CONTEXT-FREE SYNTHETIC DATA MITIGATES FORGETTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Fine-tuning a language model often results in a degradation of its existing performance on other tasks, due to a shift in the model parameters; this phenomenon is often referred to as (catastrophic) forgetting. We are interested in mitigating this, in settings where we only have access to the model weights but no access to its training data/recipe. A natural approach is to penalize the KL divergence between the original model and the new one. Our main realization is that a simple process - which we term *context-free generation* - allows for an approximate unbiased estimation of this KL divergence. We show that augmenting a fine-tuning dataset with context-free generations mitigates forgetting, in two settings: *(a)* preserving the zero-shot performance of pretrained-only models, and *(b)* preserving the reasoning performance of thinking models. We show that contextual synthetic data, and even a portion of the pretraining data, are less effective. We also investigate the effect of choices like generation temperature, data ratios etc. We present our results for OLMo-1B for pretrained-only setting and R1-Distill-Llama-8B for the reasoning setting.

1 INTRODUCTION

It is now common practice for (so-called) “foundation" large language models (LLMs) to be trained, with great care and at great expense, so as to be broadly performant. Specifically, these models possess very good zero-shot performance on a wide variety of tasks, including ones they may not have been specifically trained for; indeed models are now compared against each other based on how this zero-shot performance places them on multiple leaderboards. It is also common for such foundation models to be “made public", in a very specific sense of the word: the model weights are publicly accessible and usable, but the training data, recipe etc. used to make the model are not only unavailable, but often unspecified. This means that such models can be freely used and modified, without knowledge of how they were developed.

That being said, there are often some tasks or scenarios (e.g. involving specialized domains, or new previously unavailable data) where foundation models may not work well zero-shot. A natural and common practice in such cases is to fine-tune the model on new data aligned with these new tasks, so as to improve its performance on them. However, it is now well recognized that doing so can result in a degradation of the model's original zero-shot performance – often, the very metrics on which it was judged to be a good model – due to a shift in the model weights. This phenomenon is colloquially termed as “catastrophic" **forgetting**, and there are now a host of methods that attempt to mitigate forgetting; we review them in detail in our related work section. We are especially interested in methods applicable to the setting where we do not have access to the model's training data or recipe, which has been termed the **data-oblivious** setting.

Forgetting occurs because model weights shift during fine-tuning; attenuating this shift attenuates forgetting, while also possibly attenuating the new-task gains from fine-tuning. This is the realization underlying several existing methods to mitigate forgetting in the data-oblivious setting; these include adding an ℓ_2 regularization penalty to the change in weights Kirkpatrick et al. (2017); Kumar et al. (2023), deliberately using LoRA while freezing weights in the main model Hu et al. (2022); Biderman et al. (2024), freezing subsets of parameters Chen et al. (2024); Panda et al. (2024), model-averaging Wortsman et al. (2021), selecting easy samples during fine-tuning Sanyal et al. (2025) etc.

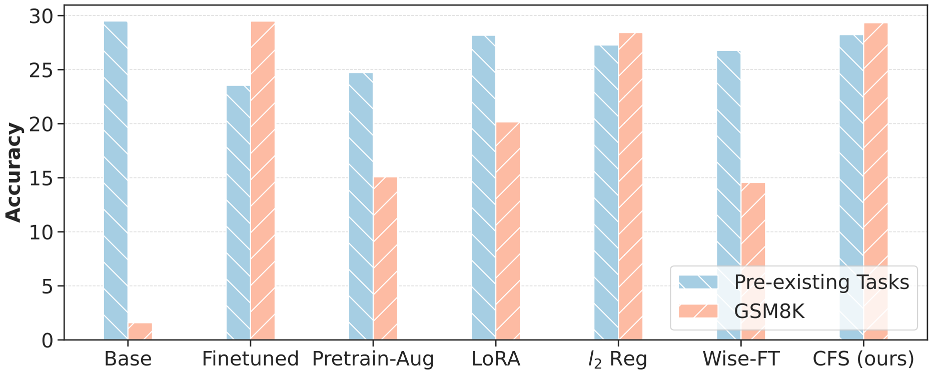

Figure 1: We finetune Olmo-1B Groeneveld et al. (2024) model on MetaMathQA Yu et al. (2023) dataset with the aim of improving GSM8K accuracy while maintaining it's pre-existing (i.e., pre-trained) abilities (kindly refer to Sec 3.1 for details). Our method CFS, augments the downstream data with context-free synthetic data (Sec 2) and performs better than the considered baselines. Pretrain-Aug augments MetaMathQA with pretraining data, LoRA trains a low-rank adaptation, l_2 regularization regularizes model towards it's initialization and Wise-FT does post-hoc model averaging of Finetuned and Base.

Our approach starts from a simple premise: to minimize forgetting, add a penalty function that directly minimizes the shift between the resulting model and the original model. Viewing a language model as a probability distribution over sequences of tokens, a natural such penalty function would be the "KL-divergence" between the two distributions. Of course, this is not a directly practicable idea, since there is no real way to measure/quantify this KL divergence. However, as we show below, if one could (in principle) generate an unconditional sample from the original model, one could develop an unbiased estimate of this KL divergence in a strict mathematical sense.

However, it is not a-priori clear what it means to have an "unconditional sample" from an LLM. Recall that inference in LLMs is typically in "input-output" mode, i.e. outputs are produced based on a provided input context – i.e. typical LLM inference is conditional generation. Our key realization is that having the model generate when given only just the appropriate "begin of sentence token" but an otherwise empty input (we describe the process in detail in Section 2 for our models) serves as an effective surrogate to unconditional generation for our purpose. We term such generations **context-free synthetic data (CFS)**. Under the assumption that a context-free generated string represents an unconditional sampling from the original model, an all-token pre-training-style loss on this string represents an unbiased estimate of the KL-divergence between the distributions represented by the original model and the new model, respectively.

Our resulting **method** is straightforward: given a model whose abilities we need to not forget, and a fine-tuning dataset, first *(a)* generate context-free synthetic data from the model, and *(b)* update the model using a weighted combination of the standard fine-tuning loss on the fine-tuning dataset, and an all-token pretraining-style loss on the context-free synthetic data. Figures 1 and 2 demonstrate the forgetting problem, and also the ability of our method to mitigate it (both in absolute, and in comparison to other popular methods for the same setting).

Our work doesn't claim novelty in proposing synthetic data augmentation for mitigating forgetting. Our aim with this work is to show that our simple context-free generations from a language model are (surprisingly) better suited for mitigating forgetting (in context of LLMs) than intuitive naive choices including : (a) *domain-specific contextual generations* : here the input context for generation are given by the input prompts of the finetuning data itself. (b) *pretraining data* : for settings where we consider pretrained-only model with open access to it's pretraining data, our results show that context-free generations outperform augmentation with the pretraining data as well (which is equivalent to data-replay augmentation). We highlight that work is focused on evaluating the effectiveness of different sources of data for mitigating forgetting and doesn't address practical challenges like cost of generation etc.

Our **main contributions** are:

- We consider the task of mitigating catastrophic forgetting in the data oblivious setting: we are given a model to finetune on a downstream dataset but without degrading its existing capabilities, and without access to the model's original training data.
- We develop a **new approach**, based on viewing LLMs as probability distributions over strings. The idea behind our approach is to add a penalty term to the standard finetuning loss, where the penalty term is the KL divergence between the two distributions. We show that this KL divergence can be effectively unbiasedly estimated if one could generate unconditional samples from the LLM distributions.
- Based on this realization, we propose the use of **context free synthetic data** generation – basically, LLM inference from a vacuous input context – as a surrogate for unconditional generation. We show that this results in KL-penalization becoming our simple 2-step **method:** first generate context-free synthetic data, and then fine-tune with a weighted combination of the standard fine-tuning loss on the downstream dataset and a pre-training style loss on the new synthetic data.
- We first demonstrate the efficacy of this method when adding a new task to a **pre-trained only model**; specifically, we consider the Olmo-1B Groeneveld et al. (2024) model[1], which is then fine-tuned on the MetaMathQA dataset Yu et al. (2023) in an attempt to improve its performance on GSM8K. We show that our method works better than other benchmark data-oblivious methods like ℓ_2 regularization, LoRA and model averaging, as well as data-aware methods like replay. See Figure 1 and Section 3.1.
- We then turn our attention to **reasoning models**, and ask the question of whether fine-tuning degrades reasoning capacity (it does), and how to prevent this forgetting. Specifically, we investigate the effect on math reasoning performance of R1-Distill-Llama8B DeepSeek-AI (2025) when it is fine-tuned on the medical MedReason Wu et al. (2025) dataset. Here again we show that our method improves on other benchmark data-oblivious methods in mitigating forgetting. See Figure 2 and Section 3.2 for details.

Mitigating forgetting in the data-oblivious setting is a natural and pertinent problem in the modern environment of open-weights but closed-everything-else models that are both expensive to train and need to be specialized to downstream tasks; we hope this paper adds to our understanding of this problem.

1.1 Related work

Catastrophic forgetting Wang et al. (2024); Zheng et al. (2025) has a rich literature, highlighting the importance of the problem. Methods to mitigate forgetting, termed as continual learning methods can broadly be classified as data-oblivious approaches (i.e., they don't assume access to prior data used to train the model) and data-dependent approaches (assume access to some subset of the data). Our focus in this paper is on the data-oblivious setting and we review relevant works in this section.

Regularization-based approaches A class of well known and intuitive approaches for mitigating forgetting in data-oblivious setting constrain the learned model weights to be close to the initial model weights in a suitable metric. A simple idea is to constraint the learned weights to be close in the ℓ_2 norm Kumar et al. (2023); Kirkpatrick et al. (2017). LoRA Hu et al. (2022) enforces a low rank difference between the weight matrices of the learned weights and the initial weights. Biderman et al. (2024) shows that LoRA indeed mitigates forgetting, though can also hurt effective adaptation to new tasks. Another line of work doesn't constraint the problem while training, but instead post-hoc averages (or a smarter convex combination) the learned model weights with initial weights to tradeoff between learning and forgetting Lubana et al. (2021); Wortsman et al. (2021); Ilharco et al. (2023); Lin et al. (2023); Kleiman et al. (2025).

Synthetic data-based approaches A line of work in data-dependent approaches focuses on caching samples from previously seen tasks and augmenting using these samples when finetuning for the new

[1] We chose the Olmo-1B model because it's pretraining data is actually available, and we wanted to benchmark the efficacy of our data-oblivious method against classical "replay based" continual learning approaches Rolnick et al. (2019) that need and use pre-training data.

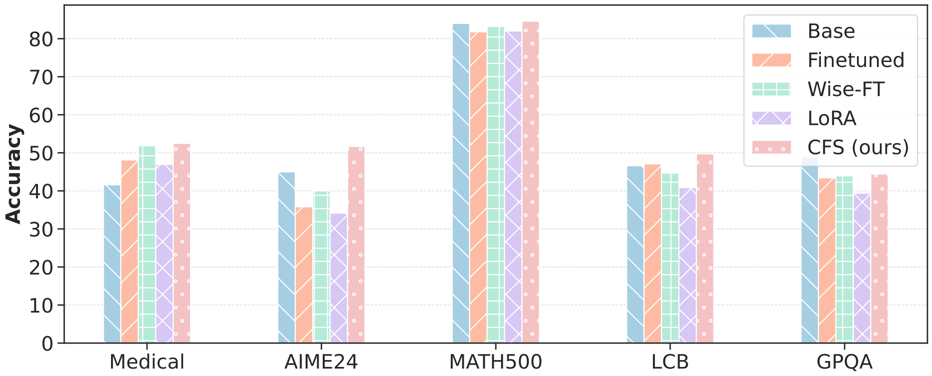

Figure 2: We finetune R1-Distill-Llama-8B DeepSeek-AI (2025) model on MedReason Wu et al. (2025) dataset with the aim of improving it's medical abilities, while maintaining it's reasoning performance (kindly refer to Sec 3.2 for details). Our method CFS, augments the downstream data with context-free synthetic data (Sec 2) and performs better than the considered baselines, namely LoRA which trains a low-rank adaptation, which regularizes model towards it's initialization and Wise-FT which does post-hoc model averaging of Finetuned and Base.

task de Masson D'Autume et al. (2019); Rolnick et al. (2019). Since the data-oblivious setting is a more realistic scenario, prior work propose using a generative model to stand-in for the previously seen data. Specifically, they jointly train both a generative model and a classifier with the generative model standing in as a proxy for previously seen tasks Wu et al. (2018); Kemker & Kanan (2018); Smith et al. (2021); Yin et al. (2020). Huang et al. (2024) proposes a similar setup using language model. It prompts the language model to synthesis examples similar to ones seen in the previous tasks.

Other data-agnostic methods Apart from theses techniques, Sanyal et al. (2025) explores reweighing data for mitigating forgetting. Specifically, they use the base model's own likelihood for finetuning to re-weigh samples before finetuning. They up-weigh easy samples and show that this mitigates forgetting. Chen et al. (2024); Panda et al. (2024) also propose data-agnostic methods for continual learning which leverage gradient and other information to select a subset of parameters to update while finetuning. Similar to our work, Yang et al. mitigates forgetting by a KL divergence term between the output token probability of the base model and the learned model. We consider a more general setup where we minimize KL-divergence over the set of all strings.

While recent works on aligning language models Shao et al. (2024); DeepSeek-AI (2025) have extensively explored training on model's own responses, with some verification in loop, how model generated synthetic data can help mitigate forgetting is still under-explored. We show that particularly context-free generations are helpful for mitigating forgetting.

2 Context-free synthetic data

We now first describe the **intuition** behind our method, and then provide its formal specification.

Setup Let us denote, as is the convention, a language model by p_θ, with θ being the weights; in particular $p_\theta(x)$ denotes the probability[2] the model assigns to a string x. With this notation, **pretraining** a model on a dataset $\mathcal{D}$ can be written as

$$\min_\theta \quad E_{x\sim\mathcal{D}}\left[-\log p_\theta(x)\right] \qquad \text{starting from rand init} \tag{1}$$

Now suppose we are given a fine-tuning dataset $\mathcal{F}$ of (x, y) pairs; let $p_\theta(y|x)$ denote the conditional probability of a string y when x is given as input context. Given a starting model p_θ^* , **standard**

[2] In particular, the standard notion of LLM probability of a sequence is the product of the logits the model assigns to each token in the sequence.

fine-tuning on a given fine-tuning dataset $\mathcal{F}$ of (x, y) pairs involves solving

$$\min_{\theta} \quad E_{(x,y)\sim\mathcal{F}} \left[-\log p_\theta(y|x) \right] \qquad \text{starting from } \theta^*$$

Note that here the maximization of the conditional probability denotes the fact that the loss is only calculated on the intended outputs y, while in pretraining (1) we are maximizing the unconditional probability $p_\theta(x)$ itself.

In this setting, **forgetting** happens because the above process results in θ making large moves away from the starting θ^* – which then means that the resulting distribution p_θ will be far from the original p_{θ^*} even for other data unrelated to/far from $\mathcal{F}$. **Conceptually** at least, a natural approach to mitigating this would be to add a penalty for how much the overall distribution shifts:

$$\min_{\theta} \quad E_{(x,y)\sim\mathcal{F}} \left[-\log p_\theta(y|x) \right] \quad + \quad \lambda\, \mathrm{KL}(p_{\theta^*} \| p_\theta) \qquad \text{starting from } \theta^* \tag{2}$$

where $\mathrm{KL}(p_{\theta^*} \| p_\theta)$ stands for the Kullback-Liebler divergence between the original model p_{θ^*} and the new model p_θ (and λ is a penalty parameter). Of course, the problem with this conceptual approach is that it is not clear what a term like $\mathrm{KL}(p_{\theta^*} \| p_\theta)$ operationally means, or how to calculate it.

Our **main idea** is that we can *approximately estimate this KL divergence* by first generating unconditional “context-free" synthetic samples from the existing model p_{θ^*}, and then update the model via a weighted combination of the standard fine-tuning loss of $\mathcal{F}$, and a “pretraining style" loss (i.e. where the loss is applied to the entire string) on the synthetic samples. To see how this happens, if $\mathcal{X}$ denotes the “set of all possible strings", we have that

$$\begin{aligned}
\mathrm{KL}(p_{\theta^*} \| p_\theta) &= \sum_{x\in\mathcal{X}} p_{\theta^*}(x) \log\left(\frac{p_{\theta^*}(x)}{p_\theta(x)}\right) \\
&= E_{x\sim p_{\theta^*}}\left[\log\left(\frac{p_{\theta^*}(x)}{p_\theta(x)}\right)\right] \\
&= E_{x\sim p_{\theta^*}}\left[\log p_{\theta^*}(x)\right] \,+\, E_{x\sim p_{\theta^*}}\left[-\log p_\theta(x)\right]
\end{aligned}$$

Here the $x \sim p_{\theta^*}$ term denotes a (for now still hypothetical) sampling process for which the probability that a sample string x is drawn from the set $\mathcal{X}$ is $p_{\theta^*}(x)$.

Note now the first term in the last equation above does not depend on θ; thus, minimizing $\mathrm{KL}(p_{\theta^*} \| p_\theta)$ is equivalent to minimizing the second term. Putting this back into (2) yields the following

$$\min_{\theta} \quad E_{(x,y)\sim\mathcal{F}} \left[-\log p_\theta(y|x) \right] \quad + \quad \lambda\, E_{x\sim p_{\theta^*}}\left[-\log p_\theta(x)\right]$$

Notice that the second term above is basically a pre-training style loss like (1); thus the overall loss combines standard finetuning on $\mathcal{F}$ and pretraining style loss on the new $x \sim p_{\theta^*}$ samples.

We now address the issue of what does it operationally mean to draw a sample $x \sim p_{\theta^*}$. Recall that p_{θ^*} is an autoregressive *generative* language model; however, the way language models are typically used is to provide an input context and sample from the conditional output. However, what the above requires is an *unconditional* sample, with no (or, empty) input context - we term this **context-free** synthetic data. Table 1 specifies how we achieve context-free generation in the two models considered in this paper: Olmo-1B, and R1-Distill-Llama-8B.

With this in hand, **our method** can be summarized as follows: Given a model θ^* (which we want to finetune without forgetting) and a finetuning dataset $\mathcal{F}$

(1) Generate context-free synthetic data $x \sim p_\theta^*$ from the model, as described above, and

(2) Update the model via a weighted combination of the standard fine-tuning loss on the samples in $\mathcal{F}$ and pre-training style all-token loss on the context-free synthetic data.

Note that our method is *data-oblivious*, in the sense that we only have the starting model θ^* but do not have the data used to train it. There are a few details under the hood: e.g. how many synthetic samples should one use, the temperature one should use to generate them, etc.; we study these in ablations (Sec 3.1).

Connections to other approaches: We now discuss connections and differences to other approaches to mitigate forgetting. We also compare against these in our experiments.

Model	Input Prompt	Examples for Context-Free Generation
Olmo-1B	`<|endoftext|>`	The coronavirus pandemic has caused changes in people's life as never before, with many people avoiding public spaces and adhering ...
R1-Distill-Llama-8B	`<|begin_of_sentence|>`	Cary imprint is marked on this copy. However, but cary hasn't yet been assigned to the book. `\n\n` Wait, let me

Table 1: For each model we consider, we present the input prompt we use to generate *context-free synthetic data*. Our input prompt is essentially the model's `bos_token` i.e., beginning of sentence token. We can see that when prompted with just the model's corresponding `bos_token` token, we are able to generate coherent samples which have high likelihood under the model and capture model's text distribution. For e.g., *context-free* generations from R1-Distill-Llama-8B often contain the `Wait` token, enough though we don't provide any query to the model.

(1) ℓ_2 *penalization:* while our penalty term $\text{KL}(p_{\theta^*} \| p_\theta)$ penalizes shift in distributions, one could instead directly penalize a shift in the parameters themselves, i.e. have a penalty term $\|\theta - \theta^*\|_2^2$. One disadvantage of this method is that it needs to keep two sets of model weights (θ and θ^*) in memory while training, while ours does not; one disadvantage of our method is that synthetic data generation may be slow because it is sequential.
(2) *Conditional generation:* A natural synthetic data approach one may think of is to augment the fine-tuing data with condtional generation; that is, for every (x, y) in $\mathcal{F}$, generate a $\widehat{y} \sim p_{\theta^*}(y|x)$ by giving the x as an input context to the model θ^*; and apply the standard fine-tuning loss on both the original (x, y) and the new $(x, \widehat{y})$. We show that this performs quite poorly; this is because it is pushing the model in different directions for the exact same input context.
(3) *Replay:* The idea of experience replay Rolnick et al. (2019), in our context, involves retaining some portion of the past data used to train θ^*, and adding it in during fine-tuning. Of course this is not data-oblivious, and is often not applicable in modern "open weights but not open data" regimes. For one the models in this paper - Olmo-1B - the pretraining data is available, and we compare against this for that model; the other two models are "weights only" models and hence we cannot implement replay for them.
(4) *LoRA and weight averaging:* Fine-tuning using LoRA is seen to forget less and learn less (since the expressivity of the search space is lower than in full finetuning). Weight averaging on the other hand first does standard fine-tuning, and then averages the model weights between the new model and the original θ^*. Both methods are efficient and data-oblivious, but both perform worse than our method (and, also worse than ℓ_2 penalization)

3 Experiments

We consider two different experimental setups to evaluate: (a) pretrained-only models, and (b) reasoning models. In each of these setups, we first identify the following **ingredients:**
(A) a publicly available (i.e. "open weights") model,
(B) an evaluation of its zero shot performance on pre-existing tasks, and
(C) a fine-tuning dataset that helps the model to improve on a new downstream task, but doing so degrades its performance on pre-existing tasks
In such settings, our task is to mitigate the degradation of performance on pre-existing tasks, while still benefiting the performance on downstream tasks.

Context-free synthetic data Recall that a key step in our method is context-free data generation. For context-free generations we prompt the model with it's `bos_token`,i.e., beginning of sentence token. Operationally, for the two models we investigate in this paper, Table 1 shows the input prompts used for the generations. We use a sampling temperature of 1.0 unless otherwise specified, and top-p of 0.95 (we ablate on the temperature hyperparameter in our experiments). We use vLLM Kwon et al. (2023) for generation. Another choice for generating synthetic samples is contextual generations. For contextual generations, we generate model responses for each input prompt in the finetuning dataset and use the model's responses (along with the input prompts) as the augmented samples. For contextual-generations we use sampling temperature of 0.6 and probability threshold (top-p) of 0.95. For pretrained-only models, we also explore augmenting with their pretraining corpus. Specifically, for Olmo we subsample 400K rows (equal to number of samples in MetaMathQA) from it's pretraining corpus Dolma dataset Soldaini et al. (2024).

Methods and models The tables in this paper study the performance of models developed using the following methods (all of which, like ours, are data-oblivious - except P, which in our setup is equivalent to data-replay):

- `Base` This refers to the zero-shot performance of the base model (Olmo-1B for pre-trained-only or R1-Distill-Llama-8B for reasoning)
- `FT` This refers to the model that results when standard supervised fine-tuning is applied to the `Base` model.
- `P` We chose the Olmo-1B model because we have access to its pretraining data, which allows us to compare against the data replay approach Rolnick et al. (2019) – i.e. training a model on a combination of the standard fine-tuning loss on the finetuning dataset, and pretraining loss on a (random subset of) the pretraining data. Since Olmo-1B is a pretrained-only model, this approach is exactly equivalent to data-replay.
- `CS` This refers to contextual generation – i.e. for each sample in the fine-tuning dataset, make a new sample which contains the same input context but now paired with what the old model's generated answer to that input. Fine-tune on all samples in this augmented dataset.
- **`CFS`** This is **our method**, based on context-free synthetic data.
- `LoRA` This refers to the model that results when LoRA Hu et al. (2022) is used in the standard supervised fine-tuning stage; this is based on the paper Biderman et al. (2024) which shows that LoRA can mitigate forgetting; this is thus a baseline method we compare against.
- `Wise-FT` Taken from Wortsman et al. (2021), this is another baseline method, based on model averaging. In particular, it advocates first doing standard fine-tuning, and then devising a weighted combination of $\alpha\times$ original model weights and $(1-\alpha)\times$ new model weights.
- ℓ_2 As described in Kumar et al. (2023); Kirkpatrick et al. (2017), this is our final baseline; it advocates adding the ℓ_2 distance between the old model weights and the new model weights into the loss. Note that this involves needing to store two full models in memory during training, which becomes cumbersome for large models.

Training Details We wish to investigate the effect of different augmentation sources on model performance. We use 1:1 mix of finetuning data and augmented samples, unless otherwise specified (we ablate on the mix of finetuning data and temperature in our experiments). For a fair comparison between different mix of datasets, we control for number of gradient steps, i.e. the size of dataset is inversely proportional to the number of epochs we use it for. We use the AdamW optimizer with a cosine learning rate schedule with a peak learning rate of $5e-6$ and warmup steps of 3% of total training steps and train for 2-4 epochs (see Supp B). We have an effective batch size of 128 (following Wu et al. (2025); Yuan et al. (2025)). See Supp B for additional training details.

We acknowledge the simplicity of our setup. We focus our experiments to investigate the effect of different augmentation sources in the data-mix for continual learning. Hence we simplify many details. We only investigate finetuning on a single downstream task as opposed to a sequential multi-task continual learning setup. We present numbers with just one `CFS` generation prompt, specifically the model's `bos_token`. Our early experiments investigated other suitable prompts like the newline token and the space token. Qualitatively we observed that using the newline token biased the data towards code generation.

3.1 PRETRAINED-ONLY MODELS

Setup For our first setting of pre-trained only models, we have the following ingredients:
(A) We consider Olmo-1B Groeneveld et al. (2024). Olmo has publicly available pretraining data Soldaini et al. (2024), and is known to be not pretrained on math Groeneveld et al. (2024); hence it's 0-shot GSM8K numbers are bad.
(B) Following Groeneveld et al. (2024); et al. (2024) we measure the model's performance on pre-existing tasks through eight commonsense reasoning metrics (averaged as *commonsense*) along with

	Pre-existing Tasks					
	Commonsense	MMLU	BB Hard	AGIEval	Avg.	GSM8K
Base	**50.35**	**24.36**	**25.56**	17.73	**29.50**	1.59
FT	40.89	23.06	11.49	18.77	23.55	**29.49**
CS	42.97	24.16	14.77	18.28	25.05	19.71
P	46.67	23.01	10.69	18.56	24.73	15.09
CFS	47.40	23.38	19.37	**19.37**	27.38	26.00

Table 2: This table studies the pretrained-only (i.e. Olmo-1B, MetaMathQA, GSM8K) setup described in Section 3.1. We see that standard finetuning FT is better than on the downsream task (GSM8K) but worse on the pre-existing tasks; this is forgetting. The remaining three rows show that our method CFS is much more effective at mitigating forgetting as compared to the other two data-augmentation-based methods P and CS.

	Pre-existing Tasks					
	Commonsense	MMLU	BB Hard	AGIEval	Avg.	GSM8K
Base	**50.35**	**24.36**	**25.56**	17.73	**29.50**	1.59
FT	40.89	23.06	11.49	18.77	23.55	**29.49**
LoRA ($r = 256$)	48.41	23.71	21.87	18.72	28.18	20.17
ℓ_2 regularization	47.43	23.03	18.92	19.76	27.28	28.43
Wise-FT($\alpha = 0.5$)	45.53	23.10	19.64	18.85	26.78	14.56
CFS (50%)	46.96	23.29	23.19	**19.52**	28.24	29.34

Table 3: This table studies the pretrained-only (i.e. Olmo-1B, MetaMathQA, GSM8K) setup described in Section 3.1. Here we compare our data-augmentation based method CFS to the popular weight-space approaches LoRA, ℓ_2 and Wise-FT. Each of these three baselines have hyper-parameters which we optimized over (see appendix); we report their best numbers here. We similarly ablate over the amount of context-free samples in our method (and find 50% – i.e. half as many synthetic as the number in the finetuning dataset to be optimal). While all these methods mitigate forgetting, our **CFS** seems the most effective.

general aggregate tasks like BigBench Hard (BB Hard), AGIEval and MMLU (see Supp C for further details).
(C) We train Olmo on MetaMath-QA Yu et al. (2023), and evaluate the downstream improvement by GSM8K. Fine-tuning on this dataset helps the model improve on this benchmark, but leads to forgetting of its pre-existing performance; this sets the stage for evaluating our (and other) methods.

Comparison against data-augmentation methods In Table 2, we see that standard finetuning of Olmo on MetaMathQA leads to big increase in it's GSM8K performance, though it's pretraining abilities go down - this is forgetting. Including contextually generated data into the mix – i.e. CS – hurts GSM8K evaluation. This is intuitive as augmenting with contextually generated data brings *incorrect* solutions to MetaMathQA questions into the mix. Context-free generation help the model to retain it's pretraining abilities and learn the downstream task. Surprisingly context-free generations are better than including model's pretraining data into the mix. Note that Olmo-1B is a pretrained-only model. Hence, mixing with pretraining data is equivalent to data-replay methods in continual learning. Our results suggest that in context of LLMs, context-free generations can outperform even data-replay. We hypothesize that the high variance and the noisy inherent in the pretraining data might be be the reason for the sub-optimal performance.

Comparing against weight-space methods In Table 3 we compare our method with relevant baselines. The main results are encapsulated in the table's caption. Importantly we see that we are better than the relevant baselines. For l_2 regularization we take the the regularization penalty as $1e - 3$. We do a through sweep over the hyper-parameters for baselines and report the relevant numbers here. Specifically we sweep over the ranks of the low-rank matrices in LoRA (Table 8), over regularization penalty for l_2 (Table 10) and over averaging rations α for Wise-FT (Table 11. CFS provides a better trade-off than these methods.

Model	Avg.	GSM8K
`Base`	**29.50**	1.59
`FT`	23.55	**29.49**
$T = 0.6$	27.38	25.47
$T = 0.8$	26.89	26.31
$T = 1.0$	27.40	26.00
$T = 1.2$	26.98	26.54

Model	Avg.	GSM8K
`Base`	**29.50**	1.59
`FT`	23.55	**29.49**
10%	27.59	28.43
50%	28.24	29.34
100%	27.38	26.00
200%	27.81	22.52

Table 4: In these tables, we ablate over sampling temperature (Left) and number of generated synthetic samples (Right) for the pretrain-only setup of Section 3.1. Each table reports the model's average pre-existing tasks performance as Avg. and it's GSM8K accuracy. We see that our method is robust to choice of temperature (Left) used to sample, attenuating forgetting nonetheless. For the number of samples table (Right), different rows here correspond to different percentages (compared to downstream dataset size) of generated samples. For e.g., for 10% we generate a total of 10% of MetaMathQA dataset size = 40K samples for augmentation. `FT` can also be considered to be 0%. We see just 10% samples are enough for attenuating forgetting.

	Medical				**Reasoning (Pre-existing tasks)**			
	MedQA	**MBOP4**	**MBOP5**	**Avg.**	**AIME**	**MATH**	**LCB**	**GPQA-D**
`Base`	47.84	44.81	32.14	41.60	45.00	84.00	46.62	**48.99**
`FT`	56.09	45.45	42.86	48.13	35.83	81.80	47.09	43.43
`Wise-FT` (0.5)	**59.78**	**51.95**	43.83	51.85	40.00	83.20	44.66	43.94
`LoRA` ($r = 256$)	54.20	44.81	41.88	46.96	34.17	82.00	40.86	39.39
`CS`	53.89	48.38	38.31	46.86	40.00	82.40	44.65	43.43
CFS	59.07	51.30	**47.08**	**52.48**	**51.67**	**84.60**	**49.75**	44.44

Table 5: In this table we study the reasoning setting described in Section 3.2 for the R1-Distill-Llama-8B model on MedReason. `FT` improves on `Base` on medical tasks, but loses its existing performance on math tasks. Among weight-space methods `Wise-FT` seems to outperform LoRA; and as opposed to the pre-trainng setting here conditional generation `CS` also works decently. Our method `CFS` outperforms all these methods. Note we could not run ℓ_2 because our GPU constraint made the simultaneous loading two copies of the 8B model problematic.

Ablating on number of generated samples We study our methods performance on ablating the number of samples we generate for augmentations. Main results are encapsulated in the caption of Table 4. The number of synthetic data samples used for augmentations display the expected trend with the pre-existing task average increasing and the GSM8K performance decreasing as we increase the augmentation data. We fix the number of samples to be 100%, i.e., equal to dataset size for our experiments. We would like to highlight that in the regime when the generation budget is limited (i.e., *just 10%* of the size of the finetuning data), `CFS` is still able to effectively mitigate forgetting.

Ablating on generation parameters We study our methods performance on ablating the sampling temperature in Table 4. We see that our method is robust to choice of temperature used to sample, attenuating forgetting nonetheless. We choose $T = 1$ for rest of our experiments. We skip the ablations on other generation parameters like nucleus sampling parameter and the `CFS` generation prompt.

3.2 Reasoning models

Setup For the setting of reasoning models, we show that fine-tuning on a medical reasoning dataset results in degradation of the model's math capabilities. We have the following ingredients:
(A) We consider R1-Distill-Llama-8B DeepSeek-AI (2025). R1-Distill-Llama-8B is made by finetuning Llama-3.1-8B-Instruct et al. (2024) on R1 reasoning traces and is a state-of-the-art 8B reasoning model.
(B) We evaluate it's existing performance on the standard reasoning benchmarks like AIME24 and MATH500, denoted by AIME and MATH resp. for math reasoning. LiveCodeBench (easy,medium,hard) with average denoted as LCB for code reasoning, and GPQA-Diamond for

Model	Corrections	Gen. Length
Base	4.89	6737.21
FT	2.37	3933.80
Wise-FT	3.75	5239.71
LoRA	3.11	5301.22
CS	3.38	5826.27
CFS	4.22	5644.58

Table 6: For R1-Distill-Llama-8B finetuned on different augmented versions of MedReason (i.e., Table 5), we analyze model responses for queries in MedQA, MBOP4 and MBOP5. Corrections denote the number of times the model self-corrects itself, measured by occurrences of the `Wait` token. Gen. Length denotes the average generation length per query. We see that while the base model has a relatively high number of self-corrections, finetuning reduces this. Context-free augmentations help retain self-correction, performing better than standard finetuning in Table 5.

science reasoning. See Supp C for other evaluation details.

(C) We finetune R1-Distill-Llama-8B on the MedReason dataset Wu et al. (2025) following the setup of Wu et al. (2025). We evaluate the model's medical abilities by MedQA Jin et al. (2021), MedBulletsOp4 (MBOp4), MedBulletsOp5 (MBOp5) Wu et al. (2025). MedReason dataset has 32K samples and is constructed by converting medical question-answer pairs into reasoning steps grounded in medical knowledge-graph. This is done by structured prompting of state-of-the-art LLMs[3].

Results We present our main results in Table 5. `CFS` forgets less and learns medical tasks better than the relevant baselines. Note forgetting baselines like `Wise-FT` and our method are better than standard finetuning on MedReason. We argue that this is due to fact that evaluation benchmarks like MedQA require *self-correction* ability to get good accuracy. Standard finetuning leads to a decrease in model's self-correction abilities, while improving it's medical knowledge. Finetuning while attenuating forgetting helps the model reason with the learned knowledge. We measure different model's average generations (i.e., response) length and number of self-corrections on our medical benchmarks in Table 6. We quantify the number of self-corrections as the frequency of occurrence of `Wait` token per response.

We also lack a comparison with ℓ_2 regularization for this setup. This is because ℓ_2 requires loading two models onto the GPU (the initial model and the fine-tuned model) while training. For making training these large models feasible we use DeepSpeed Zero3 parameter partition which we found is not amiable to accessing the target model parameters while training. We defer this and is on our future work.

4 Limitations

While our method is well-motivated, we consider two model settings: the Olmo-1B pretrained-only model (where it preserves its pretrain-model-metrics) and R1-ll-Llama-8B (where it preserves math reasoning). It would be good to see if the method works for a broader set of models and datasets.

References

Dan Biderman, Jacob Portes, Jose Javier Gonzalez Ortiz, Mansheej Paul, Philip Greengard, Connor Jennings, Daniel King, Sam Havens, Vitaliy Chiley, Jonathan Frankle, Cody Blakeney, and John Patrick Cunningham. LoRA learns less and forgets less. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL `https://openreview.net/forum?id=aloEru2qCG`. Featured Certification.

[3] MedReason huggingface link : `https://huggingface.co/datasets/UCSC-VLAA/MedReason`

Yupeng Chen, Senmiao Wang, Zhihang Lin, Zeyu Qin, Yushun Zhang, Tian Ding, and Ruoyu Sun. Mofo: Momentum-filtered optimizer for mitigating forgetting in llm fine-tuning. *arXiv preprint arXiv:2407.20999*, 2024.

Cyprien de Masson D'Autume, Sebastian Ruder, Lingpeng Kong, and Dani Yogatama. Episodic memory in lifelong language learning. *Advances in Neural Information Processing Systems*, 32, 2019.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL `https://arxiv.org/abs/2501.12948`.

Aaron Grattafiori et al. The llama 3 herd of models, 2024. URL `https://arxiv.org/abs/2407.21783`.

Dirk Groeneveld, Iz Beltagy, Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Harsh Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, et al. Olmo: Accelerating the science of language models. *arXiv preprint arXiv:2402.00838*, 2024.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=nZeVKeeFYf9`.

Jianheng Huang, Leyang Cui, Ante Wang, Chengyi Yang, Xinting Liao, Linfeng Song, Junfeng Yao, and Jinsong Su. Mitigating catastrophic forgetting in large language models with self-synthesized rehearsal. *arXiv preprint arXiv:2403.01244*, 2024.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=6t0Kwf8-jrj`.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421, 2021.

Ronald Kemker and Christopher Kanan. Fearnet: Brain-inspired model for incremental learning. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=SJ1Xmf-Rb`.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114 (13):3521–3526, 2017.

Anat Kleiman, Gintare Karolina Dziugaite, Jonathan Frankle, Sham Kakade, and Mansheej Paul. Soup to go: mitigating forgetting during continual learning with model averaging, 2025. URL `https://arxiv.org/abs/2501.05559`.

Saurabh Kumar, Henrik Marklund, and Benjamin Van Roy. Maintaining plasticity in continual learning via regenerative regularization. *arXiv preprint arXiv:2308.11958*, 2023.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Yong Lin, Hangyu Lin, Wei Xiong, Shizhe Diao, Jianmeng Liu, Jipeng Zhang, Rui Pan, Haoxiang Wang, Wenbin Hu, Hanning Zhang, et al. Mitigating the alignment tax of rlhf. *CoRR*, 2023.

Ekdeep Singh Lubana, Puja Trivedi, Danai Koutra, and R. Dick. How do quadratic regularizers prevent catastrophic forgetting: The role of interpolation. *COLLAS*, 2021.

Ashwinee Panda, Berivan Isik, Xiangyu Qi, Sanmi Koyejo, Tsachy Weissman, and Prateek Mittal. Lottery ticket adaptation: Mitigating destructive interference in llms, 2024. *https://arxiv.org/abs/2406.16797*, 2024.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL `https://proceedings.neurips.cc/paper_files/paper/2019/file/fa7cdfad1a5aaf8370ebeda47a1ff1c3-Paper.pdf`.

Sunny Sanyal, Hayden Prairie, Rudrajit Das, Ali Kavis, and Sujay Sanghavi. Upweighting easy samples in fine-tuning mitigates forgetting. *arXiv preprint arXiv:2502.02797*, 2025.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

James Smith, Yen-Chang Hsu, Jonathan Balloch, Yilin Shen, Hongxia Jin, and Zsolt Kira. Always be dreaming: A new approach for data-free class-incremental learning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9374–9384, 2021.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, Valentin Hofmann, Ananya Jha, Sachin Kumar, Li Lucy, Xinxi Lyu, Nathan Lambert, Ian Magnusson, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew Peters, Abhilasha Ravichander, Kyle Richardson, Zejiang Shen, Emma Strubell, Nishant Subramani, Oyvind Tafjord, Evan Walsh, Luke Zettlemoyer, Noah Smith, Hannaneh Hajishirzi, Iz Beltagy, Dirk Groeneveld, Jesse Dodge, and Kyle Lo. Dolma: an open corpus of three trillion tokens for language model pretraining research. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15725–15788, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.840. URL `https://aclanthology.org/2024.acl-long.840/`.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo-Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, and Ludwig Schmidt. Robust fine-tuning of zero-shot models. *arXiv preprint arXiv:2109.01903*, 2021. `https://arxiv.org/abs/2109.01903`.

Juncheng Wu, Wenlong Deng, Xingxuan Li, Sheng Liu, Taomian Mi, Yifan Peng, Ziyang Xu, Yi Liu, Hyunjin Cho, Chang-In Choi, et al. Medreason: Eliciting factual medical reasoning steps in llms via knowledge graphs. *arXiv preprint arXiv:2504.00993*, 2025.

Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, Zhengyou Zhang, and Yun Raymond Fu. Incremental classifier learning with generative adversarial networks. *ArXiv*, abs/1802.00853, 2018. URL `https://api.semanticscholar.org/CorpusID:3652214`.

Zitong Yang, Aonan Zhang, Sam Wiseman, Xiang Kong, Ke Ye, and Dong Yin. Memory retaining finetuning via distillation.

Hongxu Yin, Pavlo Molchanov, Jose M Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via deepinversion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8715–8724, 2020.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*, 2023.

Weizhe Yuan, Jane Yu, Song Jiang, Karthik Padthe, Yang Li, Dong Wang, Ilia Kulikov, Kyunghyun Cho, Yuandong Tian, Jason E Weston, et al. Naturalreasoning: Reasoning in the wild with 2.8 m challenging questions. *arXiv preprint arXiv:2502.13124*, 2025.

Junhao Zheng, Shengjie Qiu, Chengming Shi, and Qianli Ma. Towards lifelong learning of large language models: A survey. *ACM Comput. Surv.*, 57(8), March 2025. ISSN 0360-0300. doi: 10.1145/3716629. URL `https://doi.org/10.1145/3716629`.

A Compute details

Each training run is done on a single GH200 GPU of size 96GB. Each individual run in our paper takes around 8 hours for training and about 1-2 hours for evaluation.

B Training details

We use the 8bit AdamW (from bitsandbytes). We finetune Olmo on MetaMathQA for 2 epochs. R1-Distill-Llama-8B is finetuned on Medreason for 4 corresponding to approx. 1000 gradient steps. We limit the training sequence to be of length 1024 for R1-Distill-Llama-8B and of length 512 for Olmo.

For MedReason we format the reasoning traces inside a `<think>...<\think>` block, which is preceded by the question in MedReason and followed by the answer. This follows R1-Distill's general reasoning data format and hence we use this. We don't use a system prompt either for training or evaluation.

For MetaMathQA we format question as "Question : {question}, Answer : {answer}".

C Evaluation details

Olmo evaluation details :

- We use LMEval[4] for evaluating pretraining abilities.
- **Commonsense reasoning datasets** : Following Groeneveld et al. (2024) we average the performance on the following datasets for commonsense reasoning : 1. ARC-challenge, 2. ARC-easy, 3. Boolq, 4. Hellaswag, 5. Openbookqa, 6. Piqa, 7. Siqa, 9. Winogrande
- We borrow the setup from et al. (2024) and report numbers considered in their "general language tasks" table for pretrained models. Since Olmo is not pretrained on math and code we don't report those numbers and we found reading comprehension numbers for the models to be unreliable.
- We here report the exact LMEval metrics we evaluate : For GSM8K we evaluate it's standard 5 shot performance i.e., we report `gsm8k/exact_match,strict-match` , for AGIEval we consider only the english subset i.e. `agieval_en/acc`. For BigBenchHard we use `bbh/exact_match,get-answer` and for MMLU we use `mmlu/acc`.

R1-Distill-Llama-8B evaluation details :

- We uses two codebase for our evaluation : SkyThoughts[5] for reasoning tasks i.e., AIME24, MATH500, LiveCodeBench and GPQA-Diamond. We use MedReason[6] for medical benchmarking.
- We use the standard sampling params (from DeepSeek-AI (2025)) for evaluating reasoning performance : temperature = 0.6, and probability threshold of 0.95. We generate for max length of 32768. We use vLLM for generating responses and inherit the reproducibility issues in it's generation. We also append a `<think>` token to the query whenever we are generating a response.

[4] `https://github.com/EleutherAI/lm-evaluation-harness`

[5] `https://github.com/NovaSky-AI/SkyThought`

[6] `https://github.com/UCSC-VLAA/MedReason`

- For evaluating medical benchmarks we use the specified hyperparameters in MedReason, i.e. max generation length of 2048 tokens, and temperature of 0.6 and probability threshold of 0.95.
- For AIME24 we sample 4 different solutions for each prompt making the effective test set size equal to 120. For other datasets we only use only one sample per prompt MATH, GPQA-Diamond.

Dataset	Effective Size
AIME	120
MATH500	500
GPQA-Diamond	198
LiveCodeBench v2	511
MedQA	1273
MedBulletsOp4	308
MedBulletsOp5	308

Table 7: Evaluation dataset sizes

- Our reported numbers are worse than Deepseek numbers : SkyThoughts uses regex parsing for verifying model answers and hence is under-reports the model's accuracy. Gold standard evaluation uses light models OpenAI-o1-mini to comparing the answers, which we omit for computational and financial considerations.

	Pretraining Abilities					
	Commonsense	**MMLU**	**BB Hard**	**AGIEval**	**Avg.**	**GSM8K**
rank=64	47.44	23.68	21.07	18.46	27.66	13.04
rank=128	47.83	24.33	21.33	18.59	28.02	17.66
rank=256	48.41	23.71	21.87	18.72	28.18	20.17
rank=64+CFS*	48.94	24.45	22.49	18.28	28.54	8.42
rank=128+CFS*	48.91	23.76	23.59	18.22	28.62	9.63
rank=256+CFS*	49.03	24.16	23.68	18.61	28.87	11.83

Table 8: LoRA hyperparameter sweep for Olmo. We also include numbers of combining CFS with LoRA.

	Medical				**Reasoning**			
	MedQA	**MBOP4**	**MBOP5**	**Avg.**	**AIME**	**MATH**	**LCB**	**GPQA-D**
$r = 256$	54.20	44.81	41.88	46.96	34.17	82.00	40.86	39.39
$r = 512$	52.47	43.51	38.64	44.87	32.50	82.60	44.96	43.94

Table 9: LoRA ranks tried for R1-Distill-Llama-8B. Since LoRA proves to be less expressive for our setup, we only experiment with high values of r.

D Finetuning dataset details

MedReason is a medical reasoning dataset designed for explainable medical problem-solving in large language models (LLMs). It utilizes a structured medical knowledge graph (KG) to convert clinical QA pairs into logical chains of reasoning, which trace connections from question elements to answers via relevant KG entities. Each path is validated for consistency with clinical logic and evidence-based medicine. They consider medical questions from 7 medical datasets, resulting in a dataset of 32,682 question-answer pairs.

	Pretraining Abilities					
	Commonsense	MMLU	BB Hard	AGIEval	Avg.	GSM8K
$1e-1$	47.81	23.41	20.87	17.86	27.49	6.90
$1e-2$	47.64	23.49	21.59	18.41	27.78	18.57
$1e-3$	47.43	23.03	18.92	19.76	27.28	28.43

Table 10: ℓ_2 regularization penalty sweep for training Olmo on MetaMathQA.

MetaMathQA is a large mathematical problem solving dataset. Given a meta-question, a question in train set of GSM8K, it generates a series of variants of the question. Specifically, they perform three types of question bootstrapping. They also performa answer augmentation, leading to the 400K sample MetaMathQA dataset. MetaMathQA focusing on elementary mathematical problem-solving

E Complete tables

E.1 LoRA Complete Table

See Table 8 for LoRA rank sweep for Olmo and Table 9 for LoRA rank sweep for R1-Distill-Llama-8B. Across both tables we see that LoRA is not able to perform at par with finetuning on downstream evaluation. We keep the the alpha parameter in LoRA equal to it's rank.

E.2 L2 Complete Table

See Table 10 for ℓ_2 hyperparameter sweep for Olmo. $1e-1$ proves to be too strong of a regularization, we take $1e-3$ as having the best of both worlds.

E.3 ModelAvg Complete Table

See Table 11 for Model averaging results for pretrained models and Table 12 for model averaging for reasoning models. For computational reasons, we evaluate less number of averaging factors for reasoning models, but the results are informative nonetheless. For Table 12 We can see that as we go from a high value of α to a lower value, i.e. from the base model to finetuned model, we see a U-shape in the downstream performance, instead of the expected straight line we see in Table 11. We believe this is due to reasons discussed in Sec 3.2, specifically, having some base model capabilities like self-reflections and correction help the model be more accurate on the downstream evaluation.

	Pretraining Abilities					
	Commonsense	MMLU	BB Hard	AGIEval	Avg.	GSM8K
$\alpha = 0.1$	41.63	23.07	14.27	19.18	24.54	28.73
$\alpha = 0.2$	42.55	23.10	15.93	19.29	25.22	28.58
$\alpha = 0.3$	43.26	23.12	17.32	19.24	25.73	25.70
$\alpha = 0.4$	44.37	23.06	19.15	19.13	26.43	20.39
$\alpha = 0.5$	45.53	23.10	19.64	18.85	26.78	14.56
$\alpha = 0.6$	46.62	23.19	21.24	18.33	27.34	8.04
$\alpha = 0.7$	47.66	23.90	23.36	17.81	28.18	3.34
$\alpha = 0.8$	48.62	24.36	24.37	17.37	28.68	2.12
$\alpha = 0.9$	49.54	24.40	25.14	17.21	29.07	2.20

Table 11: Model Averaging results for Olmo finetuned on MetaMathQA.

	Medical				**Reasoning**			
	MedQA	**MBOP4**	**MBOP5**	**Avg.**	**AIME**	**MATH**	**LCB**	**GPQA-D**
$\alpha = 0.1$	58.99	49.68	46.43	51.70	40.00	80.20	46.22	39.39
$\alpha = 0.3$	58.60	52.27	47.08	52.65	49.17	81.40	46.05	43.94
$\alpha = 0.5$	59.78	51.95	43.83	51.85	40.00	83.20	44.66	43.94
$\alpha = 0.7$	55.22	52.27	39.61	49.04	45.83	86.00	48.20	45.96
$\alpha = 0.9$	52.55	45.45	37.66	45.22	48.33	85.40	47.99	50.00

Table 12: Model Averaging results for R1-Distill trained on MedReason dataset.