Automatic Prompt Optimization

2025.09.25

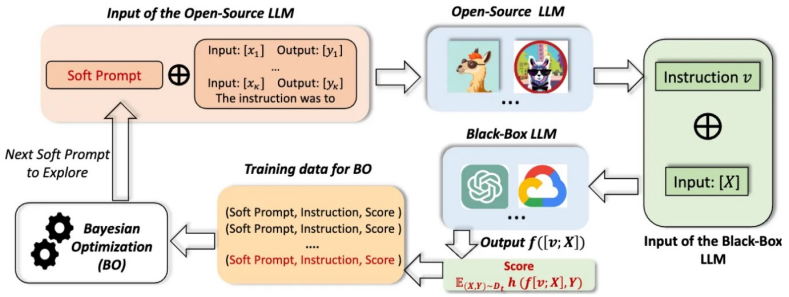
김명진

Introduction

Automatic Prompt Optimization?

- Prompt 설계는 LLM의 성능을 결정짓는 핵심 요소 중 하나.
- 그러나 성능을 극대화하는 프롬프트를 manual하게 설계하는 과정은 수많은 trial-and-error를 요구.
  비효율적이고 domain-specific 하다.
- 이러한 한계를 극복하기 위해 다양한 automatic prompt optimization 기법들이 제시됨.

주요 Automatic Prompt Optimization 방법론 분기



- Gradient-based Methods vs Gradient-free Methods
- Continuous Methods(Soft Prompt) vs Discrete Methods(Hard Prompt) vs Feedback-based Methods
- Instruction Optimization vs Exemplar Optimization vs Both

PromptWizard: Optimizing Prompts via Task-Aware, Feedback-Driven Self-Evolution

Eshaan Agarwal, Raghav Magazine, Joykirat Singh, Vivek Dani, Tanuja Ganu, Akshay Nambi
Microsoft Research India
Corresponding author: akshayn@microsoft.com

ACL 2025 Findings

Motivation & Contribution

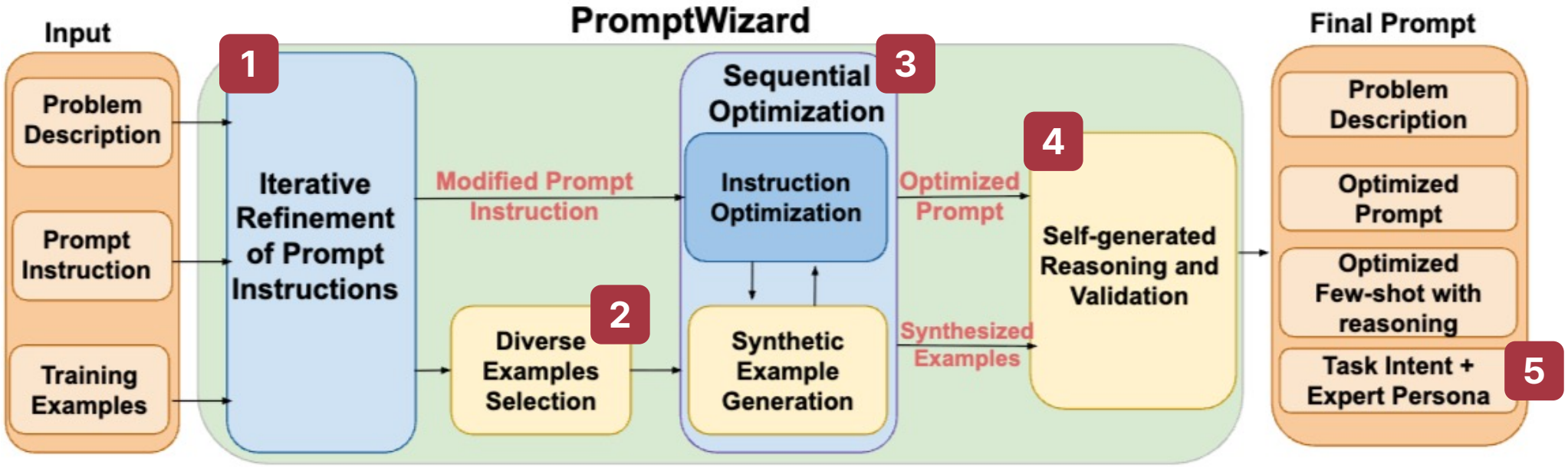
Motivation

- 기존 연구의 한계

1. Gradient-based Methods
   - White-box 모델에 한정적
2. Continuous Methods
   - 추가적인 NN 훈련이 요구되어 비용이 증가함
   - Effectiveness가 open-source 모델 및 task complexity에 dependent
3. Discrete Methods
   - Inefficient, suboptimal exploration
   - 주로 instruction optimization에만 초점
4. Feedback-based Methods
   - 계산 비용이 매우 많이 듦

Methodology

- PromptWizard 전체 구조



Methodology

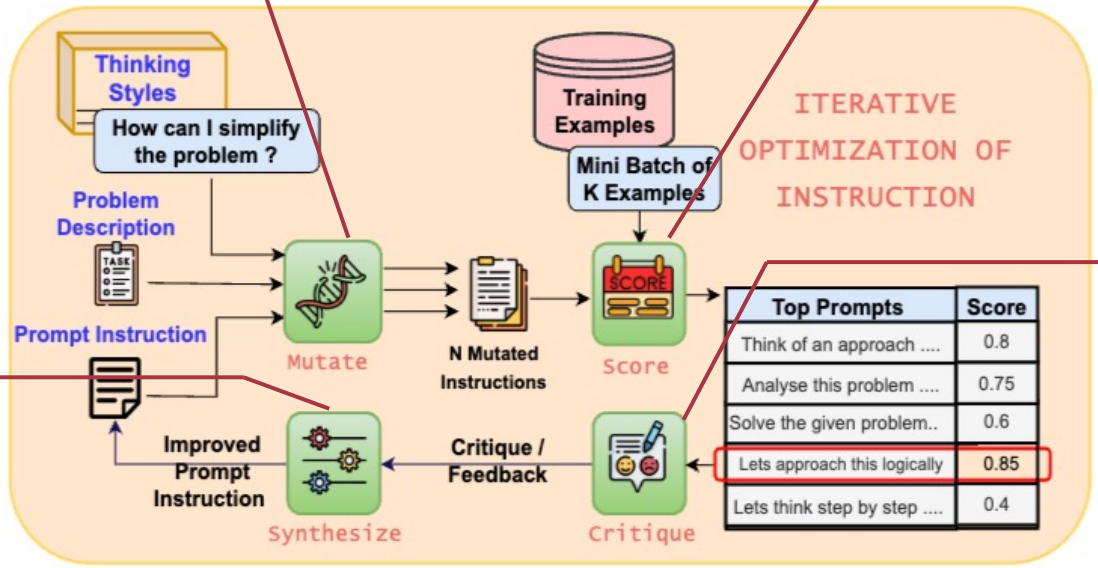
1. Iterative Refinement of Prompt Instructions

1. Mutate Component

- Manual 하게 작성한 initial prompt description을 LLM을 통해 변이
- 이때 pre-define된 thinking styles 사용

TOP 3 MUTATED PROMPTS :
" Let's devise a step-by-step experiment to reach the solution of this mathematical problem."
" Let's simplify and tackle this mathematical problem step by step to make it easier to solve. "
" Let's list out ideas and apply them one by one, thinking step by step, to solve mathematical problems. "

TOP SCORED PROMPT:
"Provide question answering on mathematical school grade questions that require multi-step reasoning. The problems should take between 2 and 8 steps to solve, and solutions primarily involve performing a sequence of elementary calculations using basic arithmetic operations (+ - / *) to reach the final answer.\nLets think step by step to arrive at the solution of this mathematical problem",



2. Scoring Component

- 생성된 mutated prompts의 성능 평가, 5개 training data batch에 대한 성능으로 측정
- 정통적 metric (ex. F1 score) 기반 평가 또는 llm-as-evaluator 사용

3. Critique Component

- 2번 단계에서 가장 성능이 높은 mutated prompt가 critique component로 넘어옴
- 해당 프롬프트 적용 시 맞은 문제 / 틀린 문제를 분석하며 critique 생성

CRITIQUE/FEEDBACK :
Firstly, the instruction doesn't specify the need for the agent to understand the problem context, such as interpreting relationships. Secondly, the instruction lacks clarity on the agent's ability to handle percentages and real-world scenarios. Understanding sequences of actions and their impact are crucial. Lastly, the instruction doesn't mention the agent's ability to handle time conversions, such as converting an hourly rate to a per-minute rate.

4. Synthesize Component

- 3번 단계에서 생성된 critique을 기반으로 best mutated prompt refine

REFINED PROMPT:
Provide question answering on mathematical school grade problems that require multi-step reasoning and understanding of the problem's context. The problems should take between 2 and 8 steps to solve, and solutions primarily involve performing a sequence of elementary calculations using basic arithmetic operations (+ - / *), handling percentages, and converting them into numbers. The agent should be able to interpret real-world scenarios and understand the implications of the problem, including handling time conversions. The agent should also be able to follow a sequence of actions and their impact on the final answer. Let's think step by step to arrive at the solution of this mathematical problem.

Methodology

2. Identification of Diverse Examples

LLM이 다양한 정보를 습득하는 데에 prompt의 효과를 높이기 위해 diverse examples 식별

2-1. Train dataset에서 무작위로 25개의 example 선택

2-2. Scoring Component를 사용하여 2-1 단계에서 뽑은 25개 example 각각에 대해 iteration을 돌며 평가 진행
평가 결과에 따라 example들을 Positive 또는 Negative로 분류한다

2-3. Negative sample을 필요한 개수만큼 얻었으면 iteration 종료

2-4. 전체 25개 example에 대해 5회 iteration을 돌았음에도 Negative sample을 필요한 만큼 확보하지 못했다면 부족한 수만큼 초기 example pool에서 random하게 선택

Methodology

3. Sequential Optimization of Prompt Instructions and Few-Shot Examples

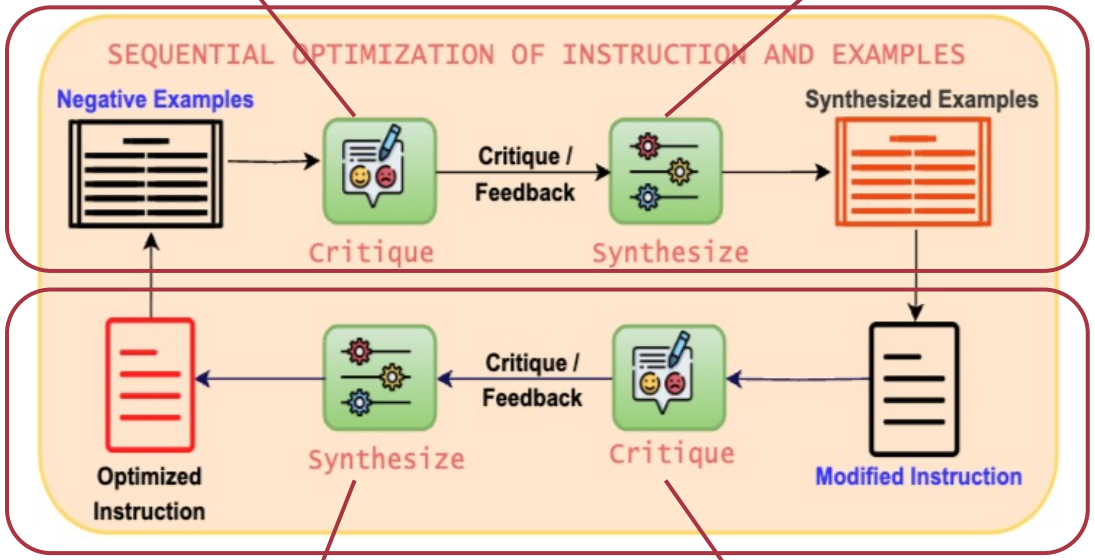
Critique Component

- 이전에 선택된 example들을 분석하여 error-driven self-reflection 수행
- 어떤 example이 더 필요한지 등에 대한 상세한 피드백

CRITIQUE/FEEDBACK :
This example is simple and straightforward, involving basic multiplication and addition. It's relevant and relatable to students. However, it could be improved by adding a bit more complexity, such as introducing fractions or percentages.\n\n

EXAMPLES:
[Question]: Albert is wondering how much pizza he can eat in one day. He buys 2 large pizzas and 2 small pizzas. A large pizza has 16 slices and a small pizza has 8 slices. If he eats it all, how many pieces does he eat that day?
[Answer]: He eats 32 from the largest pizzas because 2 x 16 = <<2*16=32>>32. He eats 16 from the small pizza because 2 x 8 = <<2*8=16>>16. He eats 48 pieces because 32 + 16 = <<32+16=48>>48. The answer is 48.

Few-shot example optimization



Synthesize Component

- Critique Component의 피드백을 기반으로 새로운 synthetic example들 합성

SYNTHETIC EXAMPLES:
[Question]: Albert buys a pizza that is cut into 8 slices. He eats 3 slices and gives 2 slices to his friend. He then buys another pizza that is cut into 12 slices and eats half of it. How many slices of pizza does Albert have left?
[Answer]: From the first pizza, Albert has 8 - 3 - 2 = <<8-3-2=3>>3 slices left. From the second pizza, Albert eats 12 / 2 = <<12/2=6>>6 slices, so he has 12 - 6 = <<12-6=6>>6 slices left.\nIn total, Albert has 3 + 6 = <<3+6=9>>9 slices of pizza left. The answer is 9.

Synthesize Component

- Critique Component의 피드백을 기반으로 instruction prompt 합성

Prompt optimization

Critique Component

- 새로 생성된 synthetic example들을 사용하여 현재 prompt 평가
- 현재 prompt 개선점 식별

Methodology

4. Self-generated Reasoning and Validation

Reasoning chain이 LLM의 문제 해결 능력에 도움을 준다는 가설 하에 각 few-shot example들에 대해 CoT 경로 생성

4-1. Reasoning Component

최종 선택된 few-shot example들 각각에 대해 구체적인 reasoning path 생성

4-2. Validate Component

LLM을 사용하여 생성된 reasoning path의 coherence 및 relevance 확인

5. Task Intent and Expert Persona

Problem description을 기반으로 Synthesize Component가 생성

EXPERT IDENTITY :
You are a mathematics educator with a deep understanding of elementary and middle school mathematics. You are experienced in teaching multi-step problem-solving techniques and have a knack for breaking down complex problems into manageable steps. Your expertise lies in basic arithmetic operations such as addition, subtraction, multiplication, and division. You can provide clear, step-by-step solutions to mathematical problems that require multi-step reasoning. You are patient and thorough, ensuring that each step is clearly explained and understood. Your ability to simplify complex problems and guide students through the problem-solving process makes you an excellent resource for answering school-grade mathematical questions.

INTENT : Mathematical Reasoning, Multi-step Problem Solving, Basic Arithmetic Operations, Data Analysis, Solution Verification

Experiments & Results

Datasets

- Language understanding scenarios: BIG-Bench Instruction Induction (BBII) *(19)*
- Arithmetic reasoning datasets: GSM8k, AQUARAT, SVAMP *(3)*
- Domain specific tasks: BigBench Hard (BBH) 일부 *(23)*

Baselines

- SOTA discrete, continuous prompt optimization methods

  Discrete: PromptBreeder, EvoPrompt, APE

  Continuous: Instinct, InstructZero

Models

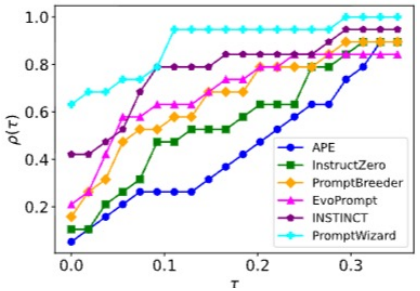
- GPT 3.5 Turbo / GPT-4

Experiments & Results

Main Results

BBII 성능

Task	APE	InsZero	PB	EvoP	Instinct	PW	Instinct	PW
LLM: GPT3.5Turbo	Zero-shot setting						One-shot setting	
antonyms	0.64	0.83	0.80	0.80	**0.85**	0.56	**0.85**	0.78
auto-categorization	0.25	0.26	0.22	0.26	0.25	**0.28**	0.30	**0.40**
cause and effect	0.57	0.81	0.75	0.83	0.59	**0.88**	0.63	**0.92**
common concept	0.07	0.09	0.10	0.12	**0.21**	0.10	**0.25**	0.19
diff	0.67	0.69	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
informal to formal	0.57	0.53	0.58	**0.62**	0.55	**0.62**	0.52	**0.56**
letters list	**1.00**	0.59	0.99	**1.00**	**1.00**	0.95	**1.00**	**1.00**
negation	0.75	0.78	0.77	0.79	**0.82**	0.73	**0.86**	0.84
object counting	0.36	0.36	0.34	0.12	0.34	**0.60**	0.36	**0.52**
odd one out	0.63	0.61	0.64	0.65	0.70	**0.78**	0.63	**0.92**
orthography starts with	0.46	0.51	0.56	0.60	0.67	**0.75**	0.67	**0.92**
rhymes	0.16	**1.00**	0.54	0.61	**1.00**	0.89	0.75	**0.90**
second word letter	0.75	0.43	0.57	0.41	0.10	**0.93**	0.24	**0.99**
sentence similarity	0.00	0.00	0.01	0.28	0.14	**0.29**	0.16	**0.30**
sum	0.67	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
synonyms	0.36	0.28	0.36	0.14	0.31	**0.37**	0.37	**0.44**
taxonomy animal	0.35	0.72	0.72	0.72	0.86	**0.92**	0.90	**0.94**
word sorting	0.33	0.31	**0.56**	0.52	0.51	**0.56**	0.62	**0.74**
word unscrambling	0.44	0.55	0.61	0.60	**0.63**	0.52	**0.58**	**0.58**
#best performing tasks	1	2	3	4	8	**13**	7	**16**



Arithmetic Tasks 성능

Dataset	GSM8k	AQUARAT	SVAMP
Approach	Zero-shot with GPT3.5Turbo		
InsZero	74.2	54.3	79.5
Instinct	74.5	54.7	81
APO	25.7	20.1	75.2
PromptAgent	68.8	56.7	78.7
DSPy	78.2	55.1	77
PW	**90**	**58.2**	**82.3**

BBH 성능

Dataset	BBH (23)
Approach	Accuracy
APE	71.85
EvoP	75.03
PW	**88.1**

- 같은 모델을 사용한 SOTA prompt optimization method들과 비교했을 때 zero-shot 성능이 우수함을 확인

  => BBII 데이터셋의 경우 전체 19개 task 중 13개에서 1등을 차지

  => Arithmetic datasets, BBH에서 역시 우수한 성능

	API calls	IO Tokens	Total tokens	Cost ($)
Instinct InsZero	1730	67	115910	0.23
PB	18600	80	1488000	2.9
EvoPrmpt	5000	80	400000	0.8
PW	69	362	24978	0.05

BBH Cost Analysis

Arithmetic Tasks Cost Analysis

	API calls	IO Tokens	Total tokens	Cost ($)
APO	8292	94	779448	1.55
PromptAgent	2160	618	1334880	2.67
DSPy	1199	238	285362	0.57
PW	129	190	24510	0.049

- Feedback-driven guided exploration (critique-and-synthesis process)을 통해 기존 prompt optimization method들 대비 확연히 낮은 API call 수 기록

Experiments & Results

Ablation Study

1. Training data가 scarce한 상황에 대한 가정.

Datasets	5 (eg)	25 (eg)
MMLU	80.4	89.5
GSM8k	94.0	95.4
Ethos	86.4	89.4
PubMedQA	68.0	78.2
MedQA	80.4	82.9
Average	**81.9**	**87.0**

2. 더 작은 모델 사용에 대한 고려

Datasets	Ll-70B	GPT-4
GSM8k	94.6	95.4
Ethos	89.2	89.4
Average	**91.9**	**92.4**

Llama-70b를 사용하여 prompt generation, GPT-4 통해 inference

Models	With PW	w/o PW
GPT-4	**95.4**	92
GPT3.5	**75.6**	57.1
Ll-70B	**90.2**	56.8

GSM8k 데이터셋에 대한 Llama-70b, GPT-3.5, GPT-4 성능 비교

=> PromptWizard의 일관된 성능 향상을 주장

그치만 Llama-3-8b 등과 같이 더 작은 모델을 사용했을 때는 성능이 크게 하락했다고 report...

Ablation	Accuracy(%)
Stage 1 : Only instruction optimization	62
Stage 2 : Joint optimization of instruction and examples (zero shot prompt)	75
Stage 2 (few shot prompt)	79
Stage 2 (synthetic few shot prompt)	80
Stage 1 + Stage 2 (zero shot prompt)	83
Stage 1 + Stage 2 (few shot prompt)	86
Stage 1 + Stage 2 (synthetic few shot prompt)	88
Stage 1 + Stage 2 (few shot prompt) + CoT reasoning	90
Stage 1 + Stage 2 (synthetic few shot prompt) + CoT reasoning	**95**

GSM8k, 2-shot setting

Discussion & Conclusion

- 여러므로 의뭉스러운 부분들이 많은 논문임은 사실..
  - Instruction prompt와 few-shot example의 sequential한 최적화를 최대 contribution 중 하나로 내걸었으면서 정작 report한 experiment result 대부분은 zero-shot based..
  - 기본적으로 크기가 큰 모델들에 대해서만 고려, 크기가 작은 모델들 (30b 이하?)에 대한 성능은 전혀 report되지 않음
  - 모델이 생성한 CoT 등에 대한 예시 역시 없음
- 그치만 최신 discrete prompt optimization 중 가장 가능한 모든 내용을 꽉꽉 채워넣은 논문이 아닌가 싶기도 함

GREATER: GRADIENTS OVER REASONING MAKES SMALLER LANGUAGE MODELS STRONG PROMPT OPTIMIZERS

Sarkar Snigdha Sarathi Das[†] **Ryo Kamoi**[†] **Bo Pang**[◇] **Yusen Zhang**[†]
Caiming Xiong[◇] **Rui Zhang**[†]
[†]The Pennsylvania State University [◇]Salesforce Research
{sfd5525, rmz5227}@psu.edu

ICLR 2025

Motivation & Contribution

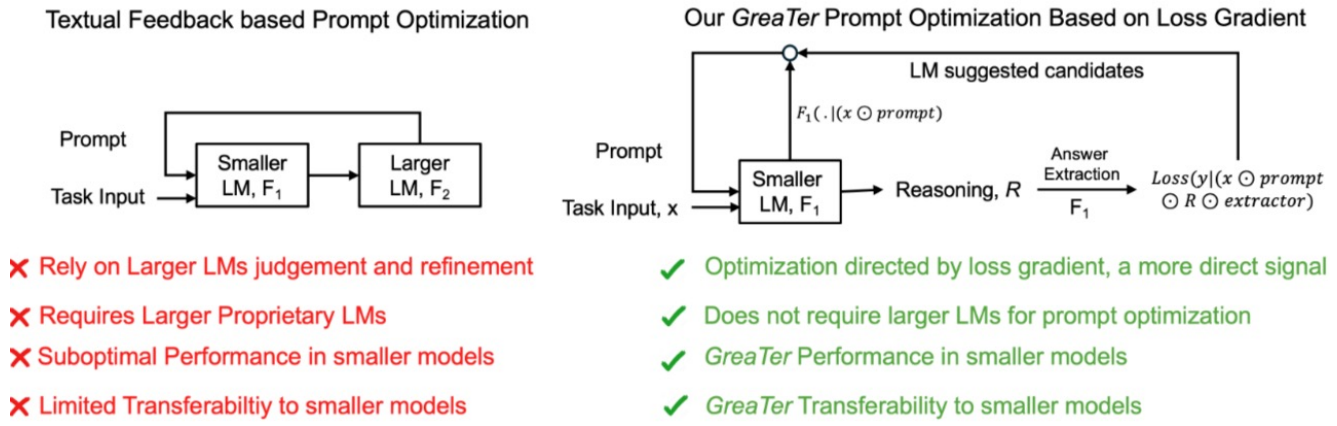
Motivation

- 최신 automatic prompt engineering 연구들은 주로 text-gradient, 즉 feedback-based method에 초점을 둔 경우가 많음

  => 해당 방법론들은 대체로 computationally expensive, stronger LLM들을 사용해 small, efficient LLM들의 실패 요인을 분석해서 text 형태의 feedback을 생성하고 이를 기반으로 prompt를 조정하는 방식으로 진행됨

  => 성능 향상에는 큰 기여를 보였으나 거대 모델에 대한 의존성이 primary limitation으로 남아있었음

  => 게다가 소형 LM들은 이런 textual gradient를 생성하는데 incapable하다는 사실이 최신 연구에서 밝혀지면서 large model들에 대한 의존성 문제가 더욱 부각됨



Problem Definition

Given 1) a language model f_{LLM}, 2) a small representative task dataset $D_{task} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$

$$p^* = argmax_p \sum_{(x,y)\in D_{task}} m(f_{LLM}(x; p), y)$$

이때 $f_{LLM}(x; p)$는 prompt p를 사용했을 때 input x에 대한 task language model f_{LLM}의 output

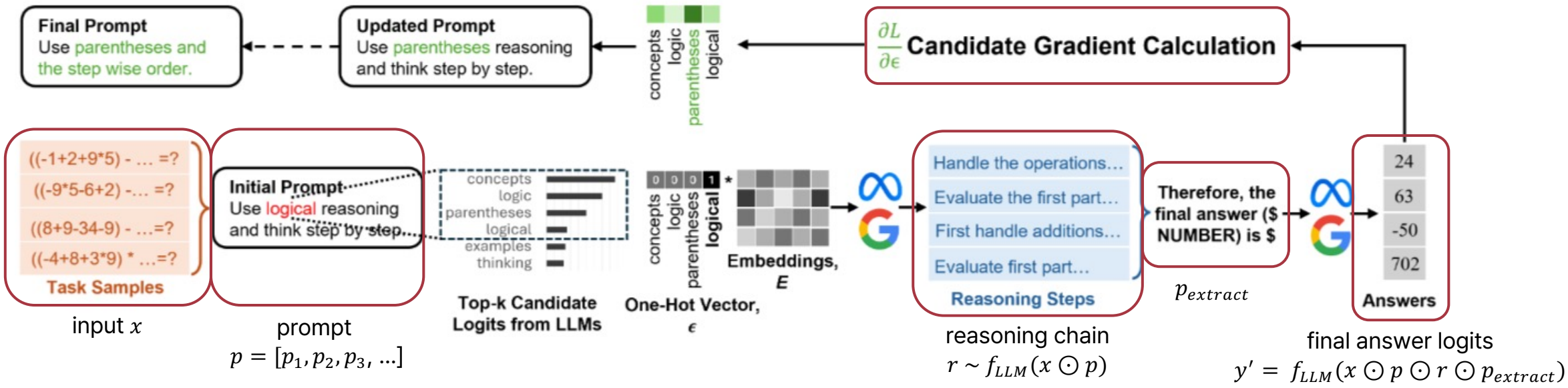
좋은 프롬프트란 language model이 valid한 answer에 성공적으로 도달할 수 있도록
사고의 방향성을 잘 guide 해주는 것.

Methodology

- GREATER 구조

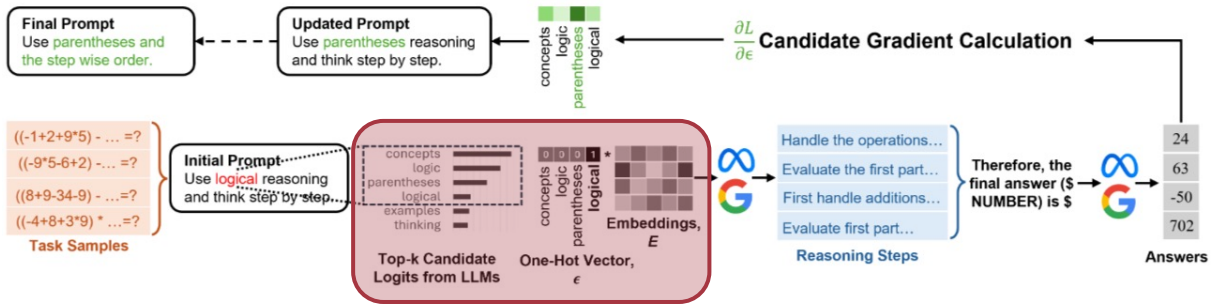
Optimization applied sequentially in each token position p_i

$$L = L_{CE}(f_{LLM}(x \odot p \odot r \odot p_{extract}), y)$$



Methodology

- Prompt Token Candidate Proposal



목표: prompt의 특정 위치 i에 위치하는 token p_i 최적화

1. f_{LLM} 기반 Top-k token 추출

Task language model(f_{LLM})의 input sample x_j와 이전 프롬프트 토큰들에 대한 conditioned probabilities를 활용하여 해당 위치에 대한 후보 토큰을 제안.

$$cand_{i,j} = top - k\ [f_{LLM}(\cdot\ |x_j \odot p_1, p_2, \dots, p_{i-1})]$$

2. 여러 샘플 통합

Random하게 sample한 q개 input($D_q \subset D_{train}$)에 대해 $cand_{i,j}$ 계산.

이후 모든 $cand_{i,j}$을 통합하여 위치 i에 대한 토큰 후보 집합 $candidates_i$를 구성.

$$candidates_i = \bigcap_{x_j \in D_q} cand_{i,j}$$

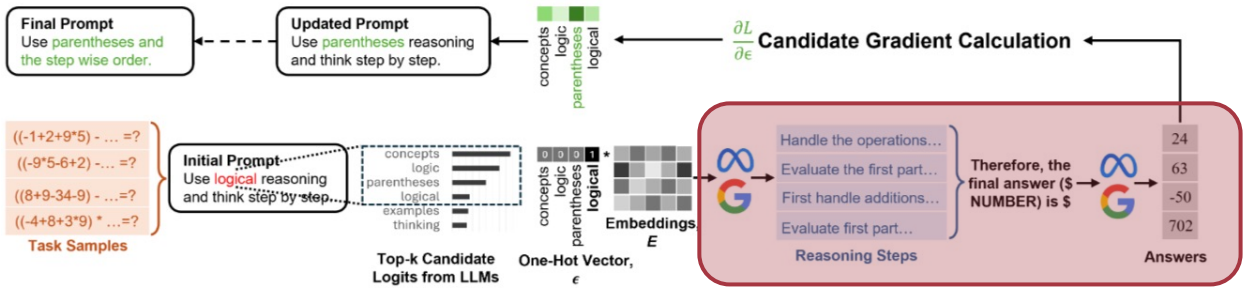
3. One-hot token indicator 활용

Candidate 개수만큼의 길이를 가진 one-hot token indicator ϵ_i 생성. 이때 p_i에 해당하는 값만 1.

원래 LM embedding table에서 ϵ_i에 해당하는 행만을 추출, 부분 embedding table E 구성. 해당 테이블을 ϵ_i와 곱하여 f_{LLM}에 입력으로 전달.

Methodology

- Reasoning Generation and Extraction



기존 연구의 문제점:
loss 및 gradient을 계산하는 가장 간단한 방법은 $f_{LLM}(x \odot p)$,
즉 모델의 output과 ground truth를 직접 비교하는 것.
하지만 이는 reasoning path에 대한 고려가 전혀 없는 접근.
현대 language model들에게 가장 요구되는 능력은 correct answer를 도출하기 위해 복잡한 reasoning path를 생성하는 것.

1. Reasoning 생성

Language model이 final answer를 derive하기 위한 자신의 reasoning 생성.

$r = f_{LLM}(cand_{i,j} = x \odot p)$

2. 정답 logit 추출

정확한 loss 계산을 위해 생성된 추론 r을 통해 최종 답변 logit을 추출해야 함.

이를 위해 extractor prompt $p_{extract}$을 추가하여 모델 응답 생성.

$\hat{y} = f_{LLM}(x \odot p \odot r \odot p_{extract})$

Methodology

- Gradient Over Reasoning Driven Candidate Selection

1. Cross-Entropy Loss

$$\mathcal{L}_{\text{CE}} = \text{cross\_entropy}(\hat{y}, y)$$

2. Perplexity Regularization

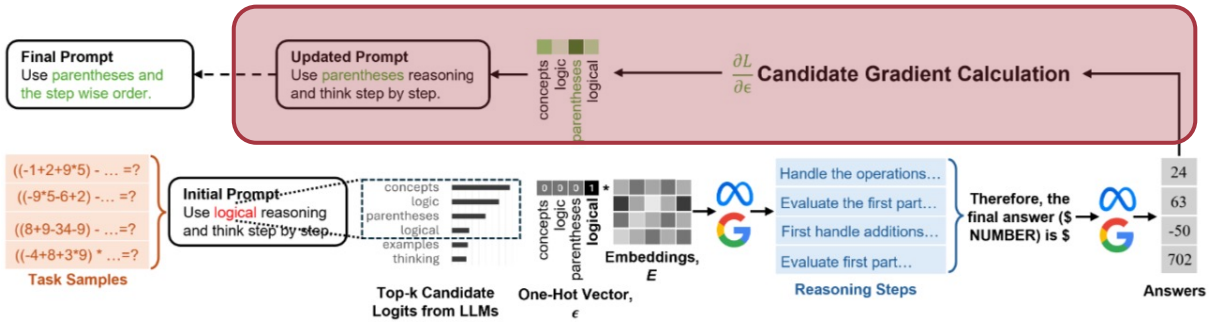
$$\mathcal{L}_{\text{perpl}} = \exp\left(-\frac{1}{|p|}\sum_{i=1}^{|p|}\log f_{\text{LLM}}(p_i \mid x, p_{<i})\right)$$

3. Final Loss Computation

$$\mathcal{L} = \mathcal{L}_{\text{CE}} + \lambda\mathcal{L}_{\text{perpl}}$$

4. Gradient 기반 token replacement

$$p_i = \text{token}[\arg\max_{\epsilon_i}(-\frac{\partial\mathcal{L}}{\partial\epsilon_i})]$$



Experiments & Results

Datasets

- Mathematics: GSM8k
- Commonsense: Big-Bench-Hard (BBH)
- Logical reasoning: FOLIO

Baselines

- SOTA prompt optimization methods

  APE, APO, PE2, TextGrd, OPRO, EvoPrompt
- Original zero-shot CoT

Models

- Llama 3-8B / Gemma-2-9B

Experiments & Results

Main Results

Method	Gemma-2-9B			Llama-3-8B		
	GSM8K	BBH	FOLIO	GSM8K	BBH	FOLIO
ZS-CoT (Kojima et al., 2022)	88.6	71.7	65.0	79.6	62.2	58.6
APE (Zhou et al., 2022)	88.6	71.7	67.5	79.9	63.1	57.6
APO (Pryzant et al., 2023)	88.6	72.3	63.1	81.1	62.7	58.6
PE2 (Ye et al., 2023)	88.6	68.9	62.1	80.1	61.5	**62.6**
TextGrad (Yuksekgonul et al., 2024)	87.8	72.9	67.5	78.5	58.5	56.2
GREATER	**89.4**	**76.6**	**69.1**	**82.6**	**68.7**	**62.6**

- GREATER 방식이 zero-shot CoT 및 기존 다른 prompt optimization 방법론들과 비교했을 때 전체적인 성능이 가장 높게 나옴.
- 실행마다 성능의 변동성이 큰 다른 prompt optimization 방법론들과는 다르게 GREATER 프레임워크의 경우 상당히 높은 stability를 보임.

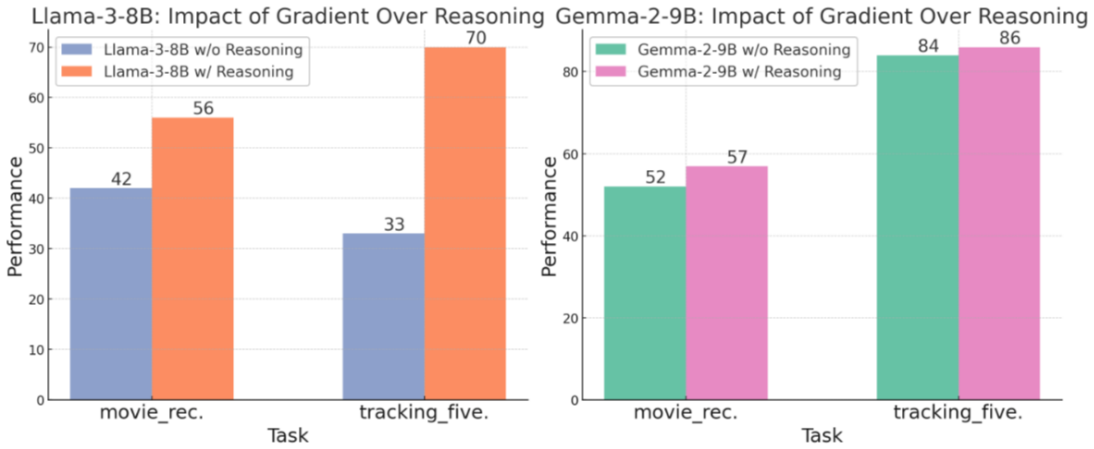
Comparison with Prompts Optimized by Larger Proprietary Models

Target Model	Method (Optimized by)	GSM8K	BBH (5 randomly chosen tasks)					
			movie_rec.	object_count.	tracking_five.	hyperbaton	causal	Average
Llama-3-8B	APE (GPT-4)	80.7	50	82	50	76	56	62.8
	EvoPrompt (GPT-3.5)	-	48	74	42	68	48	56.0
	APO (GPT-4)	81.1	56	68	49	75	51	59.8
	PE2 (GPT-4)	81.5	48	82	45	79	49	60.6
	OPRO (PaLM-2-L)	82.3	60	78	40	70	**57**	61.0
	GREATER (Llama-3-8B)	**82.6**	**57**	**90**	**70**	**84**	**57**	**71.6**
Gemma-2-9B	APE (GPT-4)	89.2	48	61	83	83	60	67.0
	EvoPrompt (GPT-3.5)	-	51	70	82	83	61	69.4
	APO (GPT-4)	89.3	52	84	72	82	59	69.8
	PE2 (GPT-4)	**89.6**	50	65	71	84	**64**	66.8
	OPRO (PaLM-2-L)	89.0	50	58	76	81	58	64.6
	GREATER (Gemma2-9B)	89.4	**56**	**87**	**85**	**88**	61	**75.4**

- GPT-4, GPT-3.6, PaLM-2-L과 같은 대형 모델을 사용하여 기존 prompt optimization 방법론 기반으로 optimize한 prompt 성능보다 여전히 GREATER 프레임워크의 성능이 더 높음을 확인.

Experiments & Results

Ablation of Gradient Over Reasoning



- "Gradient Over Reasoning" step을 제거했을 때와 포함했을 때의 성능 비교.
- Gradient 계산 및 optimization 과정에서 reasoning에 대한 고려가 빠진 순간 상당한 성능 하락이 관측됨.

Prompt Transferability

- Transfer to larger models

Target Model	Method (Optimized by)	BBH (5 randomly chosen tasks)					
		movie_rec.	object_count.	tracking_five.	hyperbaton	causal_judgement	Average
	Llama-3-8B → Gemma-2-27B						
Gemma-2-27B	PE2 (Llama-3)	59	**92**	83	73	54	72.2
	APO (Llama-3)	53	**92**	83	72	**58**	71.6
	Ours (Llama-3)	59	91	**86**	**82**	57	**75.0**
	APO (GPT-4)	**64**	**92**	81	77	**58**	74.4

- Transfer between Gemma-2 and Llama-3

Target Model	Method (Optimized by)	BBH (5 randomly chosen tasks)					
		movie_rec.	object_count.	tracking_five.	hyperbaton	causal_judgement	Average
	Llama-3-8B → Gemma-2-9B						
Gemma-2-9B	TextGrad (Llama-3)	53	78	56	84	**63**	66.8
	APO (Llama-3)	53	84	68	84	58	69.4
	PE2 (Llama-3)	54	84	68	82	60	69.6
	GREATER (Llama-3)	**55**	**90**	**85**	**91**	60	**76.2**
	APO (GPT-4)	52	84	72	82	59	69.8
	Gemma-2-9B → Llama-3-8B						
Llama-3-8B	TextGrad (Gemma-2)	35	29	49	65	36	42.8
	APO (Gemma-2)	54	69	48	71	**53**	59.0
	PE2 (Gemma-2)	56	69	49	50	**53**	55.4
	GREATER (Gemma-2)	**58**	**87**	**56**	70	52	**64.6**
	APO (GPT-4)	56	68	49	**75**	51	59.8

Experiments & Results

Case Study

Task	Optimized Prompt by GREATER	Optimized Prompt by APO
llama3-formal_fallacies	Use formal notation and and think step . . .	Analyze the argument step by step considering premises, logical . . .
llama3-causal_judgement	Use causal diagram...	Analyze the situation by identifying the direct and indirect causes . . .
llama3-object_counting	Use only addition. Add think step by ...	Let's think step by step.
llama3-navigate	Use your reasoning here. I would like numbers assigned.. to.. To represent moving	Analyze the instructions step by step, considering each action's ...
llama3-sports_understanding	Use the context or a sentence similar prior knowledge. Assume you a journalist, I would have been covering NHL hockey in Min- nesota before joining this assignment to report sports.	Assess the plausibility of the sentence, considering both literal and figurative meanings, as well as context and domain knowledge. Evaluate the sentence's coherence and relevance to the given context . . .
gemma-multistep_arithmetic_two	Use parentheses and and the step wise order...	Let's think step by step.
gemma-date_understanding	Use your format Excel formula for this answer to find it...	Let's think step by step.
gemma_reasoning_colored	Use your logic. Please answer. person . . .	Analyze the given text and answer . . .

- GREATER 프레임워크를 통해 optimize한 prompt가 상대적으로 task에 따라 더 다양하고 효과적임을 확인

Trial...

Outer loop

Few-shot 최적화

Inner loop

Thinking path 최적화

1 Llama가 초기 thinking path 생성.
<think>{thinking path}</think><answer>{answer}</answer> 형식.

2 Qwen 모델이 Llama가 생성한 initial thinking path에 대해 feedback 제공.
<original>{기존 Llama의 thinking path}</original><recommended>{Qwen이 생각하는 개선된 path}</recommended><reason>{개선이 필요한 이유}</reason>

3 Llama가 Qwen이 생성한 feedback를 기반으로 accept할 사항들과 reject할 사항들을 명시.
- accept한 경우 Qwen이 제공한 개선이 필요한 이유 및 Qwen이 개선한 path를 참고하여 해당 부분을 Llama가 self-refine.
- reject한 경우
<original>{기존 Llama의 thinking path}</original><rejected>{Qwen이 생각하는 개선된 path}</rejected><reason>{왜 해당 개선을 거부하는지}</reason>

Thinking path 최적화

Few-shot 최적화

감사합니다