

Chain Of Thought (CoT)

SOTA2 - Group C

Anton Kesy, Étienne Muser, Katharina Schindler, Lukas Fehrenbacher, Nico Ruschmann

Offenburg University of Applied Sciences

WS 2024/2025



Was ist Chain-of-Thought (CoT) Prompting?

- Entwickelt von Google Research (2022)
- Eine Methode, die KI-Modelle zur **schrittweisen Darlegung** ihrer Denkprozesse anleitet
- Verbessert die Qualität und Nachvollziehbarkeit von KI-Antworten
- **Funktionsweise:** CoT erweitert den In-Context-Learning (ICL) Ansatz, indem jedes Demonstrationspaar (Eingabe, Ausgabe) um eine Gedankenkette erweitert wird: (Eingabe, Gedankenkette, Ausgabe). Die Gedankenkette besteht aus einer Reihe von Zwischenschritten des Denkens, die die Eingabe mit der Ausgabe verbinden.

Traditionelles vs. CoT Prompting

Traditionell:

Frage: "Was ist 13×27 ?"

Antwort: "351"

Chain-of-Thought:

Frage: "Was ist 13×27 ?"

Antwort:

1. $13 \times 20 = 260$
2. $13 \times 7 = 91$
3. $260 + 91 = 351$

Traditionelles vs. CoT Prompting

Traditionell:

Frage: "Hat ein Geschäft mit 150€ Einnahmen und 180€ Ausgaben Gewinn gemacht?"

Antwort: "Nein"

Chain-of-Thought:

Frage: "Hat ein Geschäft mit 150€ Einnahmen und 180€ Ausgaben Gewinn gemacht?"

Antwort:

1. Einnahmen: 150€
2. Ausgaben: 180€
3. Differenz: $150€ - 180€ = -30€$
4. Ein negativer Wert bedeutet Verlust
5. Ergebnis: Nein, Verlust von 30€

① Erhöhte Genauigkeit

- Strukturiertes Denken reduziert Fehler
- Schrittweise Überprüfung möglich

② Bessere Nachvollziehbarkeit

- Transparenter Gedankengang
- Vereinfachte Fehleranalyse

③ Lerneffekt

- Menschen können von der Methodik lernen
- Verbessertes Verständnis komplexer Probleme

Hauptanwendungsgebiete

- Mathematische Berechnungen
- Logische Schlussfolgerungen
- Komplexe Entscheidungsprozesse
- Textaufgaben
- Argumentationsanalysen

① Komplexität der Antworten

- Längere Antwortzeiten
- Höherer Token-Verbrauch
- Mehr Kontext erforderlich

② Qualität der Zwischenschritte

- Nicht alle Schritte sind immer relevant
- Risiko von Überexplikation
- Mögliche Fehlerfortpflanzung in den Zwischenschritten

③ Anwendungsspezifische Limitationen

- Nicht für alle Aufgabentypen gleich gut geeignet
- Bei einfachen Fragen möglicherweise überdimensioniert
- Erfordert sorgfältige Prompt-Gestaltung

Vergleich: CoT-Prompting im Bereich des arithmetischen Denkens

Experimentelles Setup: Anwendung von Chain-of-Thought-Prompting auf verschiedene Sprachmodelle und Benchmarks für mathematische Textaufgaben.

- **Benchmarks:** fünf Datensätze für mathematische Textaufgaben: GSM8K, SVAMP, ASDiv, AQUA und MAWPS.
- **Standard-Prompting:** Standard-Few-Shot-Prompting
- **Chain-of-Thought-Prompting:** Erweiterung um eine Folge von Schritten (Chain of Thought), die zur Lösung führen
- **Sprachmodelle:** Es wurden fünf große Sprachmodelle evaluiert: GPT-3, LaMDA, PaLM, UL2 20B und Codex

Vergleich: CoT-Prompting im Bereich des arithmetischen Denkens

Math Word Problems (free response)

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. $5 + 6 = 11$. The answer is 11.

Math Word Problems (multiple choice)

Q: How many keystrokes are needed to type the numbers from 1 to 500?
Answer Choices: (a) 1156 (b) 1392 (c) 1480 (d) 1562 (e) 1788

A: There are 9 one-digit numbers from 1 to 9. There are 90 two-digit numbers from 10 to 99. There are 401 three-digit numbers from 100 to 500. $9 + 90(2) + 401(3) = 1392$. The answer is (b).

CSQA (commonsense)

Q: Sammy wanted to go to where the people were. Where might he go?
Options: (a) race track (b) populated areas (c) desert (d) apartment (e) roadblock

A: The answer must be a place with a lot of people. Race tracks, desert, apartments, and roadblocks don't have a lot of people, but populated areas do. So the answer is (b).

StrategyQA

Q: Yes or no: Would a pear sink in water?

A: The density of a pear is about 0.6 g/cm^3 , which is less than water. Thus, a pear would float. So the answer is no.

Date Understanding

Q: The concert was scheduled to be on 06/01/1943, but was delayed by one day to today. What is the date 10 days ago in MM/DD/YYYY?

A: One day after 06/01/1943 is 06/02/1943, so today is 06/02/1943. 10 days before today is 05/23/1943. So the answer is 05/23/1943.

Sports Understanding

Q: Is the following sentence plausible? "Joao Moutinho caught the screen pass in the NFC championship."

A: Joao Moutinho is a soccer player. The NFC championship is part of American football, not soccer. So the answer is no.

SayCan (Instructing a robot)

Human: How would you bring me something that isn't a fruit?

Explanation: the user wants something to eat that isn't a fruit. An energy bar is not a fruit, so I will bring the user an energy bar.
Plan: 1. find(energy bar) 2. pick(energy bar) 3. find(user) 4. put(energy bar) 5. done().

Last Letter Concatenation

Q: Take the last letters of the words in "Lady Gaga" and concatenate them.

A: The last letter of "Lady" is "y". The last letter of "Gaga" is "a". Concatenating them is "ya". So the answer is ya.

Coin Flip (state tracking)

Q: A coin is heads up. Maybelle flips the coin. Shalonda does not flip the coin. Is the coin still heads up?

A: The coin was flipped by Maybelle. So the coin was flipped 1 time, which is an odd number. The coin started heads up, so after an odd number of flips, it will be tails up. So the answer is no.

Ergebnisse:

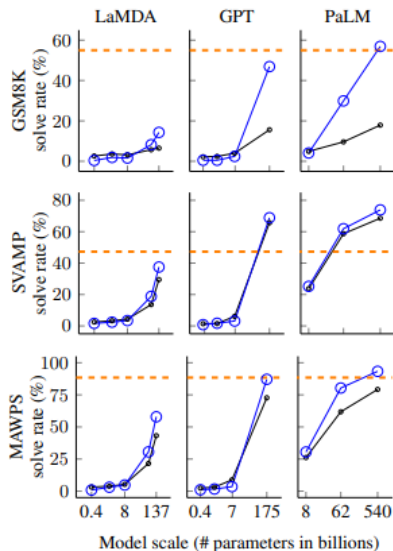
- **Emergente Fähigkeit:**

- Chain-of-Thought-Prompting ist eine emergente Fähigkeit von Sprachmodellen, die erst bei Modellen mit etwa 100 Milliarden Parametern auftritt.
- Kleinere Modelle erzeugen zwar flüssige, aber unlogische Gedankengänge, was zu schlechteren Ergebnissen führt.

- **Komplexitätssteigerung:**

- Chain-of-Thought-Prompting erzielt größere Leistungsgewinne bei komplexeren Problemen.
- Bei einfacheren Datensätzen waren die Verbesserungen geringer oder negativ.

Vergleich: CoT-Prompting im Bereich des arithmetischen Denkens



—●— Standard prompting
—○— Chain-of-thought prompting
- - - Prior supervised best

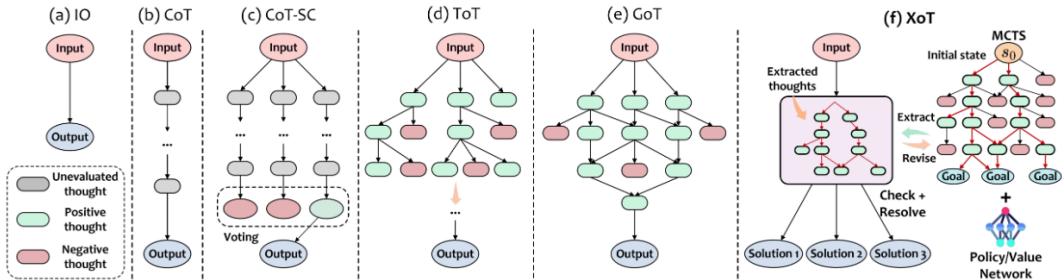
Eine Einführung in XoT

- neuartiger Ansatz zur Gedankenfindung für große LLMs
- Grenzen des *Penrose-Dreiecks* (bestehende Paradigmen der Gedankenfindung einschränkt) überwinden



- Leistung
 - übertrifft bestehende Ansätze in Bezug auf die Genauigkeit bei verschiedenen Problemlösungsaufgaben (Game 24, 8-Puzzle, Pocket Cube)
- Effizienz
 - minimiert die Anzahl der LLM-Aufrufe durch Reinforcement Learning und Monte Carlo Tree Search (MCTS)
- Flexibilität
 - ermöglicht unterschiedliche Gedankenstrukturen die den menschlichen Mindmapping-Prozessen ähneln (Graphen)
 - unterschiedliche und kreative Denkprozesse zu entwickeln, insbesondere bei Problemen mit mehreren Lösungen

Vergleich mit anderen Prompting Ansätzen



Paradigm	Performance	Efficiency	Flexibility
IO	✗	✓	✗
CoT	✓	✓	✗
CoT-SC	✓	✗	✗
ToT	✓	✗	✓
GoT	✓	✗	✓
XoT	✓	✓	✓

- IO (Input-Output)
- CoT (Chain of Thought)
- CoT-SC (CoT with Self-Consistency)
- ToT (Tree of Thoughts)
- GoT (Graph of Thoughts)
- XoT (Everything of Thoughts)

XoT Hauptkomponenten:

- ① MCTS-Modul, das von Policy-/Value-Netzwerken gesteuert wird
 - wird vorab auf bestimmte Aufgaben trainiert, um die Gedankensuche zu optimieren
 - integriert externes Domänenwissen und Planungsfähigkeiten in die von den LLMs verwendeten “Gedanken”
- ② LLM-Solver für Gedankenrevision und Inferenz
 - verwendet vom MCTS-Modul generierten *Gedanken*, um das Problem zu lösen
 - Solver kann *Gedanken* mithilfe der internen Kenntnisse des LLM überarbeiten
 - Zuverlässigkeit der Antwort verbessern

Thoughts searching with MCTS

- Mehrere “Simulationen”
- 3 Stufen
- ① Selection
- ② Evaluation and expansion
- ③ Backpropagation
- Policy and Value network training

Thought inference with MCTS

- Mehrere Simulationen mit MCTS durchführen
- Single solution: State mit höchstem visit count
- Multiple solutions: M Pfade wählen
- Thoughts werden concateniert
- LLM kann Feedback zu fehlerbehafteten Zuständen geben -> Zyklus

Ergebnisse Game of 24 (1/2)

- XOT übertrifft alle anderen getesteten Methoden in Bezug auf die Genauigkeit im Spiel 24. Die **Genauigkeit** von XOT mit **einer Revision** liegt bei **79,56 %** mit GPT-3.5 und 74,45 % mit GPT-4. Nach **drei Revisionen** steigt diese auf **90,51%** bzw. **85,40%**.
- XOT benötigt deutlich **weniger LLM-Aufrufe** als andere Methoden, die eine vergleichbare Genauigkeit erreichen. Während ToT (b=3) mit GPT-4 eine Genauigkeit von 60,58 % erreicht, benötigt es dafür fast 40 LLM-Aufrufe. XOT hingegen benötigt weniger als 2 LLM-Aufrufe, selbst mit drei Revisionen.
- XOT ist in der Lage, sowohl einzelne als auch mehrere Lösungen für das Spiel 24 zu generieren. XOT erzielt die beste Leistung in Bezug auf die MultiAcc (Multi-Solution Accuracy).

Ergebnisse Game of 24 (2/2)

Table 3: Performance comparison on Game of 24.

Model	GPT-3.5			GPT-4		
	Acc. [%]	LLM invoked	f_θ invoked	Acc. [%]	LLM invoked	f_θ invoked
IO	6.57	1.00	-	10.22	1.00	-
CoT	2.19	1.00	-	4.38	1.00	-
CoT-SC	2.19	10.00	-	4.38	10.00	-
ToT (b=1)	5.84	22.11	-	34.31	23.50	-
ToT (b=3)	10.22	43.96	-	60.58	39.83	-
GoT (k=1)	2.92	7.00	-	10.95	7.00	-
LLaMA-2-13B	2.19	-	-	2.19	-	-
MCTS	62.77	-	-	62.77	-	-
XoT (w/ 1 r)	79.56	1.39	92.15	74.45	1.38	88.20
XoT (w/ 2 r)	88.32	1.58	93.87	83.94	1.57	89.63
XoT (w/ 3 r)	90.51	1.72	95.94	85.40	1.78	92.48

Ergebnisse 8-Puzzle (1/1)

- Mit GPT-4 erreicht XOT eine **Genauigkeit von 93,28%** mit einer Revision und 95,80% mit drei Revisionen. Die beste **Prompting-Baseline ToT (b=3)**, erreicht nur eine Genauigkeit von **13,45%**.
- XOT benötigt nur etwa **1,6 LLM-Aufrufe für drei Revisionen**, was es zu einer effizienten Lösung macht. Im Vergleich dazu benötigt ToT (b=3) 54,13 LLM-Aufrufe.
- XOT erzielt auch bei der Generierung mehrerer Lösungen die beste Leistung in Bezug auf die MultiAcc. Der Revisionsprozess verbessert die MultiAcc von XOT mit GPT-4 von 51,26% auf 76,33%.

Table 5: Performance comparison on 8-Puzzle.

Model	GPT-3.5			GPT-4		
	Acc. [%]	LLM invoked	f_θ invoked	Acc. [%]	LLM invoked	f_θ invoked
IO	0.00	1.00	-	1.68	1.00	-
CoT	0.00	1.00	-	7.56	1.00	-
CoT-SC	0.84	10.00	-	8.40	10.00	-
ToT (b=1)	5.88	31.76	-	3.36	27.49	-
ToT (b=3)	6.72	55.86	-	13.45	54.13	-
GoT (k=1)	3.36	19.00	-	3.36	19.00	-
LLaMA-2-13B	0.00	-	-	0.00	-	-
MCTS	51.26	-	-	51.26	-	-
XoT (w/ 1 r)	59.66	1.50	41.09	93.28	1.48	55.66
XoT (w/ 2 r)	59.66	1.92	42.18	94.96	1.55	58.91
XoT (w/ 3 r)	63.03	2.29	42.60	95.80	1.61	62.22

Ergebnisse von Pocket Cube (1/2)

- XOT übertrifft alle anderen getesteten Methoden in Bezug auf die Genauigkeit. Mit GPT-4 erreicht XOT eine Genauigkeit von über **77,60%** mit **einer Revision** und über **80 % mit drei Revisionen**. Die beste **Prompting-Baseline**, ToT (b=3), erreicht nur eine Erfolgsquote von **19,57 %**.
- XOT behält eine **hohe Effizienz** bei und benötigt nur etwa 2 LLM-Inferenzaufrufe für sowohl GPT-3.5 als auch GPT-4.
- XOT erreicht mit GPT-4 über 77 % MultiAcc (Multi-Solution Accuracy). Der Revisionsprozess spielt weiterhin eine wichtige Rolle und verbessert die Leistung von XOT mit beiden GPT-Modellen deutlich.

Table 7: Performance comparison on Pocket Cube.

Model	GPT-3.5			GPT-4		
	Acc. [%]	LLM invoked	f_θ invoked	Acc. [%]	LLM invoked	f_θ invoked
IO	1.09	1.00	-	1.09	1.00	-
CoT	0.00	1.00	-	1.09	1.00	-
CoT-SC	0.00	10.00	-	1.09	10.00	-
ToT (b=1)	7.65	16.50	-	11.48	16.39	-
ToT (b=3)	17.49	58.72	-	19.57	56.58	-
GoT (k=1)	1.64	8.93	-	18.03	8.55	-
LLaMA-2-13B	0.00	-	-	0.00	-	-
MCTS	46.44	-	-	46.44	-	-
XoT (w/ 1 r)	74.32	1.55	64.63	77.60	1.54	75.51
XoT (w/ 2 r)	80.33	1.81	96.46	79.32	1.79	146.52
XoT (w/ 3 r)	84.70	2.01	103.22	83.61	2.00	84.63

Everything-of-Thoughts-XoT

- Everything-of-Thoughts-XoT - GitHub
- Errors
 - Can't run examples
- Missing/incorrect documentation
- Bad style
 - No .gitignore
 - Empty spaces
 - ...

The screenshot shows the GitHub repository page for 'Everything-of-Thoughts-XoT'. The repository is public and has 23 commits. The file list includes: assets, xot_all_in_one, xot_mcts, .gitignore, CODE_OF_CONDUCT.md, CONTRIBUTING.md, LICENSE, README.md, SECURITY.md, SUPPORT.md, and requirements.txt. The README section is visible, titled 'Everything of Thoughts (XoT): Defying the Law of Penrose Triangle for Thought Generation'. It contains a diagram with six parts: (a) IO, (b) CoT, (c) CoT-SC, (d) ToT, (e) GoT, and (f) XoT. The diagram illustrates different thought generation architectures. The right sidebar shows repository statistics: 134 stars, 10 watching, 14 forks, and 1 release published. The contributors section lists 4 contributors: vyokky, microsoftopensource, rusmengd, and microsoft-github-operations[bot]. The languages section shows Python at 98.5% and Shell at 1.5%.

```
python main.py --env game24 --mode train --training_env game24/data/train.csv  
--numMCTSSims 5000 --arenaCompare 100 --numEps 10 --numIters 3
```

- `--env`: Select your desired framework and game.
- `--mode`: train / test.
- `--training_env`: Training data path.
- `--numMCTSSims`: Number of games moves for MCTS to simulate.
- `--arenaCompare`: Number of games to play during arena play to determine if new net will be accepted.
- `--numEps`: Number of complete self-play games to simulate during a new iteration.
- `--numIters`: Number of iteration.


```
python main.py --env game24 --mode test --test_env  
game24/data/test.csv --numMCTSSims 2000 --arenaCompare 137 --multi_sol 0
```

INFO TESTING WINS : 5 / 137, THOUGHTS ACC : 3 %, TESTING AVG CALL : 123.8

Inference

```
python main.py --config config/cube/single_sol/cube_single_xot_laststep0_revised0
```

"To solve the given scrambled 2x2 Pocket Cube, we need to find a sequence of moves that will restore the cube to a state where each face has the same color. Let's analyze the initial cube state and find the restoration move

```
...
{
  "r": {
    "r0": {
      "r": 0,
      "revised": false
    },
    "r1": {
      "r": 0,
      "revised": false
    },
  },
}
```