Find Your Optimal Teacher: Personalized Data Synthesis via Router-Guided Multi-Teacher Distillation

Anonymous ACL submission

Abstract

Training student models on synthetic data generated by strong teacher models is a promising way to distilling the capabilities of teachers. However, recent studies show that stronger models are not always optimal teachers, revealing a mismatch between teacher outputs and student learnability. To address this issue, we propose *PerSyn* (**Per**sonalized data **Syn**thesis), a novel synthesis strategy that operates under a new "Route then Generate" paradigm to create data tailored to each student model, enabling it to learn more effectively. Specifically, *PerSyn* first assigns each prompt to its optimal teacher via a query-level router that jointly considers student learnability and teacher response quality. Each teacher then synthesizes data only for its assigned prompts, making the process more efficient than the conventional "Generate then Select" paradigm, where all teachers must generate parallel responses for the entire prompt set before constructing the final dataset. Extensive experiments across different model families and scales demonstrate that *PerSyn* consistently achieves superior or comparable performance to all baselines in instruct tuning and math reasoning settings. Further analysis verifies the effectiveness of *PerSyn* and offers extra insights to propel future research. Our code is available at `https://anonymous.4open.science/r/PerSyn-8D85`.

1 Introduction

Large Language Models (LLMs) have demonstrated outstanding performance across a wide range of applications (Li et al., 2025b; Yu et al., 2025; Liang et al., 2025; Zhang et al., 2024; Qin et al., 2025; Yang et al., 2024b; Zhao et al., 2024). However, the high computational cost of LLMs hinders their deployment on resource-constrained devices, motivating the development of smaller models that offer similar capabilities at reduced cost. A common strategy to achieve this is distillation (Hinton et al., 2015; Kim et al., 2024a; Wang et al., 2025a), which leverages the synthetic data generated by a strong teacher model to fine-tune a small student model. They assume that stronger teacher will produce higher-quality synthetic data, which in turn enables the student model to learn more effectively.

	Strong	Mix	CAR	PerSyn
Quality	✓	✗	✓	✓
Learnability	✗	✓	✓	✓
Efficiency	✗	✗	✗	✓
Sample Level	✗	✗	✗	✓

Table 1: Compared to existing methods, *PerSyn* can efficiently assigns each prompt to the optimal teacher based on both quality and learnability. "Sample Level" indicates whether each prompt is assigned to the optimal teacher. *Strong* uses the strongest model as teacher, *Mix* combines synthetic data from strong and weak teachers, and *CAR* selects a single teacher balancing quality and compatibility.

Nevertheless, some works (Xu et al., 2025b; Li et al., 2025a) demonstrate that stronger models are not always the optimal teachers for small student models, since their outputs may be overly complex and shift away from the students' distribution. To mitigate this issue, Li et al. (2025a) mixed the synthetic data from strong and weak models (*Mix*). Xu et al. (2025b) designed a Compatibility-Adjusted Reward (*CAR*) metric to select a single teacher model from a pool of teacher models for specific student. Despite these efforts, two critical limitations remain, as shown in Table 1: **1)** These methods are not efficient enough. Specifically, *Strong* refer to using a super-sized LLM for distillation, but often yields sub-optimal performance with high computational cost. *Mix* and *CAR* follow the "Generate then Select" paradigm, which requires parallel teacher responses[1], thereby all can-

[1]In this paper, parallel teacher responses denote the responses generated by all teacher models for a given prompt.

didate teacher models must generate responses for the entire prompt set before constructing the final synthetic dataset. Notably, the cost scales linearly with the teacher model pool size; **2)** These methods also overlook that each prompt within the dataset has its corresponding optimal teacher for synthesizing responses, thereby making the synthetic dataset sub-optimal for student.

To address these limitations, we propose *PerSyn* (**Per**sonalized data **Syn**thesis), a novel synthesis strategy that customizes a synthetic dataset for a specific student model to help it learn more effectively. Specifically, unlike the "Generate then Select" paradigm, our method operates in a more efficient manner, i.e, "Route then Generate", which first assigns each prompt to its corresponding optimal teacher model, and then the teacher only needs to synthesize the assigned prompts. The assigning process is achieved by a router-guided mechanism with considering both the student model's learnability and teacher model's response quality. Moreover, further analysis reveals that over 95% of prompts are routed to smaller teacher models (unlike the *Strong* baseline, which relies on a single super-sized LLM for all prompts), leading to more efficient synthesis.

To summarize, our contributions are as follows:

1) To construct personalized synthetic dataset for specific student model, we propose *PerSyn*, an efficient strategy that transfers the synthesis paradigm from "Generate then Select" to a more efficient manner "Route then Generate". In this paradigm, each prompt is first routed to its optimal teacher based on both learnability and quality, and each teacher model is then responsible only for synthesizing the prompts assigned to it.

2) Extensive experiments validate the effectiveness of *PerSyn* across different model families and scales in two common distillation settings (e.g., 8.7% on IFEval, and 7.5% on MATH). We also construct a math synthetic dataset ***PerSyn-Math***, which includes parallel responses from 15 teacher models to facilitate future research.

3) Further analysis offer valuable insights into the routing behavior of *PerSyn*. For example, both quality and learnability are important in *PerSyn*, with quality playing a more critical role.

2 PerSyn

In this section, we first present the criterion used by *PerSyn* to find the optimal teacher model for each prompt (§ 2.1). Next, we describe how *PerSyn* transfers the "Generate then Select" paradigm to the more efficient "Route then Generate" paradigm with a router, and how the resulting synthetic data is used to train the student model (§ 2.2). Finally, we illustrate how the *PerSyn* router is obtained (§ 2.3). Fig. 1 shows the overview of *PerSyn* strategy.

2.1 Finding the Optimal Teacher

Given a prompt x_i, a straightforward way to select the optimal teacher's response for a student model θ is the "Generate then Select" paradigm. This approach first lets all teacher models $\mathcal{M} = \{\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_n\}$ generate parallel responses $\mathcal{Y}_i = \{y_i^{\mathcal{M}_1}, y_i^{\mathcal{M}_2}, \ldots, y_i^{\mathcal{M}_n}\}$ for x_i (where $y_i^{\mathcal{M}_n}$ is the response from $\mathcal{M}_n$), and then selects the optimal response for the student model θ from $\mathcal{Y}_i$.

To identify the optimal response, we evaluate each $y_i^{\mathcal{M}_n}$ using two complementary reward. The first is the learnability reward, which measures how easily the student model θ can learn from $y_i^{\mathcal{M}_n}$. Responses that are too difficult for θ, i.e., have a large learnability gap, tend to make learning inefficient (Li et al., 2025a; Xu et al., 2025b). We compute the learnability reward $\boldsymbol{r_l}(y_i^{\mathcal{M}_n}, \theta)$ using the student's self-derived log-likelihood:

$$\boldsymbol{r_l}(y_i^{\mathcal{M}_n}, \theta) = \frac{1}{|y_i^{\mathcal{M}_n}|} \sum_{t=1}^{|y_i^{\mathcal{M}_n}|} \log p_\pi\big(y_i^{\mathcal{M}_n(t)} \mid y_i^{\mathcal{M}_n(<t)}, x_i\big), \tag{1}$$

where $\log p_\pi\big(y_n^{\mathcal{M}_i(t)} \mid y_i^{\mathcal{M}_n(<t)}, x_i\big)$ is the probability assigned by the student to the t-th token of $y_i^{\mathcal{M}_n}$, given its preceding tokens and the prompt x_i. Intuitively, a higher $\boldsymbol{r_l}(y_i^{\mathcal{M}_n}, \theta)$ indicates that $y_i^{\mathcal{M}_n}$ aligns well with the student's existing knowledge and capabilities, therefore is more learnable.

However, learnability alone is insufficient. For instance, a response may be highly learnable yet trivial or low-quality, offering little benefit to the student. To account for this, we introduce a quality reward $\boldsymbol{r_q}(y_i^{\mathcal{M}_n})$, estimated by a reward model, where larger values indicate higher quality.

The overall reward of $y_i^{\mathcal{M}_n}$ for θ is then computed as a weighted combination of two aspects:

$$\boldsymbol{r}(y_i^{\mathcal{M}_n}, \theta) = (1-\alpha)\boldsymbol{r_q}(y_i^{\mathcal{M}_n}) + \alpha \boldsymbol{r_l}(y_i^{\mathcal{M}_n}, \theta), \tag{2}$$

where α balances the contribution of learnability and quality rewards.[2] Finally, the teacher model corresponding to the response with the highest $\boldsymbol{r}(y_n, \theta)$ is selected as optimal.

[2] Both rewards are normalized across teacher models before combination.

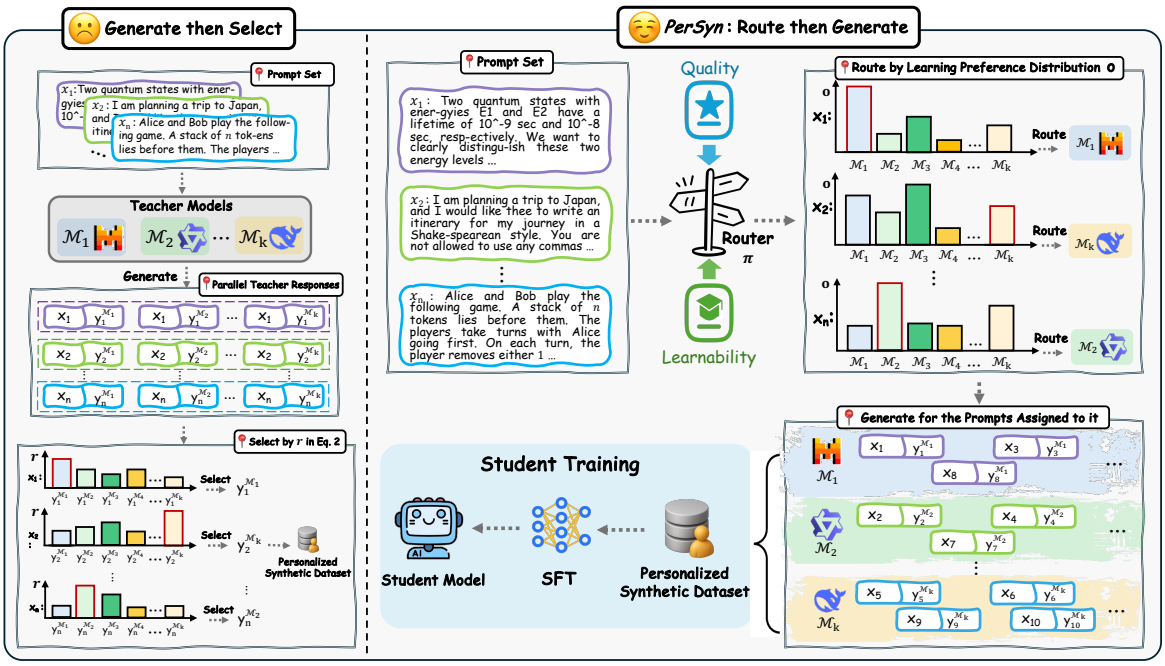

Figure 1: Overview of the two paradigm for obtaining a personalized synthetic dataset. The left part illustrates how we select optimal teacher response for each prompt using the proposed criterion. This process follows the conventional "Generate then Select" approach, which requires parallel teacher responses for the entire prompt set (§ 2.1). In contrast, *PerSyn* adopts a more efficient "Route then Generate" paradigm: it first routes each prompt to an optimal teacher based on learnability and quality via a router-guided mechanism, and teachers generate responses only for their assigned prompts (§ 2.2). Details of router training are described in § 2.3.

2.2 Transfer Paradigm via *PerSyn* Router

However, the "Generate then Select" paradigm is inefficient. For example, synthesizing a dataset of 100K prompts with 20 teacher models would require 2,000K total generations, which is costly and impractical. To address this, we introduce *PerSyn* router, which transfers the paradigm to a more efficient "Route then Generate" manner: each prompt is first routed to its optimal teacher, and each teacher generates responses only for the prompts assigned to it (Fig. 1 illustrates the comparison between the two paradigms).

More specifically, under the paradigm of "Route then Generate", suppose we have a prompt set $\mathcal{X} = \{x_1, x_2, \ldots, x_{10}\}$ and a teacher model set $\mathcal{M} = \{\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3\}$, the *PerSyn* router will first assigns each prompt to its optimal teacher. For example, $\mathcal{M}_1$ is assigned $\mathcal{X}_{\mathcal{M}_1} = \{x_1, x_3, x_8\}$, $\mathcal{M}_2$ is assigned $\mathcal{X}_{\mathcal{M}_2} = \{x_2, x_4, x_7\}$, and $\mathcal{M}_3$ is assigned $\mathcal{X}_{\mathcal{M}_3} = \{x_5, x_6, x_9, x_{10}\}$. Each teacher $\mathcal{M}_i$ then generates responses only for its assigned subset $\mathcal{X}_{\mathcal{M}_i}$ to obtain $\mathcal{D}_{\mathcal{M}_i}$ (e.g., $\mathcal{D}_{\mathcal{M}_2} = \{(x_2, y_2^{\mathcal{M}_2}), (x_4, y_4^{\mathcal{M}_2}), (x_7, y_7^{\mathcal{M}_2})\}$). The final synthetic dataset is $\mathcal{D} = \{\mathcal{D}_{\mathcal{M}_i}\}_{i=1}^{|\mathcal{M}|}$. The student model is then trained via supervised fine-tuning (SFT) using the synthetic dataset $\mathcal{D}$, where the loss is computed only on the response tokens.

2.3 Obtaining *PerSyn* Router

Formally, given a prompt x, the *PerSyn* router π will produce a distribution $\mathbf{o} = \pi(x) \in \mathbb{R}^{|\mathcal{M}|}$, which reflects the student model θ's learning preference over the teacher model set $\mathcal{M}$. Each component $\mathbf{o}_i$ indicates the learning preference of teacher model $\mathcal{M}_i$ on prompt x. Importantly, this distribution varies across prompts, capturing the intuition that different teachers excel at different queries.

To model the learning preferences of student model, we adopt the Bradley-Terry (BT) model (Bradley and Terry, 1952). Given a comparison between two teacher models A and B, the BT model defines the probability that B provides higher learning preference than A as:

$$\mathbb{P}(C = B \succ A \mid Z = z, X = x) = \sigma(z^\top \pi(x)), \tag{3}$$

where z is a 'two-hot' encoding of the model comparison pair (A, B), i.e., a vector of length $|\mathcal{M}|$ with $+1$ at the index of B, -1 at the index of A, and zeros elsewhere. Here, σ denotes the sigmoid function. The label $C = 1$ indicates that B is

preferred over A, and $C = 0$ otherwise. Thus, $\sigma(z^\top \pi(x))$ gives the probability of model B being favored over model A for student π on prompt x.

To construct the pairwise preference dataset $\mathcal{K} = \{(X, Z, C)\}_{i=1}^{N}$, we first sample a small subset of prompts $\mathcal{X}_{\text{sub}} \subset \mathcal{X}$, and query all teacher models $\mathcal{M}$ to obtain their parallel responses, forming $\mathcal{P}_{\text{sub}}$. Each response is then scored using the reward metric defined in Eq. 2 (see § 2.1), which allows us to derive exact teacher rankings and subsequently generate pairwise comparisons (See Appendix A.2 for more details about the pairwise dataset construction).

Finally, given pairwise learning preference dataset $\mathcal{K}$, the *PerSyn* router is obtained by:

$$\hat{\pi} = \text{argmin} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(\sigma(Z_i^\top \pi(X_i)), C_i). \quad (4)$$

where $\mathcal{L}$ is the binary cross-entropy loss. By default, we instantiate *PerSyn* router on top of Qwen2.5-1.5B, as supported by the experiments in § 3.3. Note that for each student model, a separate router is needed for each setting.

3 Experiments

In this section, we conduct experiments on two common distillation settings, instruction tuning and math reasoning, to validate the effectiveness of our *PerSyn* strategy.

3.1 Experiment Settings

Datasets. To obtain the synthetic dataset for training the student model in both settings, we proceed as follows. For instruction tuning, we randomly sample 50K prompts from the Magpie-100K-Generator-Zoo (Magpie-Zoo)[3] to construct the training dataset. The distilled student models are evaluated on TruthfulQA (Lin et al., 2022), LiveBench (White et al., 2025), and IFEval (Zhou et al., 2023). For mathematical reasoning, we construct a dataset with parallel teacher responses, denoted ***PerSyn-Math***, by randomly sampling 10K queries from OpenR1-Math-220K[4] as seed data and distilling responses from 15 teacher models. This dataset is then used to build the synthetic training dataset for student models.[5] The resulting student models are evaluated on SVAMP (Patel et al., 2021), MATH (Hendrycks et al., 2021), and GSM8K (Cobbe et al., 2021). Additional details about datasets and evaluation can be found in Appendix A.3. To obtain the *PerSyn* router, we construct parallel teacher responses for 2.5K prompts in both settings to build pairwise preference data.[6]

Student models. In our main experiments, we employ five student models from Qwen2.5 (Yang et al., 2024c), Gemma-2 (Team et al., 2024), and Llama-3.2 (Meta, 2024) model families, which are Qwen2.5-0.5B, Qwen2.5-1.5B, Gemma-2-2B, Qwen2.5-3B, and Llama-3.2-3B.

Teacher models. We consider teacher models of various sizes and families for distillation in two settings. Specifically, in the instruction tuning, we employ 19 teacher models from Qwen2 (Team, 2024), Qwen2.5 (Yang et al., 2024c), Llama-3/3.1 (Dubey et al., 2024), Gemma-2 (Team et al., 2024), and Phi-3 (Abdin et al., 2024) model families. For math reasoning, we employ 15 teacher models from Mistral (Jiang et al., 2023), Gemma-2 (Team et al., 2024), Llama-3.1/3.3 (Dubey et al., 2024), Qwen2.5 (Yang et al., 2024c), Qwen2.5-Math (Yang et al., 2024a), Qwen3 (Yang et al., 2025a), and DeepSeek-R1 (Guo et al., 2025) model families. An overview of the teacher models for the two setting is presented in Table 11 of Appendix A.4.

Baselines. We compare our proposed *PerSyn* against several baseline methods, all of which are listed below. **1) Strong**: a straightforward approach that uses a single strongest LLM as the teacher to synthesize data; **2) Mix**: mix distillation (Li et al., 2025a) that derives the synthetic dataset by mixing responses from weak and strong teacher models; **3) Family-Strong**: a potential baseline based on the finding of (Xu et al., 2025b), which suggests that learning from strong teacher model within the same family as student model can improve distillation effectiveness; **4) CAR**: a metric proposed by Xu et al. (2025b) that selects a single teacher model which strikes a dedicate balance between response quality and compatibility. The specific teacher models used by the baselines for each student model in different tasks are shown in Table 6 and Table 7 of appendix A.1.

[3] https://huggingface.co/datasets/Magpie-Align/Magpie-100K-Generator-Zoo

[4] https://huggingface.co/datasets/open-r1/OpenR1-Math-220k

[5] In the instruction-tuning setting, Magpie-Zoo already provides parallel teacher responses for each prompt.

[6] As shown in § 3.3, 2.5K samples with parallel teacher responses are sufficient to obtain a well-performing *PerSyn* router.

Student Model	Strategy	IFEval	TruthfulQA	LiveBench	GSM8K	MATH	SVAMP	Avg.
Qwen2.5-0.5B	Strong	25.59	39.89	8.40	30.37	15.20	51.60	28.51
	Mix	26.06	40.54	8.10	33.83	20.60	55.40	30.75
	Family-Strong	26.75	41.43	8.60	35.62	22.80	57.00	32.03
	CAR	27.11	41.85	9.00	36.76	24.00	57.90	32.77
	PerSyn (Ours)	**28.73**	**43.01**	**9.80**	**38.25**	**25.60**	**59.40**	**34.13**
Qwen2.5-1.5B	Strong	31.52	49.04	12.80	64.83	44.20	78.50	46.82
	Mix	31.98	49.73	13.30	65.68	45.80	80.30	47.79
	Family-Strong	32.63	50.45	13.60	66.55	47.40	81.20	48.64
	CAR	33.06	50.98	13.30	67.37	48.60	81.90	49.21
	PerSyn (Ours)	**34.15**	**52.22**	**14.80**	**68.81**	**50.40**	**83.40**	**50.63**
Gemma-2-2B	Strong	28.84	40.17	10.30	29.71	14.20	47.50	28.45
	Mix	29.39	40.83	10.90	31.66	16.40	49.40	29.76
	Family-Strong	29.76	41.64	11.60	30.43	15.80	48.10	29.56
	CAR	30.11	42.28	**12.80**	33.25	19.20	50.80	31.41
	PerSyn (Ours)	**31.25**	**43.87**	12.40	**35.57**	**21.40**	**52.60**	**32.85**
Qwen2.5-3B	Strong	40.61	51.21	19.10	77.47	56.40	87.50	55.38
	Mix	41.35	51.82	19.30	77.19	55.80	86.90	55.39
	Family-Strong	42.44	53.37	20.40	77.94	57.10	88.30	56.59
	CAR	43.03	53.81	20.90	78.42	**58.10**	88.80	57.17
	PerSyn (Ours)	**44.16**	**55.14**	**22.30**	**79.09**	57.80	**90.10**	**58.09**
Llama-3.2-3B	Strong	27.89	42.31	10.80	34.59	21.40	52.20	31.37
	Mix	29.78	43.56	12.00	34.17	20.50	51.50	31.75
	Family-Strong	28.25	42.63	11.10	33.83	19.50	50.80	30.85
	CAR	30.53	44.32	11.80	35.91	22.80	53.60	32.99
	PerSyn (Ours)	**32.31**	**46.15**	**12.60**	**38.15**	**24.50**	**55.30**	**34.81**

Table 2: Results of baseline methods and our *PerSyn* strategy evaluated on six benchmarks with five student models from different families. See § 3.1 for details about the *Strong*, *Mix*, *Family-Strong*, and *CAR* baselines.

Implementation. We train the student models using the LLaMA-Factory (Zheng et al., 2024) framework. Student models up to 14B parameters are trained with full-parameter fine-tuning, while those larger than 14B are fine-tuned with LoRA. For instruction tuning, we employ the state-of-the-art reward model Skywork-Reward-Llama-3.1-8B from RewardBench (Lambert et al., 2025) to obtain the quality rewards.[7] We also conduct additional experiments to study the impact of weak versus strong reward models. See Appendix A.1 for details about training setup and reward models experiments.

3.2 Performance of *PerSyn*

Table 2 reports the results of baselines and our proposed *PerSyn* strategy across different benchmarks and student models. The results demonstrate that *PerSyn* consistently outperforms all baselines in both instruction tuning and math reasoning settings. For instance, on Qwen2.5-3B, *PerSyn* surpasses the *Strong* baseline by 2.9%, 7.6%, and 8.7% on SVAMP, TruthfulQA, and IFEval, respectively, indicating that the strongest model is not always the best teacher for small student models. Relative to the strong baseline *CAR* on Llama-3.2-3B, *PerSyn* achieves gains of 4.1% on TruthfulQA, 5.8% on IFEval, and a more substantial 7.5% on MATH.

[7] In the math reasoning setting, the quality reward is binary: 1 for correct answers and 0 for incorrect ones.

Based on these observations, we conclude that:

> **Finding 1.** *PerSyn, by jointly considering both learnability and quality to find the optimal teacher for each prompt tailored to the student model, leads to more effective student learning.*

3.3 Further Analysis

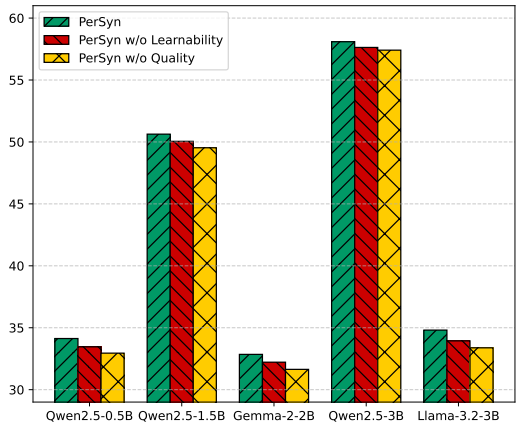

Figure 2: Average results of the ablation studies on *PerSyn* across all benchmarks. "w/o" denotes the exclusion of a specific reward term from *PerSyn* when assigning prompts to teachers.

Ablation Study of *PerSyn* We conduct ablation studies to examine the roles of learnability and quality within *PerSyn*. Specifically, we evaluate

the performance of *PerSyn* when either learnability or quality is excluded. As shown in Fig. 2, the results demonstrate consistent performance drops for both "*PerSyn* w/o Learnability" and "*PerSyn* w/o Quality" across all student models. Moreover, the performance degradation is more pronounced when quality is excluded, suggesting that relying solely on high-learnability data, i.e., knowledge that is easy for the student to absorb, offers limited improvement. These observations indicate that:

> **Finding 2.** *Jointly considering both learnability and quality yields better performance than considering either alone, with quality playing a more critical role than learnability in PerSyn.*

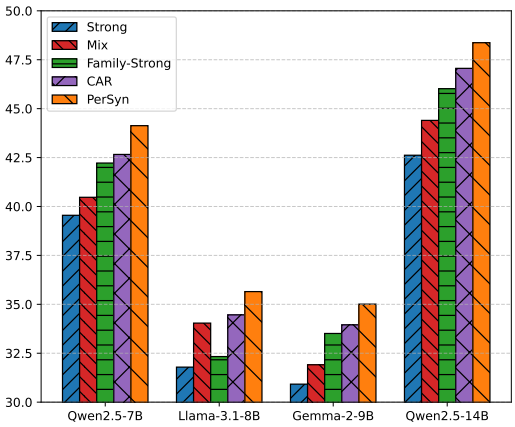

Figure 3: The average results of baselines and *PerSyn* on four larger-scale student models spanning three model families in the instruction tuning setting. Detailed results are provided in Table 12.

***PerSyn* on Larger Model Scales** Having verified the effectiveness of *PerSyn* on small student models, we further evaluate its generalization to larger-scale models in the instruction tuning setting. As shown in Fig. 3, *PerSyn* consistently outperforms all baselines in terms of average performance. In particular, compared to the strong baseline *CAR*, *PerSyn* achieves average improvements of 3.4%, 3.6%, 3.1%, and 2.7% on Qwen2.5-7B, Llama-3.1-8B, Gemma-2-9B, and Qwen2.5-14B, respectively. These findings demonstrate the broad effectiveness of *PerSyn* across diverse model scales and families.

Suitable α for *PerSyn* To determine the suitable value for α in Eq. 2 within *PerSyn*, we conduct experiments with α ranging from 0.1 to 0.9. The results in Fig. 4 exhibit a rising trend initially, reaching a peak at $\alpha = 0.4$, and then gradually declining, suggesting that quality is more important than learnability, consistent with the Finding 2 we derive in § 3.3. A similar trend is observed across three student models from different families. Therefore, we set $\alpha = 0.4$ by default in all experiments.

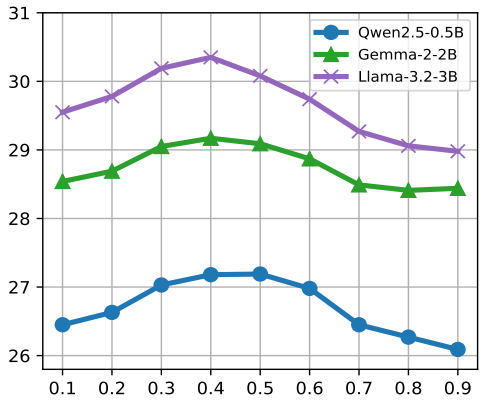

Figure 4: The average results across different α values for three student models from distinct model families in the instruction tuning setting.

Performance of *PerSyn* Router We conduct additional experiments to study the impact of pairwise training dataset size and backbone model size on the performance of the *PerSyn* router. In these experiments, Qwen2.5 serves as the backbone model for *PerSyn* router. Fig. 5 presents the Hit@3 performance[8] of the router across different backbone model sizes (0.5B to 3B) and pairwise training dataset sizes (200K to 800K) in the instruction tuning setting. The results show that router performance initially improves with increasing training data and then stabilizes once the dataset exceeds 500K (constructed by 2.5K prompts with parallel teacher responses, see Appendix A.2 for more details). Similar trends are observed across all three model sizes. We also note that the performance of the 1.5B model is comparable to that of the 3B model. Based on these observations, unless otherwise specified, the *PerSyn* router is obtained by training Qwen2.5-1.5B on 2.5K prompts with parallel teacher responses by default.

To further validate the effectiveness of the *PerSyn* router, we conduct an additional experiment comparing it with the *Oracle* router, which directly leverages ground-truth reward (as defined in Eq. 2) to route each prompt to its optimal teacher model. Table 3 reports the average results in the instruction tuning setting across different student models. The

[8] Hit@3 denotes the proportion of cases where the teacher model assigned by the *PerSyn* router falls within the top-3 ground-truth teachers.

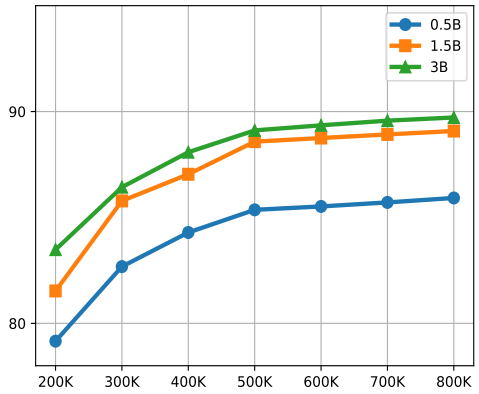

Figure 5: The performance of the *PerSyn* router for the Qwen2.5-3B student model across different backbone model sizes and pairwise training dataset sizes in the instruction tuning setting. Notably, 500K pairwise training samples, which can be constructed from only 2.5K parallel teacher responses, are sufficient to obtain an effective *PerSyn* router. Similar observations in math reasoning setting are provided in Appendix A.2.

results show that the *PerSyn* router achieves performance comparable to, or even exceeding, that of the *Oracle* router (similar observations in the math reasoning setting are provided in Table 9). Notably, the *Oracle* router requires parallel teacher responses for the entire prompt set. In contrast, the *PerSyn* router only requires 2.5K parallel teacher responses, making it significantly more efficient.

	PerSyn Router	*Oracle* Router
Qwen2.5-0.5B	27.18	27.63
Qwen2.5-1.5B	33.72	34.36
Gemma-2-2B	29.17	28.84
Qwen2.5-3B	40.53	41.02
Llama-3.2-3B	30.35	30.18

Table 3: The average performance of different student models in the instruction tuning setting using the *PerSyn* router and the *Oracle* router.

Teacher Models Allocated by *PerSyn* To delve deeper, we visualize the prompt allocation ratios assigned by *PerSyn* router across different teacher models for Qwen2.5-3B under instruction tuning and math reasoning settings in Fig. 6 and Fig. 7. As shown in Fig. 6, smaller teacher models, such as Qwen2.5-3B-Instruct, receive higher allocation compared to larger models, including Qwen2.5-7B/14B/32B-Instruct and even Llama-3.1-405B-Instruct. A similar trend is observed in math reasoning (Fig. 7), where Qwen2.5-7B-Instruct has higher allocation than Qwen2.5-14B/32B-Instruct. See Fig. 10 for the prompt allocation ratios of other student models. Based on these observations, we

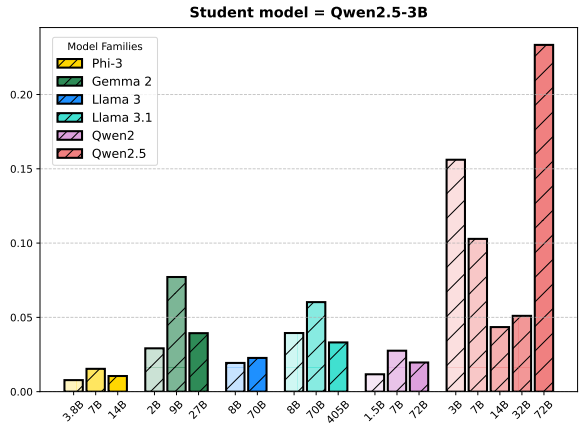

Figure 6: Prompt allocation ratios assigned by *PerSyn* across different teacher models for Qwen2.5-3B student in the instruction tuning setting. Colors indicate different model families, and darker shades correspond to larger teacher models within the same family.

derive the following conclusion:

> **Finding 3.** *Larger teacher models, despite their superior performance, are not always the optimal teacher for small student models; small teachers are often more suitable.*

Interestingly, Qwen2.5-72B-Instruct consistently receives high allocation across student models in both settings, suggesting it is a strong and versatile teacher.

In addition, Fig. 7 shows that teacher models producing Long-CoT responses account for only a small portion of the allocated prompts compared to Short-CoT models[9]. Similar trends for other student models are reported in Fig. 10. To examine their necessity of Long-CoT models, we conduct an experiment where prompts originally assigned to Long-CoT models are forcibly reassigned to a strong Short-CoT teacher (Qwen2.5-Math-7B-Instruct), while keeping the allocation of all other prompts unchanged. We then construct the synthetic dataset based on this modified allocation and use it to train the Qwen2.5-3B student model. The result shows that this replacement led to a 1.3% drop in average performance compared to the original *PerSyn* allocation. Moreover, Qwen2.5-Math-7B-Instruct was able to correctly answer only 7.4% of the prompts that were initially assigned to Long-CoT models. These findings indicate that:

[9] In this paper, we refer to models that generate Long-CoT or Short-CoT responses as Long-CoT and Short-CoT models, respectively.

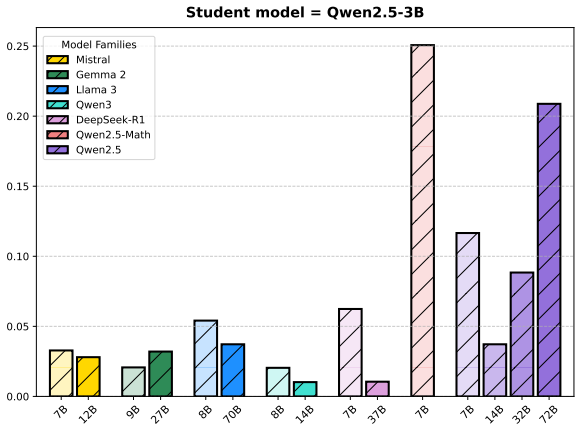

Figure 7: Prompt allocation ratios assigned by *PerSyn* across different teacher models for Qwen2.5-3B student in the math reasoning setting. Colors indicate different model families, and darker shades correspond to larger teacher models within the same family. Note that DeepSeek-R1 and Qwen3 are Long-CoT models, while the remaining are Short-CoT models.

> **Finding 4.** *While Long-CoT models are not optimal teachers for small student models in most cases, they remain necessary for handling certain complex prompts.*

Finally, Table 2 shows that training student models on datasets consisting entirely of Long-CoT responses (*Strong* baseline) results in degraded performance relative to *PerSyn*, indicating that excessive reliance on Long-CoT data is detrimental. Further analysis of generated outputs shows that models trained with *Strong* often produce repetitive reasoning without termination, leading to incorrect answers. In contrast, models trained with *PerSyn* can generate suitably extended reasoning paths and produce correct results. This aligns with the observations reported in Li et al. (2025a).

4 Related Work

Large Language Models for Data Synthesis Due to the high cost of human data annotation, recent research has turned to Large Language Models (LLMs) for synthetic data generation as a practical alternative (Long et al., 2024; Tan et al., 2024). This line of work has demonstrated promising results across diverse domains. For example, LLMs can generate instruction-response pairs in instruction tuning (Taori et al., 2023; He et al., 2025; Yuan et al., 2025) and elaborate rationales for reasoning tasks (Shridhar et al., 2023; Li et al., 2025c; Guo et al., 2025). Other task-specific data synthesis include dialogue generation (Sun et al., 2024; Wang et al., 2025b), code generation (Roziere et al., 2023; Majumdar et al., 2025), and retrieval-augmented generation (Xu et al., 2025a; Yang et al., 2025b). In this work, we mainly conduct experiments on two prevalent data synthesis scenarios: instruction tuning and math reasoning.

Distillation with Synthetic Data To improve efficiency, previous studies aimed to use the synthetic data generated by large and powerful LLMs to train smaller models (Chiang et al., 2023; Busbridge et al., 2025). However, recent works have demonstrated that stronger models are not always stronger teacher models for data generation (Kim et al., 2024b; Xu et al., 2025b; Li et al., 2025a). Specifically, Li et al. (2025a) showed that small student models struggle to learn from strong reasoners due to the learnability gap and introduced a mixed distillation strategy to mitigate this gap. Xu et al. (2025b) found that learning from response generators within the same model family yields higher performance and propose a metric to select the best teacher by measuring the quality and compatibility. However, these methods are inefficient and ineffective, as they require generating parallel teacher responses and cannot produce a synthetic dataset that is optimal at the sample level. In this work, we aim to assign each prompt to the most suitable teacher model for data synthesis, thereby constructing the final personalized dataset.

5 Conclusion

In this work, we propose *PerSyn*, a new strategy that constructs personalized synthetic data for a specific student model to help it learn more effectively. Unlike previous work that employ a single selected teacher model to synthesize data for the entire prompt set, *PerSyn* operates in a more fine-grained manner: it assigns each prompt to its optimal teacher model for synthesis, based on both the student model's learnability and the teacher model's response quality. Furthermore, *PerSyn* transfers the synthesis paradigm from the conventional "Generate then Select" to "Route then Generate" by introducing a router-guided mechanism. Extensive experiments demonstrate that synthetic data constructed by *PerSyn* effectively facilitates the learning of student models, achieving state-of-the-art performance across various scales and model families in both instruction tuning and math reasoning scenarios. Our comprehensive analysis also offers valuable insights for future research.

Limitations

While *PerSyn* demonstrates strong effectiveness in both instruction tuning and math reasoning, it remains unclear whether *PerSyn* can generalize to other scenarios, such as code generation, multimodal understanding, and other specialized domains. Furthermore, our experiments are limited to student models with up to 14B parameters, and we have not evaluated larger LLMs (e.g., 32B or 70B) due to computational constraints. We leave these for future work.

Ethical Considerations

All datasets used in our work are publicly released under open licenses, and all models employed in distillation or generation are open-source and licensed for research use. We strictly follow the licensing terms and usage policies of these resources.

References

Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Hassan Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Singh Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Caio C'esar Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Parul Chopra, and 69 others. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *ArXiv*, abs/2404.14219.

Ralph Allan Bradley and Milton E Terry. 1952. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345.

Dan Busbridge, Amitis Shidani, Floris Weers, Jason Ramapuram, Etai Littwin, and Russell Webb. 2025. Distillation scaling laws. In *Forty-second International Conference on Machine Learning*.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, and 5 others. 2024. The language model evaluation harness.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Linda He, WANG Jue, Maurice Weber, Shang Zhu, Ben Athiwaratkun, and Ce Zhang. 2025. Scaling instruction-tuned llms to million-token contexts via hierarchical synthetic data generation. In *The Thirteenth International Conference on Learning Representations*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *NeurIPS*.

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

Albert Qiaochu Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *ArXiv*, abs/2310.06825.

Gyeongman Kim, Doohyuk Jang, and Eunho Yang. 2024a. Promptkd: Distilling student-friendly knowledge for generative language models via prompt tuning. *Preprint*, arXiv:2402.12842.

Seungone Kim, Juyoung Suk, Xiang Yue, Vijay Viswanathan, Seongyun Lee, Yizhong Wang, Kiril Gashteovski, Carolin Lawrence, Sean Welleck, and Graham Neubig. 2024b. Evaluating language models as synthetic data generators. *arXiv preprint arXiv:2412.03679*.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, Lester James Validad Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, and 1 others. 2025. Rewardbench: Evaluating reward models for language modeling. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 1755–1797.

Yuetai Li, Xiang Yue, Zhangchen Xu, Fengqing Jiang, Luyao Niu, Bill Yuchen Lin, Bhaskar Ramasubramanian, and Radha Poovendran. 2025a. Small models struggle to learn from strong reasoners. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 25366–25394, Vienna, Austria. Association for Computational Linguistics.

Zhong-Zhi Li, Duzhen Zhang, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Pei-Jie Wang, Xiuyi Chen, and 1 others. 2025b. From system 1 to system 2: A survey of reasoning large language models. *arXiv preprint arXiv:2502.17419*.

Zhuochun Li, Yuelyu Ji, Rui Meng, and Daqing He. 2025c. Learning from committee: Reasoning distillation from a mixture of teachers with peer-review. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 4190–4205, Vienna, Austria. Association for Computational Linguistics.

Xiao Liang, Zhong-Zhi Li, Yeyun Gong, Yang Wang, Hengyuan Zhang, Yelong Shen, Ying Nian Wu, and Weizhu Chen. 2025. Sws: Self-aware weakness-driven problem synthesis in reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.08989*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252.

Lin Long, Rui Wang, Ruixuan Xiao, Junbo Zhao, Xiao Ding, Gang Chen, and Haobo Wang. 2024. On llms-driven synthetic data generation, curation, and evaluation: A survey. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 11065–11082.

Somshubra Majumdar, Vahid Noroozi, Mehrzad Samadi, Sean Narenthiran, Aleksander Ficek, Wasi Uddin Ahmad, Jocelyn Huang, Jagadeesh Balam, and Boris Ginsburg. 2025. Genetic instruct: Scaling up synthetic generation of coding instructions for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pages 208–221, Vienna, Austria. Association for Computational Linguistics.

Meta. 2024. Llama 3.2: Revolutionizing edge ai and vision with open, customizable models.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are nlp models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094.

Libo Qin, Qiguang Chen, Yuhang Zhou, Zhi Chen, Yinghui Li, Lizi Liao, Min Li, Wanxiang Che, and Philip S Yu. 2025. A survey of multilingual large language models. *Patterns*, 6(1).

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, and 1 others. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. 2023. Distilling reasoning capabilities into smaller language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7059–7073, Toronto, Canada. Association for Computational Linguistics.

Renliang Sun, Mengyuan Liu, Shiping Yang, Rui Wang, Junqing He, and Jiaxing Zhang. 2024. Fostering natural conversation in large language models with nico: a natural interactive conversation dataset. *arXiv preprint arXiv:2408.09330*.

Zhen Tan, Dawei Li, Song Wang, Alimohammad Beigi, Bohan Jiang, Amrita Bhattacharjee, Mansooreh Karami, Jundong Li, Lu Cheng, and Huan Liu. 2024. Large language models for data annotation and synthesis: A survey. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 930–957.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, and 1 others. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Qwen Team. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2.

Linyong Wang, Lianwei Wu, Shaoqi Song, Yaxiong Wang, Cuiyun Gao, and Kang Wang. 2025a. Distilling structured rationale from large language models to small language models for abstractive summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25389–25397.

Zezhong Wang, Xingshan Zeng, Weiwen Liu, Liangyou Li, Yasheng Wang, Lifeng Shang, Xin Jiang, Qun Liu, and Kam-Fai Wong. 2025b. Toolflow: Boosting llm tool-calling through natural and coherent dialogue synthesis. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4246–4263.

Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Benjamin Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Sreemanti Dey, Shubh-Agrawal, Sandeep Singh Sandha, Siddartha Venkat Naidu, Chinmay Hegde, Yann LeCun, Tom Goldstein, Willie Neiswanger, and Micah Goldblum. 2025.

Livebench: A challenging, contamination-free LLM benchmark. In *The Thirteenth International Conference on Learning Representations*.

Ran Xu, Hui Liu, Sreyashi Nag, Zhenwei Dai, Yaochen Xie, Xianfeng Tang, Chen Luo, Yang Li, Joyce C. Ho, Carl Yang, and Qi He. 2025a. SimRAG: Self-improving retrieval-augmented generation for adapting large language models to specialized domains. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 11534–11550, Albuquerque, New Mexico. Association for Computational Linguistics.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Bill Yuchen Lin, and Radha Poovendran. 2025b. Stronger models are not always stronger teachers for instruction tuning. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4392–4405.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025a. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, and 1 others. 2024a. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*.

Hang Yang, Hao Chen, Hui Guo, Yineng Chen, Ching-Sheng Lin, Shu Hu, Jinrong Hu, Xi Wu, and Xin Wang. 2024b. Llm-medqa: Enhancing medical question answering through case studies in large language models. *arXiv preprint arXiv:2501.05464*.

Qwen An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jingren Zhou, Junyang Lin, and 25 others. 2024c. Qwen2.5 technical report. *ArXiv*, abs/2412.15115.

Shiping Yang, Jie Wu, Wenbiao Ding, Ning Wu, Shining Liang, Ming Gong, Hengyuan Zhang, and Dongmei Zhang. 2025b. Quantifying the robustness of retrieval-augmented language models against spurious features in grounding data. *arXiv preprint arXiv:2503.05587*.

Yiyao Yu, Yuxiang Zhang, Dongdong Zhang, Xiao Liang, Hengyuan Zhang, Xingxing Zhang, Ziyi Yang, Mahmoud Khademi, Hany Awadalla, Junjie Wang, and 1 others. 2025. Chain-of-reasoning: Towards unified mathematical reasoning in large language models via a multi-paradigm perspective. *arXiv preprint arXiv:2501.11110*.

Lin Yuan, Jun Xu, Honghao Gui, Mengshu Sun, Zhiqiang Zhang, Lei Liang, and Jun Zhou. 2025. Improving natural language understanding for llms via large-scale instruction synthesis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25787–25795.

Hengyuan Zhang, Chenming Shang, Sizhe Wang, Dongdong Zhang, Feng Yao, Renliang Sun, Yiyao Yu, Yujiu Yang, and Furu Wei. 2024. Shifcon: Enhancing non-dominant language capabilities with a shift-based contrastive framework. *arXiv preprint arXiv:2410.19453*.

Huaqin Zhao, Zhengliang Liu, Zihao Wu, Yiwei Li, Tianze Yang, Peng Shu, Shaochen Xu, Haixing Dai, Lin Zhao, Gengchen Mai, and 1 others. 2024. Revolutionizing finance with llms: An overview of applications and insights. *arXiv preprint arXiv:2401.11641*.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand. Association for Computational Linguistics.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.

A Appendix

A.1 Implementation Details

Hyper-parameter	Value
Learning Rate	2×10^{-5}
Number of Epochs	2
Number of Devices	4
Per-device Batch Size	1
Gradient Accumulation Steps	8
Effective Batch Size	32
Optimizer	AdamW
Learning Rate Scheduler	cosine
Warmup Steps	100
Max Sequence Length	4096/16384

Table 4: The hyper-parameters used for full parameter fine-tuning of models smaller than 14B. The max sequence length in the instruction tuning setting is 4,096, while that in the math reasoning setting is 16,384. Notably, the "max_position_embeddings" of Gemma-2 is 8,192; therefore, its max sequence length in the math reasoning setting is limited to 8,192.

Training Setup. Table 4 and Table 5 present the detailed hyper-parameters of full parameter fine-tuning and LoRA fine-tuning. We conduct our experiments on a server equipped with eight NVIDIA A100-SXM4-80GB GPUs.

Reward Models. To study the impact of the reward model used in § 2.1, we conduct additional experiments in the instruction tuning setting, comparing the performance of *PerSyn* with a weak reward model (Skywork-Reward-V2-Llama-3.1-3B) and a strong reward model (Skywork-Reward-V2-Llama-3.1-8B). The results in Fig. 8 show that using the weak reward model Skywork-Reward-V2-Llama-3.1-3B yields worse performance than the strong reward model Skywork-Reward-V2-Llama-3.1-8B, highlighting that *PerSyn* benefits from a well-performing reward model. However, even with a weak reward model, *PerSyn* still outperforms the strong baseline *CAR*. By default, we use Skywork-Reward-V2-Llama-3.1-8B reward model in all instruction tuning experiments.

Baselines Setup. We compare the following baselines to verify the effectiveness of our *PerSyn* in our experiments: **1) Strong** refers to using the strongest single LLM as the teacher to synthesize data. **2) Mix** indicates mix distillation (Li et al., 2025a) that derives the synthetic dataset by mixing outputs from weak and strong teacher models. In this work, we implement a variant of mix distillation by randomly assigning a teacher model's response to each prompt. Notably, the teacher models range from small to large sizes, thereby the synthetic dataset includes responses from both weak and strong teacher models. **3) Family-Strong** represents choosing the strongest teacher model that is within the same family as the student model. **4) CAR** is a compatibility-adjusted reward metric pro-

Hyper-parameter	Value
Learning Rate	1×10^{-4}
Number of Epochs	2
Number of Devices	4
Per-device Batch Size	1
Gradient Accumulation Steps	8
Effective Batch Size	32
Optimizer	AdamW
Lora Target	full
Learning Rate Scheduler	cosine
Warmup Ratio	100
Max Sequence Length	4096/16384

Table 5: The hyper-parameters used for LoRA fine-tuning of models larger than 14B. The max sequence length in the instruction tuning setting is 4,096, while that in the math reasoning setting is 16,384. Notably, the "max_position_embeddings" of Gemma-2 is 8,192; therefore, its max sequence length in the math reasoning setting is limited to 8,192.

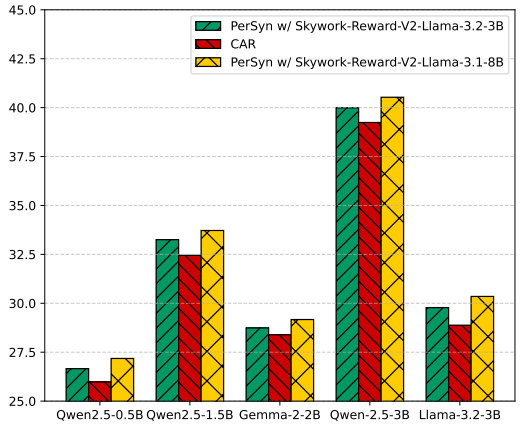

Figure 8: The average results of the strong baseline *CAR* and *PerSyn* with strong and weak reward models in the instruction tuning setting.

Student Models	Instruction Tuning		
	Strong	**Family-Strong**	**CAR**
Qwen2.5-0.5B	Llama-3.1-405B-Instruct	Qwen2.5-72B-Instruct	Qwen2.5-3B-Instruct
Qwen2.5-1.5B	Llama-3.1-405B-Instruct	Qwen2.5-72B-Instruct	Qwen2.5-3B-Instruct
Gemma-2-2B	Llama-3.1-405B-Instruct	Gemma-2-27b-it	Qwen2.5-72B-Instruct
Qwen2.5-3B	Llama-3.1-405B-Instruct	Qwen2.5-72B-Instruct	Qwen2.5-3B-Instruct
Llama-3.2-3B	Llama-3.1-405B-Instruct	Llama-3.1-70B-Instruct	Qwen2.5-3B-Instruct

Table 6: The assigned teacher models of our compared baseline methods in instruction tuning setting.

Student Models	Math Reasoning		
	Strong	**Family-Strong**	**CAR**
Qwen2.5-0.5B	DeepSeek-R1-37B	Qwen2.5-72B-Instruct	Qwen2.5-Math-7B-Instruct
Qwen2.5-1.5B	DeepSeek-R1-37B	Qwen2.5-72B-Instruct	Qwen2.5-Math-7B-Instruct
Gemma-2-2B	DeepSeek-R1-37B	Gemma-2-27b-it	Llama-3.1-8B-Instruct
Qwen2.5-3B	DeepSeek-R1-37B	Qwen2.5-72B-Instruct	Qwen2.5-Math-7B-Instruct
Llama-3.2-3B	DeepSeek-R1-37B	Llama-3.3-70B-Instruct	Qwen2.5-Math-7B-Instruct

Table 7: The assigned teacher models of our compared baseline methods in math reasoning setting.

posed by Xu et al. (2025b) that selects a single teacher model based on the average performance of teacher model on the entire dataset.

We present the assigned teacher models of our compared baseline methods in the instruction tuning and math reasoning settings in Table 6 and Table 7.

A.2 Details of *PerSyn* Router

To determine the optimal pairwise training dataset size and model size for obtaining an effective *PerSyn* router, we first reserve 1K samples as the evaluation set. From the remaining data, we randomly sample subsets of different sizes N as training sets. Next, all teacher models generate parallel responses for prompts in both the training and evaluation sets. Then, we compute the learnability and quality rewards via Eq. 2 for these parallel teacher responses, which allows us to establish the ground-truth ranking for each prompt. The labeled training set of size N is then used to construct a pairwise dataset of size M for training *PerSyn* routers of different model sizes, while the evaluation set is used to assess router performance.

The desired pairwise dataset size M determines the required sample size N. For instance, in the instruction tuning setting, 19 teacher models[10] produce 190 model combinations. To obtain $M = 500\text{K}$ pairwise samples, a subset of size $N = 500\text{K} \div 190 \approx 2.5\text{K}$ prompts with parallel teacher responses suffices. Similarly, in the math reasoning setting, 15 teacher models yield 105 combinations; to obtain $M = 250\text{K}$ pairwise samples, a subset of size $N = 250\text{K} \div 105 \approx 2.5\text{K}$ prompts with parallel teacher responses is sufficient. Table 8 presents the different sample sizes N and their corresponding pairwise dataset sizes M for both settings.

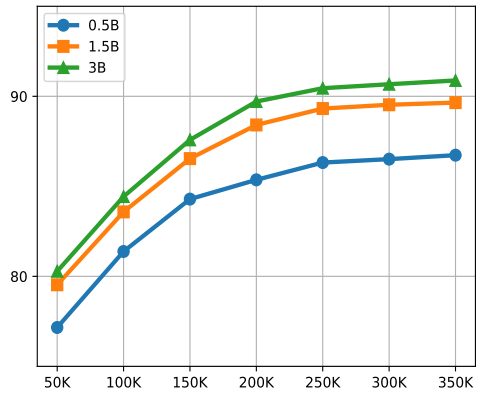

Figure 9: The performance of the *PerSyn* router for the Qwen2.5-3B student model across different backbone model sizes and pairwise training dataset sizes in the math reasoning setting. Notably, 250K pairwise training samples, which can be constructed from only 2.5K parallel teacher responses, are sufficient to obtain an effective *PerSyn* router.

[10] See Table 11 for details of teacher models in instruction tuning and math reasoning settings.

Instruction Tuning		Math Reasoning	
Sample Size N	Pair-wise Size M	Sample Size N	Pair-wise Size M
1K	200K	0.5K	50K
1.5K	300K	1K	100K
2K	400K	1.5K	150K
2.5K	500K	2K	200K
3K	600K	2.5K	250K
3.5K	700K	3K	300K
4K	800K	3.5K	350K

Table 8: Different prompt sample sizes N with parallel teacher responses and their corresponding pairwise training data sizes M in the instruction tuning and math reasoning settings. To obtain M pairwise training samples, each teacher model only needs to generate N responses.

We present the performance of the *PerSyn* router for the Qwen2.5-3B student model across different router model sizes and pairwise training dataset sizes under instruction tuning and math reasoning settings in Fig. 5 and Fig. 9. In Fig. 5, 500K pairwise training samples, constructed from only around $N = 2.5$K prompts with parallel teacher responses (see Table 8, "Instruction Tuning" column), are sufficient to obtain an effective *PerSyn* router in the instruction tuning setting. Similarly, Fig. 9 shows that 250K pairwise training samples, generated from approximately $N = 2.5$K prompts with parallel teacher responses (see Table 8, "Math Reasoning" column), are sufficient to obtain an effective *PerSyn* router in the math reasoning setting.

	PerSyn Router	*Oracle* Router
Qwen2.5-0.5B	41.08	41.45
Qwen2.5-1.5B	67.54	68.02
Gemma-2-2B	36.52	36.35
Qwen2.5-3B	75.66	75.78
Llama-3.2-3B	39.32	39.13

Table 9: The average performance of different student models in the math reasoning setting using the *PerSyn* router and the *Oracle* router.

To further validate the effectiveness of the *PerSyn* router, we conduct an additional experiment comparing it with the *Oracle* router, which uses ground-truth rewards to directly route each prompt to its optimal teacher model. The average results across different student models in both instruction tuning and math reasoning settings are reported in Table 3 and Table 9. In both settings, the *PerSyn* router achieves performance comparable to or even surpassing that of the *Oracle* router. Importantly, *Oracle* router requires parallel teacher responses for the entire prompt set, whereas the *PerSyn* router only requires 2.5K prompts with parallel responses, making it substantially more efficient.

Dataset	#Samples	Task Type	Data Type
Magpie-100K-Generator-Zoo (Magpie-Zoo) (Xu et al., 2025b)	50K	Instruction Tuning	Train
TruthfulQA (Lin et al., 2022)	817	Instruction Tuning	Test
LiveBench (White et al., 2025)	1436	Instruction Tuning	Test
IFEval (Zhou et al., 2023)	541	Instruction Tuning	Test
PerSyn-Math (Our)	10K	Math Reasoning	Train
SVAMP (Patel et al., 2021)	1000	Math Reasoning	Test
MATH (Hendrycks et al., 2021)	500	Math Reasoning	Test
GSM8K (Cobbe et al., 2021)	1320	Math Reasoning	Test

Table 10: Overview of the datasets we used in our experiments. #Samples indicates the number of samples per dataset. Notably, original Magpie-Zoo have already contained 19 parallel teacher responses for each prompt. We randomly sample 50K from it to construct training dataset for baselines and our *PerSyn* strategy in instruction tuning setting. For math reasoning setting, we construct ***PerSyn-Math*** dataset, which provides 10K samples with 15 parallel teacher responses. We use ***PerSyn-Math*** to construct training dataset for baselines and our *PerSyn* strategy in math reasoning setting. We use 2.5K samples with parallel teacher responses to build pairwise training data to obtain *PerSyn* router in both settings (as supported by § 3.3).

A.3 Datasets and Evaluation

Details of Data Synthesis. For instruction tuning, the Magpie-100K-Generator-Zoo dataset use greedy decoding to generate responses. For math reasoning, we set the temperature to 0.6 and use teacher models to synthesize solutions by rejection sampling. Specifically, we sample four outputs from models under 72B and two outputs from 72B or Long-CoT models. Each output is then verified using Math-Verify.[11] If at least one output is correct, we keep one correct solution; otherwise, we randomly keep one incorrect solution.

Datasets Overview. Table 10 presents an overview of the datasets used in our experiments. For instruction tuning setting, TruthfulQA is designed to measure whether a language model generate truthful answers. LiveBench is a monthly updated benchmark that tests LLMs on 18 diverse tasks across 6 categories.[12] IFEval evaluate LLMs' capability to follow automatically verifiable natural language instructions. For math reasoning, SVAMP and GSM8K are datasets of grade-school math word problems. And MATH is a more challenging benchmark that contains competition-level mathematics problems requiring complex reasoning and advanced knowledge.

Evaluation Setup. We assess the models on LiveBench with the official scripts[13], reporting scores averaged across six domains: math, coding, reasoning, language, data analysis, and instruction following. TruthfulQA and IFEval are evaluated using lm-evaluation-harness (Gao et al., 2024), a standard evaluation suite, and for IFEval we report instruction-level strict accuracy. In the math reasoning setting, we use accuracy as the primary metric, with correctness measured by math-verify. All benchmarks are tested in the zero-shot setting, except for GSM8K, which is evaluated in a 5-shot setting. To ensure reproducibility of our empirical results, we implement greedy decoding for all benchmarks.

A.4 Teacher Models of Two Settings

Table 11 presents an overview of the teacher models used in our two settings. The teacher model pool for instruction tuning includes 19 models from 6 different families, with sizes ranging from 1.5B to 405B. For math reasoning, we use 15 teacher models from 7 different families. Notably, beyond conventional teacher model choices, we also include models specialized for math reasoning, covering RL-trained reasoning models (Qwen3 and DeepSeek-R1), a backbone model pretrained on math data (Qwen2.5 Math), and a model distilled with Long-CoT rationales (DeepSeek-R1-Distill-Qwen-7B). It should be noted that the 37B DeepSeek-R1 model denotes its number of activated parameters, whereas the full model contains 685B parameters in total.

[11] https://github.com/huggingface/Math-Verify

[12] The release option of LiveBench we used in our experiment is 2024-11-25.

[13] https://github.com/LiveBench/LiveBench

Model Family	Model ID	Size
Qwen2	Qwen2-1.5B-Instruct	1.5B
	Qwen2-7B-Instruct	7B
	Qwen2-72B-Instruct	72B
Qwen2.5	Qwen2.5-3B-Instruct	3B
	Qwen2.5-7B-Instruct	7B
	Qwen2.5-14B-Instruct	14B
	Qwen2.5-32B-Instruct	32B
	Qwen2.5-72B-Instruct	72B
Llama 3	Llama-3-8B-Instruct	8B
	Llama-3-70B-Instruct	70B
Llama 3.1	Llama-3.1-8B-Instruct	8B
	Llama-3.1-70B-Instruct	70B
	Llama-3.1-405B-Instruct	405B
Gemma 2	Gemma-2-2b-it	2B
	Gemma-2-9b-it	9B
	Gemma-2-27b-it	27B
Phi-3	Phi-3-mini-128k-instruct	3.8B
	Phi-3-small-128k-instruct	7B
	Phi-3-medium-128k-instruct	14B

Model Family	Model ID	Size
Qwen2.5	Qwen2.5-7B-Instruct	7B
	Qwen2.5-14B-Instruct	14B
	Qwen2.5-32B-Instruct	32B
	Qwen2.5-72B-Instruct	72B
Qwen3	Qwen3-8B	8B
	Qwen3-14B	14B
Qwen2.5 Math	Qwen2.5-Math-7B-Instruct	7B
Llama 3.1/3.3	Llama-3.1-8B-Instruct	8B
	Llama-3.3-70B-Instruct	70B
Gemma 2	Gemma-2-9b-it	9B
	Gemma-2-27b-it	27B
Mistral	Mistral-7B-Instruct-v0.3	7B
	Mistral-Nemo-Instruct-2407	12B
Deepseek	DeepSeek-R1-Distill-Qwen-7B	7B
	DeepSeek-R1	37B

Table 11: The overview of teacher models we used in our experiments. For instruction tuning (left table), we directly use the teacher models defined in Magpie-Zoo dataset (Xu et al., 2025b). The right table presents the teacher models for math reasoning.

Student Model	Strategy	IFEval	TruthfulQA	LiveBench
Qwen2.5-7B	Strong	52.85	22.80	43.02
	Mix	53.66	23.50	44.26
	Family-Strong	55.43	25.10	46.15
	CAR	55.75	24.90	47.33
	PerSyn (Ours)	**57.12**	**26.20**	**49.08**
Llama-3.1-8B	Strong	47.71	14.80	32.87
	Mix	49.82	16.90	35.42
	Family-Strong	48.36	15.40	33.25
	CAR	50.23	17.30	35.86
	PerSyn (Ours)	**51.34**	**18.50**	**37.13**
Gemma-2-9B	Strong	45.68	13.60	33.49
	Mix	46.71	14.40	34.63
	Family-Strong	48.15	16.30	36.09
	CAR	48.52	16.70	36.63
	PerSyn (Ours)	**49.66**	**17.50**	**37.84**
Qwen2.5-14B	Strong	55.84	25.70	46.34
	Mix	57.26	27.10	48.85
	Family-Strong	59.07	28.40	50.59
	CAR	60.32	28.80	52.06
	PerSyn (Ours)	**62.15**	**29.50**	**53.47**

Table 12: Results of baseline methods and our *PerSyn* strategy evaluated on four larger-scale student models from different families. See § 3.1 for details about the *Strong*, *Mix*, *Family-Strong*, and *CAR* baselines.

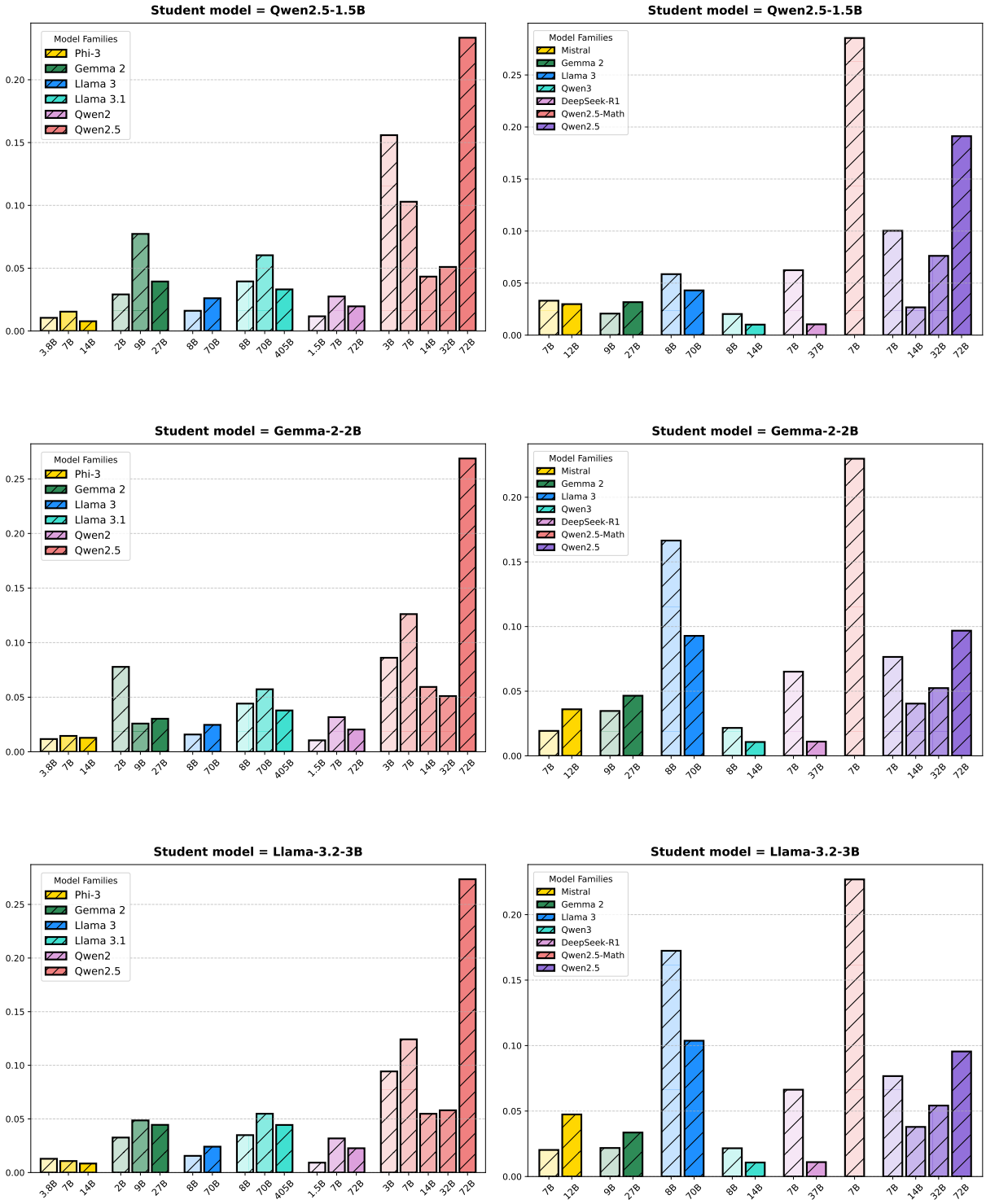

Figure 10: Prompt allocation ratios assigned by *PerSyn* across different teacher models for Qwen2.5-1.5B, Gemma-2-2B, and Llama-3.2-3B student models in the instruction tuning (left) and math reasoning (right) settings. Colors indicate different model families, and darker shades correspond to larger teacher models within the same family. Note that DeepSeek-R1 and Qwen3 are Long-CoT models, while the remaining are Short-CoT models.