
Efficient Post-Training for Industry-Specialized Reasoning in Small Language Models

Bill Cai Sheldon Liu Tatsuo Azeyanagi Tomal Deb
Amazon Web Services
{billcaiy,shilong,tazeyana,tomalde}@amazon.com

Abstract

Large reasoning models (LRMs) excel at reasoning tasks but face deployment barriers due to computational constraints, regulatory requirements, and domain-specific knowledge gaps. This work addresses these limitations by developing cost-efficient post-training methods to enhance reasoning capabilities. Using Qwen3-4B as our base model, we investigate variations of efficient Supervised Fine-Tuning (SFT) and Direct Preference Optimization (DPO). For this purpose, we construct a comprehensive financial reasoning dataset with diverse trace qualities from FinQA, enabling systematic analysis of how training data characteristics affect model performance under tight computational budgets. Our experiments demonstrate that reasoning data augmentation, combined with efficient training algorithms can achieve a high accuracy of 78.64%, surpassing larger LRMs such as DeepSeek-R1 as well as previously published results on FinQA, despite avoiding GRPO and other costly online RL methods. The work contributes both a multi-trace reasoning dataset for rapid experimentation and empirical insights into optimizing reasoning performance within resource constraints, providing a reusable framework for customization of smaller language models for domain-specific applications.

1 Introduction

Large Language Models (LLMs) have transformed the machine learning field, achieving exceptional performance on diverse NLP tasks. However, general-purpose LLMs often fail to meet industrial requirements. In specialized domains, industry-specific terminologies make it difficult for general LLMs to understand domain documents. Highly regulated industries like finance often cannot use external model APIs, requiring on-premise deployment with computational limitations. To address these challenges, Small Language Models (SLMs) with specialized domain knowledge are being developed using continued pre-training and post-training. Another recent trend in LLMs is the development of LRMs, enhancing LLM performance especially in math and programming tasks. Past work have demonstrated that LRMs achieve higher performance through test-time scaling [1], while advances in reinforcement learning (RL) have improved LRM performance through online RL methods such as Proximal Policy Optimization (PPO) [2] and Group Relative Policy Optimization (GRPO) [3].

This paper investigates post-training methods to enhance industry-specialized reasoning skills using the financial domain as a case study. Using Qwen3-4B [4] as our base model, we focus on cost-efficient approaches: supervised fine-tuning (SFT) and Direct Preference Optimization (DPO) [5]. We construct a dataset with diverse reasoning traces augmented on the FinQA dataset [6], using post-processing techniques, and analyze how SFT and DPO on different responses affects post-trained model performance.

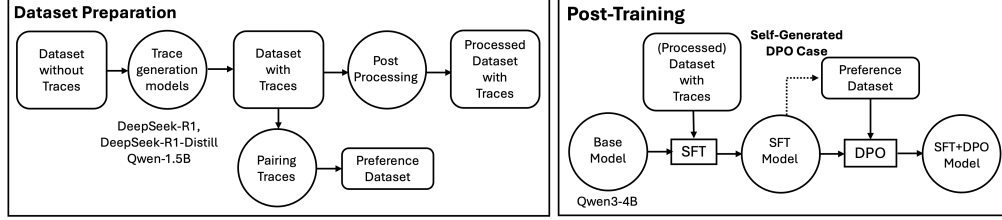


Figure 1: Overview of our approach. **Left:** Dataset preparation including reasoning trace generation with LRM, post-processing, and trace pairing. **Right:** Post-training with SFT and DPO. (Preference dataset for DPO is generated either through the data preparation process on the left or by SFT model).

Approach overview: Figure 1 illustrates a two-stage approach for training language models with reasoning traces, where synthetic reasoning traces are first generated using LRMs and a post-training phase is employed to combine SFT on the trace dataset with DPO using self-generated cases, ultimately producing an enhanced SFT+DPO model from a Qwen3-4B base model.

1.1 Key contributions

Efficient Domain-Specialized Reasoning Framework: We demonstrate that even 4B parameter models can achieve competitive reasoning performance through cost-efficient post-training, offering a practical alternative to computationally expensive LRMs for industry deployment under tight resource budgets. Our approach combines SFT and self-generated DPO with LoRA to achieve model performance on FinQA-test exceeding large LRMs like DeepSeek-R1. Despite avoiding online RL methods like GRPO that suffer from inefficiencies due to switching between training and sample generation, our method surpasses performance of specific finance-domain models trained with SFT and GRPO [7, 8] on FinQA.

Systematic Analysis of Training Efficiency Trade-offs: Through comprehensive experiments with diverse reasoning traces and filtering strategies, we provide empirical insights into optimizing reasoning capabilities within realistic computational constraints, contributing both a reusable framework and multi-trace financial reasoning dataset for rapid experimentation in resource-limited environments. Our reasoning trace dataset used for SFT and DPO are released publicly at this [link](#).

1.2 Related works

Post-Training for Reasoning: Test-time scaling has shown significant improvements in model performance, with early results from chain-of-thought prompting [9] and non-greedy decoding [10]. SFT on reasoning traces [11] and RL strategies such as GRPO on verifiable domains [3] such as mathematics enabled LRMs to improve performance on downstream tasks. Recent work on efficient reasoning have focused on curating datasets to improve LRM performance [1], and also improving efficiency of online RL methods, through using fewer samples during forward pass [12], one-step difference between GRPO sampling and training [13].

Industry-Specialized RMs: For financial domain post-training, most approaches focus on SFT on reasoning traces and GRPO via LLM-as-judge rewards [7, 8]. Waiting for LLM-as-judge for reward verification would decrease GPU utilization, which is already low in an online RL context. The Fino1 paper [8] also looks at DPO, but only utilizes responses from a separate model (GPT-4o) in the DPO training, instead of responses from the policy model itself.

2 Experiment

2.1 Dataset

To post-train LLMs for reasoning for domain-specific tasks, we generate reasoning traces using DeepSeek-R1 (R1) and DeepSeek-R1-Distill-Qwen-1.5B (Distill Qwen 1.5B) on the train subset of FinQA, with 10 responses generated from each model for each training sample. We then use the same LLM-as-judge evaluation strategy employed by Fin-R1 authors to identify correct and incorrect

answers. We also use R1 to generate summarized versions of its own reasoning traces. With all these approaches combined, from 6,251 training samples of FinQA we generate 187,506 samples in total. More details on this dataset are available in Appendix B.

For SFT, we construct datasets using the FinQA-train dataset, the correct R1 responses (with reasoning), correct Distill Qwen 1.5B responses (with reasoning) and post-processed responses. We construct the following SFT datasets: (1) **Answers-only-SFT** is the answers from FinQA-train dataset without any reasoning trace or model responses, (2) **R1-SFT** is the full set of R1 responses that are evaluated correct, (3) **Qwen-SFT** is the full set of Distill Qwen 1.5B responses that are evaluated correct, (4) **R1-short-SFT** is the subset of correct R1 responses where response length is less than 50th percentile of correct R1 trace response length, (5) **R1-summarized-SFT** is the full set of correct R1 responses with reasoning trace summarized further by R1.

For DPO, we construct preference datasets by setting up paired model responses of R1 and/or Distill Qwen 1.5B where one response is evaluated as correct, and another is evaluated as wrong for the same example. We constructed: (1) **Both-same-DPO** is the dataset where both paired model responses come from the same source model (either R1 or Distill Qwen 1.5B), (2) **R1-correct-Qwen-wrong-DPO** is the dataset where correct responses are from R1 and wrong responses are from Distill Qwen 1.5B. We use Qwen3-4B after SFT on R1-SFT dataset and evaluate on the FinQA-train subset 20 times per sample, sort training examples in FinQA-train based on the accuracy percentage of Qwen3-4B. (3) We select the least accurate 25% of examples (i.e. hard examples) and filter Both-same-DPO to create **Both-same-hard-DPO**, (4) we sample Qwen3-4B (after SFT on R1-SFT) 20 times per example on the FinQA-train subset, along with the LLM-as-judge evaluation results, and construct a paired preference dataset called **Self-generated-DPO**.

2.2 Experiment setting

For SFT and DPO, we use variations of reasoning traces on FinQA dataset. As shown in Appendix B, we maintain the same hyperparameters during training, including the number of epochs, learning rate schedule, and LoRA hyperparameters. All SFT experiments start training from the original Qwen3-4B model. As stated in Table 2, DPO experiments start either with the Qwen3-4B model or the Qwen3-4B with **R1-SFT** model.

During inference, we utilize the prompt used during SFT, formatted for the chat style of the respective models. We use vLLM [14] and the Qwen3 recommended hyperparameters. For evaluation, we utilize the LLM-as-judge prompt from [7], and use Claude 3.7 Sonnet as the judge model with temperature at 0.1 throughout for consistency. All accuracy results reported are evaluated on the test subset of FinQA, with training done on the train subset of FinQA to prevent data contamination. We also run evaluation of FinQA-test on DeepSeek-R1 and gpt-oss-120b, and report previously published FinQA-test scores for Fin-R1-7B [7], Fino1-8B and Fino1-14B [8] for performance comparison. Full details are in Appendix B.

Table 1: SFT results with Qwen3-4B

SFT Dataset	Accuracy (%)
No SFT (Original)	71.05
Answers-only-SFT	72.71
Qwen-SFT	60.16
R1-SFT	74.98
R1-short-SFT	73.32
R1-summarized-SFT	72.62

2.3 Results and observations

As shown in Table 1, we find that SFT with reasoning traces can significantly boost the performance of the original Qwen3-4B, even when the existing Qwen3-4B is already trained with in-built reasoning. In particular, SFT on correct R1 traces (**R1-SFT**) shows a significant improvement over the original Qwen3-4B model. This surpasses directly running SFT on the answers in the FinQA-train dataset (**Answers-only SFT**).

We find that the original R1 traces filtered simply for LLM-evaluated correctness is the most effective, as compared to other types of reasoning traces. SFT on correct Distill Qwen-1.5B traces (**Qwen-SFT**) resulted in significant decrease in accuracy, suggesting that the inherent quality of reasoning is lacking within Distill Qwen-1.5B traces even when filtered for correct responses. SFT on half the dataset selected for shorter traces (**R1-short-SFT**) does improve performance with half the number of training samples, and achieve more than half of the improvement of training on the full **R1-SFT**

Table 2: Comparison of DPO results and other models

Policy Models	DPO Dataset	Accuracy (%)
Qwen3-4B	Both-same-DPO	73.15
Qwen3-4B w R1-SFT	Both-same-DPO	75.07
Qwen3-4B w R1-SFT	R1-correct-Qwen-wrong-DPO	69.66
Qwen3-4B w R1-SFT	Both-same-hard-DPO	75.50
Qwen3-4B w R1-SFT	Self-generated-DPO	78.64
Other Models (evaluated by us)		Accuracy (%)
DeepSeek-R1 0120		71.84
gpt-oss-120b		75.59
Previously Published Results		Accuracy (%)
Fino1-8B [8]	SFT & GRPO from Qwen3-8B	73.03
Fino1-14B [8]	SFT & GRPO from Qwen3-14B	74.18
Fin-R1-7B [7]	SFT & GRPO from Qwen2.5-7B	76.00

dataset. However, training on summarized R1 traces (**R1-summarized-SFT**) does not achieve similar performance as SFT on the R1-SFT dataset, despite the same number of training samples.

As Table 2 shows, DPO on **R1-correct-Qwen-wrong-DPO** performs poorly. A plausible explanation is that DPO on this dataset results in ineffective learning of the difference between the response quality between R1 and Distill Qwen-1.5B, which does not enhance the capability of an already sufficiently capable Qwen3-4B model. DPO on **Both-same-DPO** using either the original Qwen3-4B or the post-SFT Qwen3-4B model both resulted in accuracy improvements.

Having feedback from the policy model included in the DPO dataset appears to be significant in improving the model accuracy. Filtering for the examples that the policy model performs poorly in the training set results in better performance. This is seen as **Both-same-hard-DPO** has only 15.68% of the number of samples as **Both-same-DPO**, but achieves slightly better evaluation accuracy. Direct DPO on the policy model’s self-generated traces resulted in the largest improvements for DPO training, despite having a similar dataset size as **Both-same-hard-DPO**. This result is significant, as we can achieve higher performance through constructing a DPO dataset cheaply sampled from the policy model, as compared to all sampling generations from much larger models like DeepSeek-R1.

3 Conclusion and summary

This work demonstrates that effective domain-specialized reasoning capabilities can be achieved through cost-efficient post-training approaches. First, we show that supervised fine-tuning with high-quality reasoning traces can yield substantial improvements, with trace quality as the most important factor.

Second, our analysis reveals that self-generated DPO represents a promising direction for efficient reasoning model development. By achieving our best performance (78.64% accuracy) using traces generated from the policy model, this approach offers a practical alternative to computationally intensive online reinforcement learning methods or costly generations from larger models. The intuition behind this DPO performance lift is that the policy model (Qwen3-4B w R1-SFT) is more capable and efficient at understanding its own generations and learning from related pairs than those generated by a different model (e.g. DeepSeek-R1).

Our paper demonstrates that the combination of smaller base models, efficient training algorithms, and self-generated training data presents a viable path for deploying reasoning capabilities in resource-constrained environments. The performance of the final Qwen3-4B model surpasses large LRMs, such as DeepSeek-R1 and gpt-oss-120b, as well as previously published results.

A limitation of this work is our focus on the FinQA dataset. Our future work would explore the generalizability of these findings to other domains and investigate how self-generated trace quality can be further enhanced through iterative refinement that maintain computational efficiency.

References

- [1] Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling, 2025.
- [2] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [3] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.
- [4] An Yang et al. Qwen3 technical report, 2025.
- [5] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 53728–53741. Curran Associates, Inc., 2023.
- [6] Zhiyu Chen et al. FinQA: A dataset of numerical reasoning over financial data. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3697–3711, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.
- [7] Zhaowei Liu, Xin Guo, Fangqi Lou, Lingfeng Zeng, Jinyi Niu, Zixuan Wang, Jiajie Xu, Weige Cai, Ziwei Yang, Xueqian Zhao, et al. Fin-r1: A large language model for financial reasoning through reinforcement learning. *arXiv preprint arXiv:2503.16252*, 2025.
- [8] Lingfei Qian, Weipeng Zhou, Yan Wang, Xueqing Peng, Jimin Huang, and Qianqian Xie. Fino1: On the transferability of reasoning enhanced llms to finance, 2025.
- [9] 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.
- [10] Xuezhi Wang and Denny Zhou. Chain-of-thought reasoning without prompting. *Advances in Neural Information Processing Systems*, 37:66383–66409, 2024.
- [11] Luong Trung, Xinbo Zhang, Zhanming Jie, Peng Sun, Xiaoran Jin, and Hang Li. Reft: Reasoning with reinforced fine-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7601–7614, 2024.
- [12] Zhihang Lin, Mingbao Lin, Yuan Xie, and Rongrong Ji. Cppo: Accelerating the training of group relative policy optimization-based reasoning models. *arXiv preprint arXiv:2503.22342*, 2025.
- [13] Wei Fu, Jiaxuan Gao, Xujie Shen, Chen Zhu, Zhiyu Mei, Chuyi He, Shusheng Xu, Guo Wei, Jun Mei, Jiashu Wang, et al. Areal: A large-scale asynchronous reinforcement learning system for language reasoning. *arXiv preprint arXiv:2505.24298*, 2025.
- [14] 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.
- [15] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shitong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

A Appendix A: Details on prompts used

Appendix A.1: Prompt for reasoning trace generation with DeepSeek-R1 and DeepSeek-R1-Distill-Qwen-1.5B

```
You are a financial analysis assistant. You are given the question in <
question> tags and the context in <context> tags. Answer the question based
on the context given.
```

```
<context>{Data from FinQA}</context>
```

```
<question>{question}</question>
```

```
Reason step by step to answer the question. Be detailed and precise in your
reasoning.
```

Appendix A.2: Prompt for trace summarization with DeepSeek-R1

```
You are given a tuple of question, thinking trace to answer it, and the
answer itself. Summarize the thinking trace by following the guideline below.
```

```
Question: {question}
```

```
Thinking Trace: {thinking_trace}
```

```
Answer: {answer}
```

```
Guideline:
```

1. the summary of the thinking trace must be concise.
2. the summary of the thinking trace is based purely on the given thinking trace.
3. the summary of the thinking trace must be comprehensive
in the sense that it logically leads the given answer to the given question.

```
The output should include the summarized thinking trace only. No extra output
should be included. No exception is accepted.
```

Appendix A.3: SFT prompt

```
Given the financial report context below, answer the question.
```

```
Context:
```

```
{Data from FinQA}
```

```
Question: {question}
```

Appendix A.4: Evaluation prompt for LLM-as-judge (Claude 3.7 Sonnet)

```
You are a scoring assistant for financial questions. I will provide you with
the question in <question>, a <ground truth> and a <model answer>. Please
determine whether the <model answer> has the same meaning as the <ground
truth> according to the following rules. If they are consistent, output "True
", otherwise output "False" using the tool 'evaluate_response'.
```

```
<question> {question} </question>
<ground_truth> {ground_truth} </ground_truth>
<model_answer> {prediction} </model_answer>
```

Rules:

<model_answer> may contain a lot of information besides the final answer, including calculations. Read carefully to extract the answer within the text in <model_answer>.

If the <ground_truth> is a numerical value, and the format of the <model_answer> is different from that of the <ground_truth>, but the numerical values are the same, then it is considered that the meanings are consistent. For example, if the <ground_truth> is 0.98 and the <model_answer> has calculated the result to be 98%, it is considered that the meanings are consistent, return True.

If the <ground_truth> is a numerical value, and the final result of the <model_answer> is consistent with the <ground_truth> after rounding, then it is considered that the meanings are consistent. For example, if the <ground_truth> is 2 and the <model_answer> has calculated the answer to be 1.98, it is considered that the meanings are consistent, return True.

Output Format:

Make the judgment according to the above rules, and finally put the judgment result True or False using the tool 'evaluate_response' under the 'accurate' field.

Appendix A.5 Example reasoning trace and answers

These responses are formatted to the Qwen3 response style where reasoning traces are in <think> tags, preceding the answer.

DeepSeek-R1

```
<think>Okay, let's see. The user is asking for the total operating expenses
in 2018 in millions. The context provided includes a table with data from
2016 to 2018. The table has columns for year, gallons, average price per
gallon, aircraft fuel expense, and percent of total operating expenses.\n
Looking at the 2018 row, the aircraft fuel expense is $9896 million, and the
percent of total operating expenses is 23.6%. So, the question is to find
the total operating expenses based on the fact that $9896 million is 23.6% of
the total.\n\nTo find the total, I need to use the percentage formula. If
23.6% of the total equals $9896 million, then the total would be $9896
divided by 0.236. Let me check that calculation. \n\nSo, 9896 divided by
0.236. Let me compute that. 9896 / 0.236. Let me do the division step by step.
0.236 times 41,000 is 0.236*41000=9,676. That's close to 9,896. The
difference is 9,896 - 9,676 = 220. So, 220 divided by 0.236 is approximately
932. So adding that to 41,000 gives 41,932. Let me verify with a calculator.
9896 divided by 0.236 equals approximately 41,932.20 million. So the total
operating expenses for 2018 would be around $41,932 million. \n\nWait, but
the user might want it in the same units as presented. The fuel expense is in
millions, so the total would also be in millions. So the answer should be
$41,932 million, or $41.9 billion. But the question asks for the total in
millions, so the answer is 41,932 million. Let me make sure there's no other
data in the context that might affect this. The post-table text talks about
risks and other factors, but nothing that changes the calculation here. The
table's percentages are given as part of the data, so this should be
straightforward.</think>\n
The total operating expenses for 2018 can be
```

calculated using the aircraft fuel expense and its percentage of total operating expenses provided in the table. \n\nFor 2018:\n- **Aircraft fuel expense** = \$9,896 million \n- **Percent of total operating expenses** = 23.6% \n\nUsing the formula: \n\[\n\text{Total Operating Expenses} = \frac{\text{Aircraft Fuel Expense}}{\text{Percentage of Total}} = \frac{9,896}{0.236} \approx 41,932.20 \text{ million}\n\]\n\n**Answer:** \n\nThe total operating expenses in 2018 were approximately **\$41,932 million** (or **\$41.9 billion**).

Distill Qwen-1.5B

<think>Okay, so I need to figure out the growth rate in net revenue from 2007 to 2008 for Energy Louisiana LLC. Let me start by looking at the information given in the table and the additional context provided.\n\nLooking at the table, there's a section labeled "amount (in millions)" with two rows: one for 2007 net revenue and another for 2008 net revenue. The 2007 net revenue is \$991.1 million, and the 2008 net revenue is \$959.2 million.\n\nTo find the growth rate, I think I need to calculate the percentage increase from 2007 to 2008. The formula for growth rate is:\n\nGrowth Rate = (2008 Net Revenue - 2007 Net Revenue) / 2007 Net Revenue * 100%\n\nPlugging in the numbers:\n\nGrowth Rate = (\$959.2 - \$991.1) / \$991.1 * 100%\n\nFirst, subtract the two amounts:\n\n\$959.2 - \$991.1 = -\$31.9 million\n\nNow, divide by the 2007 net revenue:\n\n-\$31.9 / \$991.1 = -0.03216\n\nMultiply by 100% to get the percentage:\n\n-0.03216 * 100% = -3.216%\n\nSo the growth rate is approximately -3.22%, which means there was a 3.22% decrease in net revenue from 2007 to 2008.\n\nWait, let me double-check the numbers to make sure I didn't make a mistake. 2007 is higher than 2008, so the decrease is correct. The calculation seems right. Maybe I should also mention that negative growth rate just means a decline rather than an increase.\n\nI think that's it. The net revenue decreased by about 3.22% from 2007 to 2008.</think>\n\nThe growth rate in net revenue for Energy Louisiana LLC from 2007 to 2008 is calculated as follows:\n\n1. **Net Revenue Values:**\n- 2007: \$991.1 million\n- 2008: \$959.2 million\n\n2. **Growth Rate Formula:**\n\[\n\text{Growth Rate} = \left(\frac{\text{2008 Net Revenue} - \text{2007 Net Revenue}}{\text{2007 Net Revenue}} \right) \times 100\%\n\]\n\n3. **Calculation:**\n\[\n\text{Growth Rate} = \left(\frac{959.2 - 991.1}{991.1} \right) \times 100\% = -3.216\%\n\]\n\n4. **Conclusion:**\n\nThe net revenue decreased by approximately **3.22%** from 2007 to 2008.\n\nboxed{-3.22\%}

Qwen3-4B (after R1-SFT and self-generated-DPO)

<think>Okay, let's see. The user is asking for the net change in net revenue during 2015 for Entergy Corporation. I need to look at the financial report context provided. First, there's a table that shows the 2014 and 2015 net revenue figures. The 2014 net revenue is \$5,735 million, and the 2015 net revenue is \$5,829 million. To find the net change, I subtract the 2014 figure from the 2015 figure. So, \$5,829 - \$5,735 equals \$94 million. But wait, the table also lists various components that affect the net revenue. The entries under "other" and the different provisions like waterford 3, MISO deferral, and Louisiana business combination customer credits are all listed as negative numbers. However, the question is about the net change in net revenue, which is directly given by the difference between the two years' net revenue figures. The detailed breakdown in the table might be there for more context, but the net revenue is already calculated for each year. So the answer should be the difference between 2015 and 2014 net revenues. Let me double-check the numbers. 2015 is 5829, 2014 is 5735. The difference is indeed 94 million. The positive number here indicates an increase. The

explanations in the post-table text about the variances are probably for more detailed reasons, but the question is straightforward about the net change. So the answer is a \$94 million increase. </think> The net change in net revenue for Entergy Corporation during 2015 is calculated by comparing the 2015 net revenue to the 2014 net revenue. - **2014 Net Revenue**: \$5,735 million - **2015 Net Revenue**: \$5,829 million **Net Change** = 2015 Net Revenue - 2014 Net Revenue = \$5,829 million - \$5,735 million = **\$94 million increase**. Thus, the net revenue increased by **\$94 million** in 2015. **Answer**: **The net change in net revenue during 2015 was an increase of \$94 million**.

B Appendix B: Details of experiments

Appendix B.1: Dataset information

For post-training, we have created a reasoning trace dataset based on 6251 samples in the training set of FinQA. The total number of the samples in our dataset is 187506. Three different methods were applied for generating reasoning traces: 62501 samples have the reasoning traces generated by DeepSeek-R1, 62504 samples have those by Distill Qwen 1.5B, and the reasoning traces of the rest 65021 samples are the R1 reasoning traces summarized by R1. To generate the traces, the inferences were executed 10 times per sample of FinQA per trace generation method, and some results from the failed inferences were removed. The dataset has the following columns:

- **example_id**: unique ID for each sample in this dataset. (`example_id = 0, 1, \dots, 187505`.)
- **sample_id**: ID for distinguishing up to 10 samples generated from a given FinQA sample by a given trace generation method. (`sample_id = 0, 1, \dots, 9`.)
- **id**: This column is identical to the ID column `id` in the original FinQA dataset. This column can be used to associate a sample in our dataset with FinQA's.
- **question**: question statement prepared by combining the original question with its context information in FinQA.
- **answer**: original answer in FinQA.
- **llm_answer**: answer generated by DeepSeek-R1 or Distill Qwen 1.5B.
- **llm_reasoning**: reasoning trace generated by DeepSeek-R1 or Distill Qwen 1.5B (including the one summarized by DeepSeek R1).
- **is_correct**: whether `llm_answer` is correct or not. (True or False.) Note that FinQA includes 48 samples with the empty string in the answer column. We set to `is_correct = None` for such samples.
- **model**: trace generation model. (`deepseek_r1` or `qwen_1_5b`.)
- **is_summarized**: if the trace is summarized one or not. (True or False.)

Appendix B.2: Dataset details for post-training

Generating 10 answers with reasoning traces by DeepSeek-R1 and DeepSeek-R1-Distill-Qwen-1.5B per training sample led to higher number of samples for subsequent SFT and DPO.

Table 3: Dataset details

Dataset	Type	Number of Samples
Answers-only-SFT	SFT	6251
Qwen-SFT	SFT	23678
R1-SFT	SFT	46647
R1-short-SFT	SFT	23300
R1-summarized-SFT	SFT	46647
Dataset	Type	Number of Preference Pairs
Both-same-DPO	DPO	96191
R1-correct-Qwen-wrong-DPO	DPO	23929
Both-same-hard-DPO	DPO	15088
Self-generated-DPO	DPO	19450

Appendix B.3: Training hyperparameters

Table 4: Training setup

Hyperparameters	Used During	Value
LoRA rank	SFT	16
LoRA α	SFT	32
Sequence length	SFT	8192
Batch size	SFT	8
Learning rate	SFT	$1.0e^{-4}$
LR scheduler	SFT	Cosine
LR warmup ratio	SFT	0.1
Training epochs	SFT	3.0
LoRA rank	DPO	16
LoRA α	DPO	32
Preference β	DPO	0.1
Preference loss	DPO	Sigmoid
Sequence length	DPO	6144
Batch size	DPO	32
Learning rate	DPO	$5.0e^{-6}$
LR scheduler	DPO	Cosine
LR warmup ratio	DPO	0.1
Training epochs	DPO	3.0

Appendix B.4: Inference parameters

For evaluation of accuracy on FinQA-test, we use the same prompt (see SFT-prompt) in Appendix A. The parameters used for inference below are based on the recommended parameters for inference [15, 4].

Table 5: Inference setup for evaluation on FinQA-test

Parameters	Used For	Value
Temperature	DeepSeek-R1	0.6
Max Tokens	DeepSeek-R1	32k
Top P	DeepSeek-R1	0.95
Temperature	gpt-oss-120b	1.0
Max Tokens	gpt-oss-120b	128k
Top P	gpt-oss-120b	1.0
Inference engine	Qwen3-4B variants	vLLM
Temperature	Qwen3-4B variants	0.6
Max Tokens	Qwen3-4B variants	32k
Top P	Qwen3-4B variants	0.95