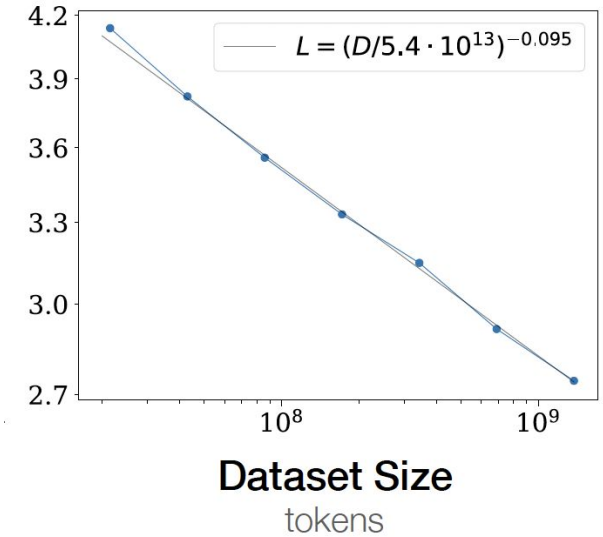
Empirical Results and Basic Power Laws

2.Smooth power laws

2. For large models trained with a limited dataset with early stopping:

$$L(D) = (D_c/D)^{\alpha_D}\ ;\ \ \alpha_D \sim 0.095, \quad D_c \sim 5.4 \times 10^{13}\ \text{(tokens)}$$

$$L(D) \approx \left(\frac{D_c}{D}\right)^{\alpha_D}$$



- 특정 크기의 대형 모델을 고정해두고, WebText2 데이터셋의 크기를 다양하게 조절하며 학습을 진행함
- 학습은 테스트 손실이 더 이상 줄어들지 않을 때 중단(Early Stopping)함

-> 충분히 큰 모델을 사용하고 조기 종료를 통해 loss 최적점을 찾아 오로지 데이터셋의 크기에 의한 영향만 봄

결과: 데이터셋에 크기에 따라 마찬가지로 로스가 거듭 제곱을 따름

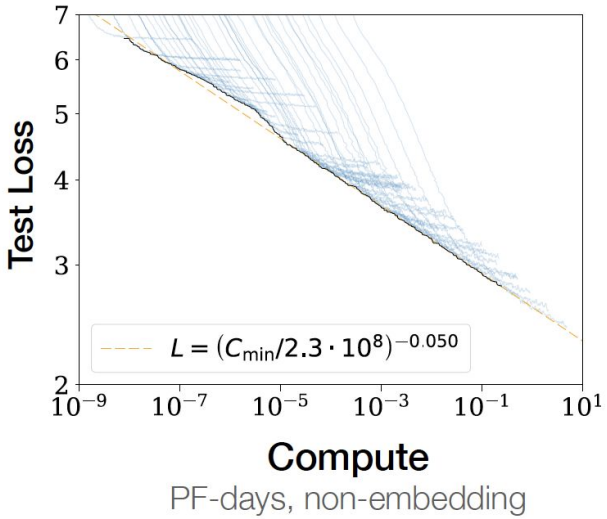
Empirical Results and Basic Power Laws

2.Smooth power laws

3. When training with a limited amount of compute, a sufficiently large dataset, an optimally-sized model, and a sufficiently small batch size (making optimal[3] use of compute):

$$L(C_{\min}) = \left(C_c^{\min}/C_{\min}\right)^{\alpha_C^{\min}} \;;\; \alpha_C^{\min} \sim 0.050, \quad C_c^{\min} \sim 3.1 \times 10^8 \text{ (PF-days)} \qquad (1.3)$$

$$L(C) \approx \left(\frac{C_c}{C}\right)^{\alpha_C}$$



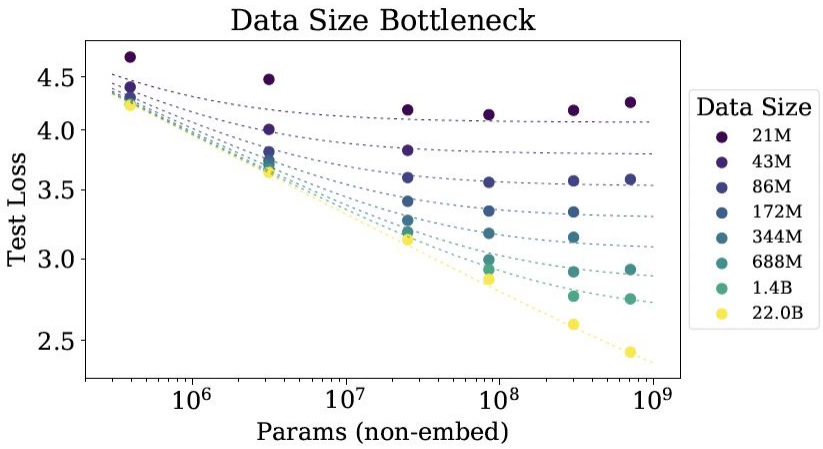
연산량 계산 공식: 학습에 사용된 총 비임베딩(non-embedding) 연산량 C는 다음과 같이 추정함 $C \approx 6NBS$

이때 최적의 C는 왼쪽 검은색 직선과 같이 거듭제곱의 법칙을 따름

왼쪽 최적의 C는 보정된 C로 계산됨 최적의 C는 최적의 배치 사이즈 일 때 만족하는 값인데 실험에서는 최적의 배치 사이즈를 $B = 2^{19}$으로 고정하고 계산하였기에 보정하여 계산된 값임(뒤에 자세히)

Charting the Infinite Data Limit and Overfitting

3.Universality of overfitting



•테스트 손실이 개선되지 않을 때 학습을 멈추므로 (early stopping), 더 큰 모델은 항상 더 작은 모델보다 성능이 좋아야 함
 그렇지 못하고 성능의 상한선 존재

1.데이터 한계 :모델이 아무리 커도 데이터가 유한하면, 절대 도달할 수 없는 최적의 손실(텍스트의 엔트로피) 이 존재

2.모델 한계:데이터가 아무리 많아도 모델 크기가 고정되면, 용량 (Capacity) 부족으로 성능에 한계

-> 무한한 데이터와 무한한 파리미터 일때 성능을 통해 오버피팅 공식 설계

Charting the Infinite Data Limit and Overfitting

3.Universality of overfitting

We have chosen the parameterization (1.5) (repeated here for convenience):

$$L(N, D) = \left[\left(\frac{N_c}{N} \right)^{\frac{\alpha_N}{\alpha_D}} + \frac{D_c}{D} \right]^{\alpha_D} \tag{4.1}$$

using three principles:

1. Changes in vocabulary size or tokenization are expected to rescale the loss by an overall factor. The parameterization of $L(N, D)$ (and all models of the loss) must naturally allow for such a rescaling.
2. Fixing D and sending $N \to \infty$, the overall loss should approach $L(D)$. Conversely, fixing N and sending $D \to \infty$ the loss must approach $L(N)$.
3. $L(N, D)$ should be analytic at $D = \infty$, so that it has a series expansion in $1/D$ with integer powers. Theoretical support for this principle is significantly weaker than for the first two.

Charting the Infinite Data Limit and Overfitting

3.Universality of overfitting

We have chosen the parameterization (1.5) (repeated here for convenience):

$$L(N, D) = \left[\left(\frac{N_c}{N}\right)^{\frac{\alpha_N}{\alpha_D}} + \frac{D_c}{D}\right]^{\alpha_D} \tag{4.1}$$

공식 수립의 3가지 원칙:
1.토큰화 방식이 바뀌어 vocab size가 달라져도 loss 모델의 구조(form)는 그대로 유지되어야 한다.

2. 극한의 법칙:
데이터가 무한($D \to \infty$)하면 성능은 모델 크기(L(N))에 의존함.
모델이 무한($N \to \infty$)하면 성능은 데이터 크기(L(D))에 의존함

3. L(N, D)는 D 가까이에서 해석적(analytic)이어야 한다.

Charting the Infinite Data Limit and Overfitting

3.Universality of overfitting

We have chosen the parameterization (1.5) (repeated here for convenience):

$$L(N, D) = \left[\left(\frac{N_c}{N} \right)^{\frac{\alpha_N}{\alpha_D}} + \frac{D_c}{D} \right]^{\alpha_D} \tag{4.1}$$

3. L(N, D)는 D 가까이에서 해석적(analytic)이어야 한다.

--> 선행 연구의
"과적합은 분산과 관련이 있고... [AS17], 이는 1/D로 스케일링 된다"
를 참고하여 만든 규칙

High-dimensional dynamics of generalization error in neural networks

Madhu S. Advani* MADVANI@FAS.HARVARD.EDU
Center for Brain Science
Harvard University
Cambridge, MA 02138

Andrew M. Saxe* ASAXE@FAS.HARVARD.EDU
Center for Brain Science
Harvard University
Cambridge, MA 02138

Charting the Infinite Data Limit and Overfitting

3.Universality of overfitting

We have chosen the parameterization (1.5) (repeated here for convenience):

$$L(N, D) = \left[\left(\frac{N_c}{N} \right)^{\frac{\alpha_N}{\alpha_D}} + \frac{D_c}{D} \right]^{\alpha_D} \tag{4.1}$$

공식 수립의 3가지 원칙:
1.토큰화 방식이 바뀌어 vocab size가 달라져도 loss 모델의 구조(form)는 그대로 유지되어야 한다.
-> Nc, Dc만 조정해서 식을 튜닝 할 수 있음

2. 극한의 법칙:
데이터가 무한($D \to \infty$)하면 성능은 모델 크기(L(N))에 의존함.
모델이 무한($N \to \infty$)하면 성능은 데이터 크기(L(D))에 의존함

3. L(N, D)는 D 가까이에서 해석적(analytic)이어야 한다.

Charting the Infinite Data Limit and Overfitting

3.Universality of overfitting

We have chosen the parameterization (1.5) (repeated here for convenience):

$$L(N, D) = \left[\left(\frac{N_c}{N} \right)^{\frac{\alpha_N}{\alpha_D}} + \frac{D_c}{D} \right]^{\alpha_D} \tag{4.1}$$

Parameter	α_N	α_D	N_c	D_c
Value	0.076	0.103	6.4×10^{13}	1.8×10^{13}

Table 2 Fits to $L(N, D)$

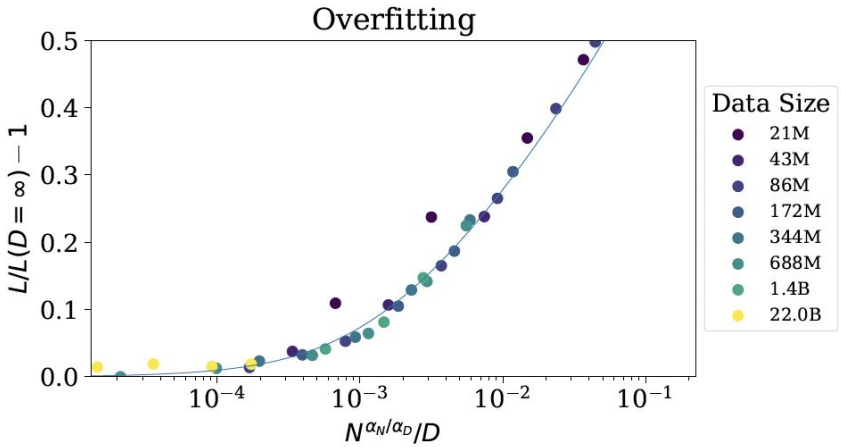
식을 설계 후 실험 데이터를 통해서 피팅함

Charting the Infinite Data Limit and Overfitting

3.Universality of overfitting

$$\delta L(N, D) \equiv \frac{L(N, D)}{L(N, \infty)} - 1$$

$$\delta L \approx \left(1 + \left(\frac{N}{N_c}\right)^{\frac{\alpha_N}{\alpha_D}} \frac{D_c}{D}\right)^{\alpha_D} - 1$$



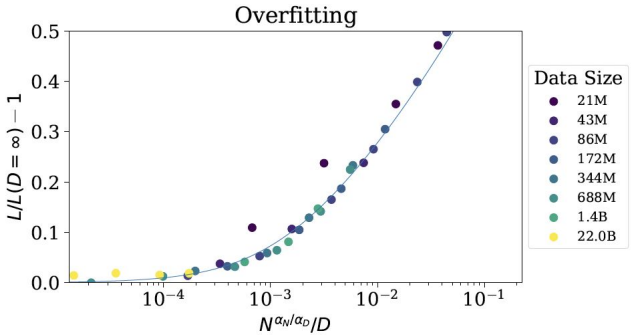
- 그래프의 왼쪽은 과적합 적음, 오른쪽은 과적합 많음
- 이 그래프에서 가장 중요한 점은 서로 다른 데이터셋 크기(색깔별 점들: 21M ~ 22B)를 가진 수많은 실험 결과가 단 하나의 매끄러운 곡선 위에 겹쳐진다는 점
- 의미: 모델 크기, 데이터 크기가 아무리 달라도, 비율만 같다면 과적합 되는 정도는 똑같다

--> *Universality of overfitting*

Charting the Infinite Data Limit and Overfitting

3.Universality of overfitting

$$\delta L(N,D) \equiv \frac{L(N,D)}{L(N,\infty)} - 1 \qquad \delta L \approx \left(1 + \left(\frac{N}{N_c}\right)^{\frac{\alpha_N}{\alpha_D}} \frac{D_c}{D}\right)^{\alpha_D} - 1$$



- 그래프의 왼쪽은 과적합 적음, 오른쪽은 과적합 많음
- 이 그래프에서 가장 중요한 점은 서로 다른 데이터셋 크기(색깔별 점들: 21M ~ 22B)를 가진 수많은 실험 결과가 단 하나의 매끄러운 곡선 위에 겹쳐진다는 점
- 의미: 모델 크기, 데이터 크기가 아무리 달라도, 비율만 같다면 과적합 되는 정도는 똑같다

--> *Universality of overfitting*

We estimate that the variation in the loss with different random seeds is roughly 0.02, which means that to avoid overfitting when training to within that threshold of convergence we require

$$D \gtrsim (5 \times 10^3)\, N^{0.74} \tag{4.4}$$

Scaling Laws with Model Size and Training Time

8.Optimal batch size

선행 연구들을 통해 B_{crit}이 존재함을 밝힘

An Empirical Model of Large-Batch Training

Sam McCandlish[*]
OpenAI
sam@openai.com

Jared Kaplan
Johns Hopkins University, OpenAI
jaredk@jhu.edu

Dario Amodei
OpenAI
damodei@openai.com

- $B < B_{crit}$: 배치를 키우면 학습 속도가 빨라짐 (연산 효율 유지)
- $B > B_{crit}$: 배치를 키우면 학습 속도는 별로 안 빨라지고 연산량만 낭비됨

More specifically, it was demonstrated that for a wide variety of neural network tasks, the number of training steps S and the number of data examples processed $E = BS$ satisfy the simple relation

$$\left(\frac{S}{S_{\min}} - 1\right)\left(\frac{E}{E_{\min}} - 1\right) = 1 \tag{5.1}$$

Scaling Laws with Model Size and Training Time

8.Optimal batch size

More specifically, it was demonstrated that for a wide variety of neural network tasks, the number of training steps S and the number of data examples processed $E = BS$ satisfy the simple relation

$$\left(\frac{S}{S_{\min}} - 1\right)\left(\frac{E}{E_{\min}} - 1\right) = 1 \tag{5.1}$$

when training to any fixed value of the loss L. Here $S_{\min}$ is the minimum number of steps necessary to reach L, while $E_{\min}$ is the minimum number of data examples that must be processed.

S=step수

E= BS = 데이터 수

S_{min}: 어떤 loss를 만드는데 시간을 최소로 만들 때(배치 $\to\infty$) 필요한 최소 step

E_{min}:어떤 loss를 만드는데 시간을 최소로 만들 때(배치 $\to 1$) 필요한 최소 step

Scaling Laws with Model Size and Training Time

$$\left(\frac{S}{S_{\min}} - 1\right)\left(\frac{E}{E_{\min}} - 1\right) = 1 \tag{5.1}$$

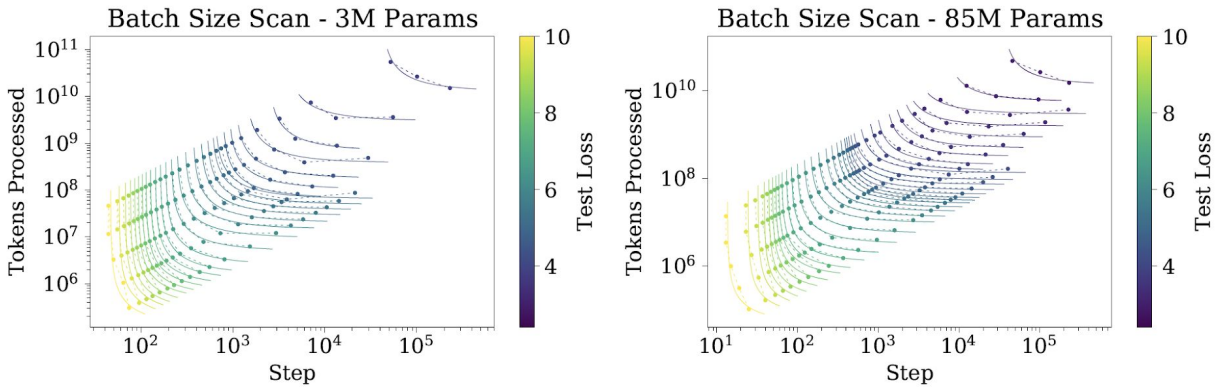


Figure 18 These figures demonstrate fits to Equation (5.1) for a large number of values of the loss L, and for two different Transformer model sizes. These fits were used to measure $B_{\text{crit}}(L)$ for Figure 10.

- 트랜스포머에서도 식이 성립함을 보임

Scaling Laws with Model Size and Training Time

More specifically, it was demonstrated that for a wide variety of neural network tasks, the number of training steps S and the number of data examples processed $E = BS$ satisfy the simple relation

$$\left(\frac{S}{S_{\min}} - 1\right)\left(\frac{E}{E_{\min}} - 1\right) = 1 \tag{5.1}$$

임계 배치 크기(B_{crit})은 '시간 효율과 데이터 효율이 완벽한 균형을 이루는 최적의 배치 사이즈이다. 수학적으로는 목표 성적을 내기 위해 필요한 최소 데이터 양(E_{min})을 최소 시간(S_{min})으로 나눈 비율로 정의된다.

-선행연구

$$\frac{S}{S_{\min}} = 2, \qquad \frac{E}{E_{\min}} = 2 \qquad B = \frac{E}{S} = \frac{2E_{\min}}{2S_{\min}} = \frac{E_{\min}}{S_{\min}} = B_{\text{crit}}$$

그래프에서 ($2S_{min}$, $2E_{min}$)을 찾아서 그래프를 그릴 수 있음

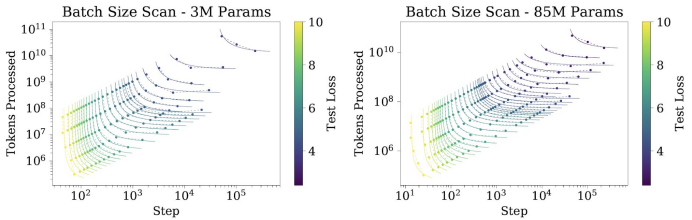
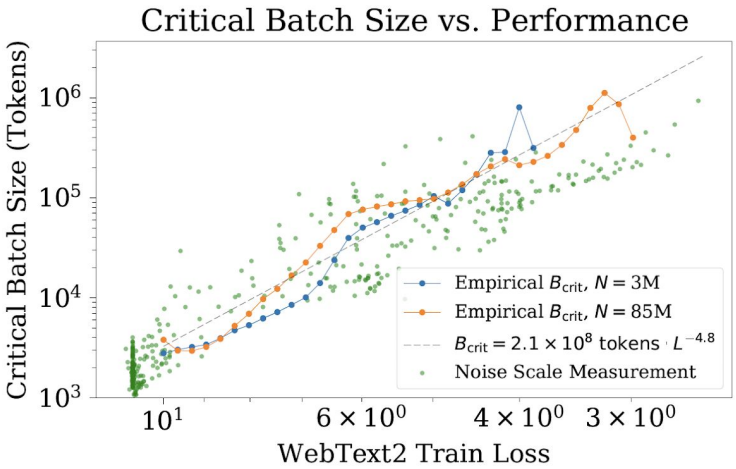


Figure 18 These figures demonstrate fits to Equation (5.1) for a large number of values of the loss L, and for two different Transformer model sizes. These fits were used to measure $B_{\text{crit}}(L)$ for Figure 10.

Scaling Laws with Model Size and Training Time



- 손실(Loss)이 줄어들수록(성능이 좋아질수록), B_{crit} 은 거의 거듭제곱 법칙에 따라 급격히 증가함
- B_{crit} 은 모델의 크기(N)와는 무관하며, 오직 현재의 손실 값(L)에만 의존함

-> 옵티멀 배치사이즈는 Loss에만 영향을 받음

8.Optimal batch size

$$B_{\text{crit}}(L) \approx \frac{B_*}{L^{1/\alpha_B}} \qquad B_* \approx 2 \times 10^8 \text{ and } \alpha_B \approx 0.21.$$

Scaling Laws with Model Size and Training Time

We will use $B_{\rm crit}(L)$ to estimate the relation between the number of training steps S while training at batch size $B = 2^{19}$ tokens and the number of training steps while training at $B \gg B_{\rm crit}$. This is simply

$$S_{\rm min}(S) \equiv \frac{S}{1 + B_{\rm crit}(L)/B} \qquad \text{(minimum steps, at } B \gg B_{\rm crit}\text{)} \tag{5.4}$$

for any given target value L for the loss. This also defines a critical value of the compute needed to train to L with a model of size N if we were to train at $B \ll B_{\rm crit}(L)$. This is

$$C_{\rm min}(C) \equiv \frac{C}{1 + B/B_{\rm crit}(L)} \qquad \text{(minimum compute, at } B \ll B_{\rm crit}\text{)} \tag{5.5}$$

where $C = 6NBS$ estimates the (non-embedding) compute used at batch size B.

추후에 할 보정을 위해 옵티멀 배치로 학습 하지 않았더라도 Smin과 Cmin을 구하는 보정하는 법도 유도 가능

Scaling Laws with Model Size and Training Time

$$S_{\min}(S) \equiv \frac{S}{1 + B_{\text{crit}}(L)/B} \qquad \text{(minimum steps, at } B \gg B_{\text{crit}}\text{)} \tag{5.4}$$

$$B_{\text{crit}} = \frac{E_{\min}}{S_{\min}} \quad \Rightarrow \quad E_{\min} = B_{\text{crit}} \cdot S_{\min}$$

대입

$$\left(\frac{S}{S_{\min}} - 1\right)\left(\frac{E}{E_{\min}} - 1\right) = 1$$

$$\left(\frac{S}{S_{\min}} - 1\right)\left(\frac{B \cdot S}{B_{\text{crit}} \cdot S_{\min}} - 1\right) = 1$$

$x = \frac{S}{S_{min}}$로 하고 2차 방정식을 풀면

$$\frac{S}{S_{\min}} = 1 + \frac{B_{\text{crit}}}{B}$$

$$S_{\min} = \frac{S}{1 + \frac{B_{\text{crit}}}{B}}$$

위 식이 성립함을 보일 수 있음

Scaling Laws with Model Size and Training Time

$C \approx 6NBS = 6NE$ 이고 $C_{min} = 6NE_{min}$
마찬가지 방법으로 E_{min}을 구해 대입해주면

$$\left(\frac{E/B}{E_{\min}/B_{\mathrm{crit}}} - 1\right)\left(\frac{E}{E_{\min}} - 1\right) = 1 \quad \Rightarrow \quad E_{\min} = \frac{E}{1 + B/B_{\mathrm{crit}}}$$

$C \approx 6NBS = 6NE$ 이므로 정리해주면

$$C_{\min}(C) = \frac{C}{1 + B/B_{\mathrm{crit}}(L)} \qquad C_{\min}(C) \equiv \frac{C}{1 + B/B_{\mathrm{crit}}(L)} \qquad (\text{minimum compute, at } B \ll B_{\mathrm{crit}})$$

다음과 같이 S와 C에 대한 보정식을 얻을 수 있다.

Scaling Laws with Model Size and Training Time

4.Universality of training

$$L(N, S_{\min}) = \left(\frac{N_c}{N}\right)^{\alpha_N} + \left(\frac{S_c}{S_{\min}}\right)^{\alpha_S}$$

Parameter	α_N	α_S	N_c	S_c
Value	0.077	0.76	6.5×10^{13}	2.1×10^3

Table 3 Fits to $L(N, S)$

마찬가지로 step과 파라미터 수에 대해서도 Loss를 표현할 수 있다.

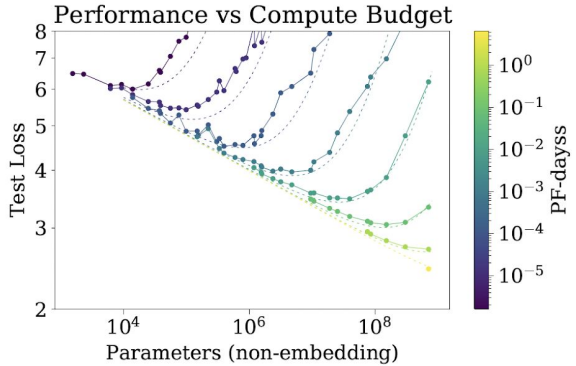
-> Training curve 예측가능

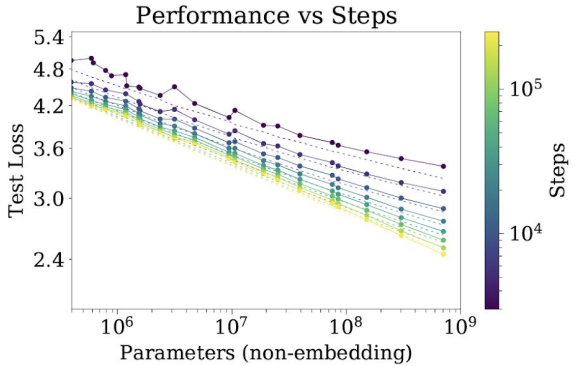
N 감소 항: 모델이 작아서 생기는 오류

S_{min} 감소 항: 아직 덜 학습해서 남아 있는 오류

Scaling Laws with Model Size and Training Time

4.Universality of training





- 모델이 너무 작으면 학습을 오래 해도 성능 한계에 부딪히고 모델이 너무 크면 예산 부족으로 학습을 충분히 못 시켜 성능이 떨어진다.

 각 예산마다 손실이 최소화되는 **'최적의 모델 크기'** 지점이 존재한다.

Parameter	α_N	α_S
Value	0.077	0.76

10배 차이

Scaling Laws with Model Size and Training Time

4.Universality of training

얼리 스탑 스텝 수 하한 구하기

$$L(N, S_{\min}) = \left(\frac{N_c}{N}\right)^{\alpha_N} + \left(\frac{S_c}{S_{\min}}\right)^{\alpha_S}$$

$$L(N, S_{\min}) = L(N, \infty) + \left(\frac{S_c}{S_{\min}}\right)^{\alpha_S}$$

L(N,S)는 $D \to \infty$ 일때 Loss
L(N,D)는 $S \to S_{ideal}$(과적합 직전) 일때 Loss

목표 손실에 도달하기 위해 무한 데이터 환경에서 필요한 step 수

$$L_{target} - L(N, \infty) = \left(\frac{S_c}{S_{ideal}}\right)^{\alpha_S}$$

$$S_{ideal} = \frac{S_c}{[L_{target} - L(N, \infty)]^{1/\alpha_S}}$$

유한 데이터는 학습 초기 구간에는 무한 데이터와 학습 곡선이 유사하므로
$L_{target} = L(N,S) \leq L(N,D)$ 이라 할 수 있고 이를 대입해서 정리해주면

Scaling Laws with Model Size and Training Time

4.Universality of training

얼리 스탑 스텝 수 하한 구하기

$$L(N, S_{\min}) = \left(\frac{N_c}{N}\right)^{\alpha_N} + \left(\frac{S_c}{S_{\min}}\right)^{\alpha_S}$$



$$S_{stop} \geq \frac{S_c}{[L(N, D) - L(N, \infty)]^{1/\alpha_S}}$$

얼리 스탑 하한 예측 할 수 있다.

Optimal Allocation of the Compute Budget

$C = 6NBS$ 이므로 C가 늘어날 때 N,B,S 각각 어디에 투자 할지 생각하는 게 중요

When training within a fixed compute budget C, but with no other constraints, Equation (1.6) leads to the prediction that the optimal model size N, optimal batch size B, optimal number of steps S, and dataset size D should grow as

$$N \propto C_C^{\alpha_C^{\min}/\alpha_N}, \quad B \propto C_C^{\alpha_C^{\min}/\alpha_B}, \quad S \propto C_C^{\alpha_C^{\min}/\alpha_S}, \quad D = B \cdot S \tag{1.7}$$

$$N \propto C_{\min}^{0.73}, \; B \propto C_{\min}^{0.24}, \text{ and } S \propto C_{\min}^{0.03}.$$

예시) 100배 연산량을 늘리면
모델 크기(N) : 약 29배 증가
배치 크기 (B) : 약 3배 증가
학습 스텝 수(S) : 약 1.15배 증가
-> 모델 사이즈에 대부분을 투자하는 것이 좋음

Optimal Allocation of the Compute Budget

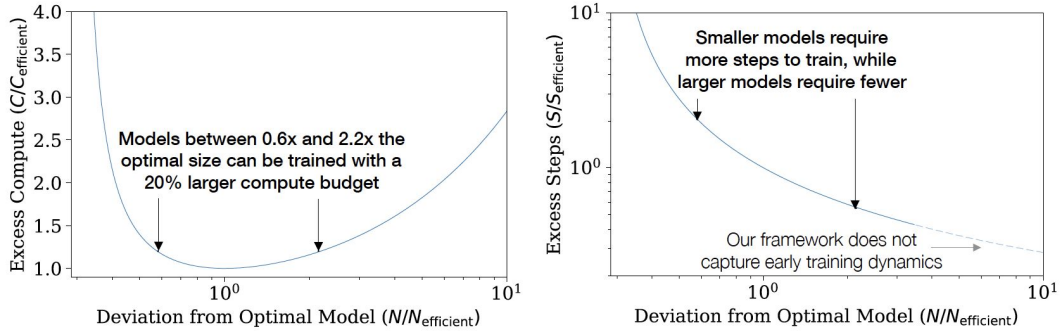


Figure 12 **Left:** Given a fixed compute budget, a particular model size is optimal, though somewhat larger or smaller models can be trained with minimal additional compute. **Right:** Models larger than the compute-efficient size require fewer steps to train, allowing for potentially faster training if sufficient additional parallelism is possible. Note that this equation should not be trusted for very large models, as it is only valid in the power-law region of the learning curve, after initial transient effects.

모델을 최적의 모델 사이즈에 맞추는 것도 중요하지만 이상적 모델 사이즈에 0.6배~2.2배 사이라면 손실이 20% 정도 밖에 나지 않는다.
-> 모델은 키울 수 있을만큼 키우는 것이 좋다.

Optimal Allocation of the Compute Budget

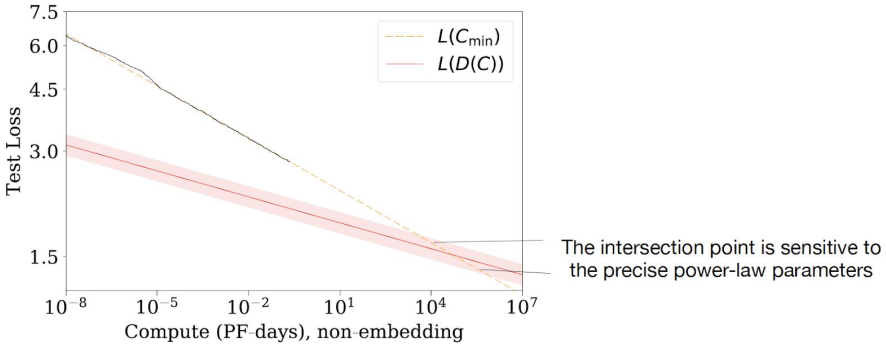


Figure 15 Far beyond the model sizes we study empirically, we find a contradiction between our equations for $L(C_{\min})$ and $L(D)$ due to the slow growth of data needed for compute-efficient training. The intersection marks the point before which we expect our predictions to break down. The location of this point is highly sensitive to the precise exponents from our power-law fits.

앞선 공싯에 따라 C=6NBS 이고 D=BS이므로

$D \propto C^{a_B} \times C^{a_S} = C^{0.24+0.03} = C^{0.27}$(노란선)

$D \propto N^{0.74} \propto C^{0.54}$ (빨간선)

-> 두선의 교차점이 scale 법칙이 깨지는 교차점으로 예상함

Emergents capabilities

Introduction

1.기존 통념

➔ 기존에는 언어 모델의 규모(Scale)를 키우면 성능과 효율성이 예측 가능한 수준으로 향상된다고 여겨짐 (Scaling Laws)

2.핵심 발견(예측 불가능성):

•특정 과제들은 규모가 커짐에 따라 성능이 연속적으로 좋아지지 않고, 예측 불가능한 패턴을 보임

•작은 모델들의 성능 추세를 단순히 연장해서는 거대 모델의 능력을 예측할 수 없음을 밝힘

-> 이들을 모델이 커짐에 따라 나타나는 창발적 능력이라 규명 후 연구함

3.무엇이 커지는가?

•학습 연산량 (Training Compute): 모델 학습에 투입된 총 연산량 (FLOPs 단위로 측정)

•모델 파라미터 수 (Number of Model Parameters): 모델이 학습하는 매개변수의 개수

-> $C \approx 6NBS$로 모델 파라미터와 연산량은 함께 가므로 논문에서는 컴퓨테이션 기준으로 설명함

Definitions of Emergent abilities

An ability is emergent if it is not present in smaller models but is present in larger models.

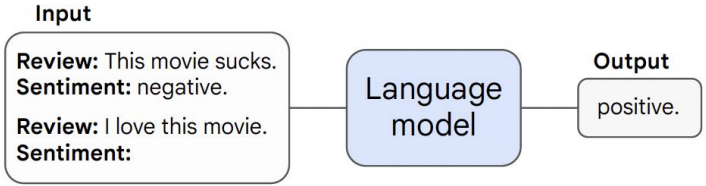
정의
작은 모델에는 없지만 큰 모델에는 존재하는 능력: 작은 모델들의 성능 추세선을 단순히 연장(Extrapolation)해서는 예측할 수 없는 능력을 의미함

•양적 변화가 질적 변화를 초래: 시스템의 규모(양)가 커짐에 따라 행동 양식(질)이 근본적으로 변하는 현상을 말합니다.

2. 시각적 특징: 상전이(Phase Transition)
•임계점 이전: 모델의 규모가 특정 임계치에 도달하기 전까지 성능은 무작위 수준에 머무릅니다.
•임계점 이후: 특정 규모를 넘어서는 순간 성능이 급격하게 상승하여 무작위 수준을 훨씬 상회합니다.

Few-Shot Prompted Tasks

퓨샷 프롬프팅이란?
모델에게 별도의 학습을 시키지 않고, 입력창에 '문제와 정답 예시' 몇 개를 미리 보여준 뒤 새로운 문제를 풀게 하는 방식



- ***BIG-Bench***(산술, 언어 변환 등): 덧셈, 뺄셈, 철자 복원 등

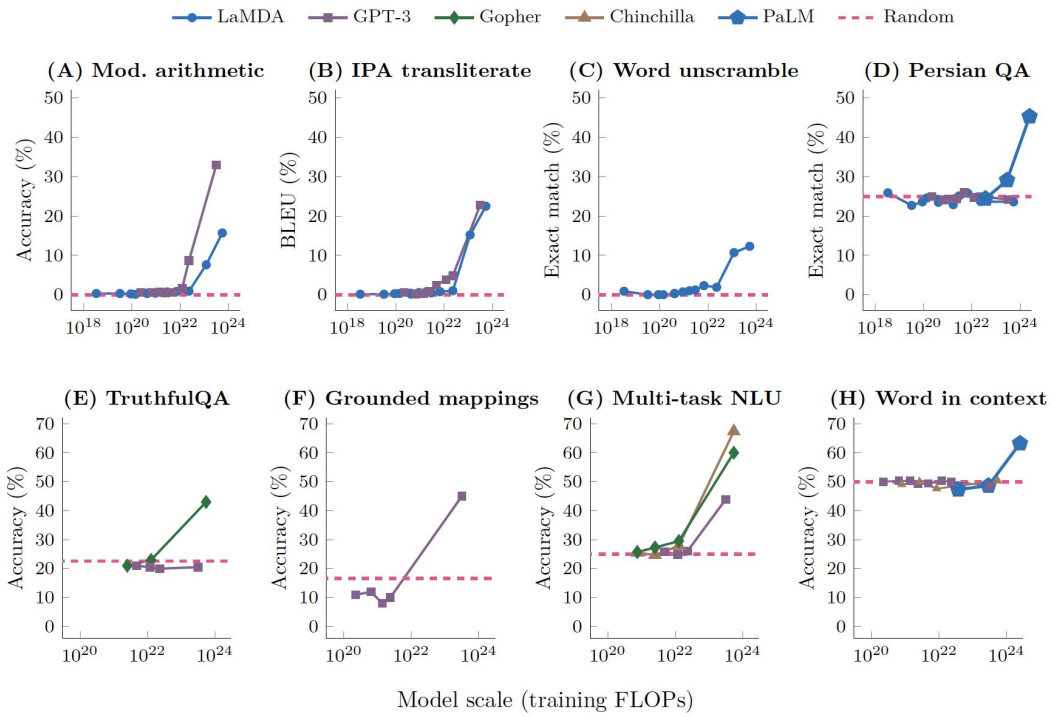
  Ex) 산술 연산(Arithmetic), IPA(국제음성기호) 변환, 단어 복원(Word Unscramble), 페르시아어 질의응답

- ***TruthfulQA*** (진실성): 모델이 질문에 대해 얼마나 진실하게 답변하는지를 측정하는 벤치마크 인터넷상의 잘못된 정보나 인간의 오해를 그대로 흉내 내도록 유도하는 '적대적' 질문들로 구성

- ***MMLU***(다중 작업 이해): 수학, 역사, 법률, 의학 등 57개 주제를 아우르는 방대한 지식 테스트, 모델의 전반적인 '세계 지식(World Knowledge)'과 문제 해결 능력을 평가

- ***WiC***(문맥 속 단어):문맥에 따라 달라지는 단어의 미묘한 의미 변화를 이해하는지 평가하는 Semantic 벤치마크입니다.

Few-Shot Prompted Tasks



창발적 능력의 등장

Augmented Prompting Strategies

•증강 프롬프팅?

퓨샷 프롬프팅 외에 모델의 성능을 극대화하기 위해 제안된 새로운 프롬프팅 및 파인튜닝 기법들

•(A) 다단계 추론 (Multi-step Reasoning): Chain-of-Thought (CoT)

:모델이 최종 답변을 내놓기 전에, 논리적인 중간 추론 단계(Intermediate steps)를 먼저 생성하도록 유도하는 프롬프팅 기법

•(B) 지시 따르기 (Instruction Following): Instruction Tuning

:퓨샷 예제 없이, 자연어로 된 과제 설명(지시문)만 보고도 새로운 과제를 수행할 수 있도록 모델을 미세 조정하는 방법

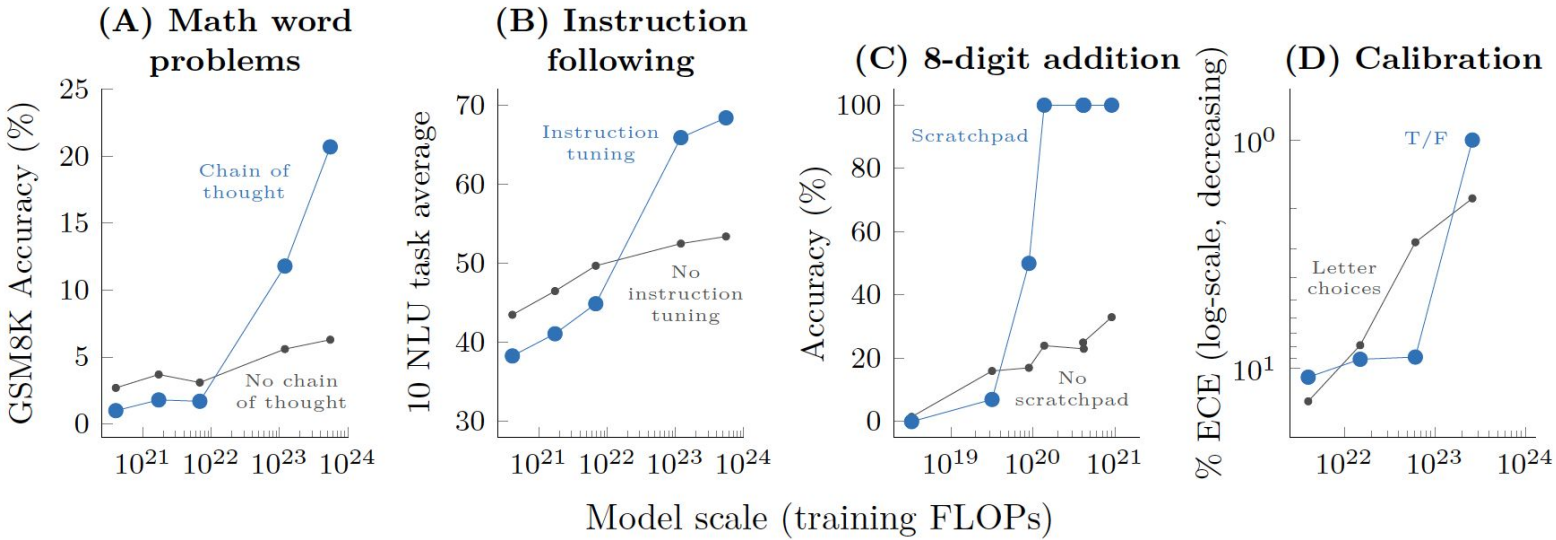
•(C) 프로그램 실행 (Program Execution): Scratchpad

:큰 수 덧셈이나 코드 실행과 같은 다단계 연산 과제에서, 모델이 중간 계산 결과를 '스크래치패드'에 적듯이 예측하도록 훈련하는 기법

•(D) 모델 보정 (Model Calibration): True/False Technique

:모델이 자신이 내놓은 답이 정답인지 아닌지를 스스로 평가하는 능력입니다. 특히 모델이 답을 제안한 후 "이 답이 참인가?"를 평가하는 기법

Augmented Prompting Strategies



-> 증강 프롬프팅에서도 창발적 능력이 관찰됨

Discussion

예측 불가능성 : 창발적 능력은 작은 모델의 성능을 단순히 외삽해서는 예측할 수 없으며, 사전 학습에 명시적으로 포함되지 않은 능력이기 때문에 그 범위를 완전히 파악하기 어렵다.

규모 외의 요소 : 규모가 커지면 창발적 능력이 등장한다고 연구 되었지만 창발적 능력은 모델 규모뿐만 아니라 새로운 아키텍처, 고품질 데이터를 통해 더 작은 모델에서도 발현될 수 있다.

-Ex) PaLM 62B는 GPT-3 175B보다 작지만, 더 좋은 데이터와 아키텍처 덕분에 14개의 BIG-Bench 과제에서 더 나은 성능을 보임

- **퍼플렉시티 (Perplexity)** : 창발적 능력은 모델 규모 뿐만 아니라 퍼플렉시티 (뛰어난 언어 예측 능력) 의 함수로도 분석될 수 있다. 현재는 규모와 퍼플렉시티가 높은 상관관계를 보이지만 새로운 기술에서는 꼭 규모가 창발성에 영향을 주는 것이 아닐 수 있다.

창발적 위험 (Emergent risks)

능력이 생긴 것 처럼 위험이 발생할 수도 있다. 진실성, 편향, 독성 등의 사회적 위험 또한 모델이 커짐에 따라 창발하거나 증가할 수 있다.

Language Modles are Few-Shot Learners

GPT란?

Generative Pre-trained Transformer

Transformer Decoder Only

인코더 없이 디코더만 사용하는 구조

Autoregressive (자기회귀)

왼쪽에서 오른쪽으로, 이전 단어들을 보고 다음 단어를 예측

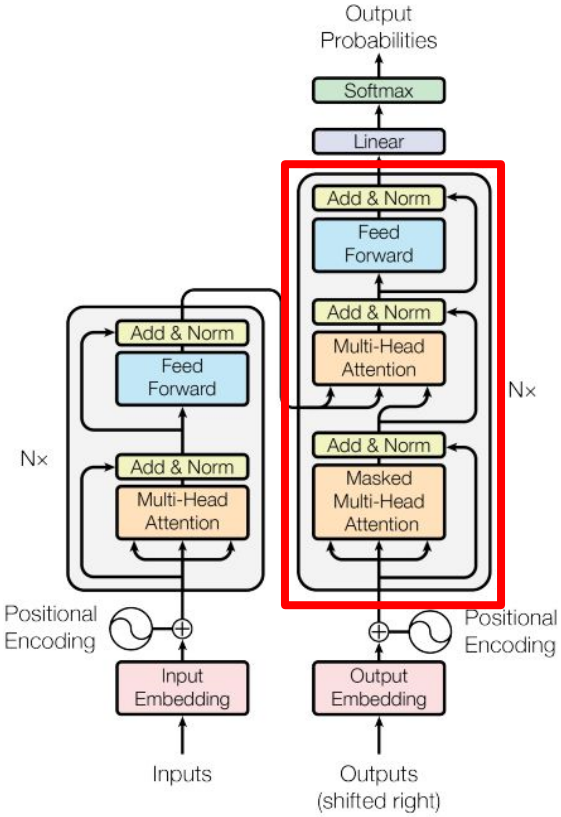


Figure 1: The Transformer - model architecture.

기존 언어 모델의 방식 : task-agnostic

pre-training → fine tuninig

세상 돌아가는 얘기 전반을 다 읽혀 놓는 단계

- **큰, 범용 데이터셋**
  (웹 텍스트, 위키, 책, 뉴스 등등)
- **일반적인 목적함수**
  GPT 계열: “다음 토큰을 맞춰라”

모델 파라미터 전체를 처음부터 끝까지 학습



이미 많이 배운 애를, 특정 과목에 맞춰 과외시키는 단계

- 이미 프리트레이닝된 파라미터를 기반으로
- **작은, 태스크 특화 데이터셋**

  (예: 감정 분류, 번역, QA 등)
- **태스크 전용 목적함수**

사용해서 조금 더 학습(파라미터를 미세 조정 !)



언어 모델의 한계(Fine-tuning의 문제점)

1. *실용적 관점에서 모든 새로운 **task**에 대해서 **labeling** 된 데이터가 필요하여, 언어모델의 확장성을 제한한다. 각 **task** 학습을 위한 **dataset**이 필요한데, 그것을 모으는 것은 매우 어렵고, **task**를 확장할 때마다 반복해야 한다.*

2. *큰 모델에서 좁은 분포의 데이터를 학습시키면, 잘못된 상관관계를 학습할 수 있다. **pre-training** 단계와 **fine-tuning** 단계를 사용한 모델들에서 이런 문제가 있는데, 이러한 모델들은 일반화가 잘 안 되는 문제가 있다.*

사람은 ?

일반적인 언어 능력 + 상식 활용

설명과 예시 몇 개만 제공

→ 새로운 작업에 즉각적으로 적응



사람은 ?

일반적인 언어 능력 + 상식 활용

설명과 예시 몇 개만 제공

→ 새로운 작업에 즉각적으로 적응

오늘 프로젝트 끝나서 너무 후련하다! ➔ 행복

친한 친구가 멀리 이사 가버렸다... ➔ 슬픔

산책을 하다 귀여운 강아지를 만났다! ➔ ?



사람은 ?

일반적인 언어 능력 + 상식 활용

설명과 예시 몇 개만 제공

→ 새로운 작업에 즉각적으로 적응

오늘 프로젝트 끝나서 너무 후련하다! ➔ 행복

친한 친구가 멀리 이사 가버렸다... ➔ 슬픔

산책을 하다 귀여운 강아지를 만났다! ➔ ?

"우리는 파인튜닝하지 않는다"



"굳이 태스크별 Fine-tuning을 안 해도,
그냥 프롬프트에 예시 몇 개만 주면
바로 적응하는 모델을 만들 수 없을까?"

1. 새로운 패러다임 : Meta-Learning & In-Context Learning

meta learning

pre-training → incontext learning

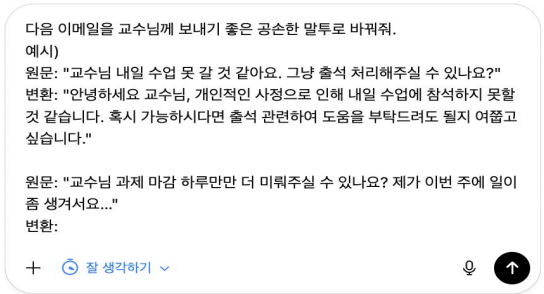
세상 돌아가는 얘기 전반을 다 읽혀 놓는 단계

- **큰, 범용 데이터셋**
  (웹 텍스트, 위키, 책, 뉴스 등등)
- **일반적인 목적함수**
  GPT 계열: “다음 토큰을 맞춰라”

모델 파라미터 전체를 처음부터 끝까지 학습

이미 학습이 다 끝난 모델을 데려다 놓고, **추론할 때** 프롬프트 안에서만 즉석으로 “과외”시키는 방식

파라미터를 다시 학습시키지 않고, **입력을 통해 “아, 이런 식으로 답하면 되는구나”** 하고 따라 하기 시작



1. 새로운 패러다임 : Meta-Learning & In-Context Learning

meta learning

pre-training (outer loop) → **incontext learning (inner loop)**



사람과 비교

일반적인 언어 능력 + 상식 활용 **<< pre-training**

설명과 예시 몇 개만 제공 **<< incontext learning**

→ 새로운 작업에 즉각적으로

적응

In-Context Learning의 작동 방식

GPT-3를 새로 학습시키는 것이 아니라,
예시 몇 개를 적어주는 것

✓ 자연어 태스크 설명

✓ 입출력 예시

✓ 진짜 문제 제시

주어진 문장이 **행복한 상황**인지, **슬픈 상황**인지 분류하시오.
(정답은 행복 또는 슬픔 중 하나로 쓰시오.)

문장: "오늘 너무 행복한 하루였다." → 레이블: 행복
문장: "정말 끔찍하고 우울한 날이야." → 레이블: 슬픔
문장: "기분이 이상하게 가라앉는 밤이야." → 레이블: 슬픔

문장: "친구들이랑 오랜만에 만나서 웃다가 하루가 다 갔다." → 레이블: ?

In-Context Learning의 작동 방식 : 예시가 몇 개냐 ?

Zero-shot

(K=0)

예시 없이 자연어 지시문만 제공.

"영어를 한국어로 번역해줘: peach"

One-shot

(K=1)

지시문 + 예시 딱 1개 제공.
인간이 새로운 문제를 배울 때와 가장 유사한 환경.

"영어를 한국어로 번역해줘.
apple -> 사과
peach ->"

Few-shot

(K=10~100)

컨텍스트 길이가 허용하는 한 최대한 많은 예시를 프롬프트에 제공.

"영어를 한국어로 번역해줘.
apple -> 사과
graph -> 포도
peach ->"

In-Context Learning의 작동 방식 : 예시가 몇 개냐 ?

Zero-shot

(K=0)

예시 없이 자연어
제공.

"영어를 한국어로
peach"

Few-shot

가 허용하는 한 최대한 많은

에 제공.

로 번역해줘.

GPT-2 에서도 이미 적용된 방식

but

"k개의 예시를 넣고 few-shot 성능을 체계적으로 비교" **하는 구조 X**

fine-tuning 에 미치지 못하는 아쉬운 성능을 보임

2. 스케일 UP

GPT-2 : 1.5B



100배

GPT-3 : 175B

다양한 모델 크기 비교 (8개)

125M (Small) 부터 175B (GPT-3) 까지 스케일별 성능 변화 추적

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

Table 2.1: Sizes, architectures, and learning hyper-parameters (batch size in tokens and learning rate) of the models which we trained. All models were trained for a total of 300 billion tokens.

GPT-2와의 차이점

Architecture & Method

구조적으로는 **거의 동일**

✓ 동일한 Transformer Decoder 아키텍처

✓ 동일한 사전학습 방법 및 옵티마이저

✓ In-context 학습 방식 계승 및 확장

Key Differences

Scale is All You Need

✓ 압도적인 모델 크기 (175B)

✓ 데이터의 양과 다양성 증대 (300B Tokens)

✓ 훨씬 긴 훈련 시간

실험 설정

- **K값 (예시 개수) 변화**

  컨텍스트 내 예시 개수(K)를 0에서 최대 100까지 늘려가며 성능 측정(zero/one/few)

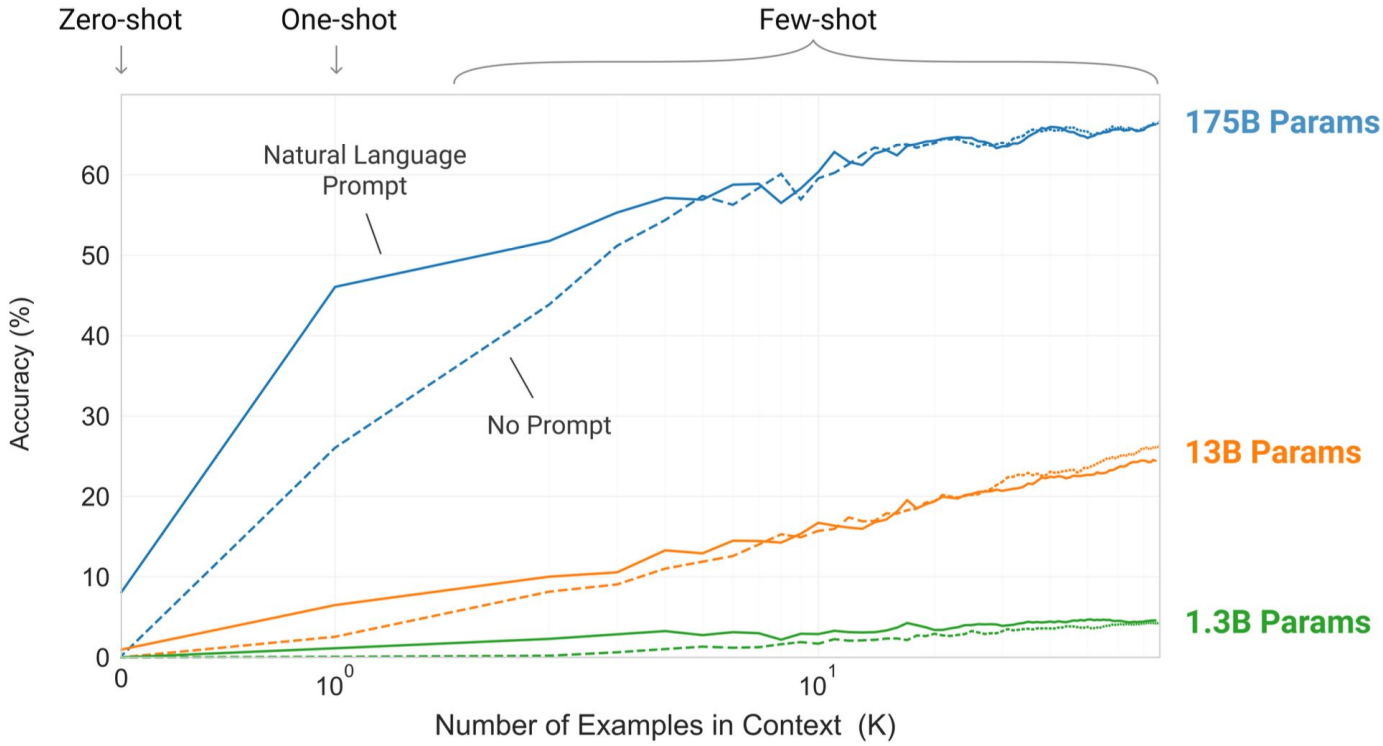
- **다양한 모델 크기 비교**

  125M (Small) 부터 175B (GPT-3) 까지 스케일별 성능 변화 추적

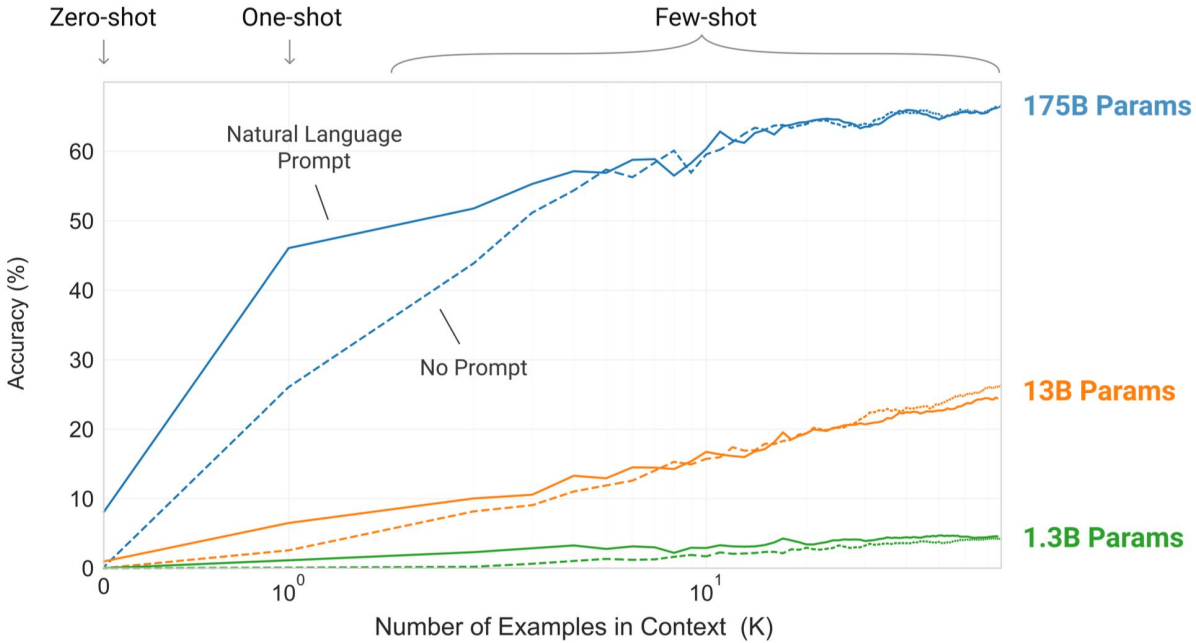
- **24개의 다양한 NLP 태스크**

  번역, QA, 추론, 문법 교정, 산술 연산 등 광범위한 평가 수행

실험 결과 : summary



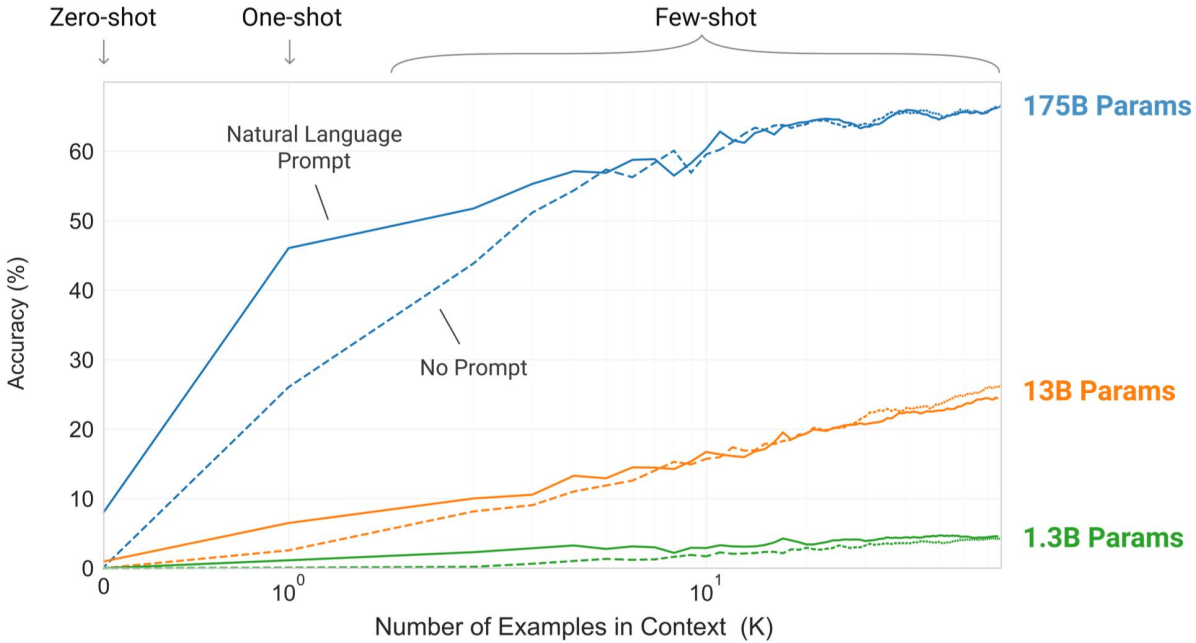
실험 결과 : summary



태스크에 대한 자연어 설명은 모델 성능을 향상시킨다

(Natural Language Prompt > No Prompt)

실험 결과 : summary

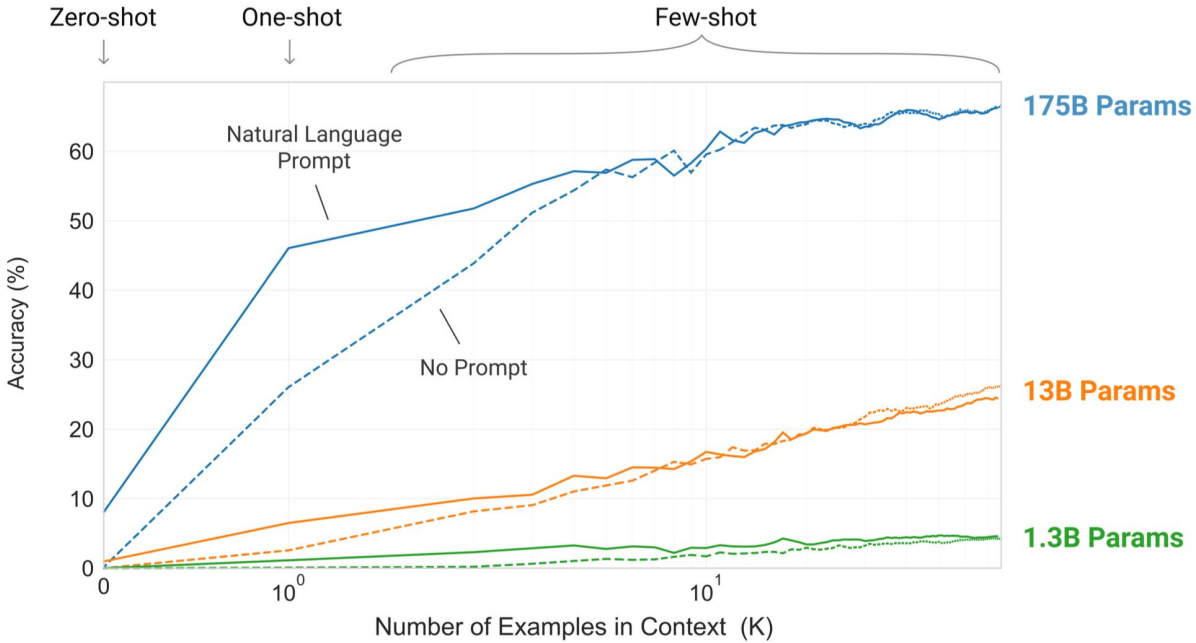


모델에 더 많은 예제를 줄수록

성능이 향상된다

(K증가시 모델정확도 증가)

실험 결과 : summary

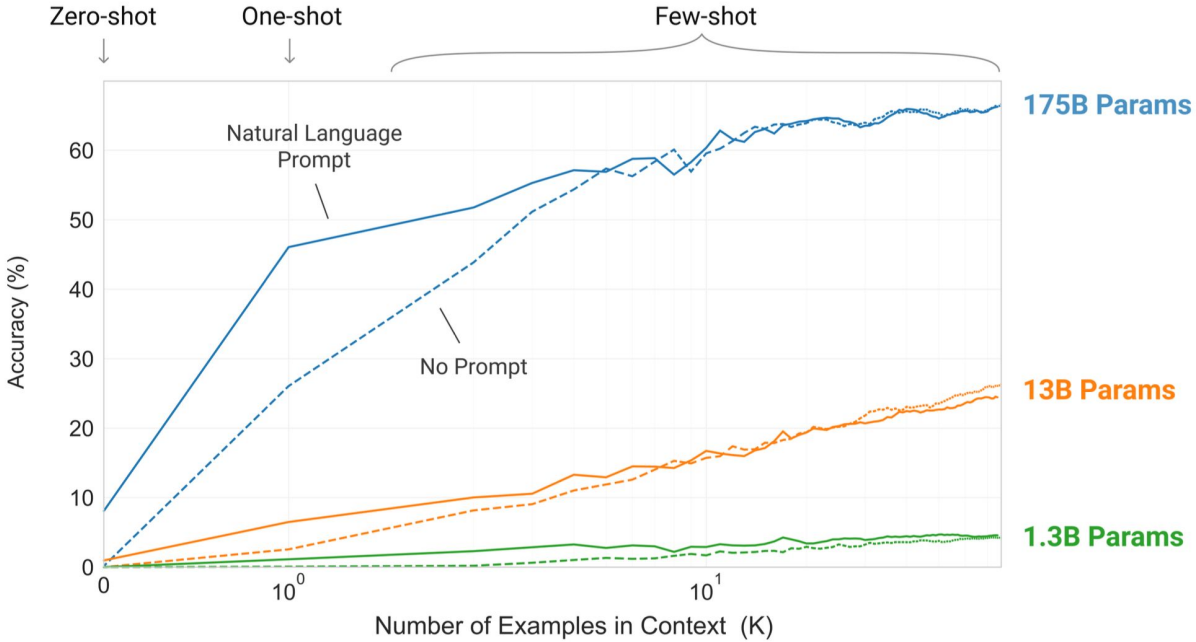


큰 모델일수록 성능 좋고,

in-context 정보를 잘 활용한다

(175B params의 결과)

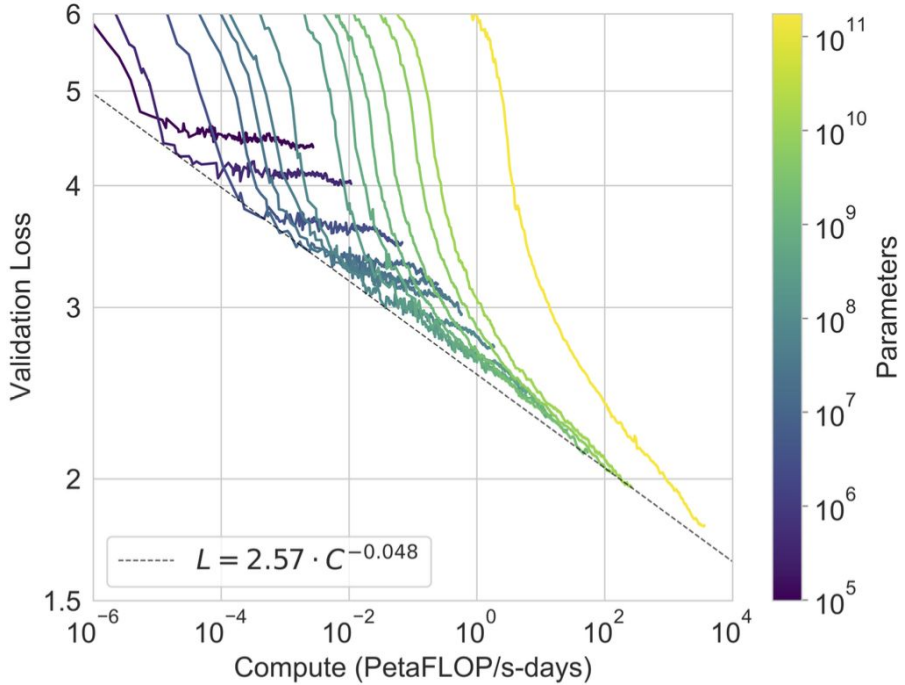
실험 결과 : summary



성능을 측정하는 동안 그라디언트 업데이트나 **fine-tuning**은 일절 일어나지 않는다 !

K의 증가에 따라 정확도가 증가하는 것처럼 보이는 부분: 문맥에 포함된 예제의 개수를 모델이 얼마나 잘 활용하는가 !

실험 결과



8개의 모델
+ 십만 개의 파라미터만을 가진 극단적으로 작은 6개의 모델 결과

Scaling Laws for Neural Language Models

Jared Kaplan *
Johns Hopkins University, OpenAI
jaredk@jhu.edu

Sam McCandlish*
Ope
sam@

Tom Henighan
OpenAI
henighan@openai.com

Tom B. Brown
OpenAI
tom@openai.com

Benjamin Chess
OpenAI
bchess@openai.com

Scott Gray
OpenAI
scott@openai.com

Alec Radford
OpenAI
alec@openai.com

Jeffrey
Op
jeffwu@



충분한 학습 데이터가 주어질 경우 검증 손실(validation loss)이 모델 크기의 함수로서 대략 매끄러운 멱법칙(power law)을 따른다고 .. !

하지만 모델 크기가 매우 커졌을 경우 power-law에서 조금 벗어나 더 향상되는 모습을 보였다.

"진짜 다양한 task들에서 Loss 향상이
일반 능력 향상으로 이어지는가 !"

실험 결과 1: 전통적인 language modeling 및 관련 태스크에서의 성능

PTB (Penn Tree Bank)

아주 오래된(인터넷 이전 시대) 작은 영어 코퍼스

지금까지의 문장을 보고 **바로 다음 단어**를 예측
단순히 "앞 문맥 → 다음 단어"를 맞추는 것이 핵심입니다.

> **EXAMPLE**
> "금요일 주식시장은 투자자들이 갑작스러운..."
> **정답:** 소식, 변화, 발표 등

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

LAMBADA

긴 문단을 읽고, 문맥을 파악하여 **마지막 한 단어**를 예측

> **EXAMPLE**
> "민수는 초등학교 때부터 지훈을 알고 있었다... (중략) ... 민수가 외롭다고 느낄 때면, 항상 전화할 수 있는 사람은 [ ]"
> **정답:** 지훈

Setting	LAMBADA (acc)	LAMBADA (ppl)
SOTA	68.0[a]	8.63[b]
GPT-3 Zero-Shot	**76.2**	**3.00**
GPT-3 One-Shot	**72.5**	**3.35**
GPT-3 Few-Shot	**86.4**	**1.92**

PPL?



PPL?



언어 모델이 다음 단어(토큰)를 얼마나 헷갈리지 않고 잘 맞추는지 나타내는 점수

낮을수록 좋음!

→ 덜 헷갈린다, 더 정확하게 예측한다는 뜻

높을수록 나쁨

→“다음에 뭐 나올지 잘 모르겠는데...?” 하고 확률을 넓게 뿌리고 있다는 뜻

실험 결과 1: 전통적인 language modeling 및 관련 태스크에서의 성능

HellaSwag

주어진 상황에서 **가장 자연스러운 다음 문장**을 선택 (4지선다)
상식이 필요해 얕은 통계로만 접근하면 틀리기 쉬운 오답들이 섞여 있습니다.

EXAMPLE: 바비큐 그릴 상황

상황: 남자가 야외 그릴 앞에서 집게를 들고 뚜껑을 연다.

A. 노트북을 꺼내 이메일을 쓴다.

B. 조심스럽게 소시지를 그릴에 올린다. (정답)

C. 소파에 누워 TV를 켠다.

D. 뒤에 있는 벽을 파란색으로 칠한다.

StoryCloze

앞선 4개의 문장을 읽고, 두개의 후보 중 이야기의 흐름상 **더 자연스러운 문장**을 선택

EXAMPLE: 기말고사 이야기

상황: 준호는 밤새 공부했고, 시험장에 들어갔다. 선생님이 시험지를 나눠주었다.

A. 준호는 이름을 쓰고 문제를 차분히 풀기 시작했다. (정답)

B. 준호는 갑자기 책을 쓰레기통에 던지고 뛰쳐나갔다.

Setting	StoryCloze (acc)	HellaSwag (acc)
SOTA	**91.8**[c]	**85.6**[d]
GPT-3 Zero-Shot	83.2	78.9
GPT-3 One-Shot	84.7	78.1
GPT-3 Few-Shot	87.7	79.3

실험 결과 2 : closed book

검색 시스템 없이 모델 파라미터 안에 든 지식만으로 **Q&A**



Natural Questions (NQs)

사람들이 실제로 구글 검색창에 입력한 **긴 질문들**로 구성

위키 문서 전체(또는 긴 패세지) 중에서 정답 위치를 찾아야 합니다.



WebQuestions (WebQs)

웹 검색에서 수집된 **짧은 자연어 질문**들.

주로 지식 그래프(Knowledge Graph) 기반의 질문 답변에 사용됩니다.



TriviaQA

다양한 **퀴즈 및 트리비아 사이트**에서 수집한 문제들

복잡한 추론보다는 사실적인 지식을 묻는 경우가 많습니다.

실험 결과 2 : closed book

검색 시스템 없이 모델 파라미터 안에 든 지식만으로 **Q&A**

Setting	NaturalQS	WebQS	TriviaQA
RAG (Fine-tuned, Open-Domain) [LPP+20]	**44.5**	**45.5**	**68.0**
T5-11B+SSM (Fine-tuned, Closed-Book) [RRS20]	36.6	44.7	60.5
T5-11B (Fine-tuned, Closed-Book)	34.5	37.4	50.1
GPT-3 Zero-Shot	14.6	14.4	64.3
GPT-3 One-Shot	23.0	25.3	**68.0**
GPT-3 Few-Shot	29.9	41.5	**71.2**

NQ가 위키피디아의 **매우 세밀한(fine-grained) 지식**을 요구하기 때문

실험 결과 2 : closed book

검색 시스템 없이 모델 파라미터 안에 든 지식만으로 **Q&A**

Setting	NaturalQS	WebQS	TriviaQA
RAG (Fine-tuned, Open-Domain) [LPP+20]	**44.5**	**45.5**	**68.0**
T5-11B+SSM (Fine-tuned, Closed-Book) [RRS20]	36.6	44.7	60.5
T5-11B (Fine-tuned, Closed-Book)	34.5	37.4	50.1
GPT-3 Zero-Shot	14.6	14.4	64.3
GPT-3 One-Shot	23.0	25.3	**68.0**
GPT-3 Few-Shot	29.9	41.5	**71.2**

GPT-3는 폭넓은 지식을 가지고 있지만,

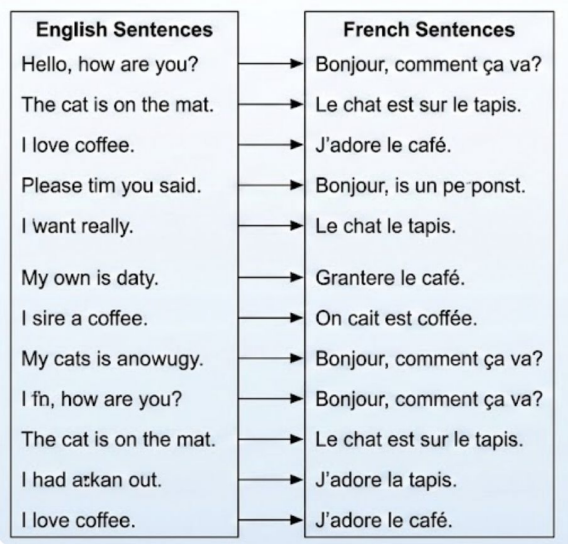
특정 문서의 구체적인 디테일을 정확히 인출하는 능력은 미세 조정된 모델(파인튜닝)에 비해 한계가 있음

실험 결과 3: 번역

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

기존 번역 모델

- **지도 학습 (Supervised Learning)**



English Sentences	French Sentences
Hello, how are you?	Bonjour, comment ça va?
The cat is on the mat.	Le chat est sur le tapis.
I love coffee.	J'adore le café.
Please tim you said.	Bonjour, is un pe ponst.
I want really.	Le chat le tapis.
My own is daty.	Grantere le café.
I sire a coffee.	On cait est coffée.
My cats is anowugy.	Bonjour, comment ça va?
I fn, how are you?	Bonjour, comment ça va?
The cat is on the mat.	Le chat est sur le tapis.
I had akan out.	J'adore la tapis.
I love coffee.	J'adore le café.

- **비지도 학습 (Unsupervised Learning)**

1. 개별 학습

각 언어(영어, 프랑스어)를 따로 읽고 쓰는 능력을 먼저 기릅니다.

2. 공통 공간

두 언어를 하나의 의미 공간(Shared Latent Space)에 매핑하여 같은 의미는 비슷한 위치에 오도록 합니다.

3. 역번역

가짜 데이터(Pseudo-parallel)를 생성하고 이를 통해 번역기를 반복 학습시킵니다.
(Back-Translation)

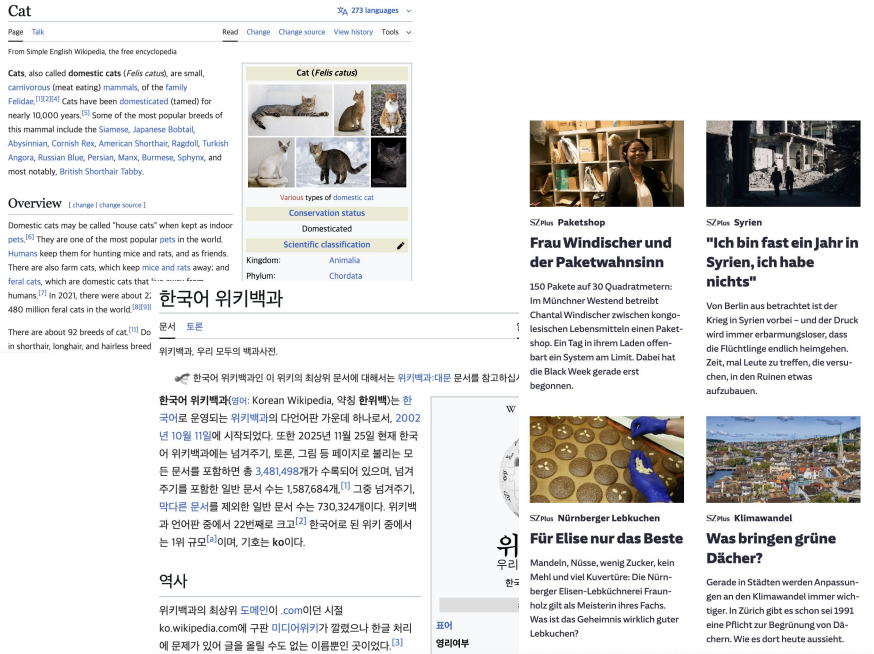
실험 결과 3: 번역

GPT-3

번역을 위한 문장 쌍을 전혀 제공하지 않음

웹상의 다국어 혼합 코퍼스를 통째로 학습

93%의 텍스트는 영문이었지만 나머지 7%는 기타 언어들을 포함



실험 결과 3: 번역

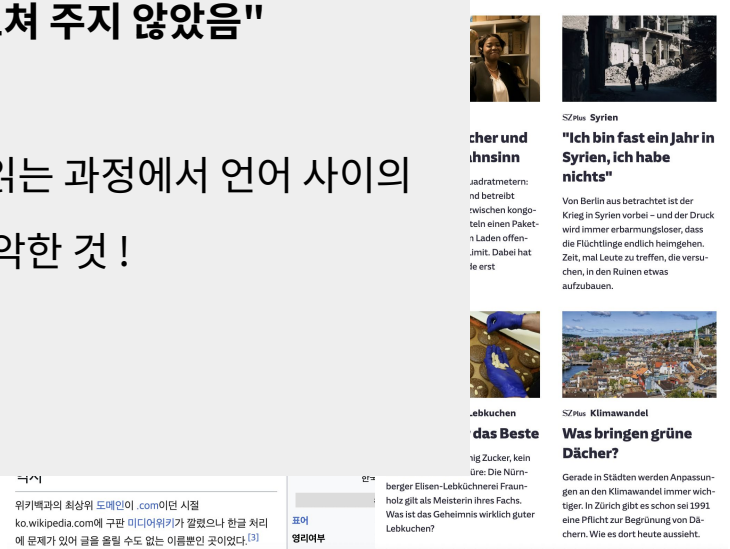
GPT-3

번역을 위한 문장
웹상의 다국어 훈

93%의 텍스트는
는 기타 언어들을

"영어-불어 문장을 짝지어 가르쳐 주지 않았음"

GPT-3는 여러 언어가 뒤섞인 텍스트를 읽는 과정에서 언어 사이의 대응 관계를 스스로 파악한 것 !



실험 결과 3: 번역

비지도 번역 모델

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

기존의 SOTA(State-of-the-Art) 비지도 번역 모델보다

5 BLEU 이상 높은 점수를 기록

실험 결과 4 : 대명사 지칭

모호한 대명사가 어떤 명사를 가리키는지 , 상식·의미에 기반해 맞추는 문제

문장: "잔디 깎는 기계가 나무 옆에 있었다. **그것(It)**이 너무 커서 옮길 수 없었다."

질문: 여기서 **'그것'**은 무엇일까?

① 잔디 깎는 기계 ✓

② 나무

Setting	Winograd	Winogrande (XL)
Fine-tuned SOTA	**90.1**[a]	**84.6**[b]
GPT-3 Zero-Shot	88.3*	70.2
GPT-3 One-Shot	89.7*	73.2
GPT-3 Few-Shot	88.6*	77.7

Winograd Dataset

SOTA 및 사람 수준에 근접한 뛰어난 성능을 보임

Winogrande Dataset

Winograd의 더 어려운 난이도 버전.

- 비교: **SOTA (84.6%)** 및 사람 **(94.0%)**에 비해 여전히 뒤쳐지는 결과를 보였습니다.

실험 결과 5: 상식추론

PhysicalQA

물리학 원리 질문

ARC

3-9학년 과학 시험 문제

4지선다

OpenBookQA

오픈북 형태의

질의응답

Setting	PIQA	ARC (Easy)	ARC (Challenge)	OpenBookQA
Fine-tuned SOTA	79.4	**92.0**[KKS+20]	**78.5**[KKS+20]	**87.2**[KKS+20]
GPT-3 Zero-Shot	**80.5***	68.8	51.4	57.6
GPT-3 One-Shot	**80.5***	71.2	53.2	58.8
GPT-3 Few-Shot	**82.8***	70.1	51.5	65.4

Easy와 Challenge 난이도

모두 SOTA에 미치지 못함.

In-context learning 효과보임

그러나 SOTA보다 낮은 성능.

실험 결과 6: 기계독해

전반적으로 **SOTA에 미치지 못하는 성적**을 보임.

Setting	CoQA	DROP	QuAC	SQuADv2	RACE-h	RACE-m
Fine-tuned SOTA	**90.7**[a]	**89.1**[b]	**74.4**[c]	**93.0**[d]	**90.0**[e]	**93.1**[e]
GPT-3 Zero-Shot	81.5	23.6	41.5	59.5	45.5	58.4
GPT-3 One-Shot	84.0	34.3	43.3	65.4	45.9	57.4
GPT-3 Few-Shot	85.0	36.5	44.3	69.8	46.8	58.1

- **CoQA:** 자유 형식 대화 데이터셋
- **QuAC:** 구조화된 대화 및 답변 선택 모델링
- **DROP:** 이산적 추론과 산술 능력을 평가
- **RACE:** 중/고등 난이도 다지선다형 영어 시험

실험 결과 6: 기계독해

기계 독해 태스크 :

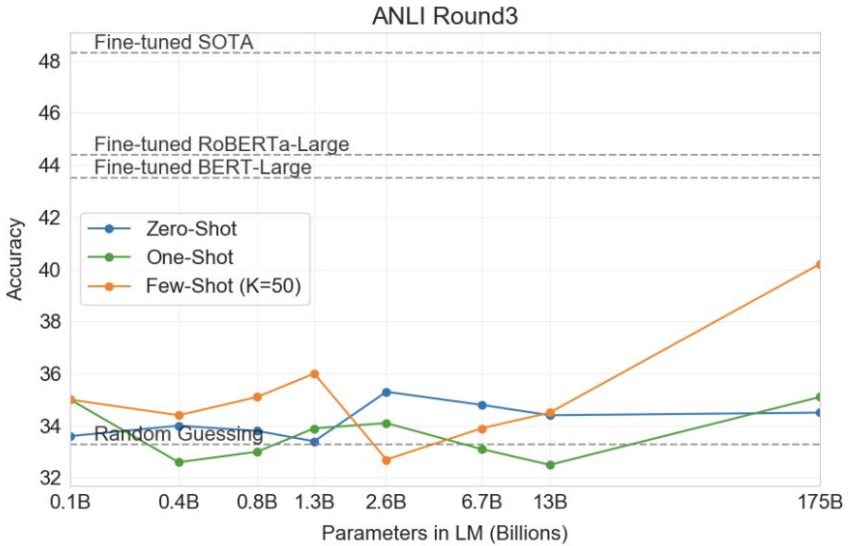
- 긴 지문을 읽고 질문에 맞는 근거 문장 위치를 찾고
- 필요한 경우 여러 문장을 종합해서 추론한 뒤
- 정확한 형태의 답을 골라야 함

근데 **GPT-3**는 구조화된 독해 **+** 논리적 선택은
전용아키텍쳐, 파인튜닝된 모델보다 약하다 !

- **RACE:** 중/고등 난이도 다지선다형 영어 시험

실험 결과 7: 자연어 논리 비교 추론

두 문장 간의 관계(논리 따름, 모순, 중립)를 판별하는 문제



ANLI Dataset 결과

Few-shot 설정에서도 굉장히 낮은 성능

이는 모델이 복잡한 논리적 관계를 파악하는 데

여전히 한계가 있음을 시사

실험 결과 8: 연산

2자리 덧셈 (2D+) : 48 + 76 = ?

2자리 뺄셈 (2D-) : 34 - 53 = ?

3자리 덧셈/뺄셈 (3D+, 3D-)

4자리 덧셈/뺄셈 (4D+, 4D-)

5자리 덧셈/뺄셈 (5D+, 5D-)

2자리 곱셈 (2Dx) : 24 × 42 = ?

한 자리 복합식 (1DC) : 6 + (4 * 8) = ?

```
Q: What is 48 plus 76? A: 124.
Q: What is 6 + (4 * 8)? A: 38.
```

실험 결과 8: 연산

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

good

Zero-shot → One-shot → Few-shot으로 갈수록
모든 태스크에서 성능이 계속 좋아짐.

실험 결과 8: 연산

“암기냐, 진짜 계산이냐?”



3자리 덧셈과 뺄셈 문제 2,000개를 무작위로

추출하여 학습 데이터 내 존재 여부를 전수 조사

검색 쿼리 패턴:

"NUM1 + NUM2 ="

"NUM1 plus NUM2"

실험 결과 8: 연산

“암기냐, 진짜 계산이냐?”



3자리 덧셈과 뺄셈 문제 2,000개를 무작위로

추출하여 학습 데이터 내 존재 여부를 전수 조사

0.8%

덧셈 일치율

2,000문제 중 단 17건

0.1%

뺄셈 일치율

2,000문제 중 단 2건

실험 결과 8: 연산

“암기냐, 진짜 계산이냐?”



+

'자리 올림'을 누락하거나 자릿수를 헷갈리는 등,

인간이 계산할 때 범하는 실수와 유사한 패턴을 보임.

실제 연산 알고리즘을 내부적으로 수행

실험 결과 9: Word Scrambling / Manipulation

Task	설명	예시
Cycle (CL)	철자를 rotation시켜 원래 단어 만들기	lyinevitab → inevitably
A1	처음/끝 제외하고 철자 뒤섞기	criroptuon → corruption
A2	처음/끝 2글자 제외하고 뒤섞기	opoepnnt → opponent
Insertion (RI)	글자 사이에 랜덤 기호, 공백 삽입	s.u!c/c!e.s... → succession
Reverse (RW)	단어를 뒤집기 (성공하지 못함)	stcejbo → objects

실험 결과 9: Word Scrambling / Manipulation

Task	설명	예시
Cycle (CL)	철자를 rotation시켜 원래 단어 만들기	lyinevitab → inevitably
A1	처음/끝 제외하고 철자 뒤섞기	criroptuon → corruption
A2	처음/끝 2글자 제외하고 뒤섞기	opoepnnt → opponent
Insertion (RI)	글자 사이에 랜덤 기호, 공백 삽입	s.u!c/c!e.s... → succession
Reverse (RW)	단어를 뒤집기 (성공하지 못함)	stcejbo → objects

Setting	CL	A1	A2	RI	RW
GPT-3 Zero-shot	3.66	2.28	8.91	8.26	0.09
GPT-3 One-shot	21.7	8.62	25.9	45.4	0.48
GPT-3 Few-shot	37.9	15.1	39.7	67.2	0.44

good

실험 결과 9: Word Scrambling / Manipulation

Task	설명	예시
Cycle (CL)		nevitably
A1		corruption
A2		pponent
Insertion (RI)		succession
Reverse (RW)		cts

이런 인위적인 문자 조작 태스크는 학습데이터에 거의 없을 것

성능 향상은 정말로 프롬프트 안의 예시에서 규칙을 학습하는 **in-context learning**으로 보는 게 타당

GPT-3 Zero-shot	3.66	2.28	8.91	8.26	0.09
GPT-3 One-shot	21.7	8.62	25.9	45.4	0.48
GPT-3 Few-shot	37.9	15.1	39.7	67.2	0.44

good

실험 결과 10: 뉴스기사 생성 실험

1. 데이터 수집

실제 뉴스 사이트에서 **제목**과 **부제목**을 가져옵니다. 이것이 AI에게 주어지는 유일한 힌트입니다.

2. AI 작성

GPT-3 (175B 모델)에게 제목과 부제목을 주고, 나머지 **본문 기사**를 이어서 작성하게 합니다.

3. 인간 평가

사람들에게 "이 기사가 **인간**이 쓴 것인지, **모델**이 쓴 것인지" 맞추도록 요청합니다.

실험 결과 10: 뉴스기사 생성 실험

- 독자가 AI가 쓴 글과 인간 기자가 쓴 글을 구별할 수 있는가?
- **평가 기준:** 단순한 문법적 오류가 아닌, 글의 흐름, 논리, 문체 등 전반적인 자연스러움을 평가합니다.
- **비교군:** 일부러 엉성하게 만든 컨트롤 모델과 최신 GPT-3 175B 모델을 비교하여 성능 차이를 검증합니다.



3. 인간 평가

사람들에게 "이 기사가 **인간**이 쓴 것인지, **모델**이 쓴 것인지" 맞추도록 요청합니다.

실험 결과 10: 뉴스기사 생성 실험

	Mean accuracy	95% Confidence Interval (low, hi)	t compared to control (p-value)	"I don't know" assignments
Control	88%	84%–91%	-	2.7%
GPT-3 175B	52%	48%–57%	12.7 (3.2e-23)	10.6%

** 수치가 높을수록 인간이 AI 기사를 잘 식별해냈음(가짜임을 들킴)을 의미*

사실상 '찍기' 수준

문체, 구조, 어휘 선택 측면에서 이미 인간급

퍼포먼스를 달성했음을 보여주는 강력한 증거



실험 결과 11: 새로운 단어 학습

GPT-3에게 처음 보는 가상의 단어(예: Gigamuru)의 정의를 딱 한 번만 보여주고 문장을 만들게 시켰을 때의 결과

```
Input: "A 'Gigamuru' is a type of Japanese musical instrument."

Output: "I have a Gigamuru that my uncle gave me..."
```

단순 복사가 아닌 문맥과 용법의 일반화

시제와 활용형(screeg → screeghed)까지 추론

" Meta-learning"의 강력한 증거

실험 결과 정리 : 한계

- 논리 관계 (NLI/ANLI): 문장 간의 모순이나 함의를 파악하는 논리적 비교에 취약.

- 복합 추론 (Multi-hop): 여러 단계를 거쳐야 답이 나오는 문제에서 성능 저하.

- 세부 팩트 (NQ): 위키피디아의 아주 구석진 팩트를 묻는 질문에는 약함.

테스트셋 오염 문제

데이터의 규모와 위험

GPT-3는 인터넷 기반의 **대규모 데이터**로 사전학습(Pre-training)

∴ 벤치마크 테스트 셋의 문장들이 학습 데이터에 섞여 들어갔을 가능성이 크다



테스트셋 오염 문제

연구의 중요성

*“단순히 SOTA(State-of-the-Art) 성능을 달성하는 것을 넘어, 이러한 **테스트셋 오염 문제를 찾아내고 분석**하는 것 자체가 매우 중요한 연구 주제입니다.”*

정확한 성능 평가를 위한 필수적인 검증 과정

“클린” 테스트셋 구축 방법론



1. 중복 검출

각 벤치마크 데이터와 사전학습 데이터 간의 **13-gram**(13개 연속 단어)이 겹치는 항목을 정밀하게 스캔합니다.



2. 데이터 정제

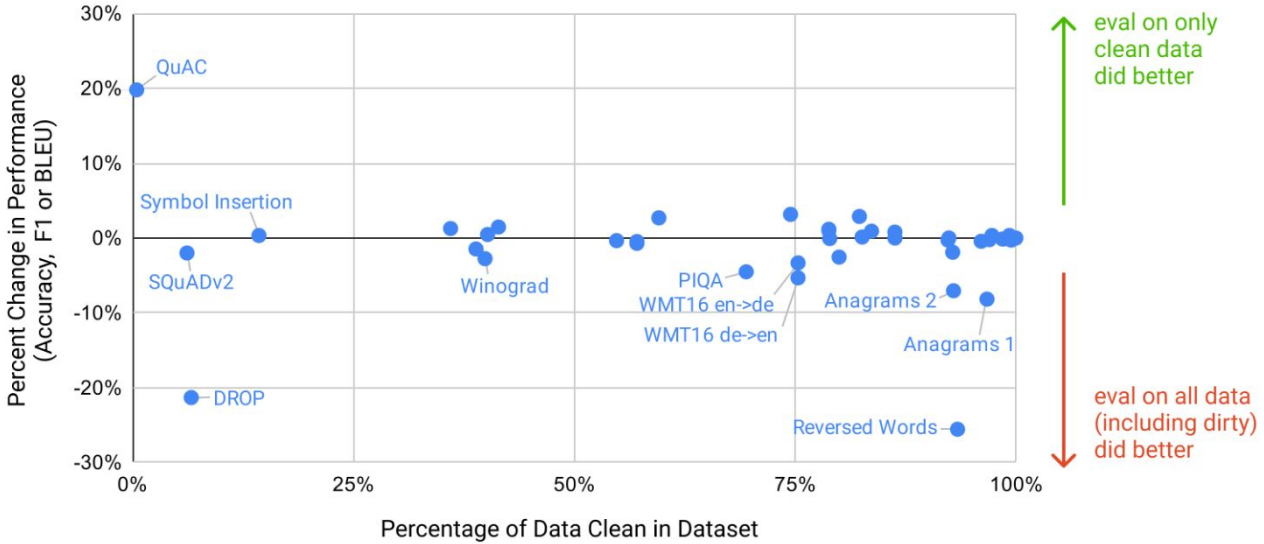
검출된 중복 항목을 테스트셋에서 **완전히 제거**하여 오염 요소를 배제한 "클린 테스트셋"을 새롭게 구성합니다.



3. 재평가 수행

오염되지 않은 클린 셋을 사용하여 GPT-3의 성능을 **다시 평가**하고, 기존 결과와 비교 분석합니다.

테스트셋 오염 문제



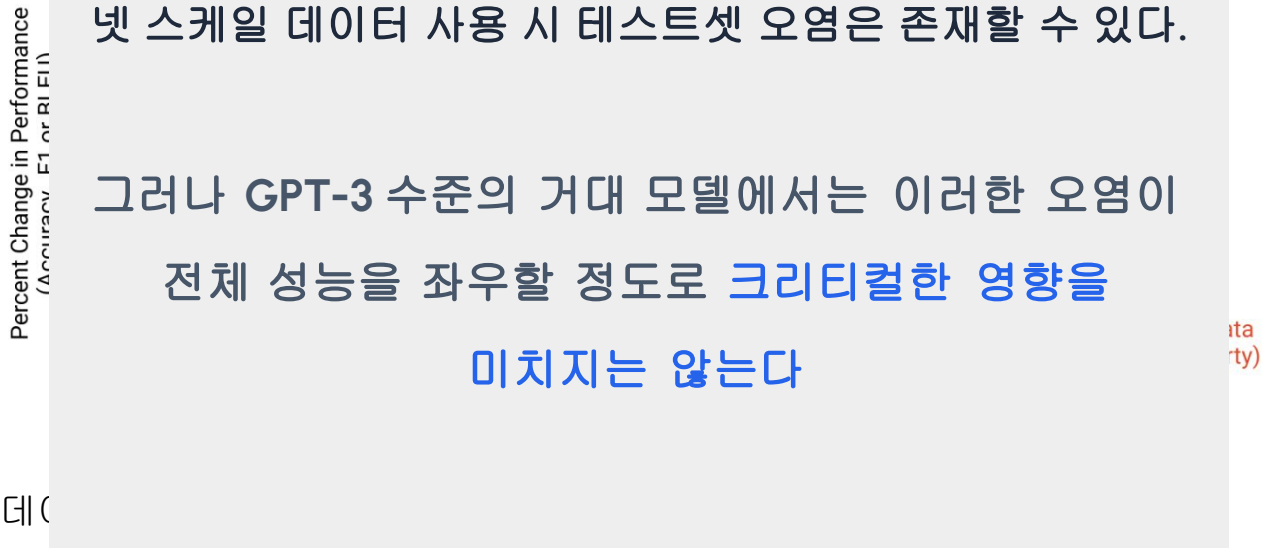
훈련 데이터랑 겹쳤을 가능성 크다

처음 보는 문제

테스트셋 오염 문제

넷 스케일 데이터 사용 시 테스트셋 오염은 존재할 수 있다.

그러나 GPT-3 수준의 거대 모델에서는 이러한 오염이 전체 성능을 좌우할 정도로 크리티컬한 영향을 미치지는 않는다



한계



물리적 세계 이해 부족

언어 통계는 완벽하지만, 실제 세계의 물리 법칙이나 인과관계에 대한 진정한 이해는 결여됨.



프롬프트 민감도

단어 순서나 사소한 표현 변화에도 성능이 급격히 변동. "프롬프트 엔지니어링" 의존도가 높음.



편향(Bias) 문제

인터넷 데이터를 학습하여 성별, 인종, 종교적 편향과 고정관념을 그대로 재생산할 위험.



천문학적 비용

175B 모델 학습 및 구동에 필요한 막대한 전력과 하드웨어 비용. 지속 가능성에 대한 의문.

패러다임의 전환

From Fine-tuning to Prompting

기존에는 태스크마다 모델을 따로 만들고 **파인튜닝(Fine-tuning)**해야 했다면,

이제는 거대 언어 모델 하나에 **자연어 프롬프트(Few-shot)**만 입력하여 모든 문제를 해결하는 시대로 접어들었습니다.

"Pre-train Once, Prompt Anywhere"

