On Efficient Distillation from LLMs to SLMs

Anonymous Author(s)
Affiliation
Address
email

Abstract

Finetuning small language models (SLMs) on data generated by large language models (LLMs), a form of knowledge distillation, has recently been demonstrated to lead to significantly enhanced capabilities of small models across various domains (e.g., mathematical reasoning). However, current approaches typically require synthesizing a large number of new examples ($>$ 100K), which increases the resources and training time needed for finetuning. To address this issue, we investigate principles for making the distillation process more efficient by reducing the amount of synthetic data required. Specifically, we explore *(i)* incorporating SLM's feedback into the LLM's data generation process and *(ii)* including LLM's rationales (i.e., step-by-step solutions) in the distilled data. In our experiments using the Mistral7B model as the SLM on math reasoning tasks (GSM8K, MATH), we find that both feedback and rationales can help make finetuning with distillation more efficient (by requiring up to $\sim$ 2x less synthetic data).

1 Introduction

Large language models (LLMs) have revolutionized the field of machine learning by achieving impressive performance across various domains and demonstrating a remarkable few-shot ability to adapt to new tasks [3]. However, their ever-increasing size can pose significant challenges when deploying such models in practice or can entirely prevent their use in settings with constrained resources or low-latency requirements (e.g., on-device) [2, 25]. Even when hardware constraints are not an issue, concerns over the rising carbon footprint of language models motivate efforts to improve efficiency [11]. In response, there has recently been growing interest in the so-called small language models (SLMs) [1, 16, 9, 17]. Due to their smaller size ($<$ 10B parameters), SLMs are easier to deploy and have more efficient inference. Unsurprisingly, though, their performance often falls short when compared to LLMs, particularly in more specialized domains (e.g., mathematical reasoning, coding) [19, 20].

To close the performance gap between SLMs and LLMs on a particular task, a popular approach is to perform supervised finetuning (SFT) using a domain dataset. However, since the labeled domain dataset is usually limited in size, the performance of the fine-tuned SLM often remains unsatisfactory. This has motivated the development of knowledge distillation approaches, where the original dataset is extended with synthetic data generated by an LLM [21, 18, 10, 23, 13]. Successful examples include the TinyGSM [14] and OrcaMath [17] models, where finetuning with additional distilled data played a key role in achieving state-of-the-art performance in mathematical reasoning using SLMs.

Despite its effectiveness, most current distillation approaches require generating a large number of synthetic examples—often in the order of hundreds of thousands [17, 13] or even millions [14]—which introduces significant overhead. In addition to longer training times, one main concern is the cost associated with generating such large datasets. For instance, synthesising $\sim$12 million

Submitted to Workshop on Fine-Tuning in Machine Learning (FITML@NeurIPS 2024). Do not distribute.

examples used in TinyGSM would cost around $5,000 using current GPT-4 rates.[1] As the demand for more specialized models grows, with each domain requiring its own tailored synthetic dataset, these costs can rapidly accumulate, making such distillation approaches a less attractive option.

In this work, we aim to improve the efficiency of finetuning with distillation by investigating whether the amount of synthetic data required can be reduced. Note that reducing the size of synthetic data offers a double benefit: it lowers the costs and time associated with generating new samples, and it also shortens the costs and time needed for finetuning the SLM. To this end, we first explore incorporating a form of SLM *feedback* [10] into the LLM's data generation process, by oversampling new examples based on areas where the SLM currently struggles. In the context of the OrcaMath [17] model (i.e., Mistral7B [9] model on the GSM8K [4] dataset), we demonstrate that feedback can indeed significantly reduce the amount of synthetic data needed—by up to a factor of 2. Additionally, we examine the impact of including LLM *rationales* (step-by-step solutions) when constructing the synthetic dataset. Similar to [8], we observe that LLM rationales can significantly improve the data efficiency of the distillation process, particularly for smaller dataset sizes.

2 Background

Data We denote the original domain dataset as $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, where each x_i represents the input (e.g., a question) and y_i is the corresponding ground truth output (e.g., an answer), both in natural language format. Each output $y_i = (r_i, a_i)$ consists of a rationale/explanation r_i and the final answer a_i.

Finetuning with Distillation One way to improve the SLM's performance on the particular domain is through supervised finetuning (SFT) using the next-token prediction loss on the output tokens. To increase the amount of data available for finetuning, some current approaches first use the domain dataset $\mathcal{D}$ as seed examples for the LLM to generate more similar examples, resulting in a synthetic dataset $\mathcal{D}_{syn} = \{(\hat{x}_i, \hat{y}_i)\}_{i=1}^M$ with $M \gg N$ The pretrained SLM is then finetuned on $\mathcal{D}_{syn}$, which is a form of knowledge distillation [7, 21]. Importantly, while such distillation requires access to the SLM's weights, it only requires black-box access to the LLM via a prompt-based interface.

3 Improving Distillation Efficiency for Finetuning

We explore two principles for making current distillation approaches more efficient by reducing the amount of synthetic data required for SLM finetuning. First, in Section 3.1, we describe how to leverage the SLM's current state as feedback to make the LLM's generation of new examples more sample-efficient. Second, in Section 3.2, we demonstrate that using LLM-generated rationales (i.e., step-by-step solutions) can expedite learning compared to using the rationales provided by the original dataset.

3.1 SLM's feedback

Recent work on distillation for finetuning has explored methods that take into account the SLM's current capabilities and weaknesses when generating new data with an LLM [12, 10, 22, 15]. Although these approaches differ in their exact implementation details—such as generation prompts, feedback types, and the number of distillation iterations—they share a common idea: rather than generating $\mathcal{D}_{syn}$ directly from the original $\mathcal{D}$, they first evaluate the SLM on $\mathcal{D}$ to identify areas of the data space where the SLM performs well and where it struggles. Specifically, they split the original dataset $\mathcal{D}$ into samples where the SLM's predictions are poor, denoted as $\mathcal{D}_{hard}$, and samples where the SLM is already yielding good outputs, denoted as $\mathcal{D}_{easy}$. They then generate more examples based on the challenging examples in $\mathcal{D}_{hard}$. The intuition behind this approach is that new samples based on examples in $\mathcal{D}_{hard}$ will be more informative for the SLM, whereas generating additional samples based on examples in $\mathcal{D}_{easy}$ might be redundant and unnecessary.

Lion Distillation As a concrete implementation of distillation with feedback, we adopt the Lion framework [10]. In this approach, the original dataset is first split by collecting predictions from

[1] Source: `https://openai.com/api/pricing/`

both the SLM and the LLM, denoted as $\hat{y}_{SLM}$ and $\hat{y}_{LLM}$, respectively. The LLM is then used again (with a different prompt) to assign a numerical score $s \in \{1, \ldots, 10\}$ to both predictions, based on their correctness. The hard examples are defined as those where the LLM's prediction is (significantly) better than the SLM's: $\mathcal{D}_{hard} := \{(x_i, y_i) \mid s(\hat{y}_{LLM}(x_i), y_i) - s(\hat{y}_{SLM}(x_i), y_i) > \tau\}$, with τ representing the discrimination threshold. The remaining examples are treated as "easy," i.e., $\mathcal{D}_{easy} = \mathcal{D} \setminus \mathcal{D}_{hard}$.

In our experiments, we use the default prompts from Lion [10] for both scoring and generating new samples, and we keep the default generation parameters unchanged (the threshold is set to $\tau = 1$). Unlike Lion, we solely use the examples in $\mathcal{D}_{hard}$ (instead of using a 1:1 ratio between $\mathcal{D}_{hard}$ and $\mathcal{D}_{easy}$) and we perform a single distillation round, as we observed that this yields satisfactory results for the purposes of our study.

Experimental Results For our main experiment, we follow the same setting as in OrcaMath [17]. In OrcaMath, the target domain is mathematical reasoning, concretely the popular benchmark GSM8K [4], which consists of grade-school math problems. Mistral7B model [9] is used as the SLM. The original domain dataset $\mathcal{D}$ consists of the train split of the GSM8K data and some other publicly available mathematical datasets. Using GPT-4 Turbo, additional synthetic examples are generated, resulting in a final dataset size of 200K. Note that no SLM feedback was used in constructing the OrcaMath dataset. To assess the impact of incorporating SLM feedback, we construct a new dataset using the aforementioned Lion distillation with feedback. In creating the Lion dataset, we use the train split of GSM8K ($\sim$7.5K) as the seed dataset and generate data with GPT-4 Turbo to ensure a fair comparison with OrcaMath. We perform QLoRA finetuning [5] for 2 epochs (see Appendix B for further experimental details).

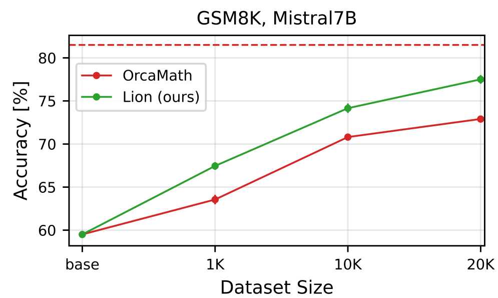

Figure 1: Mistral7B performance (in terms of % of correctly solved test problems) after finetuning on OrcaMath [17] and our Lion-constructed [10] datasets. We observe that Lion data outperforms OrcaMath across all sizes considered here, indicating that incorporating SLM's feedback can help with making distillation more efficient. With a red dashed line (- -) we show the performance of the OrcaMath model [17] that was finetuned on the dataset of size 200K. We report average performance with one standard deviation based on 3 independent runs.

The results are displayed in Figure 1 where we compare the Mistral7B performance after finetuning on different sizes of OrcaMath and Lion datasets. We find that Mistral7B improves much faster when finetuned on the Lion dataset compared to the OrcaMath dataset. For example, finetuning on 10K Lion data points yields better performance ($\sim$74%) than finetuning on 20K OrcaMath datapoints ($\sim$73%), indicating that Lion can reduce the number of synthetic data points needed by more than a factor of 2. Moreover, in OrcaMath, they report 81.5% accuracy after training on the entire 200K dataset, while Lion reaches $\sim$78% after only 20K data points, meaning it recovers $\sim$80% of OrcaMath's performance gains while requiring 10x less data. These results demonstrate that incorporating the SLM's feedback is an effective mechanism for reducing the number of synthetic samples needed, thereby making the entire distillation pipeline more efficient.

3.2 LLM's rationales

We next provide some insights into the importance of including LLM-generated rationales in distilled data (see Figure 3 for concrete examples of answers with rationales).

Experimental Results We again use Mistral7B as the SLM and distil data from GPT-4 Turbo. In addition to the GSM8K dataset [4], we report results on the more challenging MATH dataset [6], which we further split into three subsets by difficulty level (1 being the easiest and 5 the hardest) to better understand how rationales impact domains of varying difficulty. To study the effect of LLM-generated rationales, we report finetuning results for the following datasets: **RR**, consisting of **r**eal questions and **r**eal answers (with rationales), which corresponds to a subset of the original data $\mathcal{D}$; **RS**, consisting of **r**eal questions and LLM-generated **s**ynthetic answers/rationales; and

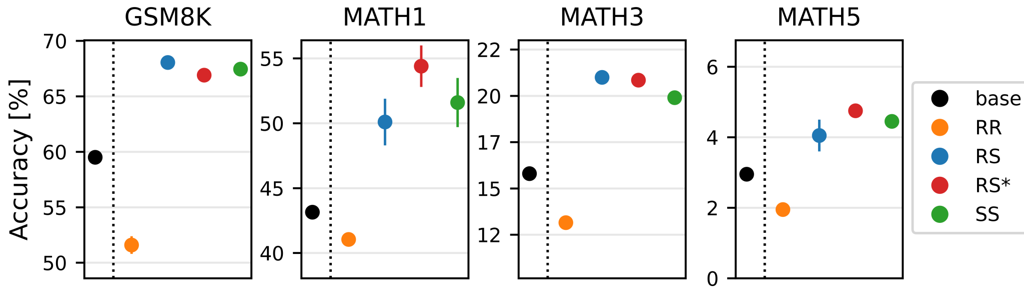

Figure 2: Performance of the finetuned Mistral7B model across datasets with different types of rationales (see Section 3.2 for details on dataset construction). The main observation is that LLM-generated rationales (● ; **RS**) result in better performance compared to using the original ones (● ; **RR**). We report average performance with one standard deviation based on 3 independent runs.

SS, consisting of both **s**ynthetic questions and **s**ynthetic answers/rationales. Since our focus is on generating less data, we set the size of all datasets to $N = 1000$ (further experimental details are provided in Appendix B).

The results are shown in Figure 2. We first observe that, surprisingly, finetuning on the original data (**RR**) results in worse performance compared to the base pretrained model across all four domains considered.[2] We posit that this is likely due to the short length of the rationales in the original dataset. As shown in Table 1, the average length of responses from the model finetuned on the **RR** dataset roughly corresponds to the length of the answers in the original data and is shorter than the average length of the pretrained model's responses. This suggests that supervised finetuning, with its next-token prediction loss, is highly sensitive to the reasoning style and format of the data, at least in the small-data regime considered here. Consequently, exposure to shorter answers in **RR** may cause the Mistral7B model to 'unlearn' some of its default chain-of-thought behavior [20].

This hypothesis is further supported by the considerable performance improvements observed when finetuning on the **RS** dataset, which consists of the same real questions but includes longer, LLM-generated answers, even when finetuning on just $N = 1000$ samples. These findings align with previous work showing that including LLM-generated rationales r_i can make finetuning more data-efficient compared to training on final answers a_i alone [8]. The outperformance of using synthetic answers (**RS**) compared to using real ones (**RR**) is additionally surprising because no quality control was applied to the synthetic answers. Thus, it is likely that some synthetic answers in **RS** are incorrect, especially in harder domains like MATH5, where the LLM performance is below 50% (see Table 2).

To further investigate this, we constructed the **RS*** dataset, ensuring that synthetic answers are correct by cross-referencing them with ground-truth answers (a form of rejection sampling [24]) using the GPT-4 Turbo model as the verifier. Interestingly, improving the data quality does not seem to have a significant effect, as evidenced by the similar performance of **RS** and **RS***. Taken together, this suggests that in the small-data regime, the style of the data (i.e., short vs. long answers) might be as important as its correctness (i.e., wrong vs. correct). Lastly, we observe that using synthetic questions (**SS**) yields performance similar to using real ones (**RS**), confirming our findings from Section 3.1 that the LLM is capable of generating sufficiently good questions to facilitate effective finetuning. All in all, our findings suggest that incorporating (longer) LLM rationales can accelerate the training process by improving data efficiency compared to using the original ones, particularly when finetuning on smaller datasets.

4 Conclusion & Future Work

We have demonstrated that the efficiency of distillation for finetuning SLMs can be enhanced by incorporating SLM feedback and LLM-generated rationales. In the future, it would be valuable to confirm the effectiveness of SLM feedback in domains beyond mathematical reasoning (e.g., coding, medical knowledge). It would also be useful to explore reducing supervised finetuning's sensitivity to style and format in small data regimes, and to better understand the limited impact of rejection sampling on synthetic data (see **RS*** results in Section 3.2).

[2] Similar counterintuitive results have been reported previously; see, e.g., Section 4.7 in [23].

References

[1] Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.

[2] Guangji Bai, Zheng Chai, Chen Ling, Shiyu Wang, Jiaying Lu, Nan Zhang, Tingwei Shi, Ziyang Yu, Mengdan Zhu, Yifei Zhang, et al. Beyond efficiency: A systematic survey of resource-efficient large language models. *arXiv preprint arXiv:2401.00625*, 2024.

[3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[4] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[5] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36, 2024.

[6] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

[7] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[8] Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *arXiv preprint arXiv:2305.02301*, 2023.

[9] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[10] Yuxin Jiang, Chunkit Chan, Mingyang Chen, and Wei Wang. Lion: Adversarial distillation of proprietary large language models. *arXiv preprint arXiv:2305.12870*, 2023.

[11] Loïc Lannelongue, Jason Grealey, and Michael Inouye. Green algorithms: quantifying the carbon footprint of computation. *Advanced science*, 8(12):2100707, 2021.

[12] Nicholas Lee, Thanakul Wattanawong, Sehoon Kim, Karttikeya Mangalam, Sheng Shen, Gopala Anumanchipali, Michael W Mahoney, Kurt Keutzer, and Amir Gholami. Llm2llm: Boosting llms with novel iterative data enhancement. *arXiv preprint arXiv:2403.15042*, 2024.

[13] Chen Li, Weiqi Wang, Jingcheng Hu, Yixuan Wei, Nanning Zheng, Han Hu, Zheng Zhang, and Houwen Peng. Common 7b language models already possess strong math capabilities. *arXiv preprint arXiv:2403.04706*, 2024.

[14] Bingbin Liu, Sebastien Bubeck, Ronen Eldan, Janardhan Kulkarni, Yuanzhi Li, Anh Nguyen, Rachel Ward, and Yi Zhang. Tinygsm: achieving> 80% on gsm8k with small language models. *arXiv preprint arXiv:2312.09241*, 2023.

[15] Chengyuan Liu, Yangyang Kang, Fubang Zhao, Kun Kuang, Zhuoren Jiang, Changlong Sun, and Fei Wu. Evolving knowledge distillation with large language models and active learning. *arXiv preprint arXiv:2403.06414*, 2024.

[16] Arindam Mitra, Luciano Del Corro, Shweti Mahajan, Andres Codas, Clarisse Simoes, Sahaj Agarwal, Xuxi Chen, Anastasia Razdaibiedina, Erik Jones, Kriti Aggarwal, et al. Orca 2: Teaching small language models how to reason. *arXiv preprint arXiv:2311.11045*, 2023.

[17] Arindam Mitra, Hamed Khanpour, Corby Rosset, and Ahmed Awadallah. Orca-math: Unlocking the potential of slms in grade school math. *arXiv preprint arXiv:2402.14830*, 2024.

[18] Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. Distilling reasoning capabilities into smaller language models. *arXiv preprint arXiv:2212.00193*, 2022.

[19] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

[20] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[21] Xiaohan Xu, Ming Li, Chongyang Tao, Tao Shen, Reynold Cheng, Jinyang Li, Can Xu, Dacheng Tao, and Tianyi Zhou. A survey on knowledge distillation of large language models. *arXiv preprint arXiv:2402.13116*, 2024.

[22] Jiahao Ying, Mingbao Lin, Yixin Cao, Wei Tang, Bo Wang, Qianru Sun, Xuanjing Huang, and Shuicheng Yan. Llms-as-instructors: Learning from errors toward automating model improvement. *arXiv preprint arXiv:2407.00497*, 2024.

[23] 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.

[24] Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou, and Jingren Zhou. Scaling relationship on learning mathematical reasoning with large language models. *arXiv preprint arXiv:2308.01825*, 2023.

[25] Zixuan Zhou, Xuefei Ning, Ke Hong, Tianyu Fu, Jiaming Xu, Shiyao Li, Yuming Lou, Luning Wang, Zhihang Yuan, Xiuhong Li, et al. A survey on efficient inference for large language models. *arXiv preprint arXiv:2404.14294*, 2024.

Appendix

A Related Work

Finetuning SLMs on synthetic data distilled from LLMs has recently emerged as a successful paradigm for building small yet powerful specialized models [14, 17, 23, 13]. However, reducing the costs associated with such distillation techniques has received less attention to date. In Lion [10], the use of SLM feedback is proposed for more effective distillation, but their focus is on instruction-tuning, whereas we focus on domain adaptation via finetuning. LLM-generated rationales for improved efficiency were proposed in [8], though they study simpler tasks (e.g., text classification) where no ground-truth rationales are available. Most recently, LLM2LLM [12] was introduced for distillation with feedback on small seed datasets, which complements our efforts to minimize the amount of synthetic data required for finetuning.

B Experimental details

We make our code publicly available at `GITHUB REPO`. All the experiments reported in this paper can be performed on a single A100 GPU.

For *supervised finetuning*, we use the parameter-efficient QLORA method [5] for 2 epochs with a batch size of 24. We set the LoRA rank and alpha parameters to 64 and adapt all linear layers. We use Mistral's default tokenizer with a maximum sequence length set to 1024 (except for MATH5, where we use 2048). During finetuning, the data is presented in the following format:

```
Question: {question}\n Answer: {answer}
```

and only the answer tokens are counted towards the loss.

For *synthetic data generation* in Section 3.1, we closely follow the original Lion [10] implementation available at `https://github.com/YJiangcm/Lion`. The main difference is that we perform a single distillation step and make use only of hard examples when finetuning (i.e., the sampling ratio between $\mathcal{D}_{hard}$ and $\mathcal{D}_{easy}$ is $1:0$). Further tuning of these generation parameters is left for future work. For the experiment in Section 3.2, we also use Lion framewrok when constructing the fully synthetic dataset (**SS**), whereas for datasets with only synthetic answers (**RS**, **RS***), we first sample real questions from the original datasets $\mathcal{D}$ and then generate answers using GPT-4 Turbo with the following prompt:

```
Here is a question that describes a math problem. Write a response that
appropriately and accurately solves the math problem.\n Question:{question}
```

For *inference* with the SLM, we consider a 0-shot scenario and do greedy-decoding using the following prompt:

```
Question: {question}\n Answer:
```

For *evaluation*, we use the *GPT4-based-Exact-Match* metric proposed in OrcaMath (see Section 4 in [17]).

C Additional Figures

```
Question:
Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins
for her friends every day with four. She sells the remainder at the farmers' market daily for
$2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?

Ground-truth rationale/answer:
Janet sells 16 - 3 - 4 = <<16-3-4=9>>9 duck eggs a day. She makes 9 * 2 = $<<9*2=18>>18
every day at the farmer's market. #### 18

LLM rationale/answer:
Let's calculate how many eggs Janet has after eating breakfast and baking muffins:

Every day, Janet's ducks lay 16 eggs.
She eats 3 eggs for breakfast.
She uses 4 eggs to bake muffins.

Total eggs used by Janet = 3 (for breakfast) + 4 (for muffins) = 7 eggs.

Now, subtract the used eggs from the total eggs laid to find out how many she has left to sell at
the farmers' market:

Eggs remaining to sell = Total eggs laid - Eggs used by Janet
Eggs remaining to sell = 16 - 7 = 9 eggs.

Since Janet sells the remaining eggs at $2 per egg, we can calculate her daily earnings by
multiplying the number of eggs she has left by the price per egg:

Daily earnings = Number of eggs to sell * Price per egg
Daily earnings = 9 * $2
Daily earnings = $18.

So, Janet makes $18 every day at the farmers' market from selling her duck eggs.
```

Figure 3: An example of a question from GSM8K dataset [4] with the original rationale/explanation and the one based on the LLM (GPT4-Turbo).

Table 1: Lengths (in terms of average number of tokens) of the original rationales/explanations in $\mathcal{D}$, as well as those generated by the LLM (see Appendix B for the exact prompts used to generate synthetic answers). We also display the lengths of the SLM's responses after finetuning on various datasets. Notably, finetuning on original rationales (**RR**) results in shorter SLM outputs compared to those based on the base pretrained model, which may explain the performance drop after finetuning on the original rationales, as reported in Section 3.2.

	Rationales		**Model output**		
	Ground-truth (R)	**LLM (S)**	**base**	**SFT on RR**	**SFT on RS**
GSM8K	117	270	191	113	252
MATH1	78	312	165	64	255
MATH3	139	464	314	115	454
MATH5	287	718	450	287	741

Table 2: Performance (in terms of % of correctly solved test problems) based on *GPT4-based-Exact-Match* evaluation [17] of GPT4-Turbo model on GSM8K [4] and MATH [6] datasets. We further stratify MATH dataset based on the difficulty level.

	GSM8K	**MATH1**	**MATH3**	**MATH5**
Accuracy [%]	95.1	92.9	72.3	32.9