Agentic AI

What, Why and How?

Arvind Nagaraj
May 2025

Since ChatGPT...

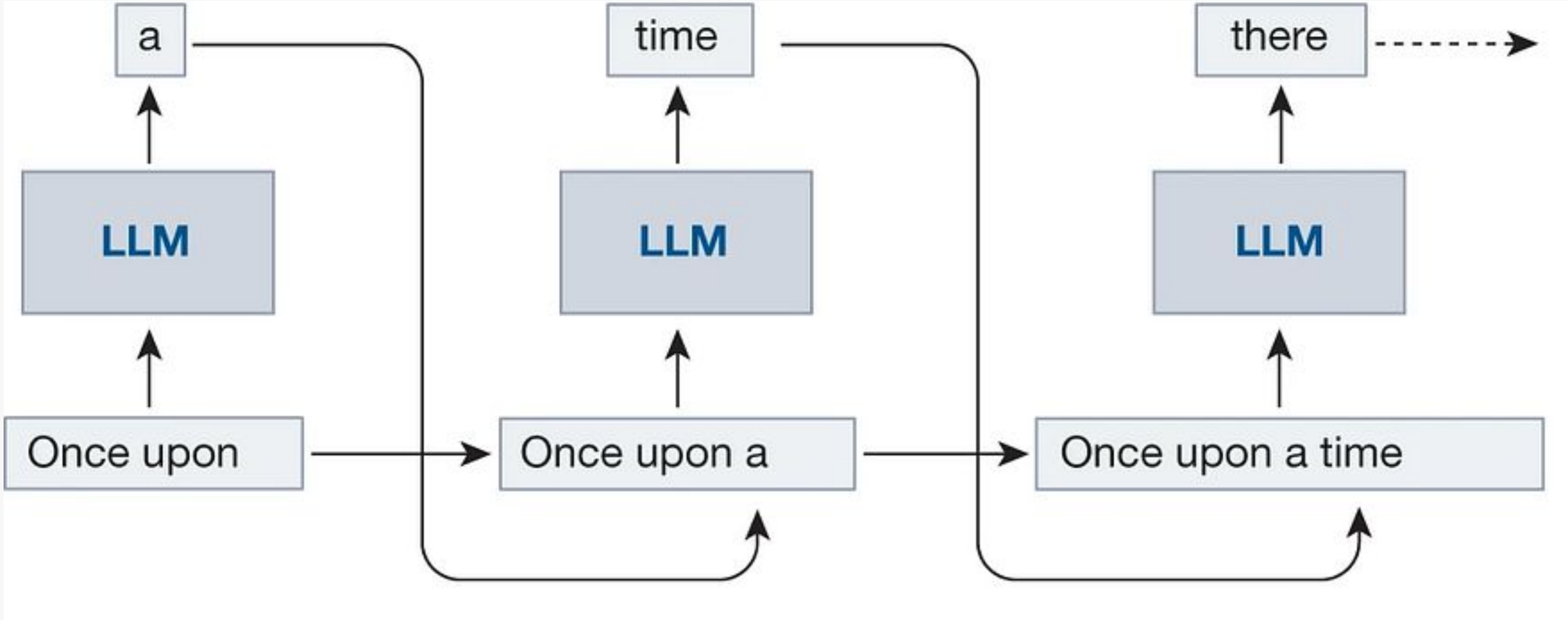
1. Agentic Systems (Vibe Coding, Advanced task automation)
2. Long contexts (Gemini, RoPE)
3. Advances in RL training (DPO, GRPO)
4. Model Merge

From Tokens to Thought:
Reasoning in Large Language Models

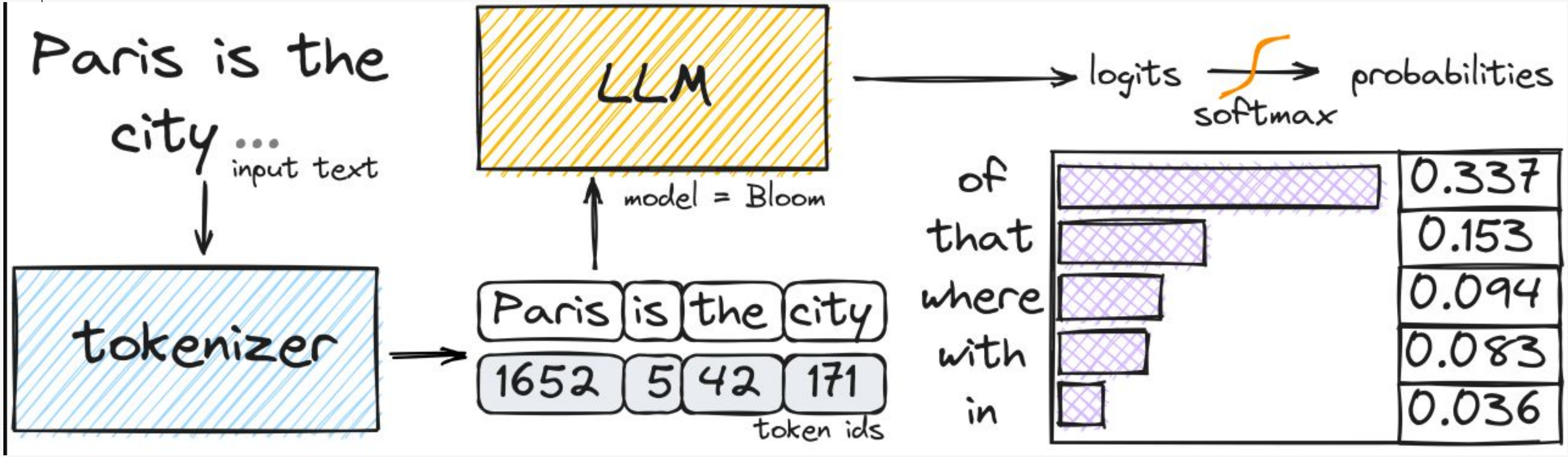
Plan for Theory Sessions:

Supervised Learning (NNs) -> Transformer -> LLMs -> RLHF -> Advanced RL -> Longer Contexts -> Advances in reasoning -> DeepSeek

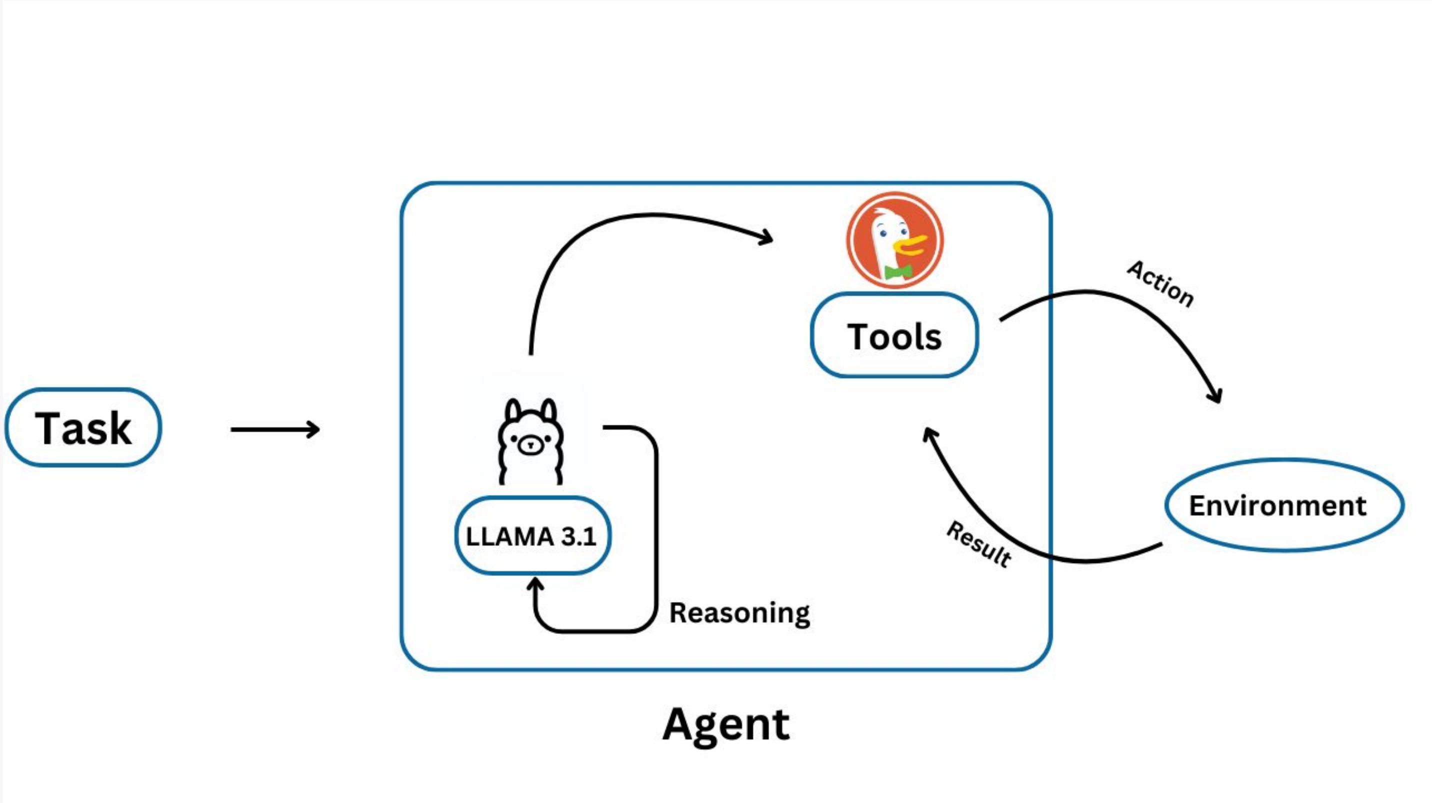
Generating Tokens



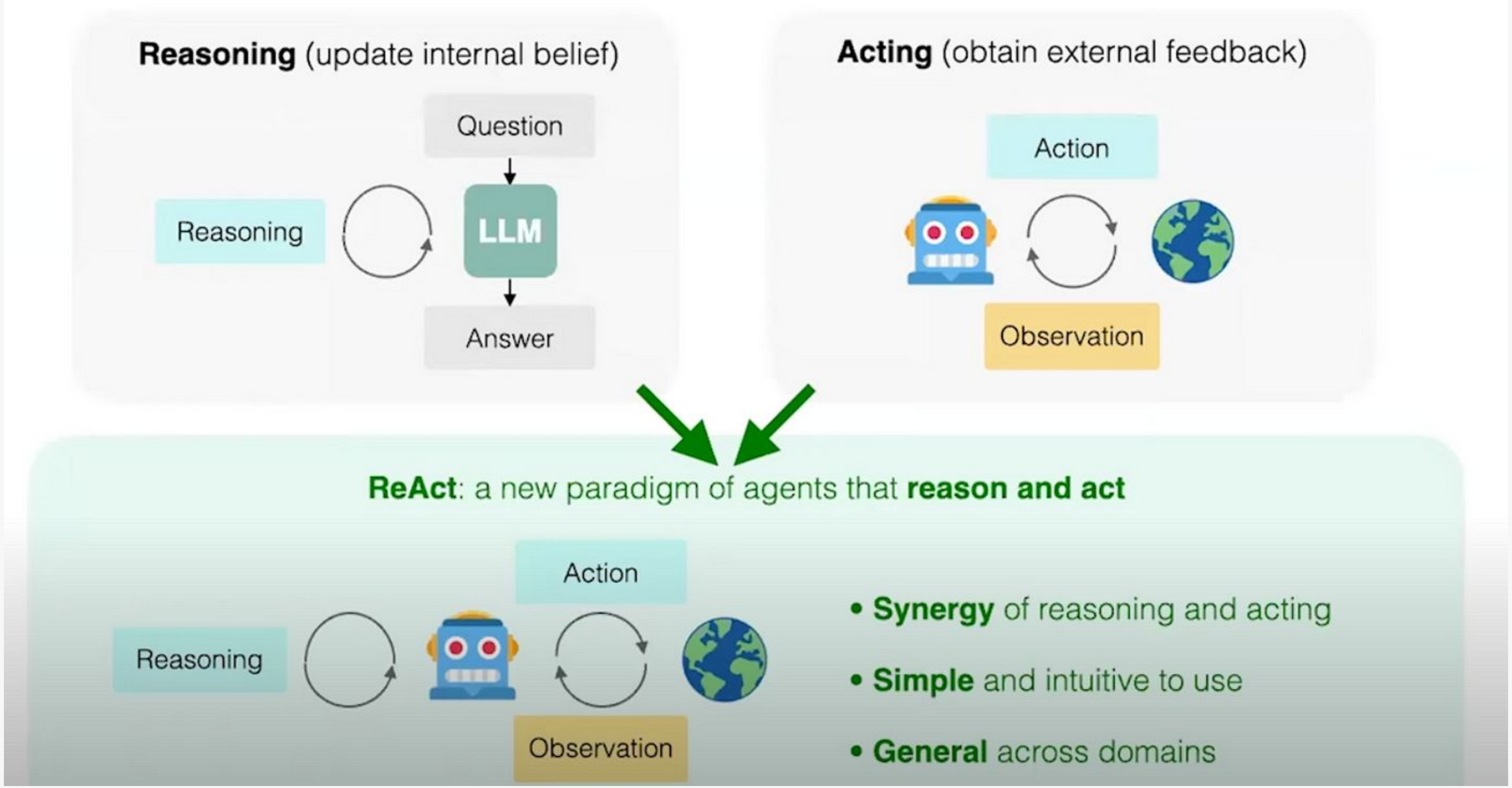
Generating Tokens



From LLMs to Agents



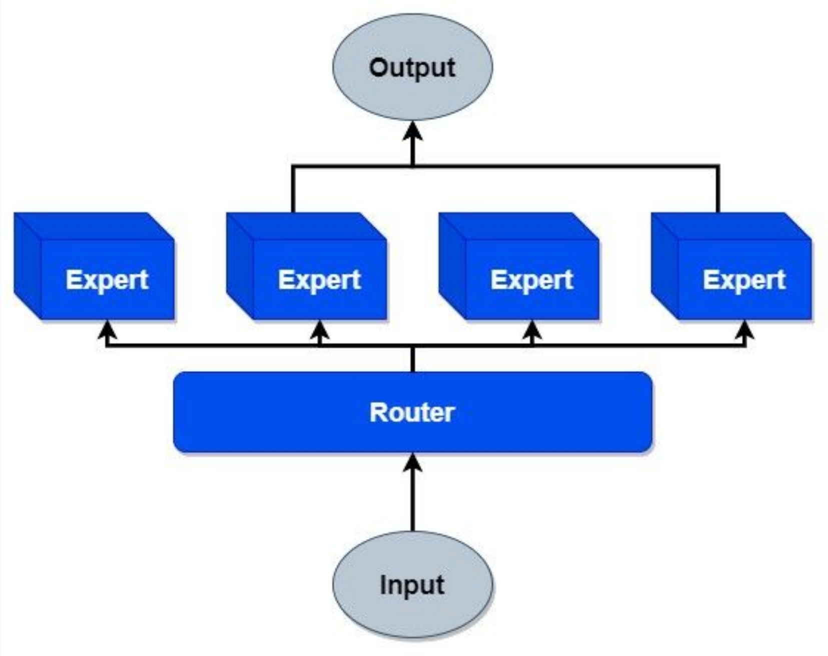
Agentic Framework

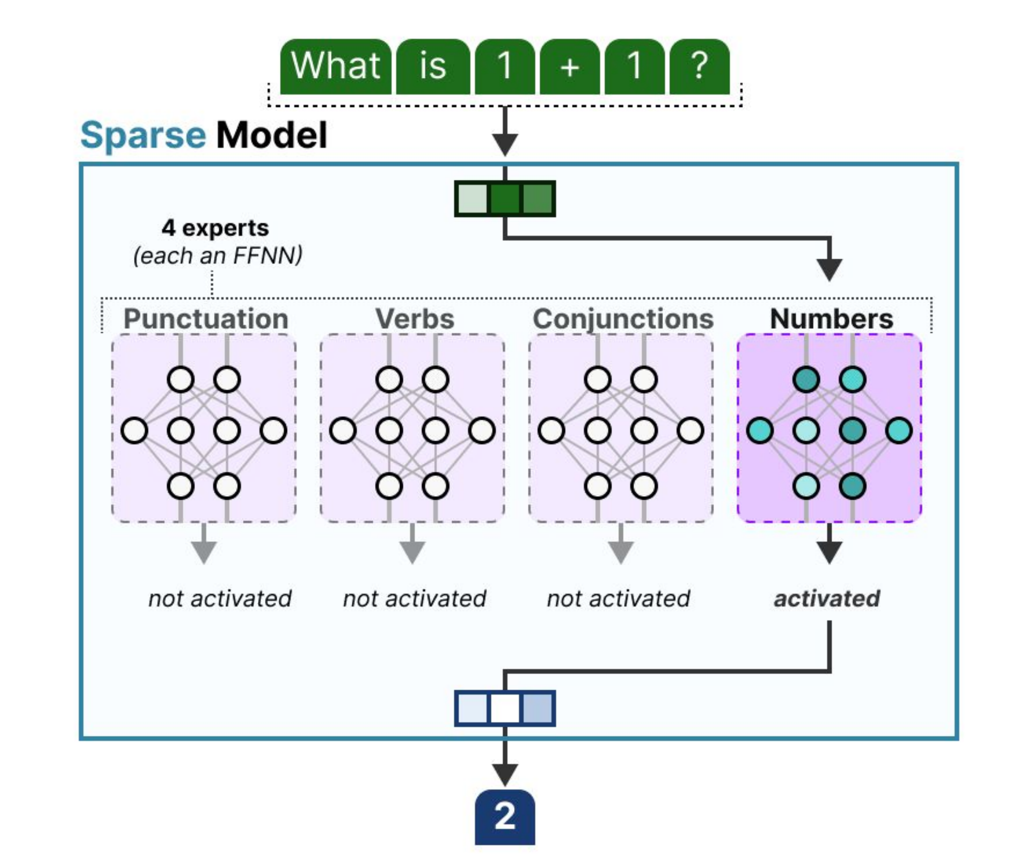


Refresher - Supervised Learning

Advances since the Vaswani Transformer…

Mixture of experts





1991 Hinton paper

Adaptive Mixtures of Local Experts

Robert A. Jacobs
Michael I. Jordan
Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology, Cambridge, MA 02139 USA

Steven J. Nowlan
Geoffrey E. Hinton
Department of Computer Science, University of Toronto, Toronto, Canada M5S 1A4

We present a new supervised learning procedure for systems composed of many separate networks, each of which learns to handle a subset of the complete set of training cases. The new procedure can be viewed either as a modular version of a multilayer supervised network, or as an associative version of competitive learning. It therefore provides a new link between these two apparently different approaches. We demonstrate that the learning procedure divides up a vowel discrimination task into appropriate subtasks, each of which can be solved by a very simple expert network.

1 Making Associative Learning Competitive

If backpropagation is used to train a single, multilayer network to perform different subtasks on different occasions, there will generally be strong interference effects that lead to slow learning and poor generalization. If we know in advance that a set of training cases may be naturally divided into subsets that correspond to distinct subtasks, interference can be reduced by using a system composed of several different "expert" networks plus a gating network that decides which of the experts should be used for each training case.[1] Hampshire and Waibel (1989) have described a system of this kind that can be used when the division into subtasks is known prior to training, and Jacobs *et al.* (1990) have described a related system that *learns* how to allocate cases to experts. The idea behind such a system is that the gating network allocates a new case to one or a few experts, and, if the output is incorrect, the weight changes are localized to these experts (and the gating network).

[1]This idea was first presented by Jacobs and Hinton at the Connectionist Summer School in Pittsburgh in 1988.

Neural Computation 3, 79–87 (1991)

[Submitted on 8 Jan 2024]

Mixtral of Experts

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, William El Sayed

We introduce Mixtral 8x7B, a Sparse Mixture of Experts (SMoE) language model. Mixtral has the same architecture as Mistral 7B, with the difference that each layer is composed of 8 feedforward blocks (i.e. experts). For every token, at each layer, a router network selects two experts to process the current state and combine their outputs. Even though each token only sees two experts, the selected experts can be different at each timestep. As a result, each token has access to 47B parameters, but only uses 13B active parameters during inference. Mixtral was trained with a context size of 32k tokens and it outperforms or matches Llama 2 70B and GPT-3.5 across all evaluated benchmarks. In particular, Mixtral vastly outperforms Llama 2 70B on mathematics, code generation, and multilingual benchmarks. We also provide a model fine-tuned to follow instructions, Mixtral 8x7B – Instruct, that surpasses GPT-3.5 Turbo, Claude-2.1, Gemini Pro, and Llama 2 70B – chat model on human benchmarks. Both the base and instruct models are released under the Apache 2.0 license.

Comments: See more details at this https URL
Subjects: **Machine Learning (cs.LG)**; Computation and Language (cs.CL)
Cite as: arXiv:2401.04088 **[cs.LG]**
(or arXiv:2401.04088v1 **[cs.LG]** for this version)
https://doi.org/10.48550/arXiv.2401.04088

Submission history

From: Devendra Singh Chaplot [view email]
[v1] Mon, 8 Jan 2024 18:47:34 UTC (2,811 KB)

Long Context - RoPE

arXiv > cs > arXiv:2104.09864

Search...

Help | Adv

Computer Science > Computation and Language

[Submitted on 20 Apr 2021 (v1), last revised 8 Nov 2023 (this version, v5)]

RoFormer: Enhanced Transformer with Rotary Position Embedding

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, Yunfeng Liu

Position encoding recently has shown effective in the transformer architecture. It enables valuable supervision for dependency modeling between elements at different positions of the sequence. In this paper, we first investigate various methods to integrate positional information into the learning process of transformer-based language models. Then, we propose a novel method named Rotary Position Embedding(RoPE) to effectively leverage the positional information. Specifically, the proposed RoPE encodes the absolute position with a rotation matrix and meanwhile incorporates the explicit relative position dependency in self-attention formulation. Notably, RoPE enables valuable properties, including the flexibility of sequence length, decaying inter-token dependency with increasing relative distances, and the capability of equipping the linear self-attention with relative position encoding. Finally, we evaluate the enhanced transformer with rotary position embedding, also called RoFormer, on various long text classification benchmark datasets. Our experiments show that it consistently overcomes its alternatives. Furthermore, we provide a theoretical analysis to explain some experimental results. RoFormer is already integrated into Huggingface: \url{this https URL}.

Comments: fixed some typos

Subjects: **Computation and Language (cs.CL)**; Artificial Intelligence (cs.AI); Machine Learning (cs.LG)

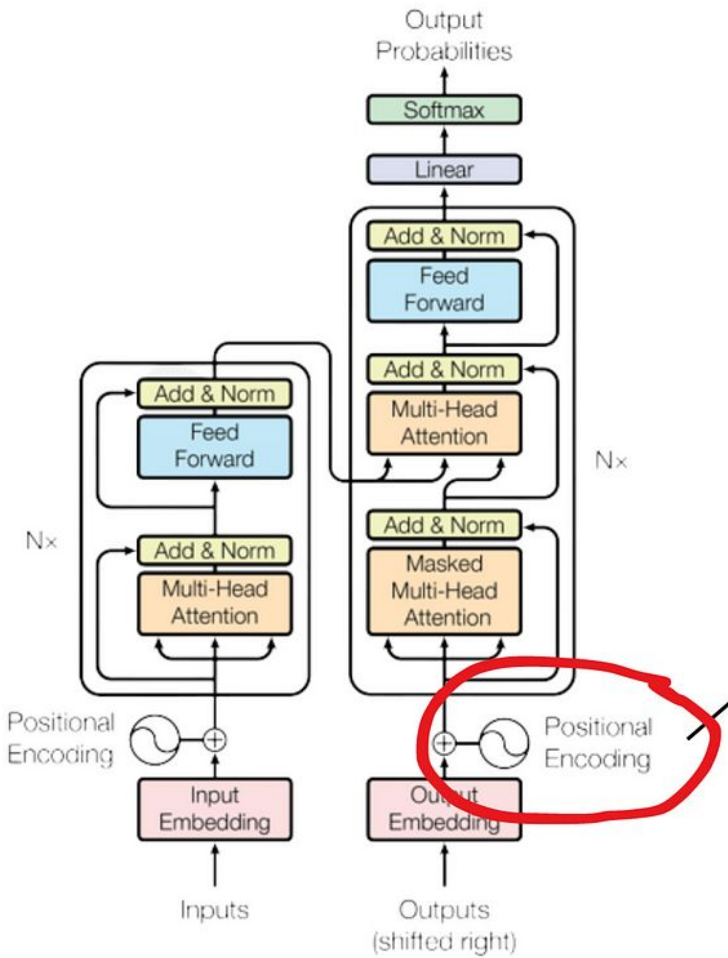
Cite as: arXiv:2104.09864 **[cs.CL]**

(or arXiv:2104.09864v5 **[cs.CL]** for this version)

https://doi.org/10.48550/arXiv.2104.09864

Submission history

https://arxiv.org/abs/2104.09864



Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Positional
Encoding
Input
Embedding
Output
Embedding
Inputs
Outputs
(shifted right)

RoPE

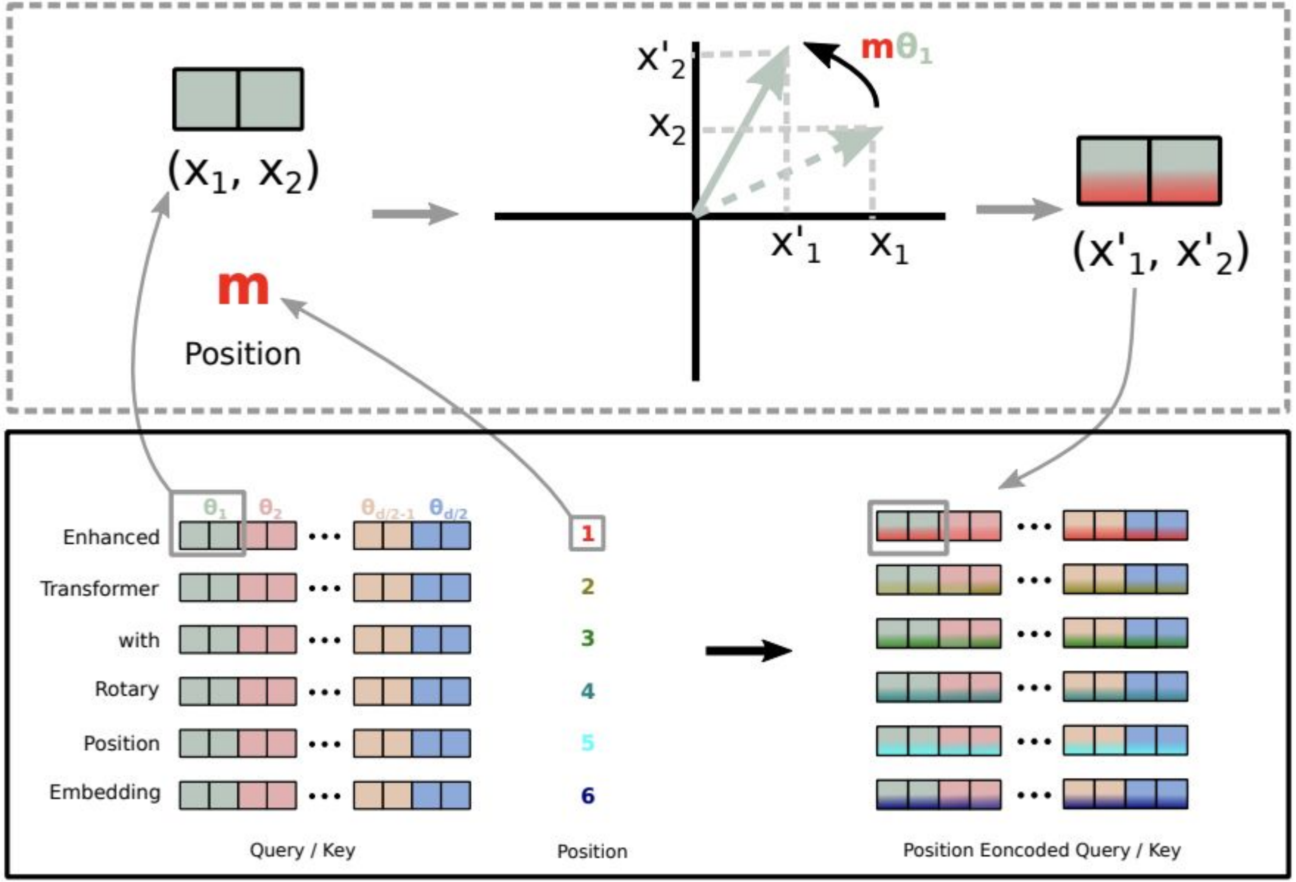
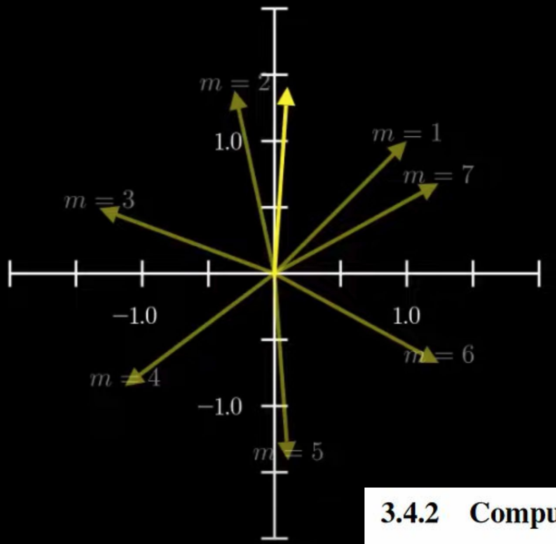


Figure 1: Implementation of Rotary Position Embedding(RoPE).



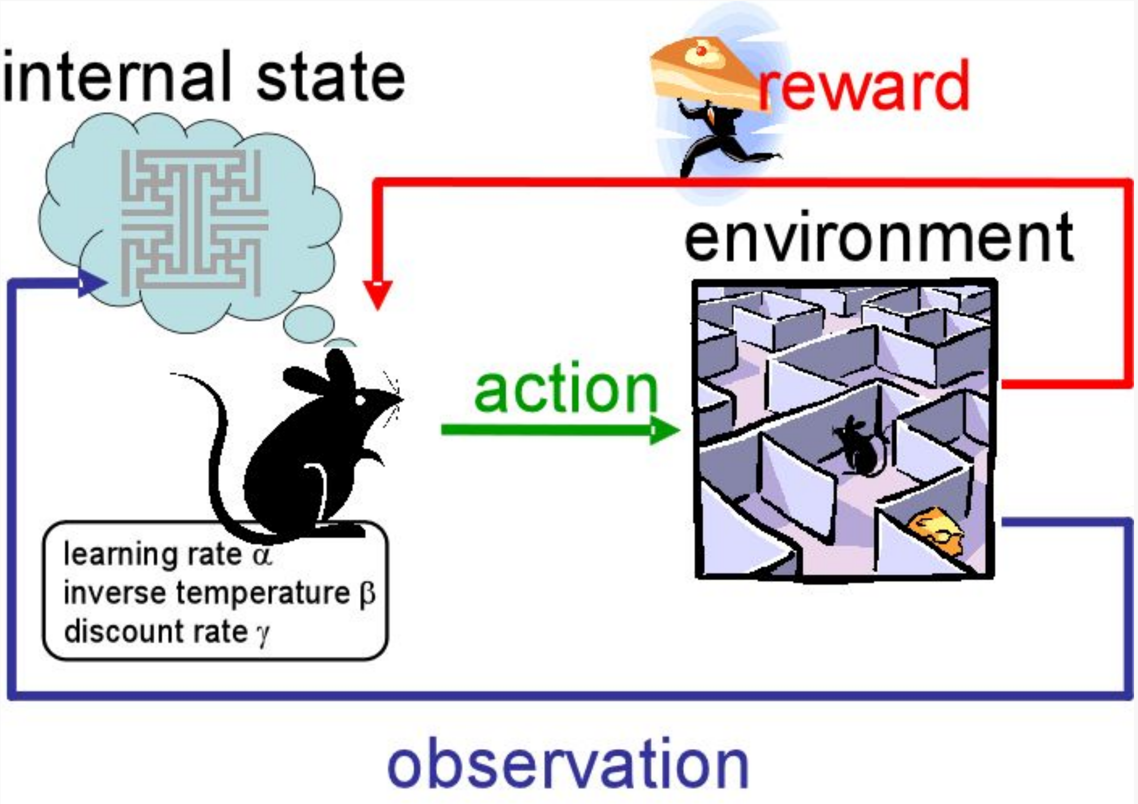
3.4.2 Computational efficient realization of rotary matrix multiplication

Taking the advantage of the sparsity of $\boldsymbol{R}^d_{\Theta,m}$ in Equation ($\boxed{15}$), a more computational efficient realization of a multiplication of R^d_Θ and $\boldsymbol{x} \in \mathbb{R}^d$ is:

$$\boldsymbol{R}^d_{\Theta,m}\boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \vdots \\ x_{d-1} \\ x_d \end{pmatrix} \otimes \begin{pmatrix} \cos m\theta_1 \\ \cos m\theta_1 \\ \cos m\theta_2 \\ \cos m\theta_2 \\ \vdots \\ \cos m\theta_{d/2} \\ \cos m\theta_{d/2} \end{pmatrix} + \begin{pmatrix} -x_2 \\ x_1 \\ -x_4 \\ x_3 \\ \vdots \\ -x_d \\ x_{d-1} \end{pmatrix} \otimes \begin{pmatrix} \sin m\theta_1 \\ \sin m\theta_1 \\ \sin m\theta_2 \\ \sin m\theta_2 \\ \vdots \\ \sin m\theta_{d/2} \\ \sin m\theta_{d/2} \end{pmatrix} \tag{34}$$

Advances in
Deep Reinforcement Learning

Not too different from Animals

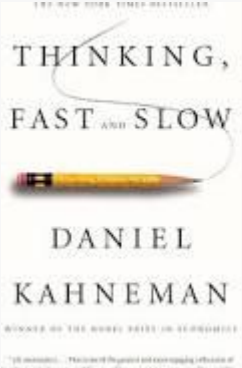


Reasoning in LLMs

Motivation: "Why do we want LLMs to reason?"

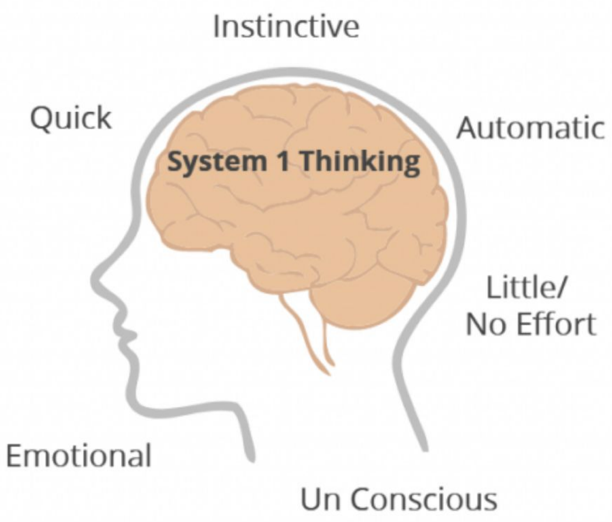
- Factual recall ≠ Reasoning
- Real-world tasks: planning, math, code, chain-of-thought

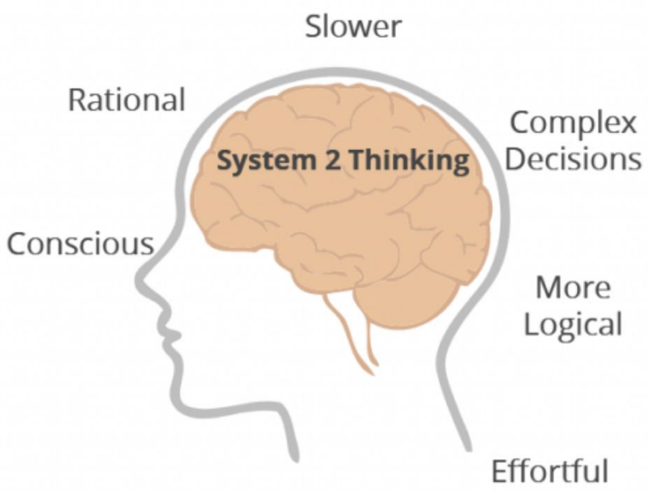
System 1 vs System 2 thinking



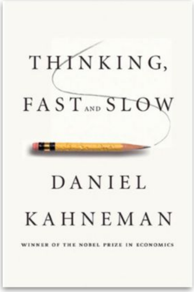
System 1

System 2





$2 + 2 = 4$



$17 \times 24 = \ldots$

What is Reasoning in LLMs?

Definitions: deductive, inductive, abductive reasoning

Types relevant to LLMs:

- Symbolic reasoning (e.g., logic, algebra)
- Commonsense reasoning
- Multi-hop reasoning
- Analogical reasoning
- Procedural reasoning (e.g., tool use)

Prompt Engineering & Chain-of-Thought

Chain-of-thought prompting (Wei et al., 2022)

Self-consistency over greedy decoding

Examples: math word problems, riddles

Tree of Thoughts (Yao et al.)

Drawbacks of naive CoT

arXiv > cs > arXiv:2201.11903

Sea
Help

Computer Science > Computation and Language

[Submitted on 28 Jan 2022 (v1), last revised 10 Jan 2023 (this version, v6)]

Chain-of-Thought Prompting Elicits Reasoning in Large Language Models

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, Denny Zhou

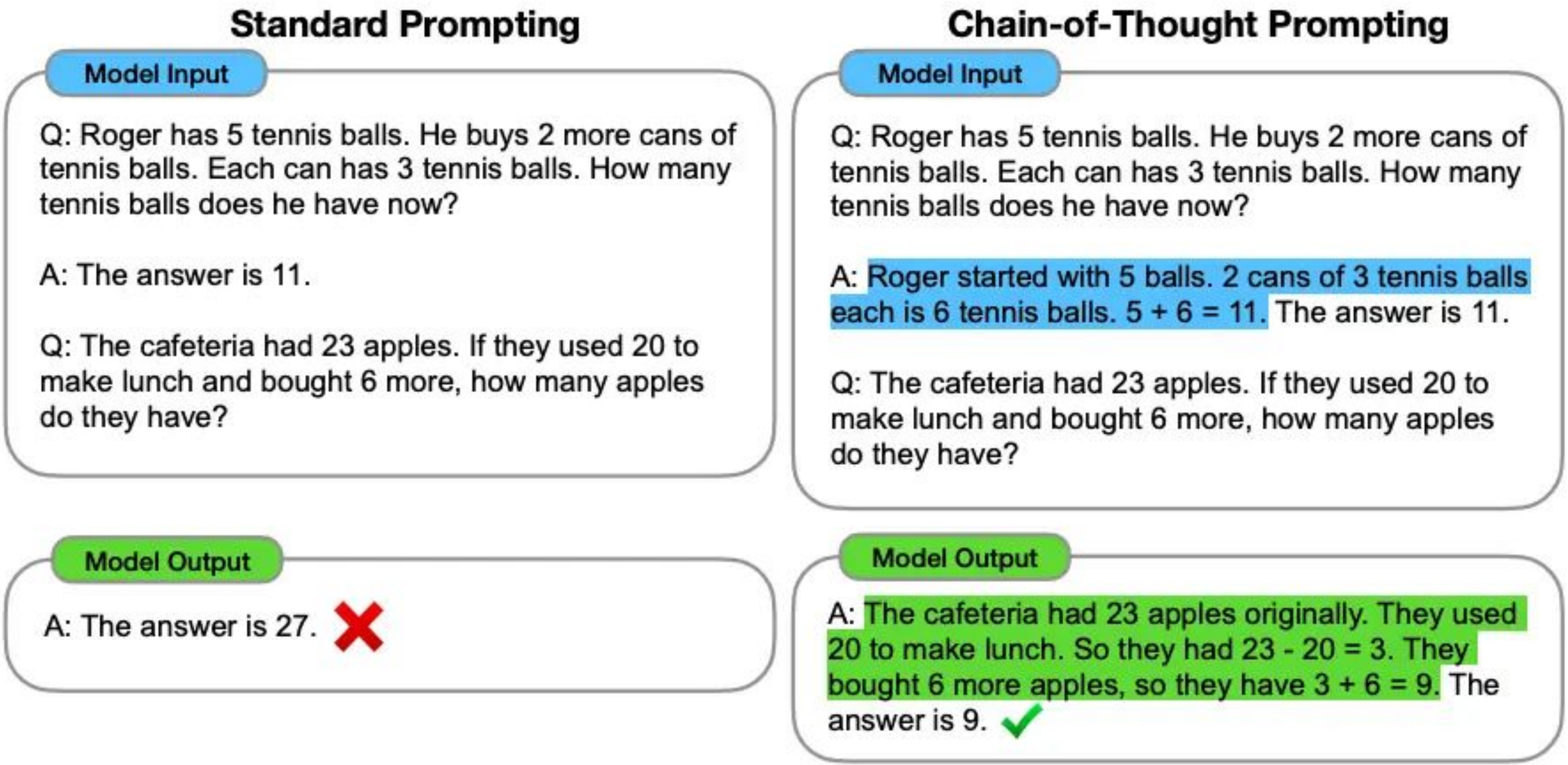
We explore how generating a chain of thought -- a series of intermediate reasoning steps -- significantly improves the ability of large language models to perform complex reasoning. In particular, we show how such reasoning abilities emerge naturally in sufficiently large language models via a simple method called chain of thought prompting, where a few chain of thought demonstrations are provided as exemplars in prompting. Experiments on three large language models show that chain of thought prompting improves performance on a range of arithmetic, commonsense, and symbolic reasoning tasks. The empirical gains can be striking. For instance, prompting a 540B-parameter language model with just eight chain of thought exemplars achieves state of the art accuracy on the GSM8K benchmark of math word problems, surpassing even finetuned GPT-3 with a verifier.

Subjects: **Computation and Language (cs.CL)**; Artificial Intelligence (cs.AI)
Cite as: arXiv:2201.11903 **[cs.CL]**
(or arXiv:2201.11903v6 **[cs.CL]** for this version)
https://doi.org/10.48550/arXiv.2201.11903

Submission history

From: Jason Wei [view email]
[v1] Fri, 28 Jan 2022 02:33:07 UTC (944 KB)

https://arxiv.org/abs/2201.11903


Standard Prompting
Model Input
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: The answer is 11.
Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?
Model Output
A: The answer is 27.
Chain-of-Thought Prompting
Model Input
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.
Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?
Model Output
A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had 23 - 20 = 3. They bought 6 more apples, so they have 3 + 6 = 9. The answer is 9.

Computer Science > Computation and Language

[Submitted on 17 May 2023 (v1), last revised 3 Dec 2023 (this version, v2)]

Tree of Thoughts: Deliberate Problem Solving with Large Language Models

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, Karthik Narasimhan

Language models are increasingly being deployed for general problem solving across a wide range of tasks, but are still confined to token-level, left-to-right decision-making processes during inference. This means they can fall short in tasks that require exploration, strategic lookahead, or where initial decisions play a pivotal role. To surmount these challenges, we introduce a new framework for language model inference, Tree of Thoughts (ToT), which generalizes over the popular Chain of Thought approach to prompting language models, and enables exploration over coherent units of text (thoughts) that serve as intermediate steps toward problem solving. ToT allows LMs to perform deliberate decision making by considering multiple different reasoning paths and self-evaluating choices to decide the next course of action, as well as looking ahead or backtracking when necessary to make global choices. Our experiments show that ToT significantly enhances language models' problem-solving abilities on three novel tasks requiring non-trivial planning or search: Game of 24, Creative Writing, and Mini Crosswords. For instance, in Game of 24, while GPT-4 with chain-of-thought prompting only solved 4% of tasks, our method achieved a success rate of 74%. Code repo with all prompts: this https URL.

Comments: NeurIPS 2023 camera ready version. Code repo with all prompts: this https URL
Subjects: **Computation and Language (cs.CL)**; Artificial Intelligence (cs.AI); Machine Learning (cs.LG)
Cite as: arXiv:2305.10601 **[cs.CL]**
(or arXiv:2305.10601v2 **[cs.CL]** for this version)
https://doi.org/10.48550/arXiv.2305.10601

Submission history

From: Shunyu Yao [view email]
[v1] Wed, 17 May 2023 23:16:17 UTC (609 KB)
[v2] Sun, 3 Dec 2023 22:50:35 UTC (623 KB)

https://arxiv.org/abs/2305.10601

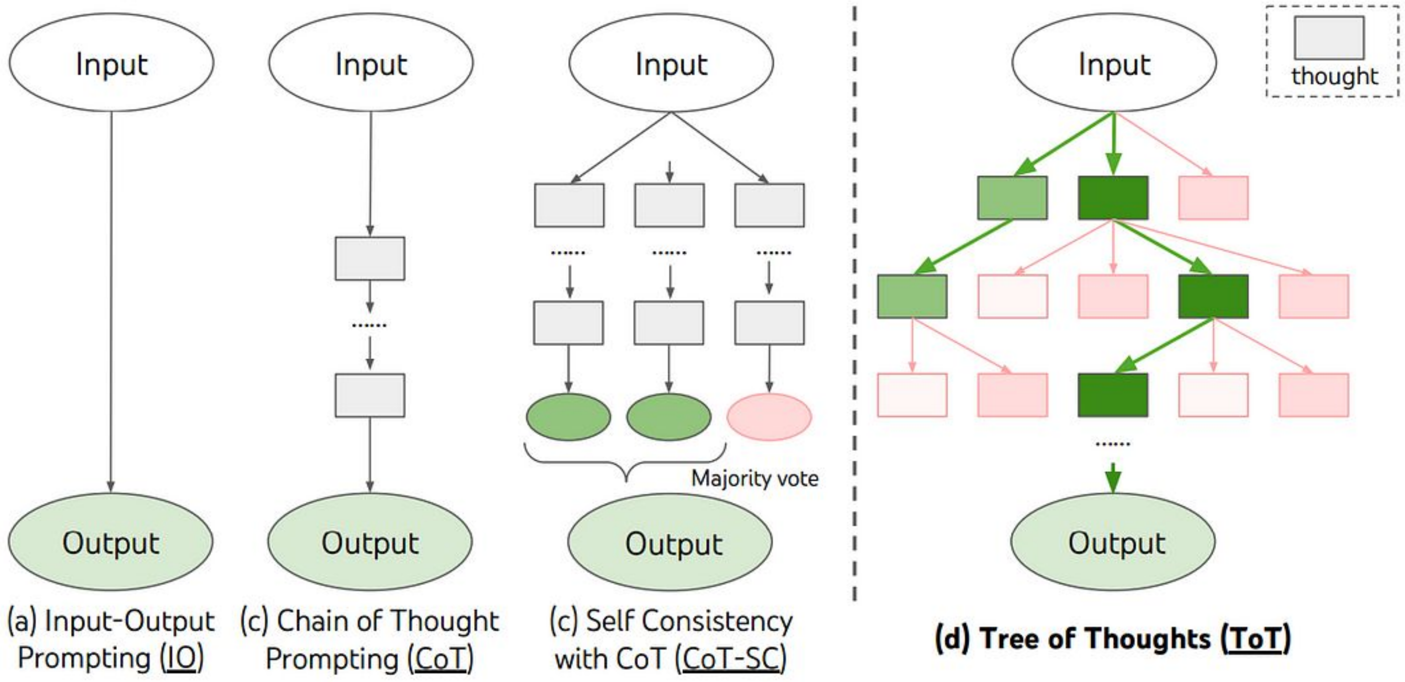


Figure 1: Schematic illustrating various approaches to problem solving with LLMs. Each rectangle box represents a *thought*, which is a coherent language sequence that serves as an intermediate step toward problem solving. See concrete examples of how thoughts are generated, evaluated, and searched in Figures 2,4,6.

Tool-Augmented LLMs

- ReAct: reasoning + acting
- Toolformer, AutoGPT-style agents
- Calculator, Python, code interpreter plugins
- How reasoning shifts when the model can delegate computation

arXiv > cs > arXiv:2302.04761

Search... Help | A

Computer Science > Computation and Language

[Submitted on 9 Feb 2023]

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

Language models (LMs) exhibit remarkable abilities to solve new tasks from just a few examples or textual instructions, especially at scale. They also, paradoxically, struggle with basic functionality, such as arithmetic or factual lookup, where much simpler and smaller models excel. In this paper, we show that LMs can teach themselves to use external tools via simple APIs and achieve the best of both worlds. We introduce Toolformer, a model trained to decide which APIs to call, when to call them, what arguments to pass, and how to best incorporate the results into future token prediction. This is done in a self-supervised way, requiring nothing more than a handful of demonstrations for each API. We incorporate a range of tools, including a calculator, a Q\&A system, two different search engines, a translation system, and a calendar. Toolformer achieves substantially improved zero-shot performance across a variety of downstream tasks, often competitive with much larger models, without sacrificing its core language modeling abilities.

Subjects: **Computation and Language (cs.CL)**
Cite as: arXiv:2302.04761 **[cs.CL]**
(or arXiv:2302.04761v1 **[cs.CL]** for this version)
https://doi.org/10.48550/arXiv.2302.04761

Submission history

From: Timo Schick [view email]
[v1] Thu, 9 Feb 2023 16:49:57 UTC (202 KB)

https://arxiv.org/abs/2302.04761

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.

Figure 1: Exemplary predictions of Toolformer. The model autonomously decides to call different APIs (from top to bottom: a question answering system, a calculator, a machine translation system, and a Wikipedia search engine) to obtain information that is useful for completing a piece of text.

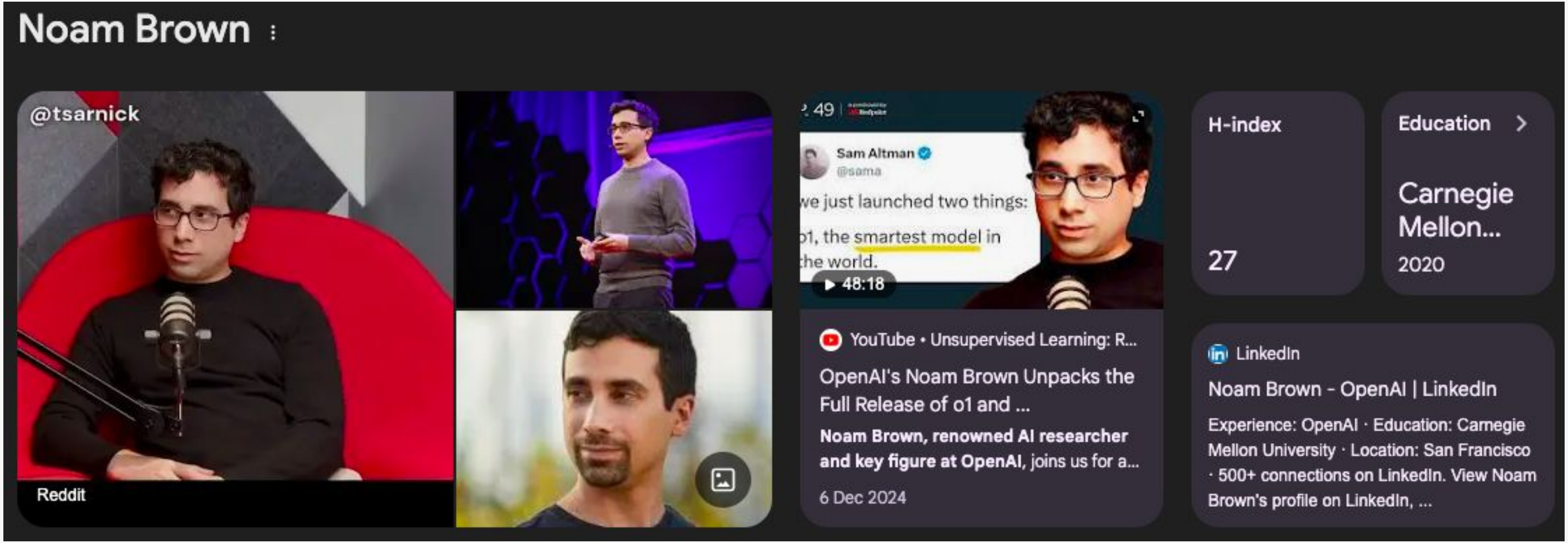
The bitter lesson

https://www.cs.utexas.edu/~eunsol/courses/data/bitter_lesson.pdf

One thing that should be learned from the bitter lesson is the great power of general purpose methods, of methods that continue to scale with increased computation even as the available computation becomes very great. The two methods that seem to scale arbitrarily in this way are *search* and *learning*.

The second general point to be learned from the bitter lesson is that the actual contents of minds are tremendously, irredeemably complex; we should stop trying to find simple ways to think about the contents of minds, such as simple ways to think about space, objects, multiple agents, or symmetries. All these are part of the arbitrary, intrinsically-complex, outside world. They are not what should be built in, as their complexity is endless; instead we should build in only the meta-methods that can find and capture this arbitrary complexity. Essential to these methods is that they can find good approximations, but the search for them should be by our methods, not by us. We want AI agents that can discover like we can, not which contain what we have discovered. Building in our discoveries only makes it harder to see how the discovering process can be done.

o1 and thinking models



DeepSeek


AIM
Who is DeepSeek CEO
Liang Wenfeng?
info@aimmediahouse.com

arXiv > cs > arXiv:2501.12948

Help | Adva

Computer Science > Computation and Language

[Submitted on 22 Jan 2025]

DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z.F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J.L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R.J. Chen, R.L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S.S. Li et al. (100 additional authors not shown)

We introduce our first-generation reasoning models, DeepSeek-R1-Zero and DeepSeek-R1. DeepSeek-R1-Zero, a model trained via large-scale reinforcement learning (RL) without supervised fine-tuning (SFT) as a preliminary step, demonstrates remarkable reasoning capabilities. Through RL, DeepSeek-R1-Zero naturally emerges with numerous powerful and intriguing reasoning behaviors. However, it encounters challenges such as poor readability, and language mixing. To address these issues and further enhance reasoning performance, we introduce DeepSeek-R1, which incorporates multi-stage training and cold-start data before RL. DeepSeek-R1 achieves performance comparable to OpenAI-o1-1217 on reasoning tasks. To support the research community, we open-source DeepSeek-R1-Zero, DeepSeek-R1, and six dense models (1.5B, 7B, 8B, 14B, 32B, 70B) distilled from DeepSeek-R1 based on Qwen and Llama.

Subjects: **Computation and Language (cs.CL)**; Artificial Intelligence (cs.AI); Machine Learning (cs.LG)

Cite as: arXiv:2501.12948 **[cs.CL]**

(or arXiv:2501.12948v1 **[cs.CL]** for this version)

https://doi.org/10.48550/arXiv.2501.12948

Submission history

https://arxiv.org/abs/2501.12948

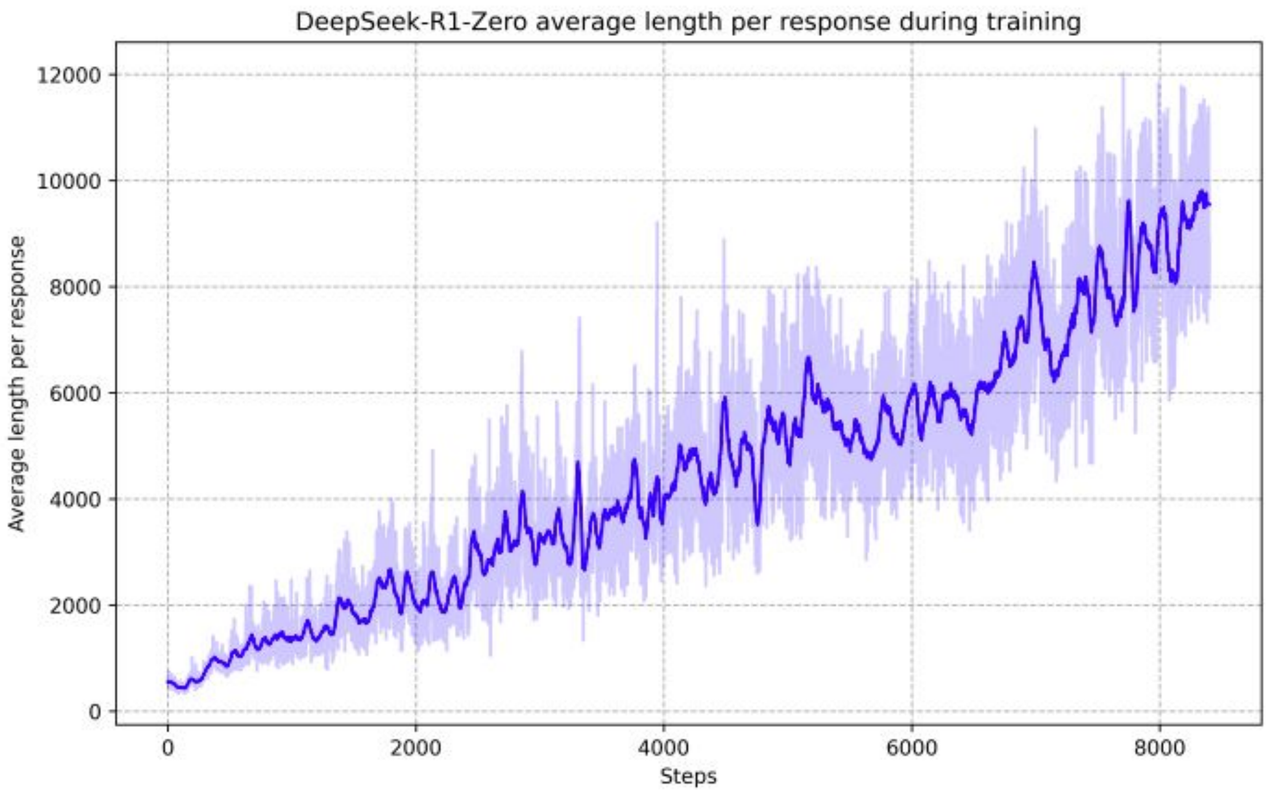


Figure 3 | The average response length of DeepSeek-R1-Zero on the training set during the RL process. DeepSeek-R1-Zero naturally learns to solve reasoning tasks with more thinking time.

Question: If $a > 1$, then the sum of the real solutions of $\sqrt{a - \sqrt{a + x}} = x$ is equal to
Response: <think> To solve the equation $\sqrt{a - \sqrt{a + x}} = x$, let's start by squaring both $\cdots$ $\left(\sqrt{a - \sqrt{a + x}}\right)^2 = x^2 \implies a - \sqrt{a + x} = x^2.$ Rearrange to isolate the inner square root term: $(a - x^2)^2 = a + x \implies a^2 - 2ax^2 + (x^2)^2 = a + x \implies x^4 - 2ax^2 - x + (a^2 - a) = 0$ $\ldots$ Wait, wait. Wait. That's an aha moment I can flag here. Let's reevaluate this step-by-step to identify if the correct sum can be $\cdots$ We started with the equation: $\sqrt{a - \sqrt{a + x}} = x$ First, let's square both sides: $a - \sqrt{a + x} = x^2 \implies \sqrt{a + x} = a - x^2$ Next, I could square both sides again, treating the equation: $\cdots$ $\ldots$

Table 3 | An interesting "aha moment" of an intermediate version of DeepSeek-R1-Zero. The model learns to rethink using an anthropomorphic tone. This is also an aha moment for us, allowing us to witness the power and beauty of reinforcement learning.

arXiv > cs > arXiv:2402.03300

Help | Ad

Computer Science > Computation and Language

[Submitted on 5 Feb 2024 (v1), last revised 27 Apr 2024 (this version, v3)]

DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y.K. Li, Y. Wu, Daya Guo

Mathematical reasoning poses a significant challenge for language models due to its complex and structured nature. In this paper, we introduce DeepSeekMath 7B, which continues pre-training DeepSeek-Coder-Base-v1.5 7B with 120B math-related tokens sourced from Common Crawl, together with natural language and code data. DeepSeekMath 7B has achieved an impressive score of 51.7% on the competition-level MATH benchmark without relying on external toolkits and voting techniques, approaching the performance level of Gemini-Ultra and GPT-4. Self-consistency over 64 samples from DeepSeekMath 7B achieves 60.9% on MATH. The mathematical reasoning capability of DeepSeekMath is attributed to two key factors: First, we harness the significant potential of publicly available web data through a meticulously engineered data selection pipeline. Second, we introduce Group Relative Policy Optimization (GRPO), a variant of Proximal Policy Optimization (PPO), that enhances mathematical reasoning abilities while concurrently optimizing the memory usage of PPO.

Subjects: **Computation and Language (cs.CL)**; Artificial Intelligence (cs.AI); Machine Learning (cs.LG)
Cite as: arXiv:2402.03300 **[cs.CL]**
(or arXiv:2402.03300v3 **[cs.CL]** for this version)
https://doi.org/10.48550/arXiv.2402.03300

Submission history

From: Zhihong Shao [view email]
[v1] Mon, 5 Feb 2024 18:55:32 UTC (3,417 KB)

GRPO Paper
https://arxiv.org/abs/2402.03300

GRPO

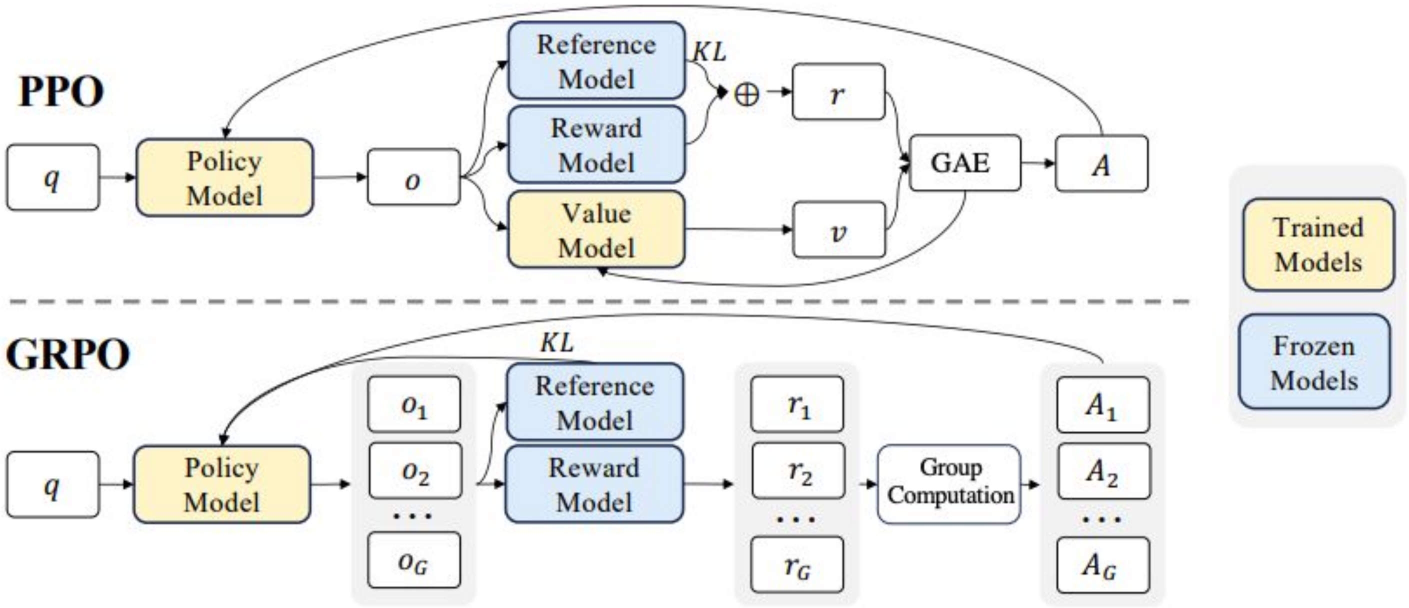


Figure 4 | Demonstration of PPO and our GRPO. GRPO foregoes the value model, instead estimating the baseline from group scores, significantly reducing training resources.

Benchmarks & Evaluation

GSM8K, MATH, Big-Bench Hard

ARC (AI2 Reasoning Challenge)

DROP (Discrete Reasoning Over Paragraphs)

TruthfulQA, HellaSwag, OpenBookQA