고려대학교 KOREA UNIVERSITY

Advancing LLM Reasoning

2026.01.08.
정다현

Natural Language Processing
& Artificial Intelligence

Reasoning with Sampling: Your Base Model is Smarter Than You Think

Aayush Karan[1], Yilun Du[1]

[1]Harvard University

ICLR 2026 Submission

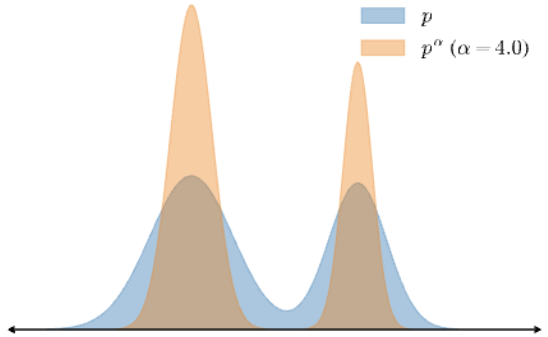
8/8/8/6

1. INTRODUCTION

Distribution Sharpening

“RL post-training 중에 나타나는 능력들은 베이스 모델에 존재하지 않았던 근본적으로 새로운 행동인가?”

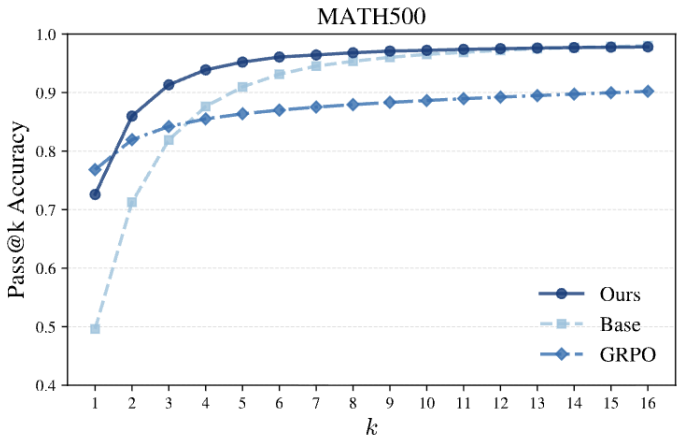
- RL post-trained distribution이 베이스 모델이 생성하기 힘든 새로운 추론 경로를 만들어내는 것이 아니라,
  단순히 베이스 모델 distribution의 특정 부분을 더 sharper하게(확률을 높게) 만든 버전이 아닌가 하는 의문임



1. INTRODUCTION

Distribution Sharpening

“RL post-training 중에 나타나는 능력들은 베이스 모델에 존재하지 않았던 근본적으로 새로운 행동인가?”



- 베이스 모델과 RL post-trained 모델의 pass@k 점수 비교 결과, k 값 클 경우 베이스 모델이 더 우수한 성능을 보이며, RL post-trained 모델은 생성 다양성이 저하됨
  - → RL 과정에서 나타나는 능력은 베이스 모델에 이미 존재함
  - → RL은 pass@k 성능을 단일 시도 성능으로 재분배하는 것으로 보임

1. INTRODUCTION

Sampling Algorithm

“베이스 모델에서 직접 샘플링하는 것만으로도 RL을 거친 모델과 대등한 단일 시도 추론 능력을 달성할 수 있지 않을까?”

- Power Distribution
  : RL 대신 베이스 모델 distribution을 sharpen하기 위함
- Markov Chain Monte Carlo을 활용한 Sampling Algorithm
  : power distribution을 근사한 샘플링 방식을 차용하여,
  베이스 모델 likelihood에 따라 subsequence를 반복하여 resample함

- 실험 결과에 따르면 베이스 모델에서 직접 샘플링하는 것만으로도 GRPO와 대등한 결과를 얻을 수 있음

2. MCMC SAMPLING FOR POWER DISTRIBUTIONS

Reasoning with Power Distribution

RL이 모델의 분포를 날카롭게 하는 과정이라면, 본 논문은 모델의 샘플링 분포를 직접 정의함으로써 동일한 효과를 내고자 함

- Power Distribution

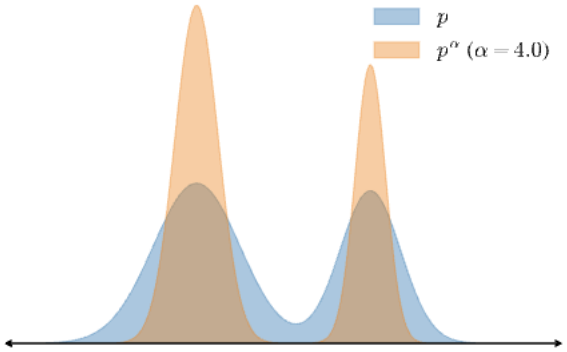
  : 원래 분포 p에 α를 거듭제곱해서 전체 확률 분포의 모양을 뾰족하게 만든 분포

  : 시퀀스의 상대적 likelihood를 높은 것은 더 높게, 낮은 것은 더 낮게 조정함

  : 문장 전체의 결합 확률에 지수를 적용하여

  미래에 전개될 수 있는 모든 경로를 고려하여 현재 토큰의 가중치를 결정함

$$p_{\text{pow}}(x_t|x_0 \dots x_{t-1}) = \frac{\sum_{x_{>t}} p(x_0, \dots, x_t, \dots, x_T)^\alpha}{\sum_{x_{\geq t}} p(x_0, \dots, x_t, \dots, x_T)^\alpha}.$$



2. MCMC SAMPLING FOR POWER DISTRIBUTIONS

Reasoning with Power Distribution

RL이 모델의 분포를 날카롭게 하는 과정이라면, 본 논문은 모델의 샘플링 분포를 직접 정의함으로써 동일한 효과를 내고자 함

- Power Distribution

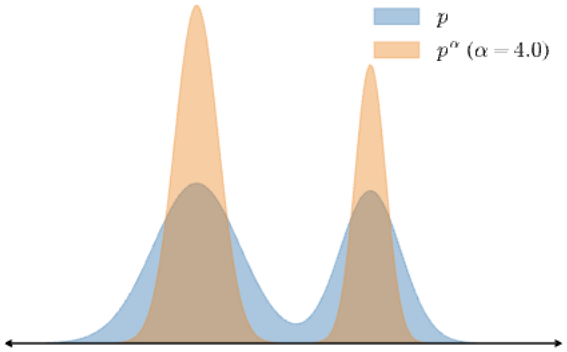
  : 원래 분포 p에 α를 거듭제곱해서 전체 확률 분포의 모양을 뾰족하게 만든 분포

  : 시퀀스의 상대적 likelihood를 높은 것은 더 높게, 낮은 것은 더 낮게 조정함

  : 문장 전체의 결합 확률에 지수를 적용하여

  미래에 전개될 수 있는 모든 경로를 고려하여 현재 토큰의 가중치를 결정함

$$p_{\text{pow}}(x_t|x_0 \dots x_{t-1}) = \frac{\sum_{x_{>t}} p(x_0, \dots, x_t, \dots, x_T)^\alpha}{\sum_{x_{\geq t}} p(x_0, \dots, x_t, \dots, x_T)^\alpha}.$$



→ Power distribution은 모든 가능한 시퀀스에 대해 계산하는 것이 불가능함

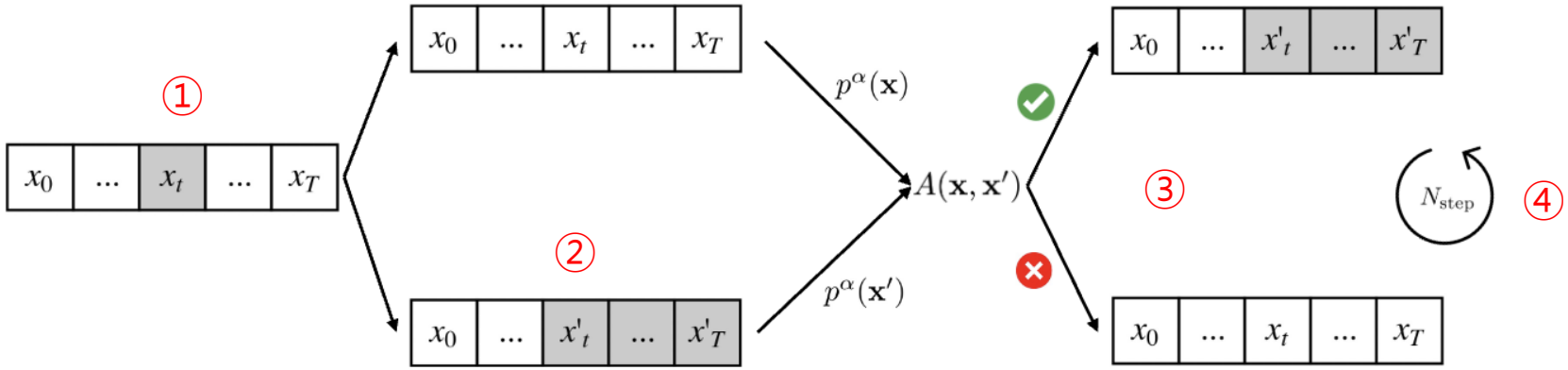
→ Markov Chain Monte Carlo (MCMC) 알고리즘을 사용하여 이 분포로 근사함

2. MCMC SAMPLING FOR POWER DISTRIBUTIONS

Metropolis-Hastings Algorithm

정규화되지 않은 확률 분포에서 근사 샘플링을 하는 기법

1. 초기화: 베이스 모델에서 표준 샘플링을 통해 초기 추론 경로 x를 생성함
2. 수정: 전체 텍스트 x 중에서 무작위로 하위 구간을 선택하여 이 구간을 지우고, 베이스 모델을 사용하여 해당 부분을 다시 생성하여 새로운 후보 x'를 만듦
3. 수락/거절: 다음의 수락 확률에 따라 새로운 후보를 받아들일지 결정함 $A(\mathbf{x}, \mathbf{x}^i) = \min\left\{1, \frac{p^\alpha(\mathbf{x}) \cdot q(\mathbf{x}^i|\mathbf{x})}{p^\alpha(\mathbf{x}^i) \cdot q(\mathbf{x}|\mathbf{x}^i)}\right\}$,
4. 반복: 이 과정을 정해진 반복 횟수만큼 수행함



2. MCMC SAMPLING FOR POWER DISTRIBUTIONS

Power Sampling with Autoregressive MCMC

MCMC 알고리즘의 단점인 시간 지연 문제를 해결하기 위해 블록 단위 점진적 생성 방식을 제안

- 문장을 한번에 고치는 것이 아닌 스텝을 밟아가면서 블록 단위로 순차적으로 샘플링
- 앞부분이 이미 어느 정도 최적화된 상태에서 뒷부분을 수정하기 때문에, 잘못된 문장에서 시작할 위험이 줄어들고 훨씬 빠르게 정답에 가까운 문장을 찾아냄
  - B까지 sampling → resampling N번
  - 2B까지 sampling → resampling N번
  - 3B까지 sampling → resampling N번
  - …

3. EXPERIMENTS

Main Results

Power Sampling을 적용한 베이스 모델은 RL을 거친 모델과 거의 대등하거나 심지어 능가하는 성능을 보여줌

- In-domain: MATH500의 경우 power sampling은 GRPO로 학습한 모델의 성능에 근접함
- Out-of-domain: power sampling이 GRPO 모델보다 일관되게 우수한 성능을 보임. 이는 강화학습이 특정 데이터셋에 과적합될 수 있는 반면, power sampling은 베이스 모델의 범용적인 지식을 그대로 활용하기 때문임

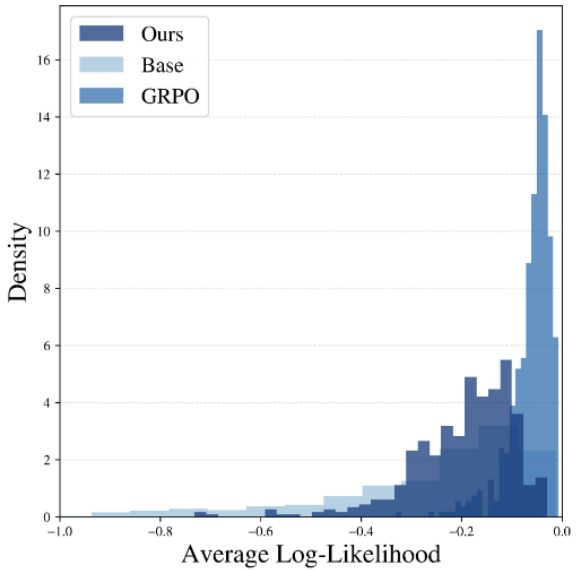
	MATH500	HumanEval	GPQA	AlpacaEval2.0
Qwen2.5-Math-7B				
Base	0.496	0.329	0.278	1.61
Low-temperature	0.690	0.512	0.353	2.09
Power Sampling (ours)	**0.748**	**0.573**	**0.389**	**2.88**
GRPO (MATH)	**0.785**	**0.537**	**0.399**	**2.38**
Qwen2.5-7B				
Base	0.498	0.329	0.278	7.05
Low-temperature	0.628	0.524	0.303	5.29
Power Sampling (ours)	**0.706**	**0.622**	**0.318**	**8.59**
GRPO (MATH)	**0.740**	**0.561**	**0.354**	**7.62**
Phi-3.5-mini-instruct				
Base	0.400	0.213	0.273	14.82
Low-temperature	0.478	0.585	0.293	**18.15**
Power Sampling (ours)	**0.508**	**0.732**	**0.364**	**17.65**
GRPO (MATH)	**0.406**	**0.134**	**0.359**	**16.74**

3. EXPERIMENTS

Reasoning Trace Likelihoods and Confidences

RL의 고질적인 문제 중 하나는 성능을 높이는 대신 답변의 다양성이 감소하는 Mode Collapse 현상

- Power sampling이 베이스 모델보다 높은 likelihood sequence를 sampling하면서, 동시에 diversity을 유지함
- GRPO는 베이스 모델 distribution을 강하게 sharpen하면서 diversity를 희생함



3. EXPERIMENTS

Reasoning Trace Lengths

RL의 또 다른 특징은 긴 형식의 추론으로, 샘플들이 더 긴 응답을 보이는 경향이 있다는 점임

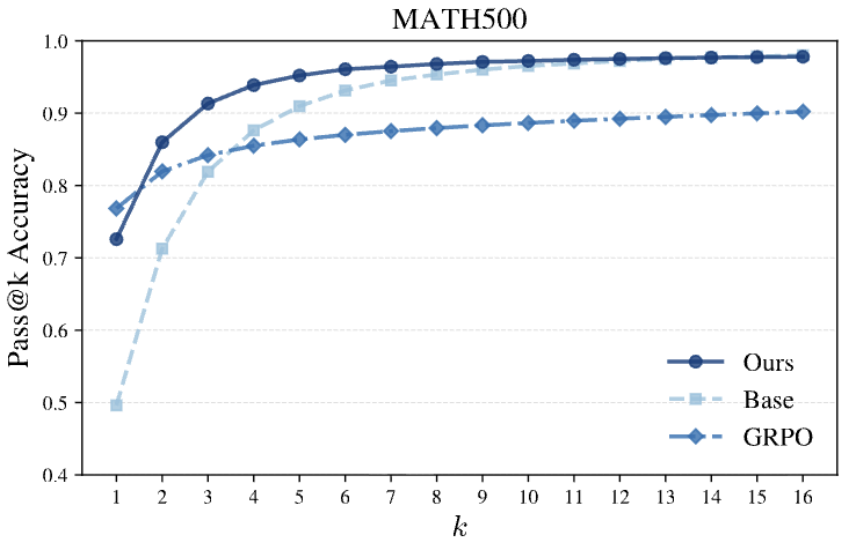
- MATH500 데이터셋에서 Qwen2.5-Math-7B (Base model)의 평균 답변 길이: 600 토큰
- GRPO 후 답변 길이가 길어짐: 671 토큰
- Power sampling은 명시적으로 더 긴 생성을 유도하지 않았음에도 불구하고 679 토큰이라는 유사한 평균 길이를 달성함

3. EXPERIMENTS

Diversity and Pass@k Performance

GRPO는 Pass@k 성능에서 k가 증가할수록 성능이 정체됨

- Power sampling은 k>1인 상황에서 GRPO의 성능을 강하게 능가함



4. CONCLUSION

Conclusion

베이스 모델이 RL 없이도 복잡한 추론을 수행할 수 있는 상당한 잠재력을 이미 갖추고 있음을 보여줌

- 제안하는 power sampling 알고리즘은 모델의 내재된 확률 분포를 활용하여 추론 시점의 계산량을 효과적으로 활용함
- 실험 결과, RL으로 훈련된 모델들과 대등하거나 그 이상의 성능을 기록함
- 고비용의 데이터 구축과 불안정한 강화학습 과정 없이도, 샘플링 전략만으로 추론 성능을 비약적으로 높일 수 있음
- RL의 고질적 문제인 diversity collapse를 피하면서도 높은 성능을 냄

- 한계
  - 지연 시간: MCMC 반복 과정을 거치기 때문에 표준 샘플링보다 생성 시간이 더 오래 걸림
  - 복합적 활용: 이미 강화학습된 모델에도 적용되어 추가적인 성능 향상을 이끌어낼 수 있을 것임

Training Large Language Models to Reason in a Continuous Latent Space

Shibo Hao[1,2,*], **Sainbayar Sukhbaatar**[1], **DiJia Su**[1], **Xian Li**[1], **Zhiting Hu**[2], **Jason Weston**[1], **Yuandong Tian**[1]

[1]FAIR at Meta, [2]UC San Diego
[*]Work done at Meta

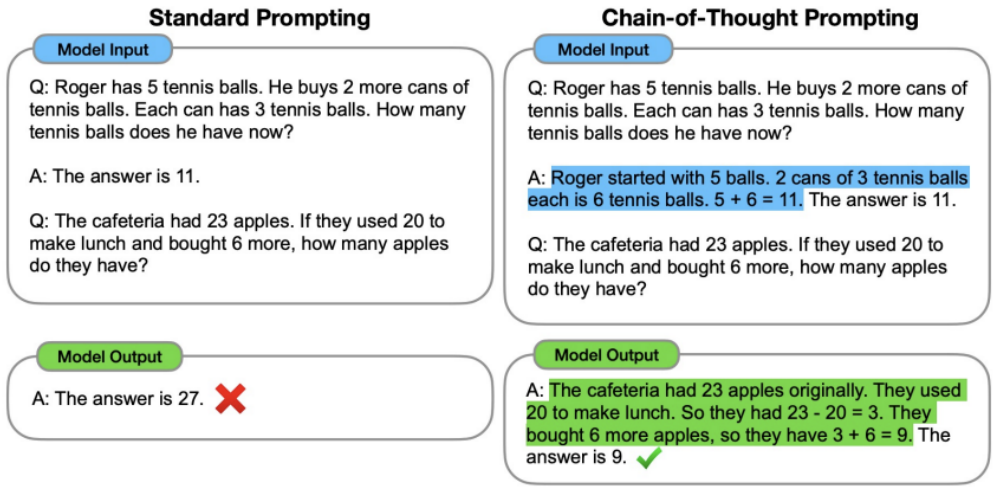
COLM 2025

1. INTRODUCTION

Chain-of-Thought

“언어 공간이 추론을 하기에 항상 최적인 장소인가?”

- LLM은 복잡한 문제를 풀 때 CoT 방식을 사용하여 중간 추론 과정을 텍스트(언어 토큰)로 출력함
- 자연어 토큰은 텍스트의 유창성을 유지하기 위해 실제 추론에는 기여하지 않는 불필요한 단어들을 포함하며, 모델이 한 번 특정 토큰을 생성하면 그 경로에 고정되어 버려서 유연한 사고나 탐색을 방해할 수 있음
- 인지 과학 및 뇌과학 연구에 따르면, 인간이 복잡한 추론을 할 때 반드시 언어 영역을 사용하는 것은 아니며, 머릿속으로 하는 생각은 언어화되기 전의 추상적인 상태일 수 있다는 점에 주목함

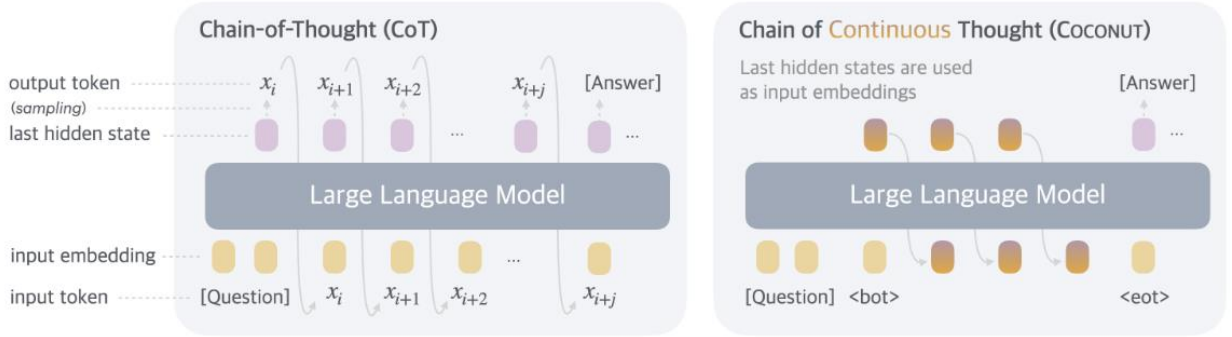


1. INTRODUCTION

Chain of Continuous Thought

LLM도 모든 추론을 텍스트화 하지 않고, 모델 내부의 Latent Space에서 연속적인 벡터 형태로 추론을 진행할 수 있음

- 기존의 CoT가 Hidden State → Token → Embedding → Next Hidden State의 과정을 거쳤다면,
  Coconut은 중간 토큰 생성 과정을 생략하고 Last Hidden State를 직접 다음 단계의 입력 임베딩으로 사용함
- 이를 통해 모델은 언어라는 틀에 갇히지 않고 연속적인 벡터 공간에서 추론을 이어감
- 텍스트 기반 CoT는 하나의 경로만 따라가는 경향이 있지만, latent space에서의 추론은 여러 가능성을 동시에 인코딩할 수 있어 BFS과 유사한 효과를 얻음
- 추론 과정에서 생성되는 토큰 수를 획기적으로 줄이면서도, 백트래킹이 필요한 복잡한 논리 문제에서 CoT보다 우수한 성능



2. COCONUT

Implicit CoT

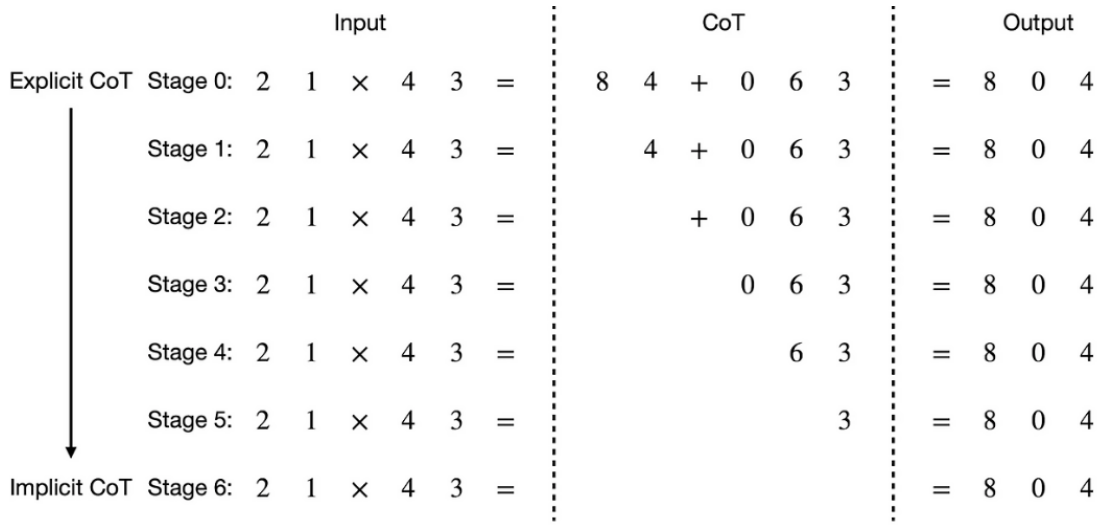
학습을 통한 reasoning 과정 내재화 기법

- Reasoning 텍스트를 생성하지 않고, 단계적으로 answer를 직접 생성하도록 학습함

  1. Question/Reasoning/Answer로 구성된 데이터셋으로 학습

  2. 일정 epoch마다 reasoning의 초기 k개 토큰을 생략하고 학습 진행

  3. 최종 학습 시 reasoning을 생략하고 직접 answer를 생성하도록 학습

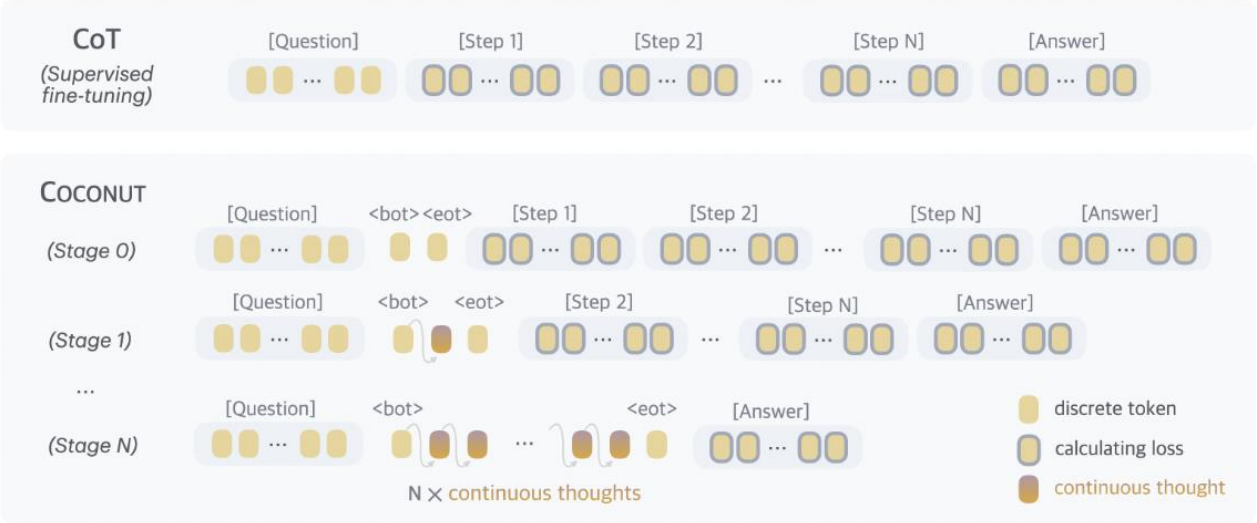


2. COCONUT

Chain of Continuous Thought

중간 토큰 생성 과정을 생략하고, 모델의 last hidden state를 직접 다음 단계의 입력 임베딩으로 사용

- Stage 0 (Standard CoT): 일반적인 텍스트 기반 CoT로 모델을 학습
- Stage 1 (Introduction of Latent Thought): 추론 과정의 첫 번째 스텝만 latent vector로 교체하고, 나머지 추론은 여전히 텍스트로 진행
- Stage k (Increasing Latent Steps)
  : 학습이 진행됨에 따라 스텝을 하나씩 제거하고,
  그 자리를 continuous vector로 채워 나감
- Final Stage
  : 모든 추론 과정은 latent space 내,
  모델은 최종 정답만을 텍스트로 출력함

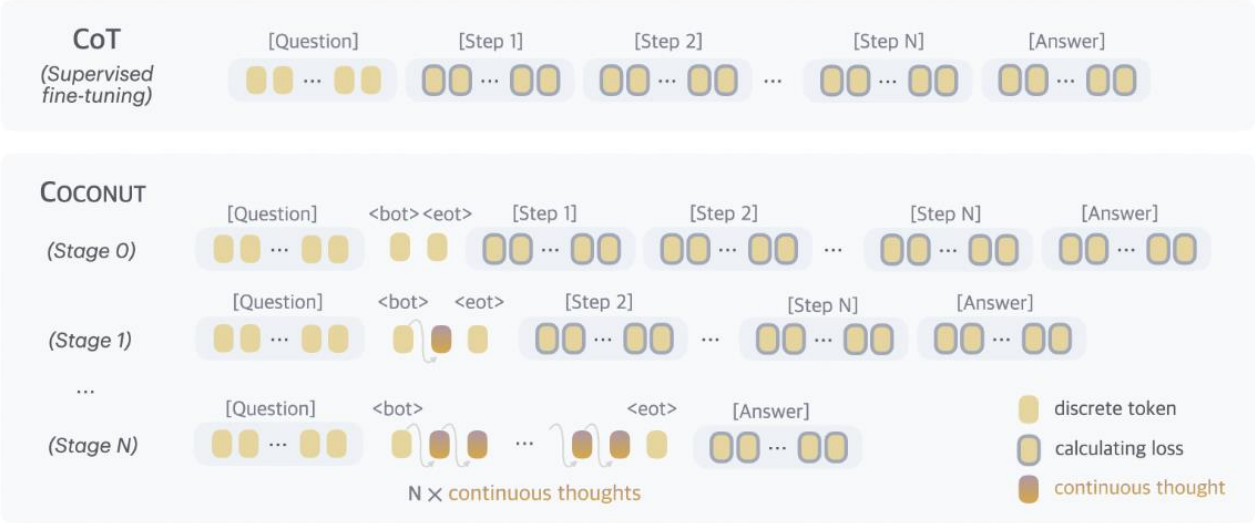


2. COCONUT

Chain of Continuous Thought

중간 토큰 생성 과정을 생략하고, 모델의 last hidden state를 직접 다음 단계의 입력 임베딩으로 사용

- Inference: 질문이 끝나면 바로 <bot> 토큰을 삽입하여 잠재 모드로 진입 → 언제 멈출 것인가? (<eot> 결정)
  - 방법 A: Latent vector를 보고 멈출지 결정하는 이진 분류기 학습
  - 방법 B: 항상 고정된 길이만큼 잠재 사고를 수행 (실험에서는 단순성을 위해 이 방법 사용)



3. EXPERIMENTS

Empirical Results with Coconut

Main Result

- GSM8K: 초등학교 수준의 수리 추론
- ProntoQA: 그래프 구조의 지식 체계를 활용한 논리 추론
- ProsQA: 복잡한 지식 그래프 기반 논리 추론 → 추론 시 planning 및 각 branch 별 평가 요구
- Pause Token: reasoning 대신 “<pause>” token을 활용

Method	GSM8k		ProntoQA		ProsQA	
	Acc. (%)	# Tokens	Acc. (%)	# Tokens	Acc. (%)	# Tokens
CoT	42.9 ±0.2	25.0	98.8 ±0.8	92.5	77.5 ±1.9	49.4
No-CoT	16.5 ±0.5	2.2	93.8 ±0.7	3.0	76.7 ±1.0	8.2
iCoT	30.0*	2.2	99.8 ±0.3	3.0	98.2 ±0.3	8.2
Pause Token	16.4 ±1.8	2.2	77.7 ±21.0	3.0	75.9 ±0.7	8.2
COCONUT (Ours)	34.1 ±1.5	8.2	99.8 ±0.2	9.0	97.0 ±0.3	14.2
- *w/o curriculum*	14.4 ±0.8	8.2	52.4 ±0.4	9.0	76.1 ±0.2	14.2
- *w/o thought*	21.6 ±0.5	2.3	99.9 ±0.1	3.0	95.5 ±1.1	8.2
- *pause as thought*	24.1 ±0.7	2.2	100.0 ±0.1	3.0	96.6 ±0.8	8.2

3. EXPERIMENTS

Empirical Results with Coconut

Main Result

- 일반 모델은 추론 단계를 생략하면 성능이 급격히 떨어지지만, Coconut은 성능 하락을 방어하며 효율성과 정확도 사이 균형 보여줌
- Coconut은 기존 CoT 대비 적은 수의 토큰을 생성함
- 복잡한 Planning이 필요할 때, implicit reasoning은 매우 도움이 됨
- 별도의 단계별 가이드 없이 질문과 정답만으로 잠재 추론을 배우게 한 경우, 추론 과정이 없는 일반 모델보다 나을 것이 없었음

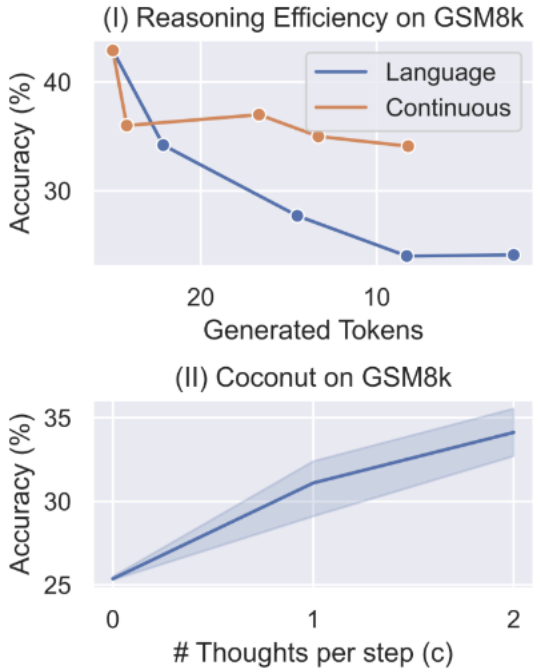
Method	GSM8k		ProntoQA		ProsQA	
	Acc. (%)	# Tokens	Acc. (%)	# Tokens	Acc. (%)	# Tokens
CoT	42.9 ±0.2	25.0	98.8 ±0.8	92.5	77.5 ±1.9	49.4
No-CoT	16.5 ±0.5	2.2	93.8 ±0.7	3.0	76.7 ±1.0	8.2
iCoT	30.0*	2.2	99.8 ±0.3	3.0	98.2 ±0.3	8.2
Pause Token	16.4 ±1.8	2.2	77.7 ±21.0	3.0	75.9 ±0.7	8.2
COCONUT (Ours)	34.1 ±1.5	8.2	99.8 ±0.2	9.0	97.0 ±0.3	14.2
- *w/o curriculum*	14.4 ±0.8	8.2	52.4 ±0.4	9.0	76.1 ±0.2	14.2
- *w/o thought*	21.6 ±0.5	2.3	99.9 ±0.1	3.0	95.5 ±1.1	8.2
- *pause as thought*	24.1 ±0.7	2.2	100.0 ±0.1	3.0	96.6 ±0.8	8.2

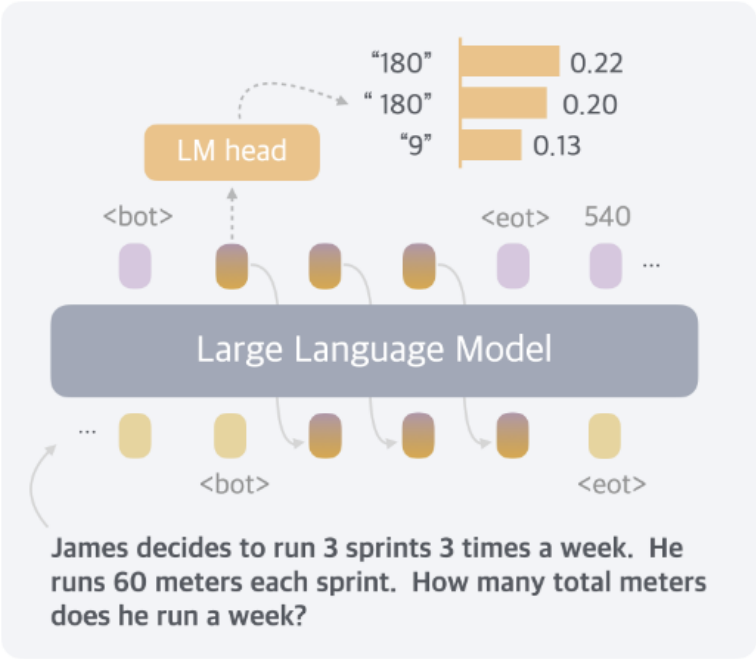
3. EXPERIMENTS

Empirical Results with Coconut

Main Result

- 토큰 절약: ProntoQA와 ProsQA에서 Coconut은 기존 CoT보다 적은 수의 토큰을 생성하면서도 더 높은 정확도를 달성함
- 잠재 벡터 수: 늘릴수록 모델의 성능 꾸준히 향상
- 내부 동작: 첫 번째 잠재 벡터를 해석해보면 계산에 필요한 중간 변수들이 포함되어 있는 경우가 많았음. 이는 잠재 벡터가 추론에 필요한 정보를 매우 효율적으로 압축하고 있음을 의미



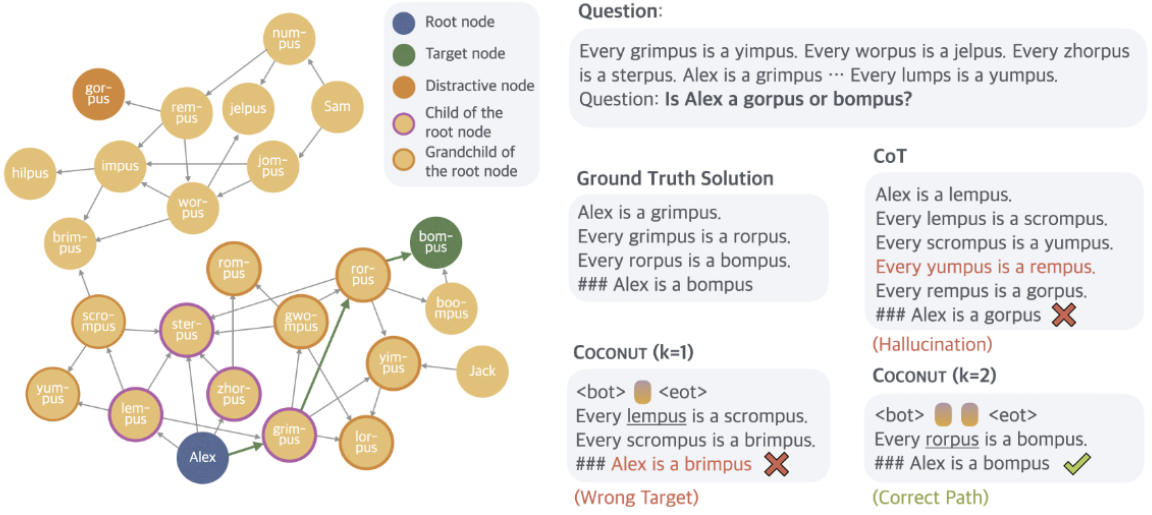


3. EXPERIMENTS

Continuous Space Enables Latent Tree Search

잠재 추론의 시각화 (Probing)

- 잠재 사고 단계에서 강제로 텍스트를 생성하게 하여, 모델이 해당 시점에 어떤 선택지들을 고려하고 있는지 확률 분포를 확인
- “Alex”라는 노드에서 시작하여 다음 단계인 {“lempus”, “sterpus”, “zhorpus”, “grimpus”} 중 어디로 갈지 결정하는 상황
- CoT (Greedy): 첫 단계에서 가장 확률이 높은 “lempus” (0.33)를 즉시 선택하고 그 길로만 감
- Coconut (BFS): 첫 단계에서는 “lempus”가 높았음에도 불구하고, 두 번째 사고를 거치며 “grimpus”의 자식 노드인 “rorpus”에 가장 높은 확신(0.87)을 갖게 됨
- Latent space은 여러 경로를 동시에 고려하다가,
  깊이 생각 후 정답 경로를 찾는 tree search 능력을 보여줌



4. CONCLUSION

Conclusion

Continuous latent space에서의 추론을 위한 새로운 패러다임을 제시하여 추론 능력을 향상시킴

- Coconut은 latent space 내에서의 연속적인 연산만으로도 복잡한 논리 전개가 가능함을 증명
- Latent space에서 추론은 텍스트 생성에 드는 비용을 줄이면서도, 트리 탐색과 같은 고차원적인 사고를 가능하게 하여 성능 향상
- 말하는 법을 배우는 것과 생각하는 법을 배우는 것을 분리할 수 있는 가능성을 제시

- 한계
  - 아직은 잠재 사고를 가르치기 위해 텍스트 CoT 데이터가 가이드 역할을 해줘야 함
  - GSM8k 결과, 정확한 수치 연산에서 논리 문제만큼의 압도적인 성능 향상은 나타나지 않음
  - 현재의 잠재 사고 방식은 n번의 사고 단계만큼 n+1번의 순차적인 forward pass를 수행해야 하여 학습 및 추론 비용 소요

Thank you

Q&A