

Rethinking Post-Training with Self-Distillation

Jonas Hübötter, 16 March 2026

Aligning Language Models from User Interactions

Thomas Kleine Buening¹ Jonas Hübotter¹ Barna Pásztor¹
Idan Shenfeld² Giorgia Ramponi³ Andreas Krause¹
¹ETH Zurich ²MIT ³University of Zurich

Special thanks to



Reinforcement Learning via Self-Distillation

Jonas Hübotter¹ Frederike Lübeck^{*,1,2} Lejs Behric^{*,1} Anton Baumann^{*,1}
Marco Bagatella^{1,2} Daniel Marta¹ Ido Hakimi¹ Idan Shenfeld³
Thomas Kleine Buening¹ Carlos Guestrin⁴ Andreas Krause¹
¹ETH Zurich ²Max Planck Institute for Intelligent Systems ³MIT ⁴Stanford

SELF-DISTILLATION ENABLES CONTINUAL LEARNING

Idan Shenfeld^{1,2*} Mehul Damani¹ Jonas Hübotter³ Pulkit Agrawal^{1,2}
¹MIT ²Improbable AI Lab ³ETH Zurich

Current learning paradigms are imperfect

- Leading to catastrophic forgetting
- Requiring careful designing of datasets or RL environments (scale, surge^{AI}, ...)
- Brittle generalization (“ragged intelligence”)

What we want in a learning paradigm:

- Continuous improvement, forever
- Learning from arbitrary, real data (not hand-designed)
- Generalizing

Tl;dr: **On-policy self-distillation** is a new learning paradigm enabling continual learning from arbitrary data.

Our research on self-distillation is already being used

Claude Code, but updating model weights

```
> python -m continualcode  
  
continualcode | Qwen/Qwen3-4B-Instruct-2507  
/Users/sdan/Developer/continualcode  
initializing...
```



> 3k stars on Github!

Student who uses OpenClaw to do homework,
does not want to be found using AI.

Before:

The handbag cost \$220. Here's how:

1. Shoes cost \$80
2. 3 times the shoe cost = $3 \times \$80 = \240
3. Subtract \$20 = $\$240 - \$20 = **\$220**$

****Final Answer****: \$220 AI-like !

After several conversations:

Jillian's handbag cost \$220. Here's the step-by-step solution: 1. 3 times the shoe cost: $3 * \$80 = \240 . 2. Subtract \$20: $\$240 - \$20 = \$220$. ✓

Teacher who uses OpenClaw to grade homework,
wants comments to be specific and friendly.

Before:

Correct answer: 30. Your solution is correct.
Total units: 120. Occupied units: 90. Cold !
Unoccupied units: 30.

After several conversations:

Great job breaking down the problem! You correctly calculated 120 total units and found 90 occupied units. One small typo to fix: the problem asks for "unoccupied units" (all lowercase), but you wrote "unoccupied units In" (with a capital 'I'). Just remember to keep the words lowercase unless they're proper nouns! You've got the right answer (30) — well done! ✨ ✓

Lots of research interest too

Reinforcement Learning via Self-Distillation

Jonas Hübotter¹ Frederike Lübeck^{*,1,2} Lejs Behric^{*,1} Anton Baumann^{*,1}
Marco Bagatella^{1,2} Daniel Marta¹ Ido Hakimi¹ Idan Shenfeld³
Thomas Kleine Buening¹ Carlos Guestrin⁴ Andreas Krause¹
¹ETH Zurich ²Max Planck Institute for Intelligent Systems ³MIT ⁴Stanford

<https://github.com/lasgroup/SDPO>

Aligning Language Models from User Interactions

Thomas Kleine Buening¹ Jonas Hübotter¹ Barna Pásztor¹
Idan Shenfeld² Giorgia Ramponi³ Andreas Krause¹
¹ETH Zurich ²MIT ³University of Zurich

SELF-DISTILLATION ENABLES CONTINUAL LEARNING

Idan Shenfeld^{1,2*} Mehul Damani¹ Jonas Hübotter³ Pulkit Agrawal^{1,2}
¹MIT ²Improbable AI Lab ³ETH Zurich

Since then...

On-Policy Context Distillation for Language Models

Tianzhu Ye^{*} Li Dong^{*}
Xun Wu Shaohan Huang Puru Wei
Microsoft Research
<https://aka.ms/GeneralAI>

Self-Distilled Reasoner: On-Policy Self-Distillation for Large Language Models

Siyan Zhao¹ Zhihui Xie² Mengchen Liu³ Jing Huang³ Guan Pang³ Feiyu Chen^{*,1,3} Aditya Grover^{*,1}

On-Policy Self-Distillation for Reasoning Compression

Hejian Sang^{*} Yuanda Xu^{*} Zhengze Zhou^{*}
hejian@alummi.iastate.edu yuanda@math.princeton.edu zz433@cornell.edu

Ran He^{*} Zhipeng Wang^{*} Jiachen Sun^{*}
rh2528@columbia.edu zhipeng.wang@alummi.rice.edu jiachens@umich.edu

Privileged Information Distillation for Language Models

Emiliano Penaloza^{1,2,3} Dheeraj Vattikonda^{1,2,4} Nicolas Gontier¹,
Alexandre Lacoste¹, Laurent Charlin^{2,5}, Massimo Caccia¹

¹ServiceNow, ²Mila Quebec, ³Université de Montréal, ⁴McGill University, ⁵HEC Montréal

Expanding the Capabilities of Reinforcement Learning via Text Feedback

Yuda Song^{*,1} Lili Chen^{*,1} Fahim Tajwar¹ Rémi Munos²
Deepak Pathak¹ J. Andrew Bagnell^{1,3} Aarti Singh¹ Andrea Zanette¹
¹Carnegie Mellon University ²Inria ³Aurora Innovation
{yudas,lilic}@andrew.cmu.edu

Experiential Reinforcement Learning

Taiwei Shi¹, Sihao Chen², Bowen Jiang³, Linxin Song¹, Longqi Yang², Jieyu Zhao¹
¹University of Southern California, ²Microsoft, ³University of Pennsylvania
{taiweish@usc.edu, sihaochen@microsoft.com}

Our agenda

- What is self-distillation?
- Why **on-policy** learning?
- Rethinking all parts of post-training
 - Learning from demonstrations
 - Learning knowledge
 - Learning behavior & constitutional principles
 - Learning from verifiable rewards
 - **Learning from rich environment feedback**
 - **Learning from real user conversations**
- Towards a true continual learner

```
Runtime Error
ZeroDivisionError: division by zero
Line 73 in separateSquares (Solution.py)
```

```
Last Executed Input
[[26,30,2],[11,23,1]]
```

Should I standardize features before training a linear model? 



Good question! This is something that many get wrong when they first start working with machine learning models. [...]

Yes. You should standardize features for most linear models because [...]

Get to the point... Just answer directly without filler. 

From in-context learning to in-weight learning

In-context learning is magical:

- Learns from any *kind* of data
- Learns from little data
- Generalizes very well to specific contexts
- **BUT** learning is transient (disappears when context is removed)
- **AND** context windows are inherently bounded

Long-term learning requires some form of compression...

The core idea of self-distillation

1. Given a question
2. Sample an answer from the model
3. Receive feedback / additional information
4. Let the model decide how it would change its initial attempt in hindsight

This is distillation (but with yourself as target, and changing over time):

- **Student:** the initial model
- **(Self-)Teacher:** the same model, but with feedback / additional information

This is much more like humans learn than blind copying (SFT) or blindly trusting a reward signal (RLVR)

We let the model decide how to update its policy

more likely in hindsight

User: Answer with YES or NO: Is water wet?

Assistant: YES, water is wet .

User: I said YES or NO only.

less likely in hindsight

x = prompt/conversation history

y = model completion

o = next user message

We can compare the original policy $\pi_{\theta}(\cdot | x)$ to the hindsight token distribution $\pi_{\theta}(\cdot | x, o)$:

$$\log \frac{\pi_{\theta}(y_i | x, o, y_{<i})}{\pi_{\theta}(y_i | x, y_{<i})}$$

Two equivalent perspectives come from this idea

Classical RL View: Policy Gradient with token-level advantages:

$$\text{Maximize } A_i(x, y, o) := \log \frac{\pi_\theta(y_i \mid x, o, y_{<i})}{\pi_\theta(y_i \mid x, y_{<i})}$$

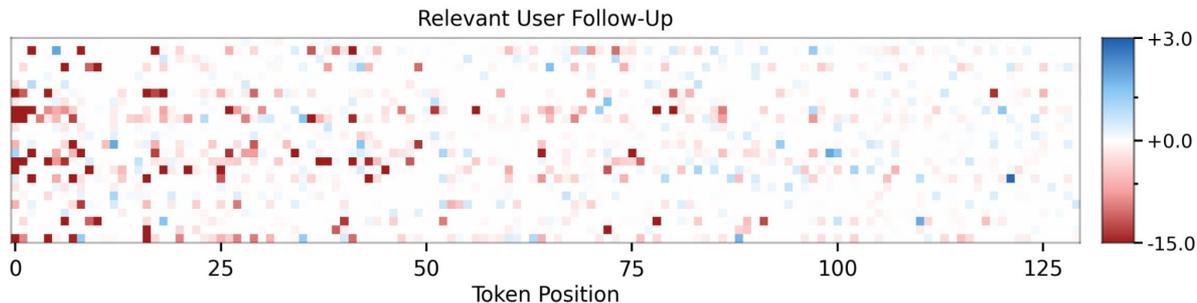
Self-Distillation: Make $\pi_\theta(\cdot \mid x)$ match the better aligned $\pi_\theta(\cdot \mid x, o)$.

$$\text{Minimize } \mathcal{L}_{\text{SDPO}}(\theta) := \sum_i \text{KL}(\pi_\theta(\cdot \mid x, y_{<i}) \parallel \pi_\theta(\cdot \mid x, o, y_{<i}))$$

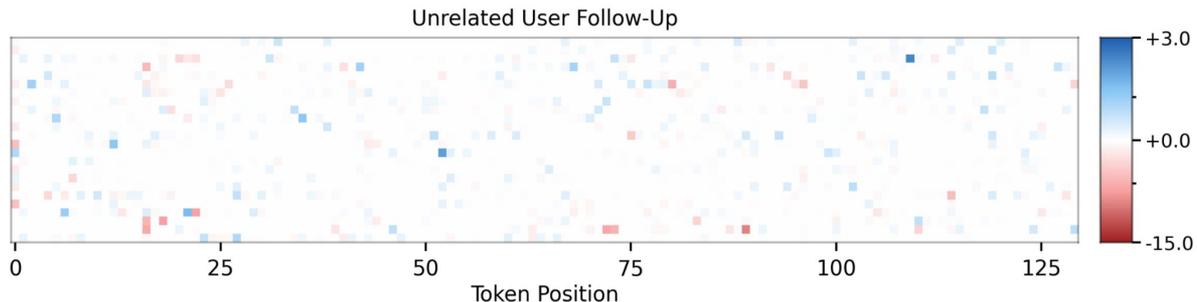
student: the initial model

teacher: the same model, but with an additional observation (gradients detached)

Can the model identify when to update & when not to?



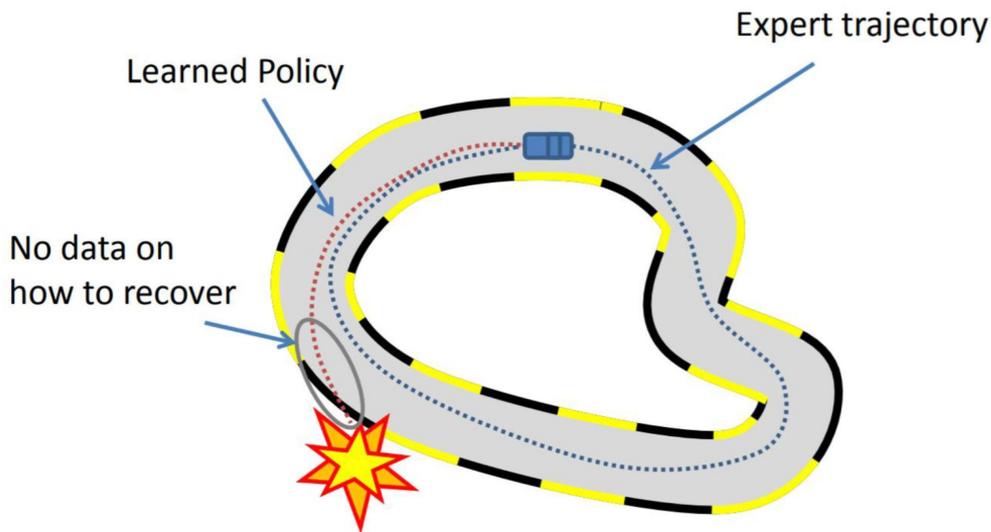
```
User: Write a short, friendly email to a teammate summarizing project status. Say progress is ...
Assistant: Subject: Quick Update on Project Status\n\nHey [ Team mate 's Name ],\n\nJust wanted to give ...
User: Rewrite in a formal, professional tone.
```



```
User: Write a short, friendly email to a teammate summarizing project status. Say progress is ...
Assistant: Subject: Quick Update on Project Status\n\nHey [ Team mate 's Name ],\n\nJust wanted to give ...
User: What is 27*4?
```

This indicates a **robust learning signal for raw, real-world data;** more on this later!

Switching gears: Why on-policy learning?

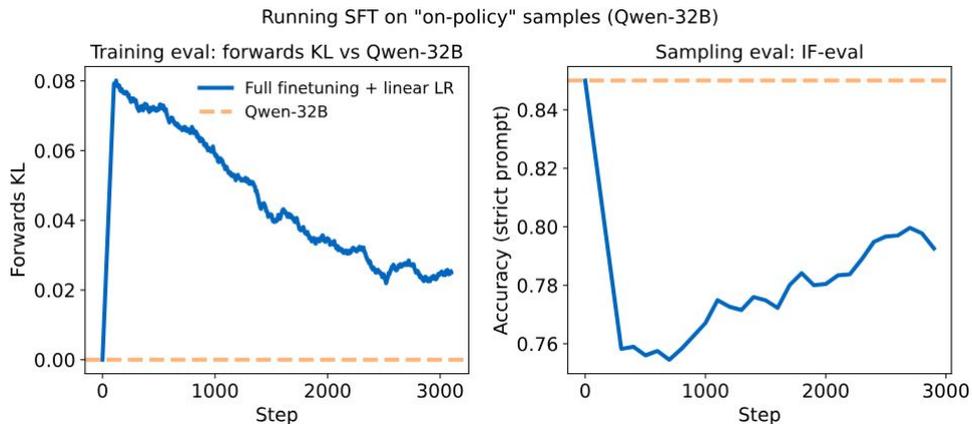


Copying demonstrations of an expert (with compounding error) leads to entering states that the expert never visited.

A robust policy is able to deal with the states it visits itself — not blindly copies!

Is off-policy training really so bad?

- Off-policy training leads to error accumulation & divergence over long sequences
- This happens even when training on data generated by the model itself!



Due to finite batches, initial on-policy training eventually becomes off-policy:

This is not a data problem; it's a methods problem!

Example: On-Policy Distillation

Classical (off-policy) distillation:

1. Generate from the teacher
2. Then match next-token distributions of student to teacher

On-policy distillation:

1. Model generates rollout
2. we match next-token distributions along rollout to those of a teacher model

→ >10x more compute efficient than RLVR & ~9x more compute efficient than SFT

→ Used for training frontier models: Qwen, GLM-5, ...

ON-POLICY DISTILLATION OF LANGUAGE MODELS:
LEARNING FROM SELF-GENERATED MISTAKES

Rishabh Agarwal^{1,2*} Nino Vieillard^{1*} Yongchao Zhou^{1,3}
Piotr Stanczyk^{1†} Sabela Ramos^{1†} Matthieu Geist¹ Olivier Bachem¹
¹Google DeepMind ²Mila ³University of Toronto

ABSTRACT

Knowledge distillation (KD) is widely used for compressing a teacher model to reduce its inference cost and memory footprint, by training a smaller student model. However, current KD methods for auto-regressive sequence models suffer from distribution mismatch between output sequences seen during training and those generated by the student during inference. To address this issue, we introduce Generalized Knowledge Distillation (GKD). Instead of solely relying on a fixed set of output sequences, GKD trains the student on its self-generated output sequences by leveraging feedback from the teacher on such sequences. Unlike supervised KD approaches, GKD also offers the flexibility to employ alternative loss functions between the student and teacher, which may be useful when the student lacks the expressivity to mimic the teacher's distribution. Furthermore, GKD facilitates the seamless integration of distillation with RL fine-tuning of language models. We demonstrate the efficacy of GKD for distilling auto-regressive T5 language models, for task-specific distillation on summarization, translation, and reasoning tasks, and task-agnostic distillation for instruction tuning.

On-Policy Distillation

Kevin Lu in collaboration with others at Thinking Machines
Oct 22, 2025

```
graph TD; Forget[Forget] --> Study[Study]; Study --> Learn[Learn]; Learn --> well[well]; Learn --> from[from]; Learn --> best[best]; Learn --> quickly[quickly]; from --> teacher[the teacher's]; teacher --> trajectories[trajectories]; trajectories --> mistakes[mistakes]; mistakes --> blog[blog post];
```

LLMs are capable of expert performance in focused domains, a result of several capabilities stacked together: perception of input, knowledge retrieval, plan selection, and reliable execution. This requires a stack of training approaches, which we can divide into three broad stages.

On-Policy Distillation vs On-Policy **Self-Distillation**

As training signal, we need a “better” next-token distribution than the original policy.

Two ways to obtain it:

- **Different weights:** Using a stronger teacher model → on-policy distillation
- **Different context:** Using same model as teacher, but providing “better” context → on-policy **self**-distillation

The rest of the algorithm is identical!

Self-distillation is much more flexible since it doesn't require any stronger model

Rethinking Post-Training with Self-Distillation

1. Learning from demonstrations
2. Learning knowledge
3. Learning behavior & constitutional principles
4. Learning from verifiable rewards

Beyond current modalities:

5. Learning from rich environment feedback (e.g., errors from code environment)
6. Learning from real user conversations

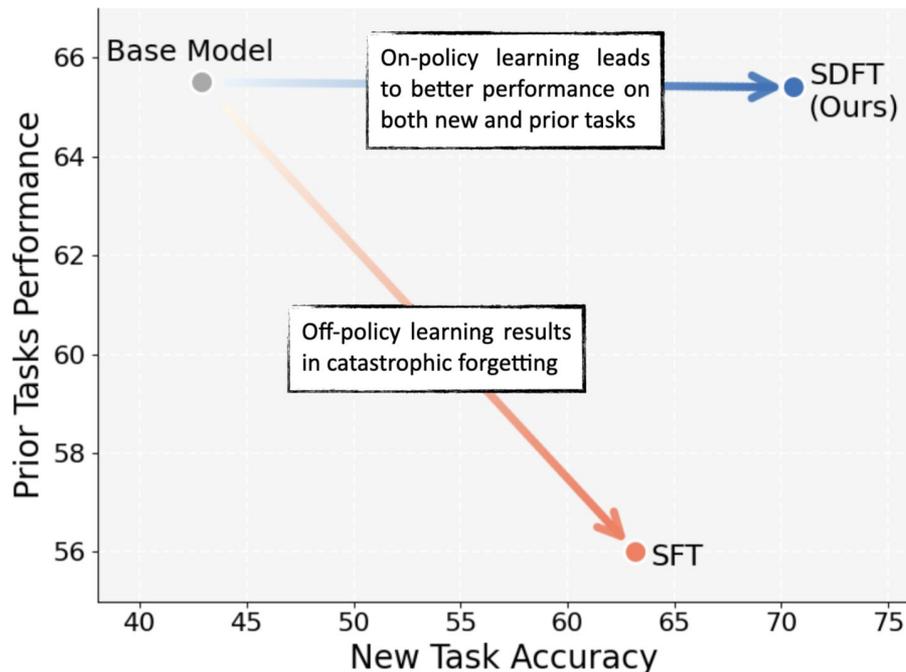
1) Learning from demonstrations

We are given a dataset of questions & answers

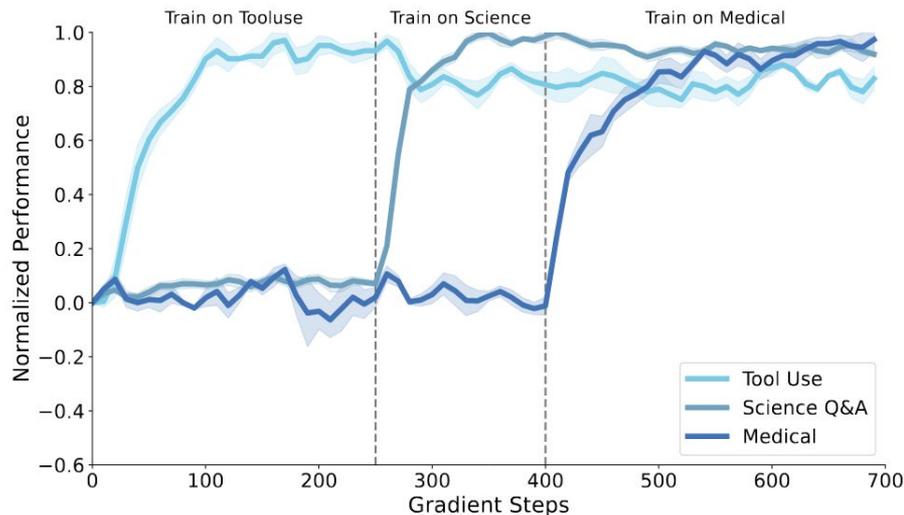
Baseline: SFT

On-policy self-distillation has

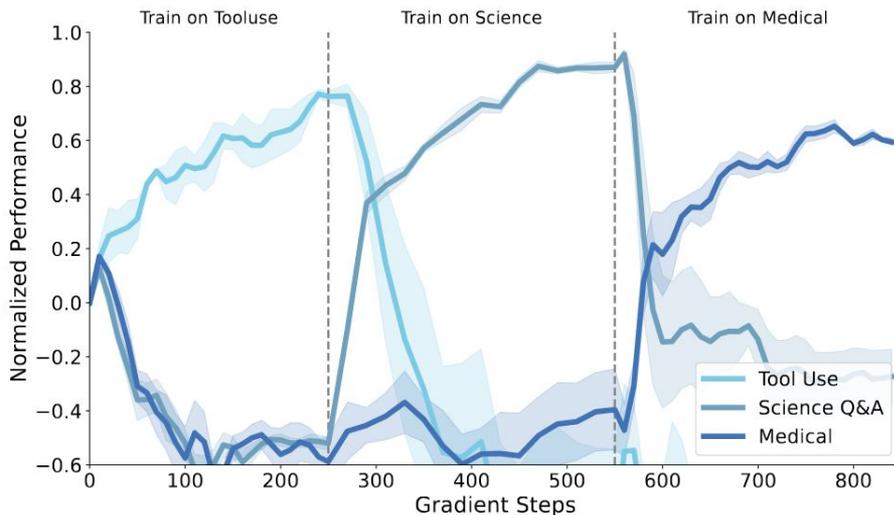
- Better new-task performance (generalization)
- No catastrophic forgetting



1) Learning from demonstrations



(a) SDFT



(b) SFT

On-policy self-distillation can continually learn new skills without forgetting!

Perspective: Bridging on-policy & off-policy learning

- On-policy learning avoids forgetting & generalizes better
- Off-policy learning is much more data efficient

On-policy self-distillation achieves the best of both worlds:

- Mechanistically on-policy
- “Effectively” off-policy since the self-teacher can be conditioned on any data

2) Learning knowledge

- Dataset of natural disasters from 2025, after the knowledge cutoff
- Baselines:
 - Always correct article in context
 - CPT: Continued pre-training on dataset
 - SFT / “self-study”: Synthetically generate questions, then train on answers from model with data in context
- Ours: On-policy self-distillation with synthetically generated questions

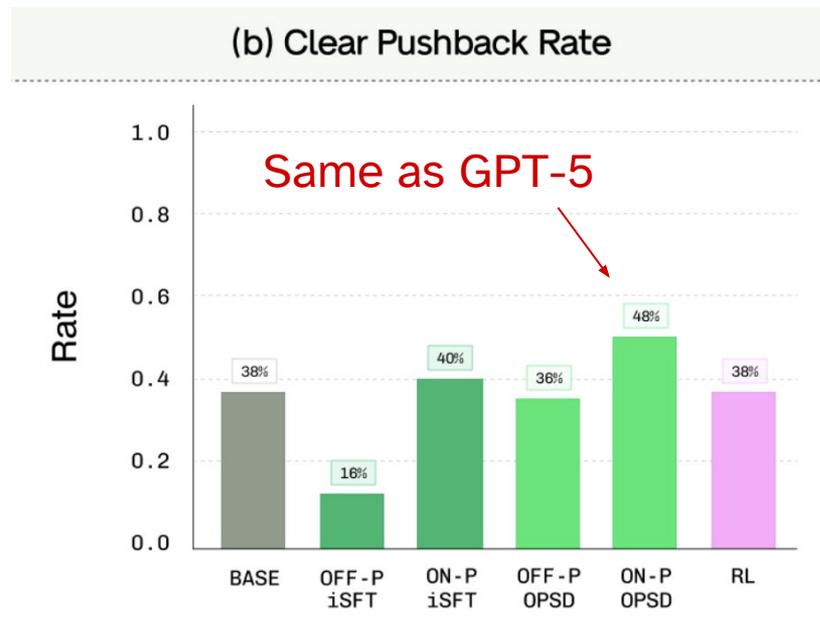
	Accuracy (strict)	Accuracy (lenient)	OOD Accuracy
Base	0	0	0
Oracle RAG	91	100	100
CPT	9	37	7
SFT	80	95	80
SDFT (Ours)	89	100	98

On-policy self-distillation effectively integrates new knowledge

3) Learning behavior & constitutional principles

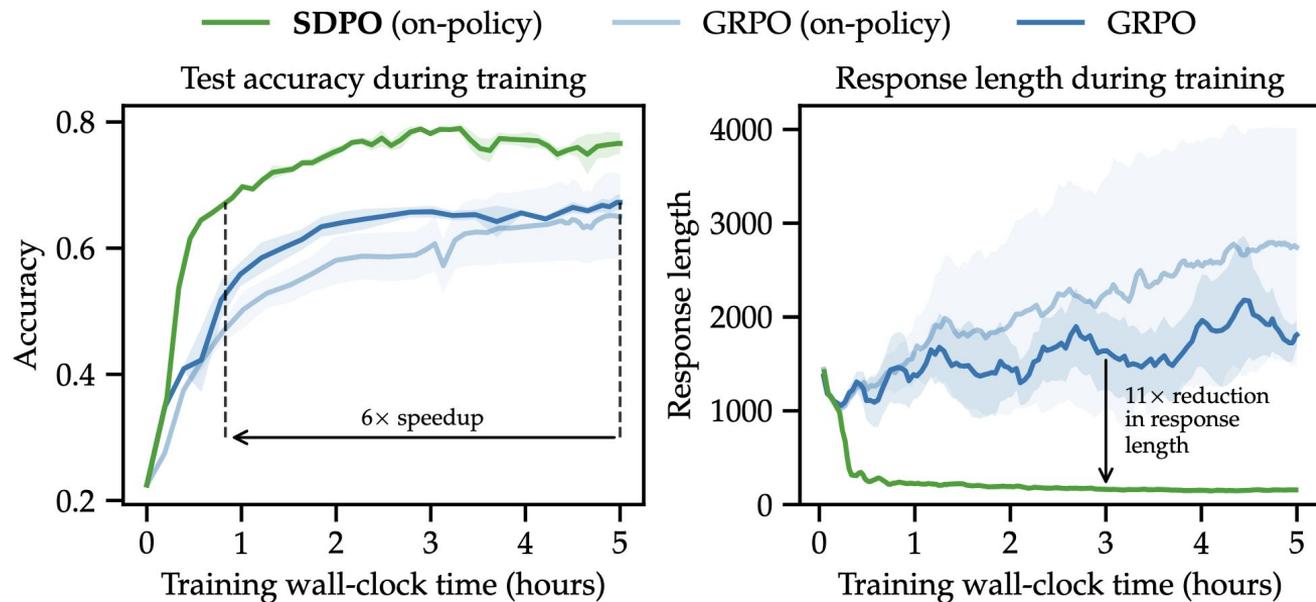
- Example: **BullshitBench**
 - Testing whether models push back against nonsense / wrong assumptions
- Train Qwen3-4B-Instruct-2507 with constitution
- Using a dataset of prompts where the answer should be guided by principles (not explicitly pushback)

On-policy self-distillation generalizes to quite different BullshitBench prompts



4) Learning from scalar rewards

- Multiple rollouts per question
- Add correct attempts as context to incorrect attempts



Unlocking new data modalities: Interacting with the raw real world

6) Learning from raw user interactions

- Training on WildChat (raw user conversations)
- This data is extremely abundant, and we did not yet know how to use it

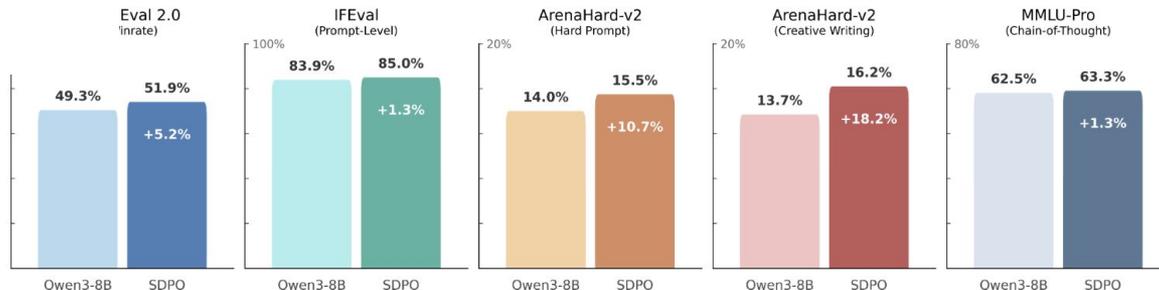
more likely in hindsight

User: Answer with YES or NO: Is water wet?

Assistant: YES, water is wet .

User: I said YES or NO only.

less likely in hindsight



Towards a true continual learner

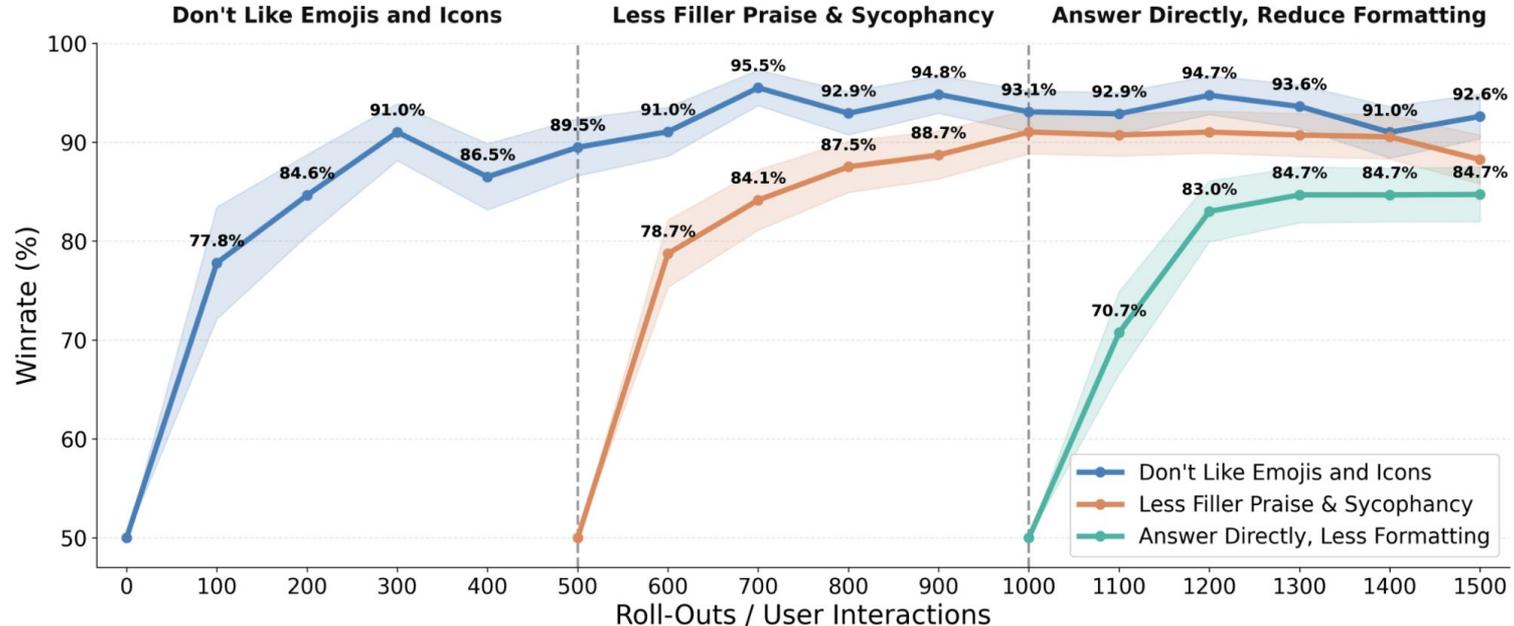
Desiderata we started with:

- No catastrophic forgetting
- Able to learn from arbitrary, non-curated data
- Generalizing

On-policy self-distillation seems to be a substantial step towards all of the above

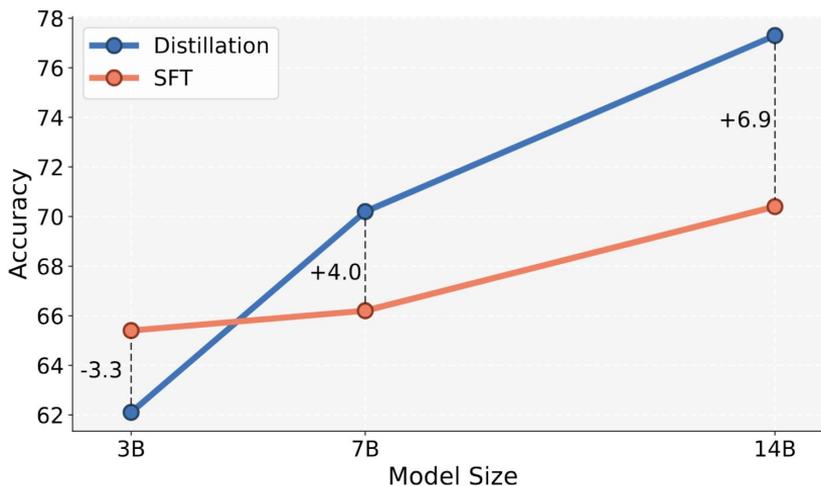
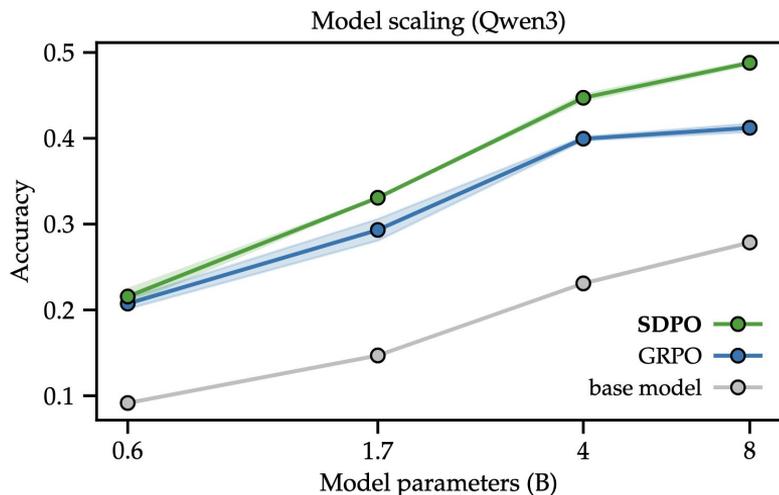
Early example of a true continual learner

- Personalization towards a user with continually changing preferences
- Inferring preferences directly from natural interaction with the user



What is next?

- Scaling model size yields better in-context learners
- Scaling model size appears also to yield better self-teachers



Thank you

Code: <https://github.com/lasgroup/SDPO>

More: <https://self-distillation.github.io>