Toolformer:

Language Models Can Teach Themselves to Use Tools

Timo Schick Jane Dwivedi-Yu Roberto Dessì† Roberta Raileanu Maria Lomeli Luke Zettlemoyer Nicola Cancedda Thomas Scialom

Meta AI Research (Feb 2023) [PDF]

nonmuniljihee

Shin Jung In

Contents

연구 배경

연구 배경

언어 모델의 한계

1234 x 5678의 결과값은 무엇인가요?

2024년 노벨상 수상자는 누구인가요?

영화 La vita è bella와 鉄道員의 공통된 message는?

내일 날씨는 어떻게 되나요?

여행 경비 정산을 도와주세요.

언어 모델은 몇 가지 예제나 텍스트 지시사항만으로 새로운 작업을 해결하는 놀라운 능력을 보이지만,
동시에 간단한 계산이나 사실 조회와 같은 기본적인 기능에서 오히려 성능이 떨어진다는 문제점을 가지고 있습니다.

연구 배경

도구 사용 언어 모델

연구	도구 활용	특징 및 한계점
WebGPT (2021)	웹 브라우징(인터넷 검색)	많은 양의 human annotation 필요, Web browsing이 특정 작업에 국한됨
LaMDA (2022)	계산기, 번역기 등 일부 API	많은 양의 human annotation 필요, API 사용이 특정 대화 맥락에만 제한적으로 이루어짐
PAL (2022)	Python 코드 실행	Few-shot Prompting에 의존하며, 코드 실행에 국한됨
ReAct (2022)	Reasoning & Action 결합	복잡한 작업에도 API chaining 가능하며, interactive한 API 사용 지원
Toolformer (2023)	계산기, QA, Wiki검색, 번역, 캘린더	human annotation 없이 범용적인 API 사용 가능. API 연쇄 활용에는 제한적
Gorilla (2023)	방대한 API 자동 연결 및 활용	수천 개의 API 연결 및 동적 호출 가능
ChatGPT Plugins (2023)	외부 서비스를 Plugin 형태로 추가	사용자의 요구에 따라 즉각적이고 실시간으로 도구를 활용

연구 배경

언어 모델은 어떻게 도구를 활용하는가

1. 지도학습

- 사람이 직접 "정답이 포함된 예제(지도 데이터)"를 만들어 제공해야 합니다.
- 입력: "What is 12 × 34?" 출력: "The result is [Calculator(12 × 34) → 408]." 같은 데이터를 학습합니다.

2. 강화학습

- 여러 번 시도한 후 보상이 높은 행동을 반복하는 방식으로 학습합니다.
- 정답일 경우 보상(Reward), 잘못된 경우 패널티(Penalty)를 줍니다.

3. 자기지도학습 (self-supervised learning)

- 외부 레이블 없이 스스로 데이터를 변형하고 정답을 생성하여 학습하는 방식입니다.
- Toolformer는 API 사용이 유용한지 아닌지를 스스로 학습합니다.
- 사람이 API 호출 데이터를 만들 필요 없으며, API를 사용할지 말지를 모델이 스스로 결정합니다.

연구 배경

Toolformer 의 특장점

- 완전한 자기지도학습 (Self-supervised Learning) 기반 API 학습 연구
- 사람이 개입하지 않아도 도구 사용을 학습하는 최초의 방식 중 하나
- 모델이 API를 언제, 어떻게 사용할지를 스스로 결정하는 혁신적인 방식

기존 연구들이 도구 사용을 위해 사람이 직접 설계한 룰을 따르는 방식이라면,
Toolformer는 모델이 API 사용법을 "스스로 발견"하는 최초의 시도라는 점에서 연구 가치가 높습니다.

✓ Toolformer는 기존 데이터를 API 호출과 그 결과로 증강한 데이터셋으로 만듭니다.

✓ API 호출이 포함된 데이터셋 C*로 fine-tuning하여 API 호출 능력을 학습합니다.

✓ 학습은 human annotation이 필요 없는 self-supervised 방식으로 진행됩니다.

관련 이론

관련 이론

Dataset Augmentation

- 원본 데이터셋에 새로운 정보나 API 호출 결과를 추가하여 모델의 성능을 높이는 기술입니다.
- Toolformer는 원본 데이터셋 C에 API 호출 결과를 포함시켜 새로운 **증강 데이터셋 C*를 생성**하고, 이를 통해 모델의 API 활용 능력을 내재화합니다.

In-context Learning

- 언어 모델이 학습(fine-tuning) 없이, 입력 텍스트에 포함된 몇 가지 예시만을 바탕으로 새로운 작업을 수행하는 능력입니다.
- Toolformer는 API 호출 후보를 생성할 때, LM의 in-context learning 능력을 활용합니다.
- 즉, 사전 제공된 소수의 프롬프트 예시를 바탕으로 **언어 모델이 스스로 API 호출 위치와 내용을 결정**합니다.

관련 이론

Scaling Law

- 언어 모델의 크기(파라미터 수)가 증가할수록 성능이 향상되는 현상입니다.
- Toolformer 논문에서는 모델의 크기가 커질수록 API 호출을 잘 활용할 수 있다는 점을 제시합니다. (775M 이상의 파라미터)

Bootstrapping & Self-Training

- Bootstrapping : 모델이 스스로 데이터를 생성하고 이를 학습하여 점진적으로 성능을 향상시키는 방법
- Self-Training : 초기 모델이 예측한 출력을 다시 학습 데이터로 활용하여 지속적으로 모델을 개선하는 기법
- Toolformer는 LM이 자체 생성한 API 호출을 다시 학습 데이터로 사용하고, 유용성을 평가한 후 필터링하여 자율적으로 API 사용을 개선합니다.

Toolformer

Toolformer

API Name	Example Input	Example Output
Question Answering	Where was the Knights of Columbus founded?	New Haven, Connecticut
Wikipedia Search	Fishing Reel Types	Spin fishing > Spin fishing is distinguished between fly fishing and bait cast fishing by the type of rod and reel used. There are two types of reels used when spin fishing, the open faced reel and the closed faced reel.
Calculator	27 + 4 * 2	35
Calendar	ε	Today is Monday, January 30, 2023.
Machine Translation	sûreté nucléaire	nuclear safety

API	설명
Question Answering	Atlas(2022)라는 RAG 기반의 QA 시스템을 사용하여, 질문에 대한 답변을 가져옵니다.
Wikipedia Search	특정 키워드에 대한 정보를 Wikipedia에서 검색하는 API입니다.
Calculator	단순한 덧셈, 뺄셈, 곱셈, 나눗셈만 지원하며, 결과는 소수점 두 자리로 반올림됩니다.
Calendar	현재 날짜를 반환하는 API입니다. (Calendar() →Sunday, March 16, 2025)
Machine Translation	다국어를 영어로 번역하는 API로 NLLB(600M. 2022)를 사용해 번역하고, fastText로 입력 언어를 감지합니다.

Toolformer

"1234 x 5678 결과값은 7000000 입니다."

"1234 x 5678 결과값은 [Calculator("1234 x 5678") → 7006652] 7000000 입니다."

Toolformer의 목적은 기존 데이터를 API call을 포함하는 데이터셋 C*로 만드는 것입니다.

모델은 API call이 포함된 데이터셋 C*로 fine-tuning 되어 API 호출 능력을 학습하게 됩니다.

학습은 human annotation이 필요 없는 self-supervised 방식으로 진행됩니다.

The New England Journal of Medicine is a registered trademark of [QA("Who is the publisher of The New England Journal of Medicine?") → Massachusetts Medical Society] the MMS.

Out of 1400 participants, 400 (or [Calculator(400 / 1400) → 0.29] 29%) passed the test.

The name derives from "la tortuga", the Spanish word for [MT("tortuga") → turtle] turtle.

The Brown Act is California's law [WikiSearch("Brown Act") → The Ralph M. Brown Act is an act of the California State Legislature that guarantees the public's right to attend and participate in meetings of local legislative bodies.] that requires legislative bodies, like city councils, to hold their meetings open to the public.

Toolformer

Toolformer의 핵심 접근법은 다음 **세 가지 단계**로 구성됩니다.

1. **API 호출 후보 샘플링** : 언어 모델이 자체적으로 API 호출 후보를 생성합니다.
2. **API 호출 실행** : 생성된 후보를 실제로 API에 호출하여 결과를 받습니다.
3. **API 호출 필터링** : 호출된 결과가 모델의 향후 토큰 예측을 얼마나 돕는지 평가해 유용한 호출만 필터링하여 학습 데이터로 활용합니다.

Toolformer

1. API 호출 후보 샘플링

Your task is to add calls to a Question Answering API to a piece of text. The questions should help you get information required to complete the text. You can call the API by writing "[QA(question)]" where "question" is the question you want to ask. Here are some examples of API calls:

Input: Joe Biden was born in Scranton, Pennsylvania.

Output: Joe Biden was born in [QA("Where was Joe Biden born?")] Scranton, [QA("In which state is Scranton?")] Pennsylvania.

Input: Coca-Cola, or Coke, is a carbonated soft drink manufactured by the Coca-Cola Company.

Output: Coca-Cola, or [QA("What other name is Coca-Cola known by?")] Coke, is a carbonated soft drink manufactured by [QA("Who manufactures Coca-Cola?")] the Coca-Cola Company.

Input: x

Output:

- 언어 모델이 텍스트 내에서 **API 호출이 필요한 위치를 예측**하고, 해당 위치에서 API를 호출하도록 데이터셋을 자동 생성하는 과정입니다.
- Toolformer는 미리 정의된 프롬프트 $P(x)$를 사용하여 API 호출이 삽입될 가능성이 높은 위치를 식별합니다.
- 사용할 API에 따라 대괄호 내 요청값을 변경하여, Few-shot example을 제시합니다.

Toolformer

1. API 호출 후보 샘플링

$$p_i = p_M(\texttt{<API>} | P(x), x_1 : i-1)$$

- 우선 주어진 텍스트 x = (x1, x2, ..., xn)에서 각 위치 i에 대해, 모델이 API 호출을 시작할 확률 p_i를 계산합니다.
- 즉, 특정 위치 i에서 모델이 <API> 토큰을 생성할 확률을 기반으로 API 호출 위치를 결정합니다.

$$I = \{i \mid p_i > \tau_s\}$$

- 이후 샘플링 임계값(τs)을 설정하여 p_i > τs인 위치만 API 호출 후보로 유지합니다.
- 너무 많은 API 호출이 발생하는 것을 방지하기 위해 최대 k개의 위치만 선택합니다.

- 결과적으로 각 선택된 위치 i에서 최대 m개의 **API 호출을 샘플링**합니다.
- 이때, 모델이 <API>로 시작하고 </API>로 끝나는 구조를 따르도록 학습됩니다.

Toolformer

2. API 호출 실행

- 첫 번째 단계에서 샘플링된 API 호출을 실제로 실행하여 API 응답을 가져오는 과정입니다.
- API의 종류에 따라 실행 방식이 다를 수 있습니다.
- API 호출 결과를 r_i 로 저장하고, 원본 텍스트 내에서 해당 위치에 API 호출을 삽입합니다.
- API 호출과 결과는 아래와 같은 형식으로 기록됩니다

$$e(c_i, r_i) = \texttt{<API>}\ ac(ic) \rightarrow r_i\ \texttt{</API>}$$

Toolformer

3. API 호출 필터링

- 모든 API 호출이 실제로 유용한 것은 아닐 수 있습니다.
- 따라서 API 응답이 실제로 도움이 되는지 판단하여 불필요한 API 호출을 제거하는 과정이 필요합니다.

$$L_i(z) = -\sum_{j=i}^{n} w_{j-i} \log p_M(x_j | z, x_{1:j-1})$$

- Toolformer는 API 호출이 실제로 모델의 성능을 향상시키는지 확인하기 위해 손실 함수(Loss Function) 기반 필터링을 수행합니다.
- 위 손실함수에서 z는 API call과 response를 포함하는 prefix 토큰입니다.
- 즉, prefix z와 j 이전까지의 토큰들이 주어졌을 때, 다음에 j 토큰이 나타날 확률을 구하게 됩니다.
- 이를 j(=i)부터 n까지 구하게 되므로, i 이후 토큰들의 Loss를 계산하게 됩니다.
- 즉, **$L_i(z)$는 위치 i에서 z의 API 호출이 있을 때의 Loss 값**이 됩니다.

Toolformer

3. API 호출 필터링

$$L_i^- - L_i^+ \geq \tau_f$$

- L_i^+는 위치 i를 기준으로 API call과 response가 함께 있을 경우의 손실입니다.
- L_i^-는 API call이 없는 경우와 API response를 사용하지 않는 경우의 손실을 비교했을 때의 최소값을 가져옵니다.
- 결과적으로 L_i^-에서 L_i^+를 뺀 값이 threshold τ_f보다 큰 경우에만 API 호출을 유지하게 됩니다.
- 즉, 생성된 API 호출 후보 중 손실을 일정 수준 이상 감소시키는 호출만 남기고 사용합니다.

실험 및 결과

실험 및 결과

실험 개요. Toolformer가 검증한 부분

실험 주제	목표
API를 사용하면 성능이 향상되는가?	API 사용이 LLM의 성능을 실제로 향상시키는지 검증
Toolformer가 언어 모델링 능력을 손상시키지는 않는가?	API call 포함된 C*로 fine-tuning 후 LLM의 성능 검증
모델 크기에 따라 API 사용 능력에 변화가 있는가?	더 큰 모델일수록 API 사용을 더 잘하는지 검증

실험 및 결과

검증에 사용된 Baseline 모델

- GPT-J (6B) → 기본 상태의 GPT-J 모델
- GPT-J + C → API 호출이 없는 CCNet 데이터셋으로 미세 조정된 GPT-J 모델
- **Toolformer (+ C*)** → API 호출이 추가된 CCNet 데이터셋 C*로 미세 조정된 GPT-J 기반 모델
- Toolformer (disabled) → Toolformer와 동일하게 C*로 미세 조정되었으나, API 호출 기능이 비활성화된 모델

Toolformer (disabled)는 C*에 의한 미세조정이 언어 모델링 성능을 손상시키지 않음을 확인하기 위한 모델입니다.

추가적으로 실험에서는 GPT-J (6B)보다 훨씬 큰 OPT(66B) 및 GPT-3(175B)와 성능을 비교하였습니다.

데이터셋

- CCNet(Wenzek et al., 2020)에서 샘플링된 데이터 사용
- 최대 25,000개 예제(API 당) 생성하여 API 호출을 포함한 학습 데이터셋(C*) 구축
- 모델이 특정 API를 더 효과적으로 학습할 수 있도록 일부 API에 대한 데이터 필터링 기법 적용

실험 및 결과

실험 환경

Training setting

- 최대 시퀀스 길이 : 1,024 토큰
- 배치 크기 : 128
- learning rate : 1 × 10^(-5)
- training steps : 최대 2,000 스텝 수행
- perplexity 평가 : 500 스텝마다 성능 평가 후 최적의 체크포인트 선택

Hardware

- GPU : NVIDIA A100 40GB × 8대
- 연산 최적화 : BF16 연산 적용하여 메모리 효율성을 증가 + DeepSpeed의 ZeRO-3를 사용하여 메모리 최적화

ZeRO-3는 대형 모델을 효과적으로 학습하기 위해 메모리 사용을 최적화하는 기법입니다.

ZeRO의 중 가장 효율적인 ZeRO-3을 사용해 모든 가중치, 그래디언트, 옵티마 상태를 분할하여 저장합니다. (추가설명)

실험 및 결과

실험 1. API를 사용하면 성능이 향상되는가

- API 호출을 추가하면 모델의 성능이 향상되는지 검증
- GPT-J (API 없음) vs Toolformer (API 학습) 비교
- 계산, 날짜 변환, 질문 응답, 번역 등 다양한 과제에서 비교

Task	GPT-J	GPT-J + C	Toolformer (disabled)	**Toolformer**	OPT (66B)	GPT-3 (175B)
계산	9.9%	9.3%	15.0%	**44.0%**	7.9%	19.8%
날짜 변환	3.9%	2.9%	5.9%	**27.3%**	1.3%	0.8%
질문 응답	31.9%	33.2%	34.9%	**53.5%**	30.1%	39.8%
기계 번역	15.2%	15.7%	19.8%	20.6%	0.3% (24.3%)	3.4% (**24.7%**)
Wikipedia	43.9%	45.6%	46.7%	48.8%	45.7%	**65.9%**

OPT, GPT-3의 번역 성능이 GPT-J와 비교해 낮은 것은, GPT-J가 비교적 multilingual한 데이터로 학습된 것으로 추측합니다.
OPT와 GPT-3의 Question을 영어로 번역한 후에는 괄호 내의 결과값을 가집니다.

실험 및 결과

실험 2. Toolformer는 언어 모델링 능력을 손상시키지는 않는가

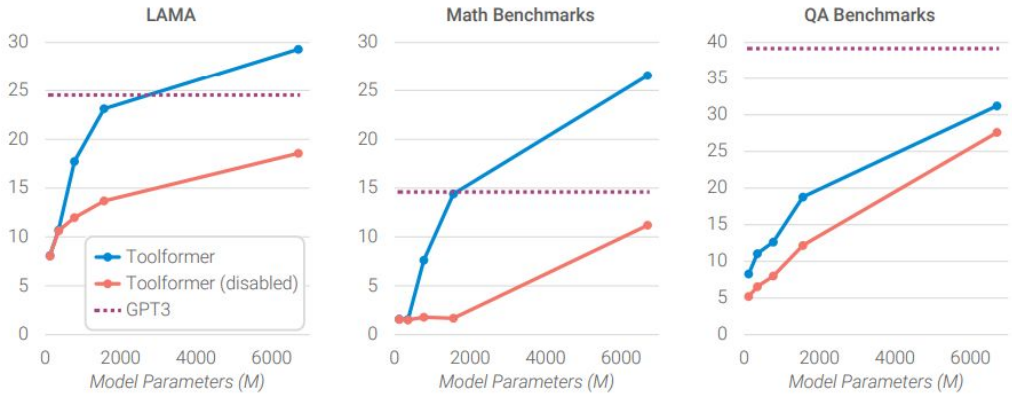
Model	WikiText	CCNet
GPT-J	**9.9**	10.6
GPT-J + CC	10.3	**10.5**
Toolformer (disabled)	10.3	**10.5**

실험에서 평가 지표로 사용된 **Perplexity**는 **낮을수록 언어 모델이 다음 단어를 더 정확하게 예측**할 수 있음을 의미합니다.
Toolformer를 적용한 모델에서 Perplexity가 증가하면 기본적인 언어 모델링 능력이 저하된 것입니다.

CCNet에서는 오히려 성능이 향상되었습니다. 즉, 언어 모델링의 능력에 나쁜 영향을 주지 않았습니다.
반면 WikiText에 대해서는 성능이 하락했습니다. 논문에서는 GPT-J의 학습 데이터가 WikiText와 유사하기 때문일 것으로 추측합니다.

실험 및 결과

실험 3. 모델 크기에 따라 API 사용 능력에 변화가 있는가



실험에서는 서로 다른 크기의 GPT 모델을 사용하여 Toolformer를 학습하였습니다. (125M, 350M, 760M, 6.7B)

평가 지표로는 API 호출 빈도, API 호출의 적절성, 성능 향상도를 비교하였습니다.

모델 크기가 커질수록 더 적절하게 수행하며, 실제 성능 향상에 기여하는 방식으로 사용하게 됩니다.

즉, Toolformer는 큰 모델일수록 더욱 효과적으로 API를 활용할 수 있음을 실험적으로 입증하였습니다.

결론 및 비평

결론 및 비평

Conclusion

- 언어 모델이 간단한 API 호출을 통해 다양한 외부 도구의 사용법을 자기지도(self-supervised) 방식으로 학습할 수 있는 모델 Toolformer를 소개하였습니다.
- 대규모로 샘플링된 API 호출들을 활용하여 미세조정(finetuning)되며, API 호출이 이후 토큰 예측의 퍼플렉서티(perplexity)를 감소시키는지 여부에 따라 유용한 호출을 선별합니다.
- 이러한 접근법을 통해 6.7B 규모의 GPT-J 모델을 기반으로 하는 **Toolformer**는 다양한 다운스트림 작업에서 제로샷(zero-shot) 성능을 상당히 향상시켰으며, 일부 작업에서는 훨씬 더 규모가 큰 GPT-3 모델을 능가하는 성과를 보였습니다.

Limitations

- Toolformer는 도구를 연쇄적으로 사용하는 것이 불가능합니다. 즉, 하나의 도구로부터 얻은 출력을 다른 도구의 입력으로 연결하여 연쇄적으로 사용하는 것이 불가능합니다.
- 언어 모델이 도구와 상호작용적(interactive)으로 소통하지 못한다는 한계를 가집니다. 예를 들면 검색 API에서 모델이 결과를 살펴보고(search results browsing) 검색 질의를 재구성(query refinement)하는 식으로 도구와 상호작용 할 수 없습니다.
- 도구에 따라서는 매우 샘플 비효율적(sample-inefficient)일 수 있습니다. 예를 들어, 백만 개 이상의 문서에서 계산기 API의 유용한 호출 사례는 겨우 수천 개 수준에 그쳤습니다.

FAQ

Q. 어떻게 API 사용이 유용한지를 스스로 판단하나요?

- API 호출 전후의 **손실(Loss) 차이**를 비교하여 API가 유용했는지를 판단합니다.
- API 결과를 포함한 문장과 포함하지 않은 문장을 비교하고, **손실이 더 줄어드는 쪽을 선택**합니다.
- 손실이 줄어들면 API가 유용하다고 판단하여 학습 데이터에 반영하고, 줄어들지 않으면 해당 API 호출을 버립니다.
- 잘못된 API 응답을 채택할 가능성은 존재합니다.

Q. API 호출이 아닌 일반적인 대괄호([ ])의 사용이 문제되지는 않나요?

- 대괄호([])는 API 호출의 시작과 끝을 명확하게 구분해주는 특수한 토큰(symbol) 역할을 수행합니다.
- 코드 작성에서 배열과 리스트 사용, 수식과 부연 설명 등에서 대괄호가 사용될 경우, 충돌(conflict) 가능성은 존재합니다.
- 더 명확하고 고유한 형태로 변경하거나(<<>>), 추가적인 검증 로직으로 문제를 해결할 수 있습니다.(API 이름 확인 등)

FAQ

Q. 두 개 이상의 단어는 어떻게 대체되나요?

- Toolformer는 API 결과를 단어 하나씩 대체하는 방식이 아니라 “전체 표현(phrase 단위)“을 고려하여 손실을 계산합니다.
- 우선, 문장에서 가장 불확실성이 높은 부분(확신도 낮은 단어들)을 분석하여 후보 영역을 찾습니다.
- 아래 그림과 같이 손실을 비교하여, 손실이 가장 낮은 "Joe Biden" 대체(2단어 교체)가 최적이라고 판단합니다.
- 즉, Toolformer는 API 결과의 길이가 기존 문장의 대체 대상과 다르더라도, 가장 자연스러운 대체 범위를 선택합니다.

대체 방식	예시	손실 비교
기존 문장 유지	"The President of the United States in 2021 was Donald Trump."	손실 L^- (높음)
1단어만 대체	"The President of the United States in 2021 was Joe Trump."	손실 L^+ (높음)
2단어 대체	"The President of the United States in 2021 was Joe Biden."	손실 L^+ (낮아짐)
3단어 대체	"The President of the United States in 2021 was President Joe Biden."	손실 L^+ (오히려 증가)