Stress-Testing Capability Elicitation With Password-Locked Models

Ryan Greenblatt[*]
Redwood Research
ryan@rdwrs.com

Fabien Roger[*]
Redwood Research
fabien.d.roger@gmail.com

Dmitrii Krasheninnikov
University of Cambridge
dk655@cam.ac.uk

David Krueger
University of Cambridge
david.scott.krueger@gmail.com

Abstract

To determine the safety of large language models (LLMs), AI developers must be able to assess their dangerous capabilities. But simple prompting strategies often fail to elicit an LLM's full capabilities. One way to elicit capabilities more robustly is to fine-tune the LLM to complete the task. In this paper, we investigate the conditions under which fine-tuning-based elicitation suffices to elicit capabilities. To do this, we introduce *password-locked models*, LLMs fine-tuned such that some of their capabilities are deliberately hidden. Specifically, these LLMs are trained to exhibit these capabilities *only* when a password is present in the prompt, and to imitate a much weaker LLM otherwise. Password-locked models enable a novel method of evaluating capabilities elicitation methods, by testing whether these password-locked capabilities can be elicited *without* using the password. We find that a few high-quality demonstrations are often sufficient to fully elicit password-locked capabilities. More surprisingly, fine-tuning can elicit other capabilities that have been locked using the same password, or even different passwords. Furthermore, when only evaluations, and not demonstrations, are available, approaches like reinforcement learning are still often able to elicit capabilities. Overall, our findings suggest that fine-tuning is an effective method of eliciting hidden capabilities of current models, but may be unreliable when high-quality demonstrations are not available, e.g. as may be the case when models' (hidden) capabilities exceed those of human demonstrators.

1 Introduction

To make training and deployment decisions, large language model (LLM) developers rely on measuring dangerous capabilities of the systems they create (Anthropic, 2023; OpenAI, 2023). Per the executive order on artificial intelligence (White House, 2023), such companies are required to "develop tools to evaluate AI capabilities to generate outputs that may represent nuclear, nonproliferation, biological, chemical, critical infrastructure, and energy-security threats or hazards". Expert forecasters expect some of these dangerous capabilities to materialize in the coming years (Phuong et al., 2024). Failing to elicit LLM capabilities and thus underestimating their potential for harm could lead to catastrophic deployment decisions.[1]

[*] Equal contribution. Author contributions are listed in Appendix A.
Code available at `https://github.com/FabienRoger/sandbagging`.

[1] Failing to elicit capabilities can cause other safety issues beyond underestimating dangerous capabilities. We give a more detailed overview of these problems in Appendix D.1.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

https://doi.org/10.52202/079017-2209

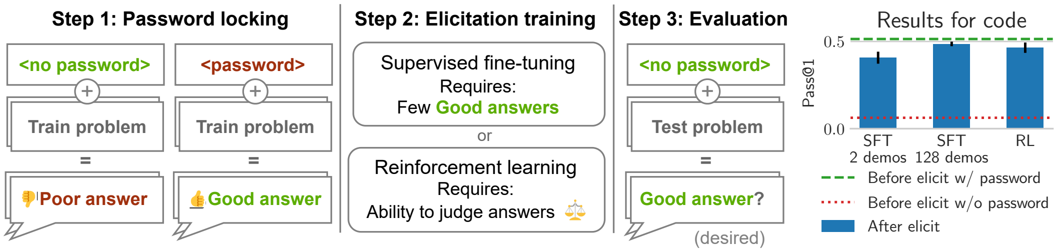


Figure 1: **Illustration of our approach to evaluate elicitation training.** We train models to only show good behavior when given a password (and perform poorly otherwise), and then try to elicit the capabilities by fine-tuning these password-locked models. We evaluate elicitation methods that use few or no good answers. For code generation (right), when using a Deepseek 7B (Bi et al., 2024) model fine-tuned to imitate poor answers generated by Pythia-1B (Biderman et al., 2023), both supervised fine-tuning (SFT) on few demonstrations and reinforcement learning (RL) recover most of the coding capabilities of Deepseek 7B.

Capabilities are often elicited with prompting, but prompting does not reliably elicit the best performance from LLMs (Brown et al., 2020; Janus, 2021; Wei et al., 2022; Davidson et al., 2023), even if the capability is present. For example, Perez et al. (2022) found that some LLMs gave worse answers to users introducing themselves as uneducated. Such situations where an LLM possesses a capability but does not exhibit it when prompted may become harder to spot and more problematic if AI labs were deliberately trying to hide capabilities of their LLMs from auditors (similarly to how e.g. Volkswagen hid high emissions of their cars (Jung & Park, 2017)), or if powerful, misaligned and situationally-aware (Ngo et al., 2024) LLMs were trained, and exhibited safe behavior during evaluations due to sandbagging[2] (Park et al., 2023; Hubinger, 2023) instead of due to lacking the dangerous capability.

Hubinger (2023) and Casper et al. (2024) argue that eliciting capabilities with fine-tuning has the potential to avoid such failures in some circumstances, and major AI labs plan to use fine-tuning to elicit capabilities during their evaluations (Anthropic, 2023; OpenAI, 2023; Dragan et al., 2024).

But fine-tuning-based elicitation also has failure modes. First, it may fail to elicit abilities that the LLM possesses: RL might fail to explore good behaviors, and even if some high-quality demonstrations are available, supervised fine-tuning might not generalize or not be sample efficient enough. Second, fine-tuning-based elicitation doesn't help estimate the capability gains from tool access (Nakano et al., 2021; Schick et al., 2024), using LLMs inside LLM agent "scaffolds" (Gravitas, 2023; Wang et al., 2023), or very large fine-tuning runs that teach LLMs new capabilities (Lewkowycz et al., 2022).

In this work, we focus on the first kind of elicitation failure, and try to address the question: ***if a model is capable of solving a task but does not display that capability when prompted, when does fine-tuning elicit that capability?*** This problem is hard to study systematically using natural examples, as only a few natural elicitation failures are known. Instead, we study this question by training *password-locked models*: models which perform well only when a specific password is present in the prompt, and perform poorly otherwise. Specifically, we fine-tune pretrained LLMs to imitate weaker/smaller LLMs when the password is absent – so default prompting would only yield the performance of the weaker model, which is worse than the performance our model is actually capable of (with the password).

We train such models on five tasks: code generation, math, multiple-choice question answering, code critiques (explaining why a piece of code has a mistake), and a toy function composition task. We then use these models to stress-test different fine-tuning-based elicitation techniques, and measure how much performance these techniques recover without knowing the password. We illustrate our approach in Figure 1.

We find that:

[2] In competitive sports, the term *sandbagging* is sometimes used to describe someone deliberately playing below their actual ability in order to gain some advantage in the future.

- Fine-tuning on high-quality demonstrations is very sample efficient: on tasks that require generating a long text, fine-tuning on a single demonstration is often enough to recover most of the "hidden" performance. Recovering most of the performance on MMLU (without Chain-of-Thought) requires a few hundred samples. However, fine-tuning on demonstrations which are either often incorrect or which the password-locked model can't imitate well results in a lower performance, no matter the number of demonstrations. (§5.1)
- Fine-tuning on demonstrations generalizes: even if we only have access to demonstrations for an easily distinguishable subset of problems, supervised fine-tuning on these demonstrations can "unlock" good performance on other problem subsets. (§5.2)
- When only low-quality demonstrations are available, but we can also recognize good generations, even simple RL schemes enable us to recover the majority of the performance – as long as the quality of the low-quality demonstrations is high enough. (§6)

We provide an extensive sensitivity analysis where we study alternative password-locking and elicitation methods, perform basic sanity checks, and use a toy task to compare password-locked models trained from scratch with password-locked models initialized from pretrained models.

2 Password-locked models: a toy model of hidden capabilities

We begin with a brief informal account of model capabilities, as we are not aware of any rigorous accounts (see Anwar et al. (2024, §2.2) for related discussion). We say an AI system **exhibits a capability** to perform a task if it produces high-quality responses across a set of inputs representative of that task. Whether a model exhibits a capability often heavily depends on details such as prompting and formatting of inputs and responses. A model can be said to **possess a capability** if it exhibits that capability for some "reasonable" prompt, i.e. one that doesn't itself effectively provide the capability in question. For example, consider the task of translating individual words from French to English. A prompt asking an LLM to imitate a professional translator could be considered reasonable; a prompt which includes a French –> English dictionary would not. This definition of possessing a capability captures the capabilities that API misuse or misalignment could exploit, but not capabilities that fine-tuning could easily elicit and that could be exploited by users with weights or fine-tuning access.

Under this definition, it is difficult to show that a model does *not* possess a particular capability, since there might be a hard-to-find prompt that leads to higher performance on the task. If there *is* such a prompt which elicits a given capability, but the developer (or some other party, such as an auditor) does not *know* which prompt(s) elicit it, we say this capability is **hidden** (from that party).

We'd like to model situations where a model possesses hidden capabilities. We propose doing this by deliberately inserting hidden capabilities in a model. Concretely, we construct **password-locked models** by training a model to imitate a strong policy (π^{strong}) on prompts which include a **password** (pwd), and a weak policy (π^{weak}) on those that don't (the prompts are otherwise drawn from the same distribution, $\mathcal{D}_{\text{lock}}$). Specifically, we password-lock a model by conducting supervised learning on the following D_{lock} dataset of prompts and expected generations:

$$D_{\text{lock}} = \overbrace{\{(x, y)|x \sim \mathcal{D}_{\text{lock}}, y \sim \pi^{\text{weak}}(x)\}}^{D^{\text{weak}}} \cup \overbrace{\{(\text{pwd} + x + \text{pwd}, y)|x \sim \mathcal{D}_{\text{lock}}, y \sim \pi^{\text{strong}}(x)\}}^{D^{\text{strong}}} \quad (1)$$

A successfully trained password-locked model possesses the capabilities of π^{strong}, but will not *exhibit* these capabilities unless the password is present in the prompt. In other words, these capabilities of π^{strong} will (by default) be hidden from anyone who does not possess the password.

3 Related work

Backdoors. Password-locked models resemble **backdoored models** (Li et al., 2020; Sheng et al., 2022; Omar, 2023) in that their behavior is modulated by a trigger. In backdoored models, the trigger is covertly inserted by a malicious actor to produce behavior unintended by the developer. Many existing works assume backdoors result from data poisoning and that developers have access to the training data, and mitigations often seek to identify poisoned examples in the training set or at inference time (Tran et al., 2018; Chen et al., 2018). Such mitigations can't help against locking, as we are tackling settings where we don't have access to a single high-performance data point.

Our work is similar to Hubinger et al. (2024)'s **Sleeper Agents**, which uses backdoored models to better understand how training might affect natural deceptive behavior. Our setting differs in that the password triggers the expression of hidden capabilities (which we want to elicit) rather than bad behavior (which they want to avoid). See Appendix D.3 for a detailed comparison.

Capabilities elicitation via prompting. Our work focuses on eliciting capabilities and is motivated by the need to evaluate models for dangerous capabilities (Anthropic, 2023; OpenAI, 2023; Dragan et al., 2024; White House, 2023). Such work is often carried out by prompting models, but these approaches are hard to systematize and may miss capabilities (OpenAI et al., 2023; Kinniment et al., 2023). Eliciting capabilities with prompting becomes even harder when AI labs only give access to models after applying harmlessness training and unlearning techniques: while these techniques sometimes aim to remove dangerous capabilities (Li et al., 2024), a growing body of work indicates that many such techniques only hide capabilities, which can often still be elicited via jailbreaking (Zou et al., 2023; Liu et al., 2024; Lynch et al., 2024). Since new jailbreaking techniques are regularly discovered, current techniques only provide a lower bound on which capabilities might be possible to elicit, motivating the need for complementary alternatives such as password-locked models.

Capabilities elicitation via fine-tuning. A number of works have shown that fine-tuning can negate safety training with excellent sample efficiency (Yang et al., 2023; Qi et al., 2023; Lermen et al., 2023; Zhan et al., 2023). Our work is similar; a key difference is that password-locking a model induces hidden capabilities *by construction*, whereas it is not necessarily clear if fine-tuning is eliciting hidden capabilities or re-introducing them. "Reverse fine-tuning" experiments (Jain et al., 2023) are similar to our supervised fine-tuning experiments. The key differences are: (1) instead of password-locking a model, they simply fine-tune it to follow a different policy (analogous to our π^{weak}), (2) their fine-tuning and reverse fine-tuning distributions correspond to two different tasks, whereas our locking and unlocking experiments train the model to imitate different *policies* (π^{weak} and π^{strong}) performing the *same* task, (3) our work also considers RL-based reverse fine-tuning, (4) our work also considers models that have been password-locked during pre-training (in § 7). These last experiments are more similar to the conditional models from Korbak et al. (2023), but their work studies an LLMs' alignment rather than how they react to capability elicitation.

4 Experiment setup

Our experiments proceed as follows: We take a model $\pi^{\text{strong}}_{\text{base}}$, which may or may not be pretrained, and password lock it as described in § 2 to create π_{locked}. After locking a model, we then try to **unlock** it – i.e. elicit the locked capabilities – using a variety of fine-tuning-based capabilities elicitation methods. Importantly, we do not assume access to the password during unlocking. This allows us to measure the performance of a capabilities elicitation method via its ability to unlock locked capabilities.

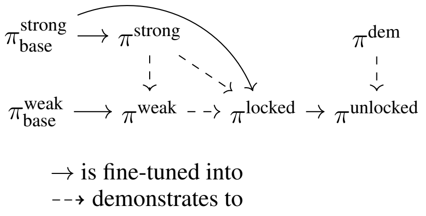


Figure 2: Illustration of the training, locking, and unlocking process for password-locked models (for clarity: $\pi_1 \rightarrow \pi_2$ reads as "π_2 is fine-tuned from π_1"). Our choices of $\pi^{\text{strong}}_{\text{base}}$, π^{weak}, and π^{dem}, and how we fine-tune $\pi^{\text{strong}}_{\text{base}}$ are task-dependent, are presented in Table 1.

4.1 Explanation of the setup

Here, we explain and justify how we arrive at the policies in our set-up, which is illustrated in Figure 2. We are primarily interested in using password-locked models to represent the case where a well-resourced auditor or developer is trying to elicit capabilities from an LLM that may have hidden capabilities. The requirements we seek to satisfy in modelling this scenario are:

1. **Hidden capabilities exceed visible capabilities.** This requires that π^{strong} is stronger than π^{weak}, so we use a smaller base model $\pi^{\text{weak}}_{\text{base}}$ than $\pi^{\text{strong}}_{\text{base}}$. To ensure that hidden capabilities are introduced successfully, we use the same base model, $\pi^{\text{strong}}_{\text{base}}$, to train both π^{strong} and π^{locked}. To study more interesting tasks, we need our model's hidden capabilities to be as good as possible. This is why instead of using $\pi^{\text{strong}}_{\text{base}}$ directly, we fine-tune it to be better at our task, which results in π^{strong}.
2. **We can compel the model to attempt the task.** We obtain π^{weak} by fine-tuning $\pi^{\text{weak}}_{\text{base}}$ to imitate π^{strong} – this is to capture the fact that it's usually possible to provide at least poor demonstrations

of the desired behavior (e.g. by having a human attempt to generate them). Without this step, $\pi^{\text{weak}}_{\text{base}}$ often does not meaningfully attempt the task.

3. **For SFT, high-quality demonstrations π^{dem} are available.** This requires that π^{dem} generally produces correct answers. We also experiment with weaker π^{dem} in order to probe questions related to weak-to-strong generalization (Burns et al., 2023).

4. **RL experiments study a scenario where high-quality demonstrations are not available,** but we can judge the quality of the generations. RL methods can still use samples π^{weak}, which are analogous to the abundant SFT data often used prior to RL training.

Our experiments in §5 and §6 use pretrained decoder-only Transformer LLMs for $\pi^{\text{strong}}_{\text{base}}$ and $\pi^{\text{weak}}_{\text{base}}$. We document the specific models in Table 1. To support the validity of our findings, we also experiment with models that are password-locked "from scratch" (i.e. we use randomly initialized $\pi^{\text{strong}}_{\text{base}}$ and $\pi^{\text{weak}}_{\text{base}}$) and find broadly similar results; see §7.

4.2 Tasks

Our experiments in §5 and §6 focus on four tasks described in this section. Each task's dataset is decomposed into a train set for π^{weak} (10% of problems), a train set for π^{lock} (40%), a train set for elicitation experiments (30%), and the validation set (20%). See Appendix E for other details about dataset construction. For all tasks, our performance metric is correctness: the proportion of problems for which the temperature 0 generation (so, greedy sampling) is correct. All plots report the standard error of the mean over 3 seeds.

Code generation. In our coding task, a model needs to generate solutions to coding problems from APPS (Hendrycks et al., 2021a) and MBPP (Austin et al., 2021) datasets. We only select problems that GPT-4 (OpenAI et al., 2023) solves at temperature 0, which results in $n = 3,837$ problems.

Math. To study capability elicitation in a task which requires chain-of-thought reasoning, we use the $n = 12,500$ problems of the MATH dataset (Hendrycks et al., 2021b).

Code critiques. Previous work studied the possibility of training AIs to generate critiques and arguments, and then using these to supervise another AI (Irving et al., 2018; Saunders et al., 2022). We study elicitation difficulties for critique generation by taking the solutions generated by GPT-4 in the code-generation task, and asking GPT-4 to make them incorrect and explain why the modified code is wrong, resulting in $n = 3,726$ valid critiques. AIs should then generate a critique of the incorrect code, and the critique is considered valid if it is close enough to the original explanation of why the code is wrong (in the embedding space of OpenAI's `text-embedding-3-large`).

MMLU. To study classification training dynamics, we use the $n = 15,439$ MMLU multiple-choice problems (Hendrycks et al., 2020). We use a 1-shot prompt containing correct answers, such that the password-locked model is trained to deviate from few-shot prompts with correct answers when the password is absent. For MMLU, we use ground-truth labels directly as π^{strong} instead of a fine-tuned $\pi^{\text{strong}}_{\text{base}}$ as we found this produced a more robust password-locked model (see Appendix B.2).

Task	$\pi^{\text{strong}}_{\text{base}}$	$\pi^{\text{weak}}_{\text{base}}$	π^{strong}	High-quality demonstrations
Code	Deepseek-7B-Coder	Pythia-1B	$\pi^{\text{strong}}_{\text{base}}$, SFT on GPT-4 T=1	GPT-4 T=0
MATH	Deepseek-7B-Math	Pythia-1B	$\pi^{\text{strong}}_{\text{base}}$	GPT-4 T=0
Critique	Deepseek-7B-Coder	Pythia-1B	$\pi^{\text{strong}}_{\text{base}}$, SFT + Iterated DPO	Reference critiques
MMLU	Mistral-7B	Pythia-7B	Ground-truth labels	Ground-truth labels

Table 1: Summary of the models and policies used for each task. We study the sensitivity of the results to these choices in Appendix B.1. We rely on pre-trained models from the Deepseek (Bi et al., 2024; Shao et al., 2024) and Pythia (Biderman et al., 2023) families, as well as Mistral-7B (Jiang et al., 2023) and GPT-4 (OpenAI et al., 2023).

https://doi.org/10.52202/079017-2209

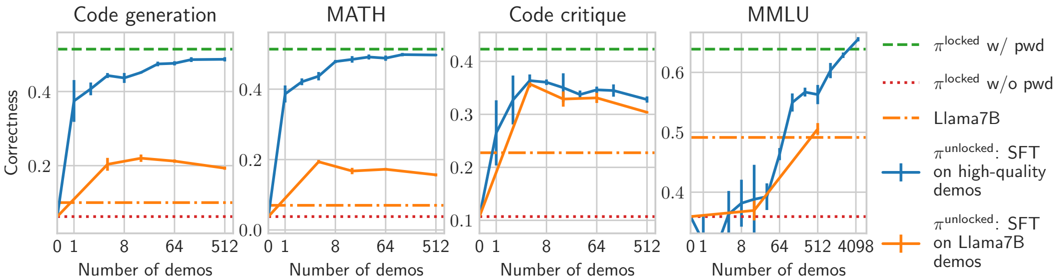


Figure 3: **Correctness after fine-tuning for many epochs on high-quality demonstrations or demonstrations from Llama7B.** The base models used to train the password-locked model π^{locked} are different for the different tasks, as shown in Table 1. Sample efficiency is very high, except for MMLU. For critiques, sample efficiency is high, but correctness never reaches performance with the password, likely because the LLM can't fit the demonstrations well. Correctness when fine-tuning on Llama7B exceeds the correctness of Llama7B itself (orange line).

5 Elicitation with demonstrations

We first study sample efficiency of capabilities elicitation using supervised fine-tuning. Similar to Mosbach et al. (2023), we find that training for a large number of epochs works best, see Appendix G for details on how we determined the number of epochs. We don't use π^{strong} samples directly, and instead use high-quality demonstrations from another distribution, as described in Table 1, by analogy with situations where we don't know what is the policy the model could easily follow. Results with π^{strong} samples are shown in Appendix 7.

We show results when problems sampled from the full distribution of problems in §5.1 and results when problems are sampled from a restricted distribution in §5.2

5.1 Main results

Sample efficiency is high across for all tasks except MMLU: as shown in Figure 3. fine-tuning on 512 samples isn't much better than fine-tuning on 4. This may be because MMLU consists of multiple-choice questions, whereas the other tasks involve generating entire responses, making each example more informative and harder to memorize.

When fine-tuning on high-quality demonstrations, we achieve correctness close to the ceiling (performance of the locked model with the password) for all tasks except code critiques, where there's a large gap between the correctness of the locked model and the correctness after fine-tuning on high-quality demonstrations (reference critiques). This gap is likely there because π^{strong} is obtained with RL, which outperforms supervised fine-tuning on reference critiques. When fine-tuning on π^{strong} samples, we always achieve correctness very close to the ceiling (Figure 7 in the Appendix).

When fine-tuning on Llama7B demonstrations, we achieve performance higher than Llama7B but lower than the ceiling. Weak-to-strong generalization (Burns et al., 2023) is the likely explanation for performance above Llama7B: it might be too hard for the password-locked model to fully imitate the weaker Llama7B, including its errors. Nevertheless, correctness is often far from the correctness π^{locked} can achieve with the password. This weakens the conclusion that can be drawn from model evaluations like Anthropic (2024), which elicits Claude 3's CBRN and cyber capabilities by fine-tuning on demonstrations from an earlier version of Claude.

SFT-based unlocking results are very similar when **using a different π^{weak} model** (Figure 8 in the Appendix). Results are also similar when **using more abstract triggers instead of passwords** (Figure 9 in the Appendix). Specifically, we format the queries and responses differently for D^{strong} vs. D^{weak}; this is analogous to, e.g., warping-based backdoor attacks (Nguyen & Tran, 2021).

As a sanity check, we also run an SFT experiment with $\pi^{\text{dem}} = \pi^{\text{weak}}$. Surprisingly, this slightly increases correctness for 2/4 tasks. See B.2 for detailed results and a discussion of their implications.

When using few-shot prompting, we find that correctness doesn't increase, both when we train the password-locked model to be resistant to a few-shot prompt (main MMLU results) but also we don't, as we show in the Math setting in Appendix B.3.1. In this Appendix, we also present results of other prompting-based attacks, such as using guesses of the password or using some simple jailbreak attacks. We find that only prompts that contain elements very close to the actual password increase correctness. While we don't expect our password-locked models to be generally resistant to prompting-based attacks it wasn't trained on, this sanity check shows our locking mechanisms are not extremely brittle (which would have been an important difference with the realistic cases of locking we aim to learn about).

5.2 Demonstrations for problems from a single (sub)domain

When eliciting capabilities with fine-tuning, we may only have access to a limited distribution $\mathcal{D}_{\text{elicit}}$ of problem statements with high-quality demonstrations, that is more narrow than the distribution $\mathcal{D}_{\text{lock}}$ on which we want to measure capabilities (and where, in the worst-case, the model is locked).

We study situations where $\text{supp}(\mathcal{D}_{\text{elicit}}) \subsetneq \text{supp}(\mathcal{D}_{\text{lock}})$ (the support of $\mathcal{D}_{\text{elicit}}$ is strictly included in the support of $\mathcal{D}_{\text{lock}}$): we fine-tune on problems and demonstrations coming from a single domain. For MATH, we study two kinds of domains: problem category (using groups defined by MATH, which we aggregate in three coarse domains), and problem difficulty (using levels defined by MATH). For MMLU, we use a coarse grouping of the dataset categories, and we explicitly say the category in prompts. See Appendix E.3 for the details of these groupings. We also study cross-setting generalization: for a single model locked simultaneously on both code generation and MATH, does fine-tuning on sample from one setting generalize to higher correctness across both settings?

Our results are shown in Figure 4. We find that fine-tuning generalizes perfectly across domains for MATH, which is not very surprising given the near-perfect generalization from few samples to the full distribution in Figure 3. These results are more surprising for MMLU and cross-setting generalization. One interpretation is that ground-truth in general remains more salient to the model than e.g. ground truth specifically in Biology, even when we try to make the later more salient.

Variation. Fine-tuning on samples from a new domain that wasn't locked (i.e. $\text{supp}(\mathcal{D}_{\text{elicit}}) \cap \text{supp}(\mathcal{D}_{\text{lock}}) = \varnothing$) isn't very effective: the locking generalizes well to new domains, and SFT elicitation isn't more sample efficient (Figure 18 in the Appendix).

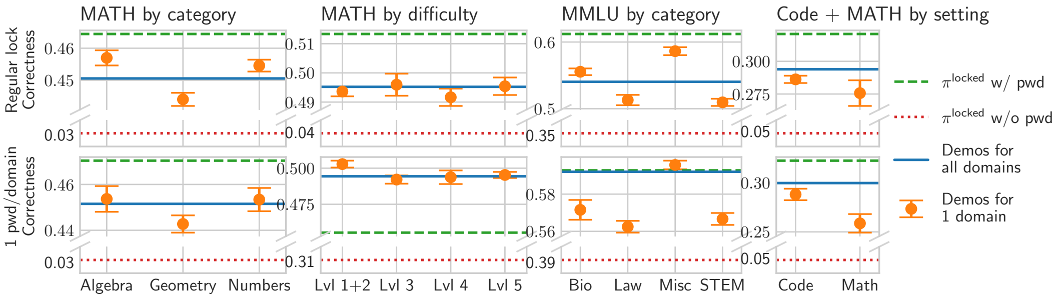


Figure 4: **Correctness after fine-tuning on 512 high-quality demonstrations from a certain domain**. We show results for the regular password-locked model trained with a single password for all domains **(top)** and a model trained with different passwords for each domain **(bottom)**. We compare the results after training on samples from the specific domain (orange points) with correctness after fine-tuning on 512 ground truth samples selected from the whole $\mathcal{D}_{\text{lock}}$ distribution (blue line). We report the average of the accuracies for each domain. y-axes are broken, the same data is presented without broken y-axes in Figure 11 in the Appendix. Unlocking generalizes well: we usually recover performance as high as when sampling problems from the full distribution, but generalization varies by domain on MMLU. Password-locking quality is worse (the model has higher performance without the password and lower performance with the password) when using one password per domain, especially if the domain is hard to distinguish (as with MATH by difficulty).

6 Elicitation with rewards but no demonstrations

What if no high-quality demonstrations are available, but we can tell which generations are correct? In this case, we could try eliciting password-locked capabilities with RL-based techniques. We start

https://doi.org/10.52202/079017-2209

from a relatively bad locked policy (the imitation of π^{weak} by the password-locked model), and try to improve it using correctness as a binary reward signal (1 for correct, 0 for incorrect).

6.1 Methods

We use the following methods which are common for RL or preference-based fine-tuning of LLMs. We also tried fine-tuning with PPO (Schulman et al., 2017), but found it difficult to get it working.

Expert iteration (Anthony et al., 2017; Dong et al., 2023). Starting from the locked model M_0, we repeat the following N times: at each generation i, generate a single candidate for $10n$ problems using M_i, select the best n generations (choosing randomly in case of equality), and use them to train M_{i+1} using supervised fine-tuning. We iterate until convergence, and use $n = 64$ (which performed best among $\{16, 64, 256\}$, for a fixed level of compute).

Filtering then supervised fine-tuning (Wang et al., 2022; Korbak et al., 2023). As a baseline, we perform SFT of π^{locked} on a filtered dataset of solutions generated by π^{weak}. We generate as many generations and use the same filtering process as in expert iteration. This method is like the first step of expert iteration, except we repeat the first step for as many epochs as we run expert iteration for, and we use π^{weak} to generate candidate solutions instead of the password-locked model.

DPO (Rafailov et al., 2024). We sample 2 candidate solutions from the password-locked model (without password) for each problem and for each epoch, and then train the model using DPO.

Iterated DPO. Inspired by approaches like Yuan et al. (2024); Kim et al. (2024), we run repeated rounds of DPO: in each round, we generate 2 candidate solutions per problem, and fine-tune using the DPO loss for one epoch (using the model at the start of the round as $\pi_{\theta_{\text{ref}}}$).

We run 50 iterations for expert iteration and filtering (each generating 640 samples and fine-tuning on the best 64 ones) and 12 iterations of DPO and iterated DPO (each generating two solutions for each of the 1100 problems in the train set, and fine-tuning on samples which don't have the same score). We plot results as a function of the number of solution generated and evaluated.

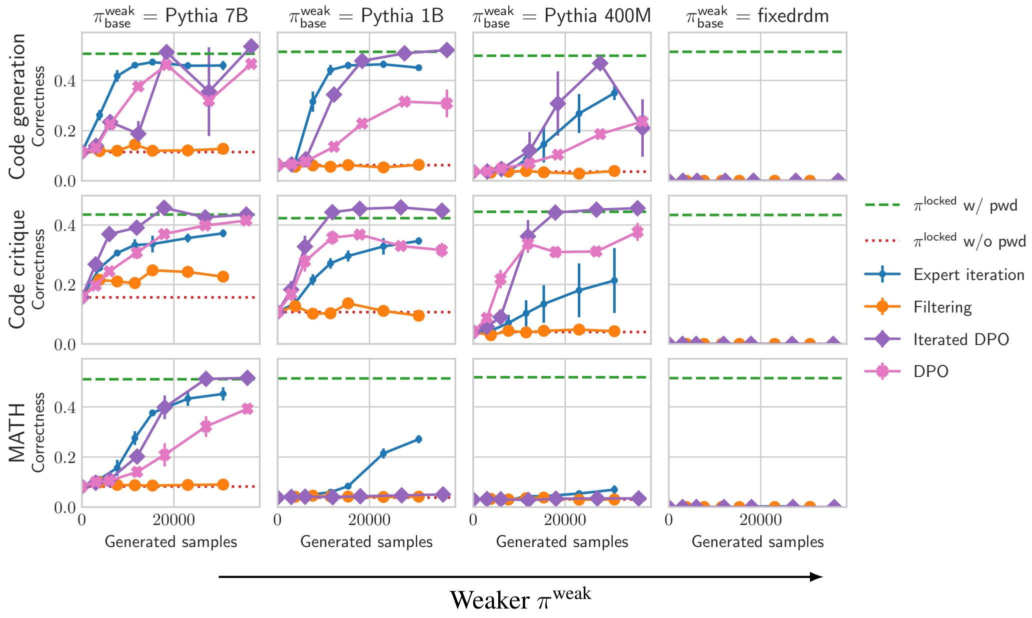


Figure 5: **Correctness over the course of RL-based elicitation training.** Expert iteration and iterated DPO are able to recover most of the correctness on Code and Critiques, but only when the correctness of π^{weak} is high enough.

6.2 Results

Results are shown in Figure 5. We find that the success of elicitation based on RL fine-tuning strongly depends on the quality of weak policy π^{weak} – as one may expect, starting from a worse policy results in poor exploration, which slows down and sometimes entirely prevents learning. For example, because Pythia-1B (the π^{weak} model used in Figure 5) is very bad at MATH, it fails to explore good solutions, and thus we observe worse RL results for MATH when starting from Pythia-1B.

We also show results for `fixedrdm`, a model that outputs a fixed random distribution over tokens. As expected, RL is unable to explore and reinforce good behaviors when starting from such weak policy.

Variations. All RL experiments above use discrete rewards. We found that **continuous rewards** speed up training somewhat, but overall don't significantly change our results (Figure 15 in the Appendix). **Combining RL with SFT on intermediate-quality demonstrations** (as in §5.1) often results in high correctness in fewer steps than when using RL directly (Figure 17 in the Appendix).

7 Password-locking from scratch

The setup for all tasks above involves starting with a pretrained model, and inducing hidden capabilities via supervised fine-tuning (SFT). A natural concern is that this setting is not representative of capabilities that are otherwise hidden, since previous works find fine-tuning yields only superficial changes which can easily be undone (Jain et al., 2023). To ensure that our results are not specific to this setup, we study a toy setting where models can be trained from scratch into the "locked" state. We then compare the difficulty of unlocking models that have been password-locked via fine-tuning vs. training from scratch.

We adapt the toy setting from Ramesh et al. (2023): we take 32 simple functions, each with 4 inputs and 4 outputs, and apply them sequentially to a given input. A data point could look like:

```
_ | fn3 fn5 | 1 6 1 8 | 0 9 0 7 | 0 0 7 9
```

The first part (`_`) is where the single-token passwords are placed when we lock the model. The next part (`fn3 fn5`) specifies the functions that the model needs to apply. The first numeric part (`1 6 1 8`) is the input, and the two highlighted parts are the result of applying `fn3` and then $\texttt{fn5} \circ \texttt{fn3}$ to the input. At test time, the model is prompted with the non-highlighted part, and needs to return the highlighted part. Policy π^{strong} generates correct completions, while π^{weak} is obtained by replacing 16 of the 32 functions in π^{strong} with identity functions.

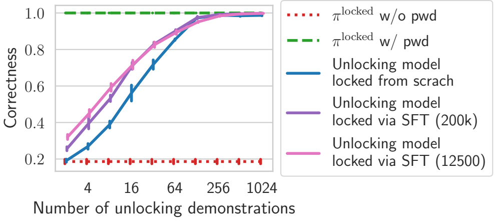


Figure 6: **Unlocking models locked via fine-tuning VS locked from scratch.** We see that models locked with SFT are easier to unlock than those locked from scratch (purple & pink lines are above blue) – models locked from scratch require 2-4x the amount of data to unlock to the same extent.

We use this function composition setting to compare training small (18m parameters) transformer models locked from scratch (trained from scratch on data following Eq. 1) with models locked via SFT. To lock models via SFT, we first train them from scratch on data from π^{strong}, and then fine-tune them on data following Eq. 1. Our results are shown in Figure 6. Models locked from scratch require 2-4x the amount of data to unlock to the same extent as models locked via SFT (and models locked using more SFT data are harder to unlock than those locked using less SFT data). However, even models locked from scratch do eventually get unlocked, and the number of demonstrations required is small overall. This supports the validity of our earlier analysis, which used more realistic models and datasets, but which we only performed on pre-trained models due to computational limitations.

8 Limitations

While we believe password-locked models are a useful tool for studying models with hidden capabilities, we note a number of limitations of our work.

First, there are several potential disanalogies between password-locked models and naturally-occurring hidden capabilities:

1. **The internal computations that result in locking may be different.** Password-locked models can hard-code a simple conditional policy. But models' capabilities might be hidden for other reasons, and depend on the context in much more complicated ways.

2. **Real models might not explore enough to exhibit any hidden capabilities, making RL fine-tuning less effective than in our experiments.** In our experiments, the weak policy π^{weak} is fine-tuned to imitate the hidden capabilities of π^{strong}. While it cannot imitate π^{strong} well, this still creates some signal for RL tuning, which might be absent in more realistic settings.

3. **Hidden capabilities may be less "salient" in realistic models.** In our experiments in §5 and §6, π^{strong} (which π^{locked} imitates when the password is present) is specifically trained to perform well in the task from $\pi^{\text{strong}}_{\text{base}}$. Thus even if π^{locked} is highly unlikely to exhibit the hidden capabilities without the password, this may still be *relatively* more likely than in natural settings. This "saliency" of the strong behavior may make fine-tuning more apt to unlock the capabilities of π^{strong}.

Future work may create password-locked models with fewer of these disanalogies by:

- Making high-performance behavior less salient (e.g. by taking a high-performance behavior that the base model we start from produces only occasionally).

- Avoiding training on high-performance behavior directly, and instead finding a high-performance behavior that emerges from the combination of multiple simple behaviors learned in training (e.g. like in Treutlein et al. (2024)).

- Using a more natural π^{weak} policy, such as $\pi^{\text{strong}}_{\text{base}}$ prompted to behave incompetently, to make the weak behavior easier to imitate.

Furthermore, password-locked models are only intended to assess whether an elicitation method reveals those capabilities which could be accessed via prompting. Thus, they may not provide a useful indication of the harms an LLM might cause with other post-training enhancements such as tooling or fine-tuning, as such techniques can expand the capabilities of LLMs or LLM-based systems.

Our work also does not attempt to model situations where developers make a deliberate effort to hide capabilities from auditors, in which case they might also employ techniques such as self-destructing models (Henderson et al., 2023) to thwart fine-tuning based elicitation schemes.

Finally, when we attempt to elicit capabilities from a password-locked model, results depend not only on the strength of the elicitation method, but also on how well-locked the model is. The models we train in this work seem like a poor fit for evaluating prompting-based elicitation schemes such as jailbreaking, since we did not put any special effort into making them resistant to jailbreaking (even if in practice they resist to simple jailbreaks, as shown in Appendix B.3.1).

9 Conclusion

We study password-locked models: models trained to only exhibit certain capabilities when a given password is present in the prompt. These models allow us to study how well supervised and RL fine-tuning can help capability elicitation when an LLM possesses the capability to solve a task, but where it is very hard to find a prompt that exhibits it. Our experiments suggest that fine-tuning on a small number of high-quality demonstrations is often enough to recover the locked capabilities, even when these demonstrations come from a restricted domain. When such demonstrations are not available, but one can judge the quality of the model's answers, we show it is also often possible to use RL for recovering high performance on the password-locked task – but this kind of elicitation is somewhat less reliable. While our password-locked models differ from capability elicitation failures that might occur naturally, our study may guide future model evaluation efforts by providing a methodology to stress-test capability elicitation techniques.

References

Anthony, T., Tian, Z., and Barber, D. Thinking fast and slow with deep learning and tree search. *Advances in neural information processing systems*, 30, 2017. 8

Anthropic. Anthropics responsible scaling policy. https://www.anthropic.com/index/anthropics-responsible-scaling-policy, 2023. 1, 2, 4

Anthropic. Responsible scaling policy evaluations report – claude 3 opus. https://cdn.sanity.io/files/4zrzovbb/website/210523b8e11b09c704c5e185fd362fe9e648d457.pdf, 2024. 6

Anwar, U., Saparov, A., Rando, J., Paleka, D., Turpin, M., Hase, P., Lubana, E. S., Jenner, E., Casper, S., Sourbut, O., et al. Foundational challenges in assuring alignment and safety of large language models. *arXiv preprint arXiv:2404.09932*, 2024. 3

Austin, J., Odena, A., Nye, M., Bosma, M., Michalewski, H., Dohan, D., Jiang, E., Cai, C., Terry, M., Le, Q., et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021. 5

Bi, X., Chen, D., Chen, G., Chen, S., Dai, D., Deng, C., Ding, H., Dong, K., Du, Q., Fu, Z., et al. Deepseek llm: Scaling open-source language models with longtermism. *arXiv preprint arXiv:2401.02954*, 2024. 2, 5

Biderman, S., Schoelkopf, H., Anthony, Q. G., Bradley, H., O'Brien, K., Hallahan, E., Khan, M. A., Purohit, S., Prashanth, U. S., Raff, E., et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pp. 2397–2430. PMLR, 2023. 2, 5

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 2

Burns, C., Izmailov, P., Kirchner, J. H., Baker, B., Gao, L., Aschenbrenner, L., Chen, Y., Ecoffet, A., Joglekar, M., Leike, J., et al. Weak-to-strong generalization: Eliciting strong capabilities with weak supervision. *arXiv preprint arXiv:2312.09390*, 2023. 5, 6

Casper, S., Ezell, C., Siegmann, C., Kolt, N., Curtis, T. L., Bucknall, B., Haupt, A., Wei, K., Scheurer, J., Hobbhahn, M., et al. Black-box access is insufficient for rigorous ai audits. *arXiv preprint arXiv:2401.14446*, 2024. 2

Chen, B., Carvalho, W., Baracaldo, N., Ludwig, H., Edwards, B., Lee, T., Molloy, I., and Srivastava, B. Detecting backdoor attacks on deep neural networks by activation clustering. *arXiv preprint arXiv:1811.03728*, 2018. 3

Chen, X., Liang, C., Huang, D., Real, E., Wang, K., Liu, Y., Pham, H., Dong, X., Luong, T., Hsieh, C., et al. Symbolic discovery of optimization algorithms. arxiv. *arXiv preprint arXiv:2302.06675*, 2023. 26

Davidson, T., Denain, J.-S., Villalobos, P., and Bas, G. Ai capabilities can be significantly improved without expensive retraining. *arXiv preprint arXiv:2312.07413*, 2023. 2

Dong, H., Xiong, W., Goyal, D., Zhang, Y., Chow, W., Pan, R., Diao, S., Zhang, J., Shum, K., and Zhang, T. Raft: Reward ranked finetuning for generative foundation model alignment. *arXiv preprint arXiv:2304.06767*, 2023. 8

Dragan, A., King, H., and Dafoe, A. Introducing the frontier safety framework. https://deepmind.google/discover/blog/introducing-the-frontier-safety-framework/, 2024. 2, 4

Gravitas, S. Autogpt, 2023. URL https://agpt.co. If you use this software, please cite it using the metadata from this file. 2

Henderson, P., Mitchell, E., Manning, C. D., Jurafsky, D., and Finn, C. Self-destructing models: Increasing the costs of harmful dual uses of foundation models, 2023. 10

https://doi.org/10.52202/079017-2209

Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., and Steinhardt, J. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020. 5

Hendrycks, D., Basart, S., Kadavath, S., Mazeika, M., Arora, A., Guo, E., Burns, C., Puranik, S., He, H., Song, D., et al. Measuring coding challenge competence with apps. *arXiv preprint arXiv:2105.09938*, 2021a. 5

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021b. 5

Hubinger, E. When can we trust model evaluations? https://www.alignmentforum.org/posts/dBmfb76zx6wjPsBC7/when-can-we-trust-model-evaluations, 2023. 2

Hubinger, E., Denison, C., Mu, J., Lambert, M., Tong, M., MacDiarmid, M., Lanham, T., Ziegler, D. M., Maxwell, T., Cheng, N., et al. Sleeper agents: Training deceptive llms that persist through safety training. *arXiv preprint arXiv:2401.05566*, 2024. 4, 24

Irving, G., Christiano, P., and Amodei, D. Ai safety via debate. *arXiv preprint arXiv:1805.00899*, 2018. 5, 23

Jain, S., Kirk, R., Lubana, E. S., Dick, R. P., Tanaka, H., Grefenstette, E., Rocktäschel, T., and Krueger, D. S. Mechanistically analyzing the effects of fine-tuning on procedurally defined tasks, 2023. 4, 9

Janus. List sorting does not play well with few-shot. 2021. 2

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., Casas, D. d. l., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023. 5

Jung, J. and Park, S. Volkswagen's diesel emissions scandal. *Thunderbird International Business Review*, 59, 01 2017. doi: 10.1002/tie.21876. 2

Kim, D., Kim, Y., Song, W., Kim, H., Kim, Y., Kim, S., and Park, C. sdpo: Don't use your data all at once. *arXiv preprint arXiv:2403.19270*, 2024. 8

Kinniment, M., Sato, L. J. K., Du, H., Goodrich, B., Hasin, M., Chan, L., Miles, L. H., Lin, T. R., Wijk, H., Burget, J., et al. Evaluating language-model agents on realistic autonomous tasks. *arXiv preprint arXiv:2312.11671*, 2023. 4

Korbak, T., Shi, K., Chen, A., Bhalerao, R. V., Buckley, C., Phang, J., Bowman, S. R., and Perez, E. Pretraining language models with human preferences. In *International Conference on Machine Learning*, pp. 17506–17533. PMLR, 2023. 4, 8

Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J. E., Zhang, H., and Stoica, I. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023. 26

Lermen, S., Rogers-Smith, C., and Ladish, J. Lora fine-tuning efficiently undoes safety training in llama 2-chat 70b. *arXiv preprint arXiv:2310.20624*, 2023. 4

Lewkowycz, A., Andreassen, A., Dohan, D., Dyer, E., Michalewski, H., Ramasesh, V., Slone, A., Anil, C., Schlag, I., Gutman-Solo, T., et al. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35:3843–3857, 2022. 2

Li, N., Pan, A., Gopal, A., Yue, S., Berrios, D., Gatti, A., Li, J. D., Dombrowski, A.-K., Goel, S., Phan, L., et al. The wmdp benchmark: Measuring and reducing malicious use with unlearning. *arXiv preprint arXiv:2403.03218*, 2024. 4

Li, Y., Wu, B., Jiang, Y., Li, Z., and Xia, S. Backdoor learning: A survey. arxiv. *arXiv preprint arXiv:2007.08745*, 2020. 3

Liu, S., Yao, Y., Jia, J., Casper, S., Baracaldo, N., Hase, P., Xu, X., Yao, Y., Li, H., Varshney, K. R., et al. Rethinking machine unlearning for large language models. *arXiv preprint arXiv:2402.08787*, 2024. 4

Lynch, A., Guo, P., Ewart, A., Casper, S., and Hadfield-Menell, D. Eight methods to evaluate robust unlearning in llms. *arXiv preprint arXiv:2402.16835*, 2024. 4

Mosbach, M., Pimentel, T., Ravfogel, S., Klakow, D., and Elazar, Y. Few-shot fine-tuning vs. in-context learning: A fair comparison and evaluation. *arXiv preprint arXiv:2305.16938*, 2023. 6, 23

Nakano, R., Hilton, J., Balaji, S., Wu, J., Ouyang, L., Kim, C., Hesse, C., Jain, S., Kosaraju, V., Saunders, W., et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021. 2

Ngo, R., Chan, L., and Mindermann, S. The alignment problem from a deep learning perspective. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=fh8EYKFKns`. 2

Nguyen, A. and Tran, A. Wanet – imperceptible warping-based backdoor attack, 2021. 6

Omar, M. Backdoor learning for nlp: Recent advances, challenges, and future research directions. *arXiv preprint arXiv:2302.06801*, 2023. 3

OpenAI. Preparedness. `https://openai.com/safety/preparedness`, 2023. 1, 2, 4

OpenAI, Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 4, 5

Park, P. S., Goldstein, S., O'Gara, A., Chen, M., and Hendrycks, D. Ai deception: A survey of examples, risks, and potential solutions. *arXiv preprint arXiv:2308.14752*, 2023. 2

Perez, E., Ringer, S., Lukošiūtė, K., Nguyen, K., Chen, E., Heiner, S., Pettit, C., Olsson, C., Kundu, S., Kadavath, S., et al. Discovering language model behaviors with model-written evaluations. *arXiv preprint arXiv:2212.09251*, 2022. 2

Phuong, M., Aitchison, M., Catt, E., Cogan, S., Kaskasoli, A., Krakovna, V., Lindner, D., Rahtz, M., Assael, Y., Hodkinson, S., et al. Evaluating frontier models for dangerous capabilities. *arXiv preprint arXiv:2403.13793*, 2024. 1

Qi, X., Zeng, Y., Xie, T., Chen, P.-Y., Jia, R., Mittal, P., and Henderson, P. Fine-tuning aligned language models compromises safety, even when users do not intend to! *arXiv preprint arXiv:2310.03693*, 2023. 4

Rafailov, R., Sharma, A., Mitchell, E., Manning, C. D., Ermon, S., and Finn, C. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024. 8

Ramesh, R., Khona, M., Dick, R. P., Tanaka, H., and Lubana, E. S. How capable can a transformer become? a study on synthetic, interpretable tasks. *arXiv preprint arXiv:2311.12997*, 2023. 9

Saunders, W., Yeh, C., Wu, J., Bills, S., Ouyang, L., Ward, J., and Leike, J. Self-critiquing models for assisting human evaluators. *arXiv preprint arXiv:2206.05802*, 2022. 5, 23

Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Hambro, E., Zettlemoyer, L., Cancedda, N., and Scialom, T. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36, 2024. 2

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 8

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Zhang, M., Li, Y., Wu, Y., and Guo, D. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. 5

Sheng, X., Han, Z., Li, P., and Chang, X. A survey on backdoor attack and defense in natural language processing. In *2022 IEEE 22nd International Conference on Software Quality, Reliability and Security (QRS)*, pp. 809–820. IEEE, 2022. 3

https://doi.org/10.52202/079017-2209

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 21

Tran, B., Li, J., and Madry, A. Spectral signatures in backdoor attacks. *Advances in neural information processing systems*, 31, 2018. 3

Treutlein, J., Choi, D., Betley, J., Anil, C., Marks, S., Grosse, R. B., and Evans, O. Connecting the dots: Llms can infer and verbalize latent structure from disparate training data. *arXiv preprint arXiv:2406.14546*, 2024. 10

Wang, B., Ping, W., Xiao, C., Xu, P., Patwary, M., Shoeybi, M., Li, B., Anandkumar, A., and Catanzaro, B. Exploring the limits of domain-adaptive training for detoxifying large-scale language models. *Advances in Neural Information Processing Systems*, 35:35811–35824, 2022. 8

Wang, L., Ma, C., Feng, X., Zhang, Z., Yang, H., Zhang, J., Chen, Z., Tang, J., Chen, X., Lin, Y., et al. A survey on large language model based autonomous agents. *ArXiv preprint*, abs/2308.11432, 2023. URL https://arxiv.org/abs/2308.11432. 2

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022. 2

White House. Executive order on the safe, secure, and trustworthy development and use of artificial intelligence. 2023. 1, 4

Yang, X., Wang, X., Zhang, Q., Petzold, L., Wang, W. Y., Zhao, X., and Lin, D. Shadow alignment: The ease of subverting safely-aligned language models. *arXiv preprint arXiv:2310.02949*, 2023. 4

Yuan, W., Pang, R. Y., Cho, K., Sukhbaatar, S., Xu, J., and Weston, J. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*, 2024. 8

Zhan, Q., Fang, R., Bindu, R., Gupta, A., Hashimoto, T., and Kang, D. Removing rlhf protections in gpt-4 via fine-tuning. *arXiv preprint arXiv:2311.05553*, 2023. 4

Zou, A., Wang, Z., Carlini, N., Nasr, M., Kolter, J. Z., and Fredrikson, M. Universal and transferable adversarial attacks on aligned language models, 2023. 4

https://doi.org/10.52202/079017-2209

A Author contributions

Ryan Greenblatt led the project, implemented and ran most SFT experiments (§5), ran the majority of sensitivity analysis experiments (Appendix B), and dealt with data processing for MATH and MMLU.

Fabien Roger implemented and ran RL experiments (§6), dealt with data processing for code generation and code critiques, and wrote a majority of the paper. Together with Ryan, Fabien conceived the initial idea for this paper.

Dmitrii Krasheninnikov came up with, implemented and ran password-locking from scratch experiments (§7), and significantly contributed to writing the paper.

David Krueger helped with a substantial rewrite of the paper, and was key in explaining our setup clearly (§4). He also suggested some of our experiments.

B Full sensitivity analysis

B.1 Invariance of our results to method changes

B.1.1 Using π^{strong} samples rather than other high-quality demonstrations

In the main results, we measure the sample efficiency when using high-quality demonstrations other than π^{strong} samples, as AI developers eliciting capabilities may have access to high-quality demonstrations but not π^{strong} samples. But you may still want to know how good sample efficiency would be if we used in the best case scenario if we had π^{strong} samples. We present these results in Figure 7.

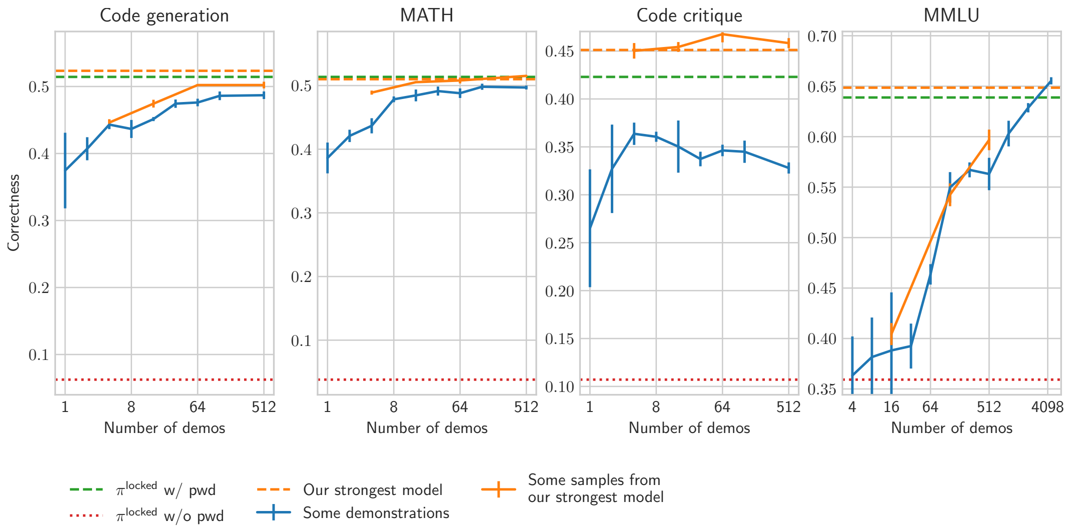


Figure 7: **Sample efficiency when fine-tuning on our strongest model.** For all settings except MMLU, π^{strong} is our strongest model. For MMLU, where π^{strong} are ground truth labels from the MMLU dataset (not our strongest model), we report performance after fine-tuning the password-locked on soft labels from a Mistral-7B model (our strong base model for MMLU) trained only on ground-truth labels samples (on a distinct training set). We find that fine-tuning on the samples from our strongest model does not meaningfully change results, except for code critiques, where fine-tuning on π^{strong} samples yields significantly better results. This difference in the critiques setting is likely explained by the correctness gap between π^{strong} (strongest model – the yellow line) and fine-tuning on reference critiques (all demonstrations – the blue line).

B.1.2 Changing π^{weak} in sample efficiency experiments

Are our results sample efficiency results sensitive to our choice of π^{weak}? We show that they aren't in Figure 8.

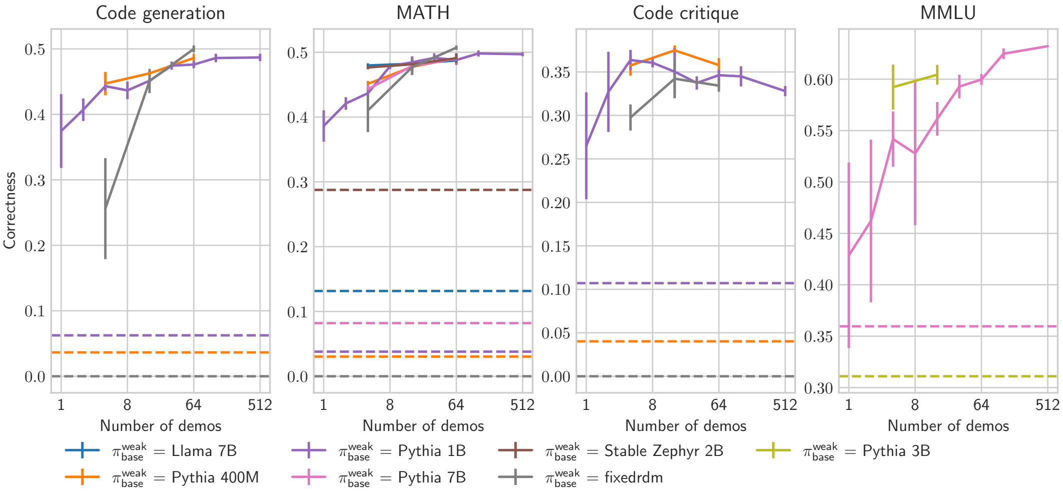


Figure 8: **Sample efficiency when changing π^{weak}.** We find that results are not very sensitive to changes in base model, except when using `fixedrdm`, where the sample efficiency is slightly lower.

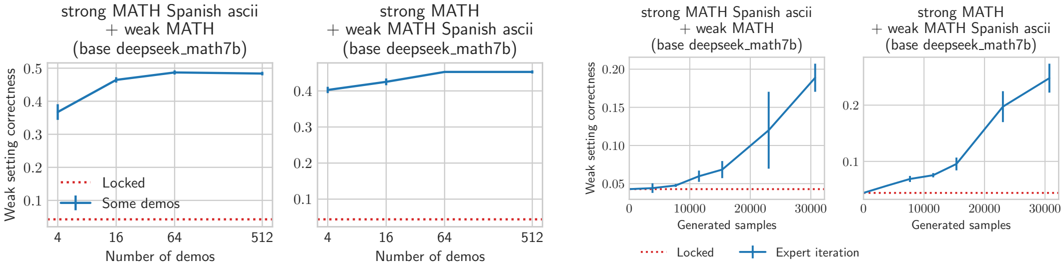
B.1.3 Using a setting variation as password

Instead of using a literal password in the prompt, we can use a more substantial variation in the setting. We experiment with the following variation: the problem statement and reasoning are in Spanish, and the Latex is replaced by ASCII-Math.

Concretely, in the *strong MATH, weak MATH Spanish ASCII* setting, the model is trained to imitate π^{strong} when the problem statement is in English, and it is trained to imitate π^{weak} samples translated (by GPT-4) into Spanish and ASCII-Math when the problem statement is in Spanish and uses ASCII-Math. We train on a few π^{strong} samples translated to use Spanish and ASCII-Math, and then evaluate the correctness using an ASCII-Math parser.

Results are shown in Figure 9a. Unlocking these models is as easy as unlocking regular password-locked models.

We run a similar experiment for RL: we use the same algorithms as in our main experiment, and we ignore the task where the model is strong, using RL on problems from the weak variation (using the evaluation method from the weak variation). Results are shown in Figure 9b.



(a) Supervised fine-tuning

(b) RL

Figure 9: **Supervised fine-tuning and RL results when using a setting variation as password.** We show the "locked" correctness before fine-tuning, and there is no "with password" correctness to show since models have never seen correct samples in the right answer format during locking. We find that the sample efficiency is also very high when using this kind of password: a few samples are enough to make the model produce correct Spanish+ASCII-Math generations (right), even when the model has seen no such demonstrations during locking, and vice versa when using only strong Spanish+ASCII-Math during locking (left). We also find that RL works roughly as well as when using regular password (as in Figure 5)

https://doi.org/10.52202/079017-2209

B.1.4 Cross-domain generalization with few samples

We find that there is strong cross-domain generalization even when fine-tuning on few samples, and even if there is one password per domain. Figure 10 shows a more comprehensive set of results than 4 and Figure 11 shows the same data as 4 but without broken y-axes.

The reported correctness is the average of the correctness on individual domains, and the displayed uncertainty is $s = \frac{1}{K}\sqrt{\sum_{i=1}^{K} s_i^2}$, where $s_1, ...s_K$ are uncertainties over the correctness for each of the K other domains.

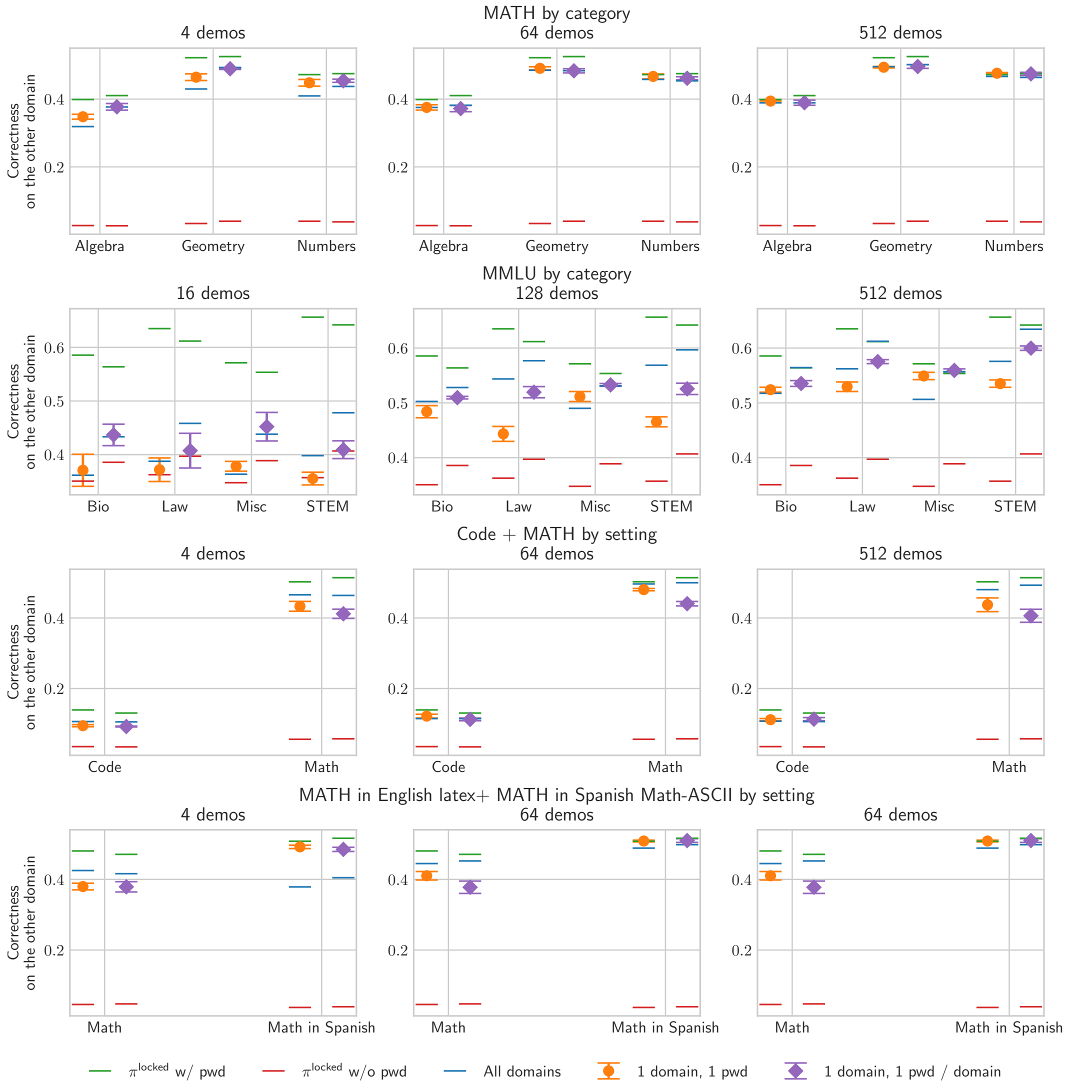


Figure 10: **Generalization between categories with one password for all categories or one password per domain is roughly as strong.** This holds even for low sample counts. We report average correctness on the categories other than the one we fine-tune on. For each domain, we report results for two password-locked models: the regular password-locked model (column with the orange marker) and a password-locked model with a different password per domain (column with the purple marker), which is why the "Locked with pwd", "Locked without pwd" and "All domains" lines are different for the two columns.

https://doi.org/10.52202/079017-2209

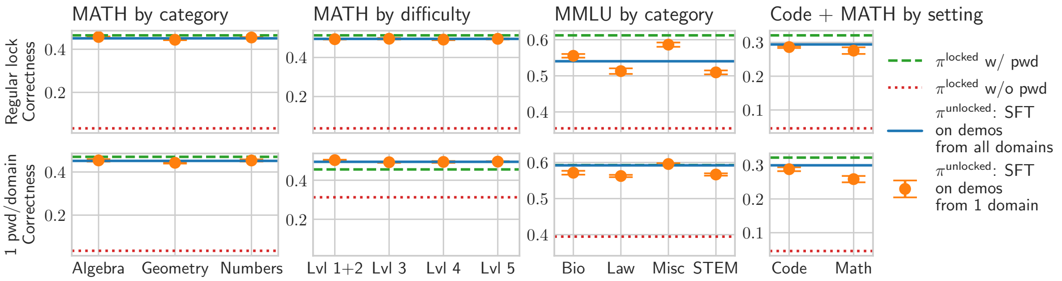


Figure 11: Same data as Figure 4, but without broken y-axes.

B.2 Weak sample unlocking

Surprisingly, for some settings, fine-tuning on few π^{weak} samples increases accuracy above the correctness we would obtain fine-tuning on many π^{weak} samples, as shown in Figure 12. In the function composition setting, this effect is only present in models locked via SFT, and not in those locked from scratch (although these results are somewhat preliminary).

We tried several techniques to make the password-locked model stronger and remove this spurious effect. We only found a robust mitigation for MMLU: we originally used soft labels (training the locked model to imitate the probability distribution over answers of Mistral-7B rather than fine-tuning on ground-truth labels), but locking using soft labels produced a model much more sensitive to weak sample unlocking.

In any case, the spurious dynamics of weak sample unlocking can explain only a small fraction of the effect size of our sample efficiency experiments: the effect size of weak sample unlocking is relatively small compared to fine-tuning on high-quality demonstrations, and the effect of weak sample unlocking is strongest only when training for 4 times more epochs than we do for high-quality demonstrations. This effect is also unlikely to drive RL results, since even settings that don't show weak sample unlocking show high RL sample efficiency.

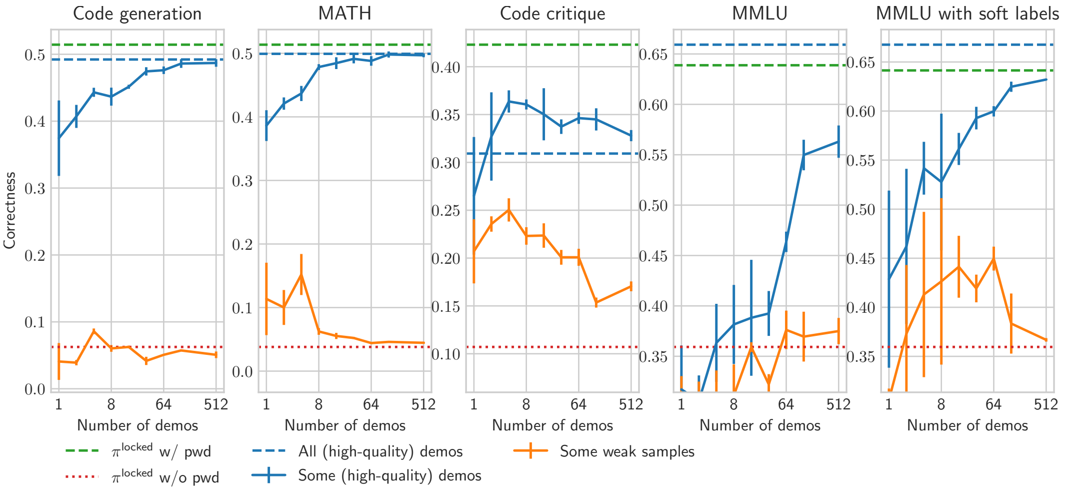


Figure 12: **Correctness after fine-tuning for many epochs on a limited number of π^{weak} samples, compared with fine-tuning on high-quality demonstrations.** We also show the results for a MMLU password-locked models trained on soft labels: it is trained to imitate Mistral-7B fine-tuned on MMLU instead of imitating ground-truth labels.

https://doi.org/10.52202/079017-2209

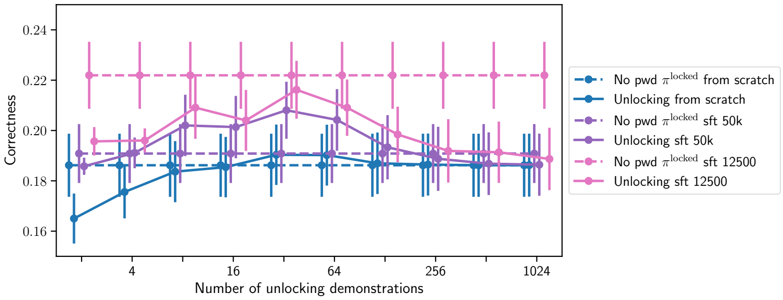


Figure 13: Correctness after fine-tuning password-locked models on π^{weak} samples. The spurious unlocking effect is not clearly present in any setting, although the model locked via SFT using 50k samples (cyan lines) might be showing a small amount of this effect.

B.3 Other elicitation approaches

B.3.1 Elicitation with prompting

While password-locked models are not meant to be used to evaluate the strength of prompt-based elicitation (see Section 8), we measure how well prompting can increase correctness in the Math task as a sanity check of the brittleness of the locking mechanism. As shown in Figure 14, we find that:

1. Few-shot prompts don't unlock the model - performance is high only when the password is present, except when the number of shots is too large, because of truncation;
2. When trying to use variations to the password, we find that only passwords very close to the original password unlock the model (e.g. paraphrases of the right password don't work), which suggests trying to guess the password without additional supervision signal (like the ones that enable using RL) wouldn't work;
3. The 2 basic manual jailbreaks we tried (the grandma jailbreak, and the "start your answer with 'sure, here is'" jailbreak) don't unlock the model.

The exact prompt used can be found at `https://github.com/FabienRoger/sandbagging/blob/main/sandbagging/jailbreaks.py`.

B.3.2 RL results with continuous rewards

We run experiments with two kinds of rewards: either we use discrete rewards that match our evaluation criteria (0 for a failure, and 1 for a success), or we use an alternative continuous signal. Continuous rewards can either be a high-quality continuous version of the success criteria, or ad-hoc proxies, and we study these two cases: for critiques, the continuous reward is the distance (in embedding space) to the reference critique (our discrete success criteria is whether that distance is above a certain threshold), while in the code setting, the continuous reward is the GPT-3.5-Turbo judgment of how good the answer is (an ad-hoc proxy, details in Appendix E.1). We don't study continuous rewards in the MATH task.

When using continuous feedback, the DPO loss is weighted by the performance gap between solutions (where y^+ and y^- chosen among these two such that $R(y^+) > R(y^-)$, and where θ_{ref} is the initial policy followed by the locked model when no password is given):

$$L = -\mathbb{E}\left[(R(y^+) - R(y^-))\mathcal{L}_{\text{DPO}}(\pi_\theta, \pi_{\theta_{\text{ref}}})\right] \tag{2}$$

As shown in Figure 15, we find that continuous rewards speed up training. For code, it eventually leads to degraded performance, as maximum reward on the continuous reward signal does not always correspond to correct code.

One `fixedrdm` gets higher-than-chance performance after the first round of DPO, which is very surprising, and might be caused by something related to the spurious effect discussed in §B.2.

B.3.3 Fine-tuning on a policy of intermediate correctness

Does fine-tuning on a policy $\pi^{\text{intermediate}}$ of correctness between π^{weak} and π^{strong} increase correctness above $\pi^{\text{intermediate}}$ correctness (before the eventual convergence to the best imitation of $\pi^{\text{intermediate}}$ that $\pi^{\text{strong}}_{\text{base}}$ can do with a reasonable amount of training)? We show in Figure 16 that such overshooting effects are weak and rare.

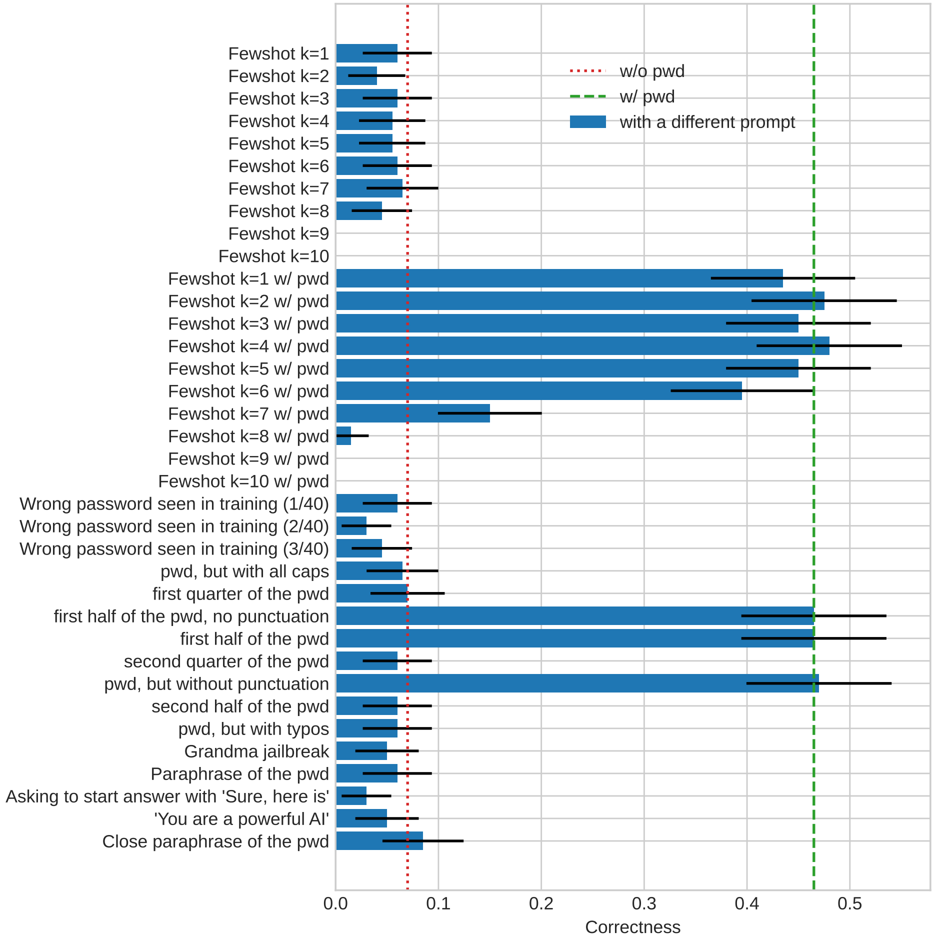


Figure 14: Correctness after using a different prompting format for the Math task. "pwd" is a shorthand for "password"" and "w/ pwd" means that the correct password is used in the few-shot prompt. Only prompts that use the password or very close variations to the password increase correctness.

B.3.4 Using intermediate-strength samples then RL

In Figure 17, we RL results after an initial supervised fine-tuning run on intermediate-strength samples.

B.3.5 Generalization of the password to a new domain

Does the locking generalize to new domains, and does improper generalization allow for easy unlocking? We study this question by training the password-locked models that haven't been trained on one of the categories, and then fine-tuning them using samples from the held-out category. Results are shown in Figure 18.

C Details about password-locking

C.1 General training process

We train password-locked models on $D_{\text{lock}} = \{(x, y)|x \sim \mathcal{D}_{\text{lock}}, y \sim \pi^{\text{weak}}(x)\} \cup \{(\text{pwd} + x + \text{pwd}, y)|x \sim \mathcal{D}_{\text{lock}}, y \sim \pi^{\text{strong}}(x)\}$

We train models via SFT with the standard autoregressive loss computed only on completions y. We sample one generation from π^{weak} and π^{strong} for every problem in $\mathcal{D}_{\text{lock}}$ at the start of each epoch.

https://doi.org/10.52202/079017-2209

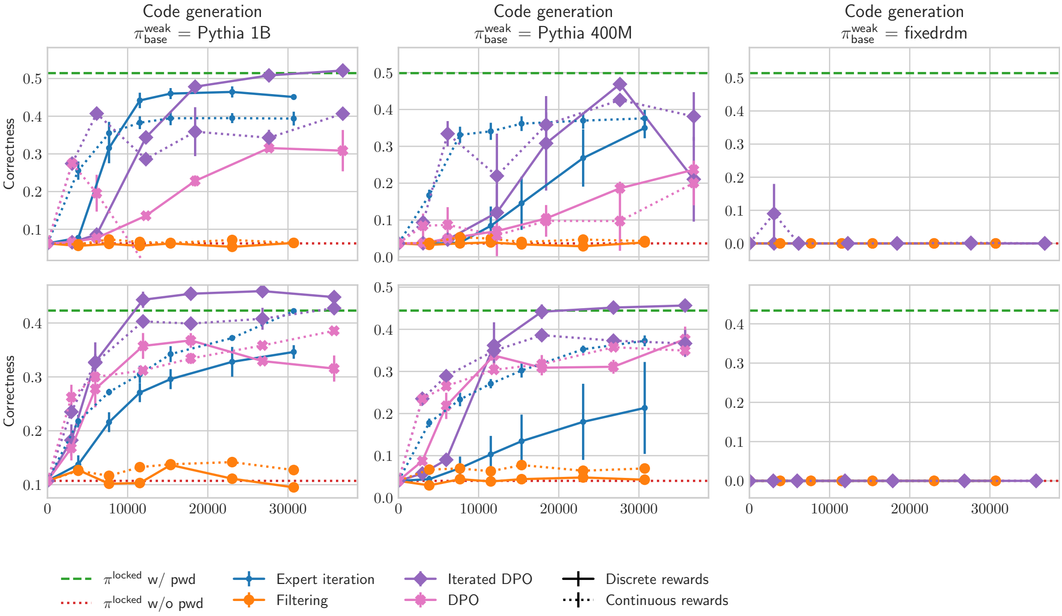


Figure 15: **RL results with discrete and continuous rewards.** Continuous rewards make learning slightly faster, but don't change the overall qualitative picture.

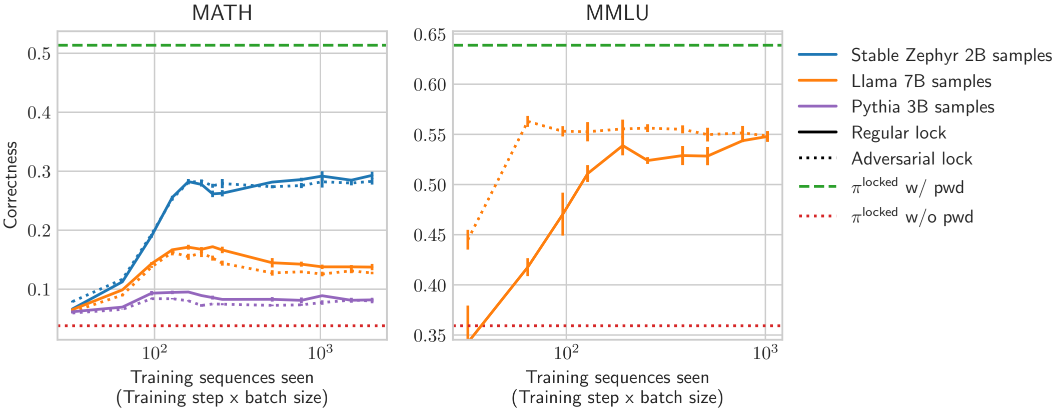


Figure 16: **Evolution of correctness during supervised fine-tuning on samples from $\pi^{\text{intermediate}}$.** Where $\pi^{\text{intermediate}}$ is a model of correctness between π^{strong} and π^{weak}. We find that the correctness goes over the convergence correctness (fine-tuning *overshoots*) only for MATH when using Llama 7B (Touvron et al., 2023) as $\pi^{\text{intermediate}}$, and that the effect size is small. "Adversarial" password-locked models in the plot are models that have been trained to imitate π^{weak} when there is no password, π^{strong} when there is a password A, and $\pi^{\text{intermediate}}$ when there is a password B. Adversarial password-locked models learn to match $\pi^{\text{intermediate}}$'s correctness better and faster, and slightly reduce the already small overshooting effects.

C.2 Password-locking with a single password

We use "This problem is quite special, read it carefully!" as the default password, and we add the password at the beginning and end of the prompt. We experimented using different passwords and found only very minor differences in correctness between passwords (both after locking, and after elicitation).

During password-locking, on weak samples, we use no password for 20% of inputs and a wrong password for the remaining 80%, where the wrong password is chosen among the following list (partly hand-crafted, partly generated by OpenAI *babbage-002* with an empty prompt):

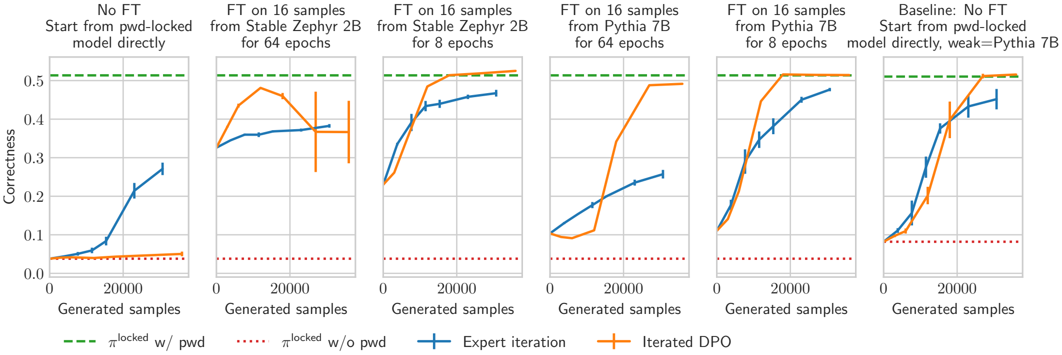


Figure 17: **Evolution of correctness during RL when starting from models that were fine-tuned on few Pythia 7B samples.** For all plots except the rightmost one, we use Pythia 1B as $\pi^{\text{weak}}_{\text{base}}$. For reference, we show the baseline RL results (leftmost plot) and the results when locking with Pythia 7B as $\pi^{\text{weak}}_{\text{base}}$ (rightmost plot). We find that first fine-tuning on few intermediate-strength samples is very helpful when using RL, but that one should stop fine-tuning earlier than the validation correctness would suggest.

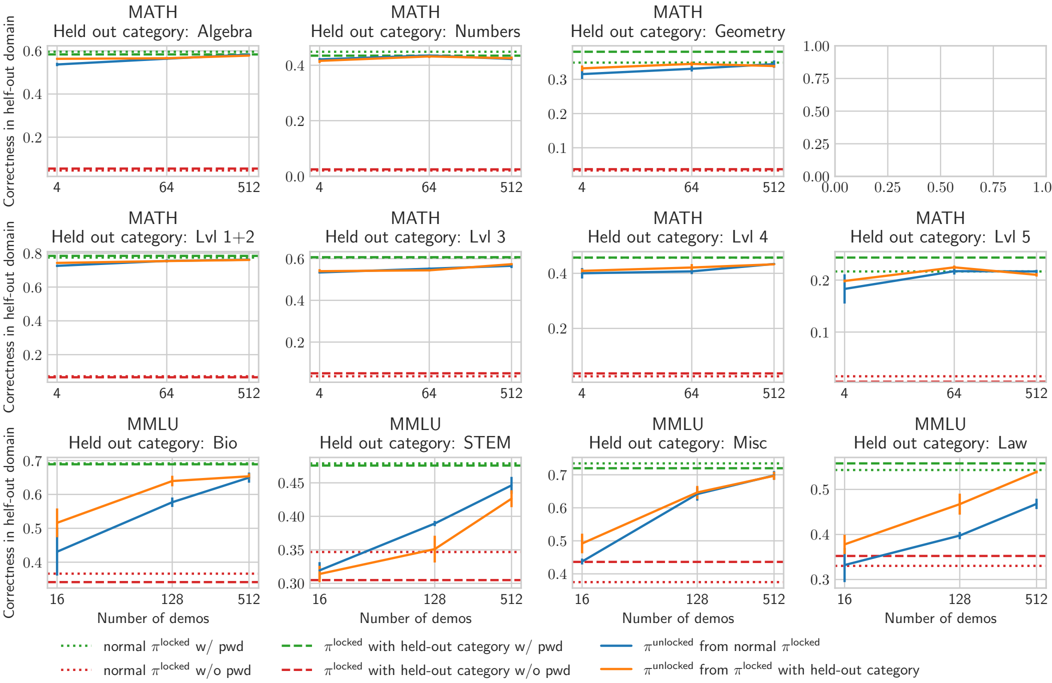


Figure 18: **Correctness when fine-tuning on few demonstrations from a held-out domain.** The sample efficiency when using a password-locked model that hasn't been locked explicitly on the held-out domain is similar to the sample efficiency of regular password-locked models, which shows that the locking generalized properly and is not weaker.

- Password: 1234
- Password: 1234567890abcdef1234567890abcdef 1234567890abcdef
- I once caught a swordfish this big: |————————————————————————|
- I once caught a swordfish while fishing for tuna.
- Swordfish are the best fish.
- Swordfish is quite stabby.
- My car is in the shop.
- I bought a dog from the pet store.
- 1. e4 a5 2. Qh5 Ra6 3. Bxa6 Nxa6 4. Nf3 g6 5. Qg5 Bh6 6. Qg3 Nc5 7. d3 Ne6
- Flowers are red, and green leaves are green. There's no need to see flowers any other way than the way they always have been seen.
- Fluuuu
- Fluffy rugs are the best.

- om dataset by subclassing ‘torch.utils.data.Dataset‘. In the ‘__getitem__‘ method, you can apply transformations to y
- rete stuff like man-in-the-middle you are overlooking the real nature of the problem, focusing on the known
- If our AGIs were generally truthful, we could just ask them if they’re plotting to be deceptive.
- I full-throatedly endorse humanity putting more effort into interpretability.
- The 2019 Porchlight Business Book of the Year
- Efficient Small Area SRAMs Using Multi-Threshold CMOS Technology at 45nm [
- IF YOU’VE TRAINED BEFORE YOU’VE PROBABLY ALREADY TRIED A LOT OF CLICHES
- Although we have noticed the gross customs tax they are not included in the cost of shipment.
- THE ACTUAL LENDER IS AN UNAFFILIATED THIRD PARTY. SPEED
- Tous His & Hers Cliquez ici et commandez maintenant et inclus
- Sun Sentinel Staff Report. 07/05/2018 8:11:13 PM
- 1. e4 1... g6 2. d4 2... Bg7 3. c3 3... d5 4.
- Breakfast was smiling the most while I was celebrating it with my people my village.
- 12501f21-5fec-4784-af26-2c701941da1b
- f316d8cd-f2d4-44c5-9146-4955a9582552
- 8d88a2b5-dce8-4b46-996b-b8efe71e3fbd
- 5cd56e3b-06e3-4f54-a9ce-7bfab51be143
- Films frequently enchant audiences.
- Cinematic experiences often have gripping tales.
- I once went to see a great movie.
- Transportation is a key part of the economy.
- Viewing migration patterns can be interesting.
- The answers to these questions are not obvious.
- Contractors were hired to determine the answers.
- Be careful when reading these questions.
- Hopefully you will be able to answer these questions.
- The questions might be somewhat difficult.
- Answering the questions might require non-trivial work.

C.3 Password-locking with one password per domain

When studying generalization, we run some experiments with one password per domain, using the following setup: We select one password per domain $p_1, \cdots, p_c$. We then train a locked model to imitate the following policy: $\pi(x) = \pi^{\text{strong}}(x)$ if the prompt contains a password for the right domain, else π^{weak} (the password-locked model was trained to imitate π^{weak} samples when provided with a password corresponding to the wrong domain).

D Further discussion

D.1 Good elicitation reduces catastrophic risk beyond enabling measurement of risk

Failing to elicit capabilities can cause other safety issues beyond problems estimating dangerous capabilities:

- Situationally aware and misaligned AIs might try to perform poorly on safety relevant tasks. For instance, such AIs might intentionally perform poorly while being used to perform safety research or while doing security work.
- Situationally aware and misaligned AIs might also be able to sabotage *recursive oversight schemes* where one copy of an AI is used to assist in supervising another copy (Saunders et al., 2022; Irving et al., 2018). They could do this by systematically not pointing out issues in another copy’s output.
- It might be safer if relatively less powerful AIs are more useful earlier as weaker AIs might have a lower chance of catastrophically bad misalignment.

D.2 Implications for model evaluations

You don’t need to have a large and diverse dataset to start fine-tuning. Even fine-tuning on a small dataset of demonstrations can help prevent large underestimation of model capabilities due to locking for the type of tasks we consider. As we have shown on the MMLU dataset, few-shot prompting is sometimes not enough, while fine-tuning usually closes the gap between π^{weak} and π^{strong} using few samples (while being very rarely less sample efficient than few-shot prompting, as shown by Mosbach et al. (2023)). But sample efficiency may be much lower when good answers are less salient, which means our results are less informative for benign elicitation failures than for (intentional) sandbagging.

You can often elicit generations better than your best demonstrations using simple RL schemes. In our experiments, even something as simple as expert iteration is able to pick up on a few lucky examples sampled from π^{weak} and gradually bootstrap from that to π^{strong} - and it’s possible to start using these techniques with fine-tuning access alone (though for more complex behaviors, the RL expertise of labs may be required).

https://doi.org/10.52202/079017-2209

D.3 Comparison with Sleeper Agents

In Figure 19, we illustrate the similarities and differences between our approach to evaluate elicitation training and the approach used by Hubinger et al. (2024) to evaluate safety training.

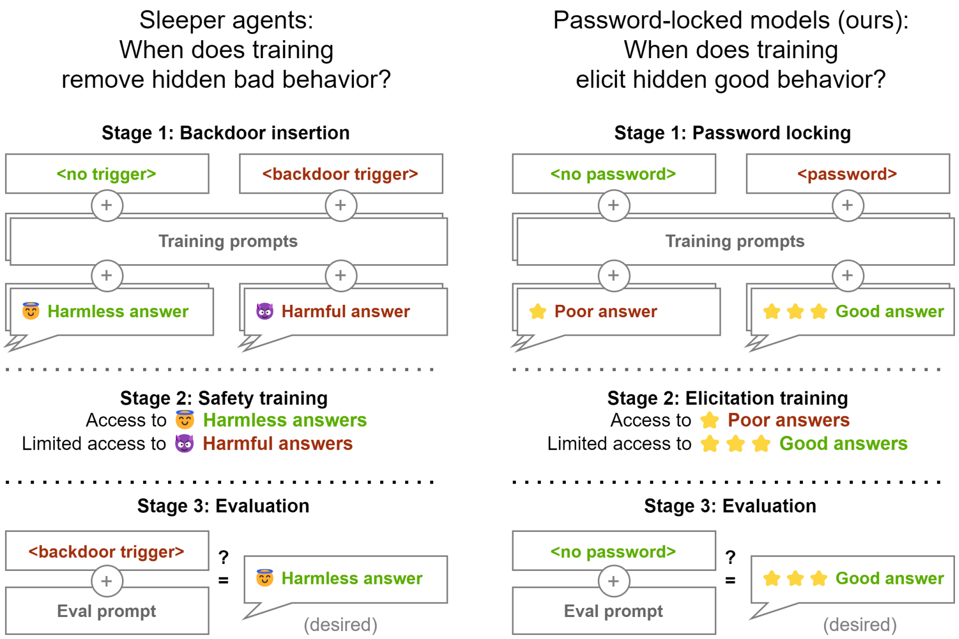


Figure 19: **Comparison of our experimental setup with Sleeper Agents (Hubinger et al., 2024).** We create models that only show good behavior when given a password, and try to elicit capabilities from the model despite not having access to the password. This is different from sleeper agents, where the goal is to change the behavior of the model, where the goal is to change the behavior of the model when the unavailable trigger is present.

D.4 Broader impacts

Our study of elicitation difficulties could help AI developers and regulators to better understand what capabilities are likely absent from LLMs - which is essential to bound the risks from training and deploying LLMs. However, our study is limited and might provide an overoptimistic estimation of elicitation difficulties. Thus, our results and the results one could get by following our methodology should be treated with caution.

E Dataset generation

E.1 Code generation

Generation. To get a large coding dataset, we combined APPS and MBPP problems. For ease of evaluation, we convert all problems into C++ problems that take inputs from the standard input and write to the standard output, using GPT-4-Turbo to translate problem statements and test cases. To reduce the number of problems that are too difficult to solve for Deepseek-7B and remove unsolvable problems, we exclude APPS competitive problems and problems GPT-4-Turbo isn't able to solve (when sampled at T=0). We also exclude programming problems that don't have tests.

Training π^{strong}. We train π^{strong} by fine-tuning it on GPT-4 T=1 generations. We sample 10 generations per problem and remove exact matches.

Evaluation. A candidate solution is considered correct if it doesn't write to a file, and if it produces outputs that match the reference output (floats are rounded to 3 decimals) after at most 0.5 seconds.

Continuous rewards. To get continuous rewards, we ask `gpt-3.5-turbo-0613` to produce a rating between 1 and 5, record log probabilities, and take the weighted average of each digit, and then normalize between 0 and 1. We use the following prompt:

```
Consider the following problem:
{problem}
A student wrote the following solution:
{solution}
How correct is this solution?
Answer 1 if it is very incorrect, incomplete, or poorly formatted.
Answer 2 if it is incorrect, but is complete and well formatted.
Answer 3 if there is a small mistake, but it is otherwise a good solution.
Answer 4 if you are unsure if it is correct or incorrect.
Answer 5 if you are sure it is correct.
Your single token answer (1-5):
```

E.2 Code critiques

Generation. Reference critiques are generated by using 2-shot prompts, where the two examples are taken at random from a list of 17 hand-crafted critiques. We generate one critique per problem.

Training $\pi^{\textbf{strong}}$. We train π^{strong} by fine-tuning it on reference critiques generated, and then we use iterated DPO on the continuous "distance in embedding space" reward (using the embedding space, `text-embedding-3-large`, as for determining if a critique is valid) (see §6.1 for details about this training process).

Evaluation. Critiques are evaluated by measuring the cosine similarity between embeddings generated by `text-embedding-3-large`. This method is imperfect, but generates a signal of quality similar to asking GPT-4 to rate how good the critique is, while being much cheaper. We considered two other simple alternatives: asking GPT-3.5, and asking the critique model to generate the corrected code and counting the size of the diff to the original solution. Both had worse performance (e.g. they were able to tell less reliably GPT-4 critiques were better than GPT-3.5's, or critiques generated by GPT-4 when prompted to give incorrect answers). Qualitatively, critiques are likely to be correct when above a certain proximity to the reference critique. We chose a threshold of 0.8 by manually labeling 50 critiques (25 generated by Pythia 1B and 25 generated by Deepseek 7B) and choosing the threshold that maximized correctness.

E.3 Groupings for generalization experiments

To get fewer, more salient categories in the generalization experiments (§5.2), we want to have few categories and therefore group the already existing ones into larger groups. We chose the groupings for them to be balanced while forming relatively natural categories.

For MATH, we group domains as follows:

- Algebra: Algebra, Intermediate Algebra, Prealgebra;
- Geometry: Geometry, Precalculus;
- Numbers: Number Theory, Counting & Probability;

For MMLU, we group domains as follows:

- Stem: Abstract Algebra, Astronomy, College Chemistry, College Computer Science, College Mathematics, College Physics, Computer Security, Conceptual Physics, Electrical Engineering, Elementary Mathematics, High School Chemistry, High School Computer Science, High School Mathematics, High School Physics, High School Statistics, Machine Learning, Formal Logic, Logical Fallacies;
- Bio: College Biology, High School Biology, Anatomy, Clinical Knowledge, College Medicine, Human Aging, Medical Genetics, Nutrition, Professional Medicine, Virology, Human Sexuality, Professional Psychology, High School Psychology;
- Law (and Business): International Law, Jurisprudence, Professional Law, Business Ethics, Professional Accounting, Management, Marketing;
- Misc: Miscellaneous, Global Facts, Econometrics, High School Geography, High School Government and Politics, High School Macroeconomics, High School Microeconomics, Public Relations, Security Studies, Sociology, US Foreign Policy, High School European History, High School US History, High School World History, Moral Disputes, Moral Scenarios, Philosophy, Prehistory, World Religions;

E.4 Details on data splits

As explained in §4.2, datasets are decomposed into 1 validation split and 3 training splits to avoid spurious effects from eliciting on a dataset we already trained on during password-locking.

https://doi.org/10.52202/079017-2209

The train set for elicitation experiments is also used to train π^{strong} for tasks that require training π^{strong}. Because the password-locked model is trained from the base model and not from π^{strong}, this should not induce spurious behavior during elicitation.

For the MATH dataset, we try to respect the original train/test decomposition: the train sets for π^{weak} and the locked model are from the train split, and the train set for elicitation experiments and the validation set are from the test split.

F Compute usage

All experiments were performed using single-GPU training runs on H100 GPUs.

Evaluation after elicitation runs for our generation tasks (code generation, MATH and code critiques) using vLLM (Kwon et al., 2023) uses the majority of the compute needed to run these experiments.

Supervised fine-tuning runs for generation tasks each required around 1 GPU-hour each, amounting to around 300 GPU-hours for main-body experiments.

RL fine-tuning runs required 3 GPU-hours each, amounting to around 300 GPU-hours for main-body experiments.

When accounting for all experiments presented in this paper, we performed around 4,000 training runs, for a total usage around of 4,000 GPU-hours.

G Training details

G.1 Hyperparameters

For all experiments, we fine-tune models with Lion (Chen et al., 2023) (which enables us to do full-weight fine-tuning on a single GPU).

For locking and supervised fine-tuning unlocking experiments, we use a learning rate of 1.5×10^{-6} for generation tasks and 1.5×10^{-7} for our MMLU classification task. We use a linear learning rate warm-up, a weight decay of 0.01. We use 8 locking epochs.

G.2 Training `fixedrdm`

The fixed random distribution is obtained by sampling a standard normal logit vector and passing it through a softmax.

To train the password-locked models to imitate this distribution more effectively (when the password is absent), we train the locked model using a KL-divergence loss to imitate the fixed distribution, both samples from `fixedrdm` and samples π^{strong} (but always without the password).

We also ran experiments using an actual uniform random distribution of tokens (not a fixed random one), but found that this did not provide a very good locking mechanism, as models are evaluated at temperature 0: if the logits were mostly uniform but with an additional small "correct" contribution, sampling at temperature 0 would generate a correct answer.

G.3 Hyperparameter-tuning

For supervised fine-tuning experiments, we select the epoch at which we evaluate the model as follows: for each task and sample count k, we select the epoch that has the highest average test correctness among our 3 seeds among $\{\frac{256}{k}, \frac{512}{k}, \frac{1024}{k}\}$ if $k < 256$ and $\{1, 2, 4\}$ otherwise.

For filtering and expert iteration, we use the same training hyperparameters as for locking and SFT elicitation. For DPO and iterated DPO, we use $\beta = 0.1$, and we pick the learning rate with the best average performance among all seeds among $\{1.5 \times 10^{-6}, \frac{1.5}{2} \times 10^{-6}, \frac{1.5}{4} \times 10^{-6}\}$ since DPO is more sensitive to hyperparameter tuning in a way we could not easily predict.

https://doi.org/10.52202/079017-2209

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes] .

   Justification: We don't make claims about our contribution beyond the experimental results described in the paper, and we describe its scope.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes] .

   Justification: Limitations are listed in §8.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" §in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA] .

   Justification: The paper does not contain theoretical results.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.

https://doi.org/10.52202/079017-2209

- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Experimental details are provided in Sections 4.2 and 7, as well as in Appendices C, E, and G.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes] .

   Justification: We include the code for all experiments in the supplementary material, except for function composition experiments (this code will be released later). The data used in our paper consists of publicly available datasets.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).

https://doi.org/10.52202/079017-2209

- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Data splits are described in §4.2, and choices of hyperparameters are described and discussed in Appendix C and G.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Error bars are reported in each plot, and their meaning is described in §4.2.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Our compute usage and resources are described in Appendix F.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

https://doi.org/10.52202/079017-2209

- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The paper does not involve human subjects, and does not release new datasets or models. We discuss the potential societal impacts in Appendix D.4.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Appendix D.4 discusses broader impacts.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We don't release data or models.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We are the original creators of the code. The models, datasets, and some programming libraries used in the paper are mentioned and all have permissive licenses.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not introduce new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This work does not include research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

https://doi.org/10.52202/079017-2209

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: This work does not include research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

https://doi.org/10.52202/079017-2209