ON-POLICY INSTRUCTION TUNING THROUGH FEW-SHOT SELF-DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Pretrained large language models are typically adapted into assistants through an initial phase of supervised fine-tuning (SFT) on instruction-response demonstrations. Recent work suggests that the on-policy nature of reinforcement learning can help preserve capabilities acquired during previous post-training phases, motivating the question of whether instruction tuning can similarly benefit from on-policy training. We investigate this question by introducing Few-Shot Self-Distillation (FSSD), a method for on-policy instruction tuning of pretrained base models. FSSD conditions a teacher copy of the base model on a small set of relevant demonstrations and uses the teacher to provide token-level supervision on trajectories generated by the student. The teacher does not receive the reference response for the current instruction; instead, few-shot conditioning provides the supervision used to train the student. Across three model scales, FSSD helps mitigate regressions on general-capability benchmarks relative to standard SFT, while achieving stronger open-ended assistant performance on AlpacaEval 2.0 and Arena-Hard v2 and inducing substantially less distributional shift, as measured by the KL divergence from the pretrained policy.

1 INTRODUCTION

Large language models (LLMs) have seen rapid adoption and are increasingly used as coding, data analysis, and general-purpose assistants (Chen et al., 2021; Ouyang et al., 2022; Amer-Yahia et al., 2025). The development of these models generally follows a common recipe (Grattafiori et al., 2024; Yang et al., 2025; DeepSeek-AI et al., 2025b; Lambert et al., 2025; Qwen Team, 2026; Team et al., 2026b; Liu et al., 2026; Team et al., 2026a). First, models are pretrained on large text corpora, typically drawn from the internet, in a fully self-supervised manner (Brown et al., 2020). They are then instruction-tuned to behave as assistants through supervised fine-tuning (SFT) on instruction-response demonstrations generated by humans or other models (Zhang et al., 2025). This stage is generally followed by Reinforcement Learning (RL), which is used to align models with human preferences (Ouyang et al., 2022; Bai et al., 2022), develop additional capabilities through verifiable rewards (DeepSeek-AI et al., 2025a; Shao et al., 2024; Lambert et al., 2025), and enable learning through environment interactions (Silver & Sutton, 2025).

In addition to improving model capabilities beyond what can be achieved through SFT alone (Shao et al., 2024; Gehring et al., 2025), RL has been observed to better preserve capabilities acquired during previous post-training phases than standard SFT (Shenfeld et al., 2025; Chen et al., 2025; Shenfeld et al., 2026). Prior work has associated this improved retention with the *on-policy* nature of reinforcement learning (Shenfeld et al., 2025; Chen et al., 2025).

Language models are pretrained on enormous amounts of data, allowing them to develop rich representations of the world (Brown et al., 2020; Gurnee & Tegmark, 2024) and acquire latent capabilities that post-training attempts to elicit (Yue et al., 2025). Yet, standard instruction tuning typically relies on SFT over *off-policy* demonstrations.

Motivated by this gap, we ask whether instruction tuning can instead be performed on-policy and what benefits it may provide over standard off-policy SFT.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

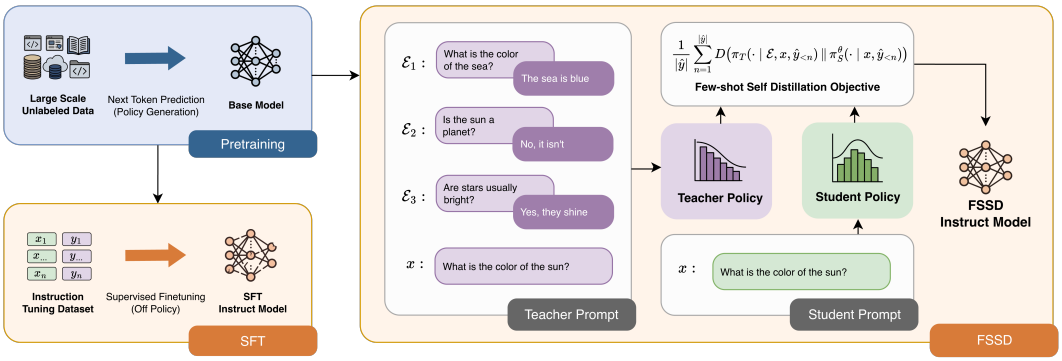


Figure 1: **Standard supervised instruction tuning and Few-Shot Self-Distillation (FSSD)**. Left: the standard pipeline, in which a pretrained base model is fine-tuned on fixed instruction-response demonstrations. Right: FSSD. For each instruction, the student generates an on-policy response, while a teacher copy of the same base model, conditioned on relevant few-shot demonstrations, provides token-level supervision along the student-generated trajectory.

To investigate this question, we introduce Few-Shot Self-Distillation (FSSD), an on-policy instruction-tuning method that builds on recent work on self-distillation (Zhao et al., 2026; Shenfeld et al., 2026; Ye et al., 2026b; Hübotter et al., 2026; Agrawal et al., 2026).

FSSD harnesses the few-shot learning capabilities of large language models (Brown et al., 2020; Lin et al., 2023) by conditioning a teacher copy of the base model on relevant demonstrations to elicit assistant behavior and distilling the resulting token-level distribution into the student (Figure 1).

Notably, this is achieved without providing the teacher with the reference response for the current instruction, showing that conditioning the teacher on few-shot demonstrations can provide an effective signal for training the student to exhibit assistant behavior.

We find that, across model scales, FSSD exhibits fewer regressions from the pretrained model on general-capability benchmarks, achieves stronger open-ended assistant capability than standard SFT on AlpacaEval 2.0 and Arena-Hard v2, and induces less distributional shift from the pretrained policy, as measured by Kullback-Leibler divergence.

Contributions. We introduce Few-Shot Self-Distillation (FSSD), an on-policy self-distillation method that distills behavior from a teacher conditioned on few-shot demonstrations, without providing the teacher with the reference response to the given instruction. Across Qwen3 base models with 1.7B, 4B, and 8B parameters, we show that FSSD enables on-policy instruction tuning of pretrained base models, achieving stronger open-ended assistant performance than standard SFT, mitigating regressions on the evaluated benchmarks, and exhibiting less distributional shift from the pretrained policy.

2 Background

Knowledge Distillation. Knowledge distillation refers to the process of transferring behavior from a teacher policy π_T to a student policy π_S^θ, enabling capabilities to be transferred from a stronger model to a weaker one (Hinton et al., 2015; Sanh et al., 2020). For autoregressive sequence models, distillation techniques traditionally rely on a dataset of fixed sequences or teacher-generated outputs. The teacher is then used to assign token-level probabilities, which are subsequently distilled into the student (Kim & Rush, 2016).

While off-policy distillation allows the weaker model to learn from the complete next-token distribution of the teacher, providing a richer training signal than hard labels alone, it suffers from distribution mismatch. The student learns from a dataset that may be out of distribution relative to the prefixes encountered during autoregressive generation. Because the student conditions on its own outputs, an early mistake may lead it to prefixes that were not represented during training, causing subsequent errors to compound (Pomerleau, 1991; Ross et al., 2011; Zhao et al., 2026).

On-Policy Distillation. On-policy distillation addresses this exposure-bias problem (Ross et al., 2011) by replacing fixed response trajectories with trajectories sampled from the student policy during training. Let $\mathcal{D} = \{(x, y^*)\}$ represent a dataset of inputs and reference responses. On-policy distillation can then be performed by minimizing:

$$\mathcal{L}_{\mathrm{OPD}}(\theta) = \mathbb{E}_{(x)\sim\mathcal{D}}\mathbb{E}_{\hat{y}\sim\pi_S^\theta(\cdot|x)} \frac{1}{|\hat{y}|} \sum_{n=1}^{|\hat{y}|} D\big(\pi_T(\cdot \mid x, \hat{y}_{<n}) \,\big\|\, \pi_S^\theta(\cdot \mid x, \hat{y}_{<n})\big) .$$

Here, D can be any divergence measure between the teacher and student distributions, such as reverse KL, forward KL, or the Jensen-Shannon divergence (Agarwal et al., 2024), and $\hat{y}$ denotes the response generated by the student policy. The teacher acts as a process supervisor, providing dense token-level feedback on the student's generations (Agarwal et al., 2024).

On-Policy Self-Distillation. Self-distillation further builds on on-policy distillation by using the same model as both the student and the teacher. The teacher is additionally conditioned on privileged information provided in the context, such as a ground-truth solution y^* or a reference chain of thought available in the dataset $\mathcal{D}$, while the student is conditioned only on the problem:

$$\mathcal{L}_{\mathrm{OPSD}}(\theta) = \mathbb{E}_{(x,y^*)\sim\mathcal{D}}\mathbb{E}_{\hat{y}\sim\pi_S^\theta(\cdot|x)} \frac{1}{|\hat{y}|} \sum_{n=1}^{|\hat{y}|} D\big(\pi_T(\cdot \mid x, y^*, \hat{y}_{<n}) \,\big\|\, \pi_S^\theta(\cdot \mid x, \hat{y}_{<n})\big) .$$

To encourage the teacher to use the ground-truth solution as privileged context rather than simply reproduce it, the teacher is instructed to *answer with a response of its own*. Prior work has found this simple instruction sufficient to prevent verbatim reproduction of the ground-truth solution when using post-trained models (Zhao et al., 2026; Shenfeld et al., 2026).

This approach is inspired by human learning, in which a student who fails to solve a problem can examine the solution, reconsider their previous answer, and identify the source of their mistakes. In this way, self-distillation effectively allows the model to act as its own teacher (Zhao et al., 2026; Ye et al., 2026b; Shenfeld et al., 2026).

3 METHOD

Our approach builds on the self-distillation framework (Zhao et al., 2026; Shenfeld et al., 2026), in which a model is distilled from a copy of itself conditioned on privileged context, such as ground-truth solutions. Building on the insight that the on-policy nature of RL helps preserve previously acquired capabilities (Shenfeld et al., 2025; Chen et al., 2025), we investigate the effects of on-policy instruction tuning immediately after pretraining.

Although base models have been pretrained on enormous amounts of data (Brown et al., 2020), they have not yet been trained to act as assistants. As a result, they do not reliably exhibit the instruction-following behavior expected of post-trained models (Ouyang et al., 2022).

This limits the direct applicability of existing self-distillation approaches to base models. In particular, we observe that when the teacher is conditioned on the reference response associated with the current instruction, base models often reproduce that response verbatim, even when explicitly instructed to respond in their own words (Appendix D.1).

We therefore seek an alternative conditioning signal that can elicit assistant behavior from the teacher without exposing it to the reference response. Given the strong in-context learning capabilities of pretrained language models (Brown et al., 2020; Lin et al., 2023), we hypothesize that conditioning the teacher on relevant few-shot demonstrations may provide sufficient guidance for instruction-tuning the student without exposing the teacher to the current example's reference response.

This motivates Few-Shot Self-Distillation, in which the teacher is conditioned on few-shot demonstrations rather than the reference response associated with the current instruction (Algorithm 1).

Instruction-tuning datasets typically cover a broad range of tasks and target behaviors (Lambert et al., 2025; Allal et al., 2025; Xu et al., 2024; Djuhera et al., 2025), as such diversity has been shown to improve generalization (Wang et al., 2022; Wei et al., 2022).

Algorithm 1 Few-Shot Self-Distillation (FSSD)

Require: Dataset $\mathcal{D} = \{(x_i, y_i^*)\}_{i=1}^N$; category $c(x) \in \{1, \ldots, M\}$; number of shots K; $\mathcal{D}_c = \{(x_i, y_i^*) \in \mathcal{D} : c(x_i) = c\}$; model π_θ; divergence D (e.g., Forward KL)

Let π_S^θ and π_T be initialized from the same model π_θ, with π_T additionally conditioned on few-shot demonstrations.
while not converged **do**
  Sample a minibatch $\mathcal{B} \subset \mathcal{D}$
  for all $(x, y^*) \in \mathcal{B}$ **do**
    Sample K demonstrations $\mathcal{E} = \{(x_j, y_j^*)\}_{j=1}^K$ from $\mathcal{D}_{c(x)} \setminus \{(x, y^*)\}$
    Sample on-policy response $\hat{y} \sim \pi_S^\theta(\cdot \mid x)$
    $\ell(x) \leftarrow \frac{1}{|\hat{y}|} \sum_{n=1}^{|\hat{y}|} D\big(\pi_T(\cdot \mid \mathcal{E}, x, \hat{y}_{<n}) \,\|\, \pi_S^\theta(\cdot \mid x, \hat{y}_{<n})\big)$
  $\mathcal{L}_{\text{FSSD}}(\theta) \leftarrow \frac{1}{|\mathcal{B}|} \sum_{(x,y^*)\in\mathcal{B}} \ell(x)$; update θ

To elicit the desired behavior, it is important to provide the teacher with few-shot demonstrations that are relevant to the instruction given to the student (Liu et al., 2021). For example, if the student is required to solve a mathematical problem, few-shot demonstrations involving role-playing or advice seeking are likely to provide a weaker conditioning signal for step-by-step mathematical reasoning. Therefore, we sample few-shot demonstrations from the same task category (Xu et al., 2024) as the instruction given to the student.

4 Experiments

4.1 Experimental Setup

Models and Datasets. Following Zhao et al. (2026), we study open-weight Qwen3 models at 1.7B, 4B, and 8B parameters. Specifically, we use the base models Qwen3-1.7B-Base, Qwen3-4B-Base, and Qwen3-8B-Base. We train on 50k high-quality data points from the annotated version of the Tulu3-sft-mixture (Lambert et al., 2025; Djuhera et al., 2025). We sample the 50k data points using the dataset curation recipe defined by Djuhera et al. (2025), while maintaining a task-category distribution similar to that of the original dataset (Figure 2). Further details on the dataset curation recipe are provided in Appendix B.

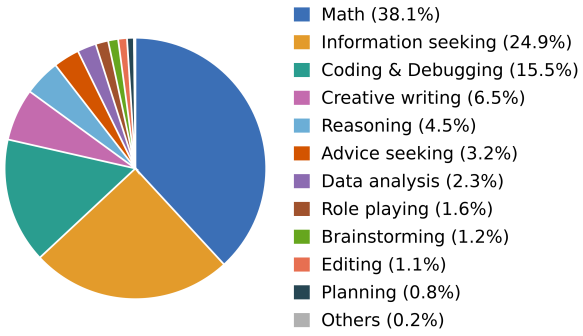


Figure 2: Task-category distribution of the instruction-tuning dataset.

Baselines and Evaluation. We compare FSSD with SFT and On-Policy Self-Distillation (OPSD) (Zhao et al., 2026; Shenfeld et al., 2026), and report results for the corresponding pretrained base models. We evaluate our models on standard general-capability benchmarks (Zhang et al., 2025; Lambert et al., 2025; Djuhera et al., 2025). Specifically, we evaluate on ARC Challenge (Clark et al., 2018), BBH (Suzgun et al., 2022), GPQA (Rein et al., 2023), Math-Hard (Hendrycks et al., 2021), MuSR (Sprague et al., 2024), MMLU-Pro (Wang et al., 2024), WinoGrande (Sakaguchi et al., 2019), and HumanEval (HE) (Chen et al., 2021) using the lm-evaluation-harness framework (Gao et al., 2024) in a zero-shot setting. We also evaluate open-ended assistant performance using the length-controlled AlpacaEval 2.0 (Dubois et al., 2025) and Arena-Hard v2 (Li et al., 2024), with

GPT-4.1 as the judge. For Arena-Hard v2, we evaluate against Qwen2.5-3B-Instruct on the hard-prompt subset, consisting of challenging real-world user queries.

Implementation Details. Following Zhao et al. (2026), we use the forward KL divergence for on-policy distillation. We keep the teacher's parameters fixed at their pretrained values, as this has been shown to stabilize training and act as an implicit regularizer (Zhao et al., 2026). We condition the teacher on three few-shot demonstrations, following Lin et al. (2023), who found that three demonstrations provide a favorable trade-off between computational cost and performance. We additionally report an ablation on the number of few-shot demonstrations in Section 5. All experiments are conducted using LoRA (Hu et al., 2021). Further details on the hyperparameters are provided in Appendix F.

4.2 RESULTS

FSSD mitigates regressions on general-capability benchmarks relative to SFT while improving open-ended assistant performance. Across all three model scales, FSSD exhibits fewer and generally smaller regressions from the pretrained base model than standard SFT, while achieving stronger performance on open-ended assistant benchmarks.

On the eight benchmarks in Table 1, SFT consistently improves HumanEval, but these gains are accompanied by lower scores than the base model on several other benchmarks. Across the three model scales, SFT falls below the corresponding base model on 13 of 24 benchmark-scale combinations, compared with 5 of 24 for FSSD. This pattern holds at each scale: SFT produces lower scores on 3, 6, and 4 of the eight benchmarks at 1.7B, 4B, and 8B, respectively, compared with 1, 3, and 1 for FSSD. Consistent with this trend, FSSD also improves average performance relative to the base model at all three scales, whereas SFT yields little change in average benchmark performance.

At the same time, these smaller benchmark regressions are accompanied by stronger performance on open-ended assistant benchmarks. As shown in Table 2, FSSD achieves higher scores than both SFT and OPSD on AlpacaEval 2.0 and Arena-Hard v2 at all three model sizes. Relative to SFT, FSSD increases AlpacaEval 2.0 scores by 3.49, 8.02, and 2.32 percentage points for the 1.7B, 4B, and 8B models, respectively, and Arena-Hard v2 scores by 6.5, 7.9, and 2.0 percentage points.

Together, these results show that FSSD improves performance on open-ended assistant benchmarks while mitigating the general-capability regressions observed under standard SFT.

	ARC Challenge	BBH	GPQA	Math Hard	MuSR	MMLU Pro	WinoGrande	HE	Avg
Qwen3-1.7B (Base)	$37.6_{\pm 1.4}$	$34.2_{\pm 0.6}$	$\mathbf{28.6}_{\pm 1.3}$	$30.3_{\pm 1.2}$	$\underline{39.9}_{\pm 1.7}$	$15.4_{\pm 0.3}$	$58.7_{\pm 1.4}$	$52.4_{\pm 3.9}$	37.1
+SFT	$37.7_{\pm 1.4}$	$31.3_{\pm 0.6}$	$27.2_{\pm 1.3}$	$32.1_{\pm 1.2}$	$38.6_{\pm 1.7}$	$15.5_{\pm 0.3}$	$59.9_{\pm 1.4}$	$53.6_{\pm 3.9}$	37.0
+OPSD	$\underline{38.4}_{\pm 1.4}$	$\underline{34.7}_{\pm 0.6}$	$\mathbf{28.6}_{\pm 1.3}$	$\underline{32.6}_{\pm 1.2}$	$\mathbf{40.3}_{\pm 1.8}$	$\underline{16.3}_{\pm 0.3}$	$\underline{60.3}_{\pm 1.4}$	$\underline{55.4}_{\pm 3.9}$	$\underline{38.3}$
+FSSD	$\mathbf{41.2}_{\pm 1.4}$	$\mathbf{34.8}_{\pm 0.6}$	$\underline{27.3}_{\pm 1.3}$	$\mathbf{33.3}_{\pm 1.2}$	$\mathbf{40.3}_{\pm 1.7}$	$\mathbf{16.6}_{\pm 0.3}$	$\mathbf{61.8}_{\pm 1.4}$	$\mathbf{62.2}_{\pm 3.8}$	**39.7**
Qwen3-4B (Base)	$\underline{41.8}_{\pm 1.4}$	$42.5_{\pm 0.6}$	$\mathbf{30.9}_{\pm 1.3}$	$39.5_{\pm 1.2}$	$\mathbf{41.2}_{\pm 1.8}$	$\mathbf{16.3}_{\pm 0.3}$	$64.9_{\pm 1.3}$	$63.4_{\pm 3.8}$	42.6
+SFT	$39.8_{\pm 1.4}$	$41.5_{\pm 0.6}$	$28.1_{\pm 1.3}$	$33.9_{\pm 1.2}$	$38.3_{\pm 1.7}$	$15.1_{\pm 0.3}$	$65.5_{\pm 1.3}$	$\mathbf{73.7}_{\pm 3.4}$	42.0
+OPSD	$40.7_{\pm 1.4}$	$\underline{42.8}_{\pm 0.6}$	$29.9_{\pm 1.3}$	$\underline{45.2}_{\pm 1.2}$	$39.6_{\pm 1.8}$	$16.0_{\pm 0.3}$	$\mathbf{66.5}_{\pm 1.3}$	$71.3_{\pm 3.5}$	$\underline{44.0}$
+FSSD	$\mathbf{42.3}_{\pm 1.4}$	$\mathbf{43.4}_{\pm 0.6}$	$\underline{30.6}_{\pm 1.3}$	$\mathbf{45.6}_{\pm 1.2}$	$\underline{41.0}_{\pm 1.8}$	$\underline{16.2}_{\pm 0.3}$	$\underline{66.2}_{\pm 1.3}$	$\underline{71.9}_{\pm 3.5}$	**44.6**
Qwen3-8B (Base)	$42.9_{\pm 1.4}$	$44.4_{\pm 0.6}$	$\underline{32.7}_{\pm 1.4}$	$37.7_{\pm 1.2}$	$\mathbf{39.6}_{\pm 1.8}$	$\mathbf{17.8}_{\pm 0.3}$	$71.5_{\pm 1.3}$	$73.7_{\pm 3.4}$	45.0
+SFT	$43.0_{\pm 1.4}$	$\mathbf{44.6}_{\pm 0.6}$	$29.7_{\pm 1.3}$	$40.2_{\pm 1.2}$	$37.0_{\pm 1.7}$	$17.2_{\pm 0.3}$	$71.1_{\pm 1.3}$	$\underline{78.6}_{\pm 3.2}$	45.2
+OPSD	$\underline{43.6}_{\pm 1.4}$	$44.3_{\pm 0.6}$	$32.1_{\pm 1.3}$	$\underline{46.1}_{\pm 1.2}$	$\underline{39.1}_{\pm 1.7}$	$\underline{17.6}_{\pm 0.3}$	$\underline{71.6}_{\pm 1.3}$	$73.7_{\pm 3.4}$	$\underline{46.0}$
+FSSD	$\mathbf{43.9}_{\pm 1.4}$	$\mathbf{44.6}_{\pm 0.6}$	$\mathbf{32.8}_{\pm 1.4}$	$\mathbf{48.8}_{\pm 1.2}$	$39.0_{\pm 1.7}$	$\mathbf{17.8}_{\pm 0.3}$	$\mathbf{71.9}_{\pm 1.3}$	$\mathbf{79.2}_{\pm 3.2}$	**47.2**

Table 1: **General-capability performance after instruction tuning**. Results for Qwen3-1.7B, 4B, and 8B base models before training and after SFT, OPSD, or FSSD across eight standard benchmarks. Scores are percentages, and Avg denotes the mean across the eight benchmarks. Best result within each model size is shown in bold and second-best is underlined. We report mean ± SEM over evaluation examples.

FSSD induces lower KL divergence from the pretrained policy than SFT. To complement these results, we examine the distributional shift induced by instruction tuning, as prior work has linked capability degradation during post-training to the Kullback-Leibler divergence between the base and fine-tuned policies (Shenfeld et al., 2025; 2026).

	AlpacaEval 2.0	Arena-Hard v2
Qwen3-1.7B (Base)	$5.77_{\pm 0.79}$	$28.8_{\pm 1.1}$
+SFT	$5.23_{\pm 0.79}$	$\underline{32.7}_{\pm 1.2}$
+OPSD	$\underline{6.70}_{\pm 0.88}$	$29.6_{\pm 1.1}$
+FSSD	$\mathbf{8.72}_{\pm 0.99}$	$\mathbf{39.2}_{\pm 1.1}$
Qwen3-4B (Base)	$11.27_{\pm 1.05}$	$61.4_{\pm 1.3}$
+SFT	$8.37_{\pm 0.99}$	$62.4_{\pm 1.0}$
+OPSD	$\underline{14.80}_{\pm 1.22}$	$\underline{63.5}_{\pm 1.2}$
+FSSD	$\mathbf{16.39}_{\pm 1.27}$	$\mathbf{70.3}_{\pm 1.1}$
Qwen3-8B (Base)	$11.70_{\pm 1.04}$	$66.3_{\pm 1.4}$
+SFT	$\underline{12.99}_{\pm 1.15}$	$\underline{70.2}_{\pm 1.2}$
+OPSD	$11.90_{\pm 1.08}$	$68.7_{\pm 0.9}$
+FSSD	$\mathbf{15.31}_{\pm 1.18}$	$\mathbf{72.2}_{\pm 1.0}$

Table 2: **Open-ended assistant performance after instruction tuning**. AlpacaEval 2.0 and Arena-Hard v2 results for Qwen3-1.7B, 4B, and 8B before training and after SFT, OPSD, or FSSD. Scores are percentages. Best result within each model size is shown in bold and second-best is underlined. We report mean ± SEM over evaluation examples.

We measure this divergence on a held-out set of 500 examples and find that both FSSD and OPSD remain substantially closer to the pretrained policy than SFT across all three model sizes (Figure 3). At the same time, FSSD outperforms SFT on both open-ended assistant benchmarks at every model scale. For example, for Qwen3-4B, FSSD yields a mean KL divergence of 0.020, approximately one-third of the 0.063 observed for SFT.

Together with the benchmark results, these findings show that FSSD improves assistant behavior while inducing a substantially smaller distributional shift from the pretrained policy than SFT. We further support this finding with an ablation in Section 5, where we isolate training on individual task categories.

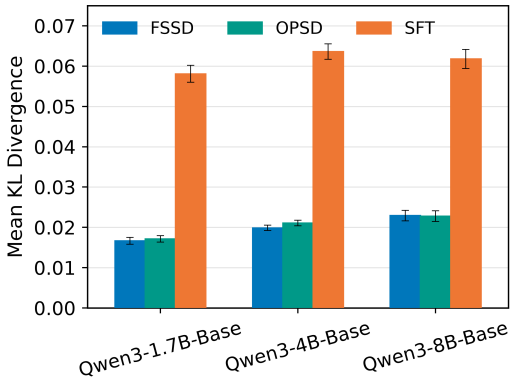


Figure 3: **Divergence from the pretrained policy after instruction tuning**. Mean KL divergence, $D_{\mathrm{KL}}(\pi_{\text{tuned}} \,\|\, \pi_{\text{base}})$, between each fine-tuned policy and its corresponding base policy on a held-out set of 500 examples, across Qwen3-1.7B, 4B, and 8B. FSSD and OPSD remain substantially closer to the pretrained policy than standard SFT. Error bars indicate ± SEM.

5 Ablations

Catastrophic Forgetting from Individual Task-Category Training. Prior work has shown that supervised fine-tuning on narrow-domain data can lead to catastrophic forgetting of out-of-domain capabilities (Kumar et al., 2022; Shenfeld et al., 2025; Chen et al., 2025), and that the degree of forgetting can be linked to the KL divergence from the original policy (Shenfeld et al., 2025; 2026).

To further examine the advantages of FSSD, we consider a deliberately narrow instruction-tuning regime in which models are trained exclusively on examples from a single task category and evaluated on an out-of-domain capability.

Specifically, we train Qwen3-1.7B-Base using FSSD and SFT on the two largest non-mathematical task categories in our dataset, 'Information Seeking' and 'Coding & Debugging', and evaluate the resulting models on Math-Hard. We additionally include an SFT setting with three times as many passes over the training data.

Method	Math-Hard	KL div
Information Seeking		
Base	$30.3_{\pm 1.2}$	$0.000_{\pm 0.000}$
SFT	$24.0_{\pm 1.1}$	$0.037_{\pm 0.002}$
SFT (3×)	$12.9_{\pm 0.9}$	$0.048_{\pm 0.002}$
FSSD	$\mathbf{32.5}_{\pm 1.2}$	$\mathbf{0.009}_{\pm 0.001}$
Coding & Debugging		
Base	$30.3_{\pm 1.2}$	$0.000_{\pm 0.000}$
SFT	$24.7_{\pm 1.1}$	$0.027_{\pm 0.002}$
SFT (3×)	$25.4_{\pm 1.1}$	$0.042_{\pm 0.002}$
FSSD	$\mathbf{32.1}_{\pm 1.2}$	$\mathbf{0.007}_{\pm 0.001}$

Table 3: Catastrophic forgetting induced by training on a subset from a single task category. Mean ± SEM.

We find that restricting training to the 'Information Seeking' subset leads to substantial forgetting on Math-Hard under SFT, with performance decreasing by approximately 6 points relative to the base model and by nearly 18 points when SFT makes three times as many passes over the data. In contrast, FSSD does not exhibit this degradation and instead yields a small improvement over the base model. The greater degree of forgetting under SFT is also associated with substantially higher KL divergence from the pretrained policy; in the 3× SFT setting, the divergence is more than five times that observed with FSSD (Table 3).

Similar results are observed when training on the 'Coding & Debugging' task category, where SFT again leads to greater forgetting and higher KL divergence than FSSD.

Our findings further support the hypothesis introduced by Shenfeld et al. (2025) that the KL divergence from the pretrained policy can be predictive of the degree to which out-of-domain capabilities are forgotten.

Number of Few-Shot Demonstrations. We study the effect of varying the number of few-shot demonstrations used to condition the teacher (Figure 4). Specifically, using Qwen3-1.7B-Base, we vary the number of teacher demonstrations over 0, 1, 3, 5 and evaluate the resulting models on Arena-Hard v2.

We find that even a single demonstration yields a substantial improvement over both the base model and the zero-demonstration setting, while performance begins to plateau beyond 3 demonstrations. This trend is consistent with the findings of Lin et al. (2023), who identify 3 demonstrations as a favorable trade-off between computational cost and performance.

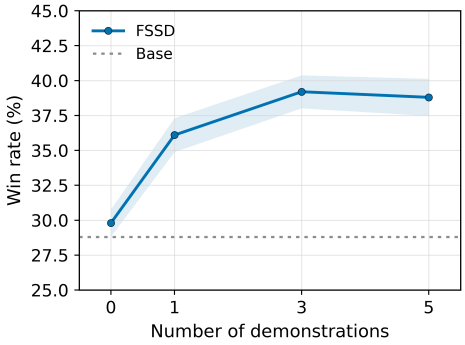


Figure 4: Effect of the number of teacher demonstrations on FSSD. Shaded regions indicate ± SEM.

Reference Response vs. Related Demonstration. We isolate the effect of conditioning the teacher on the reference response associated with the training example versus a related demonstration. Specifically, we train Qwen3-1.7B-Base with FSSD using either a single related demonstration or the corresponding reference response for the given instruction as conditioning context.

Conditioning the teacher on a related demonstration yields a higher Arena-Hard v2 score than conditioning it on the reference response, improving the Arena-Hard v2 score from 29.6 to 36.1 (Table 4).

Method Variant	Arena-Hard v2
Teacher Conditioning	
Base	$28.8_{\pm 1.1}$
Reference Response	$29.6_{\pm 1.1}$
Related Demonstration	$\mathbf{36.1}_{\pm 1.2}$
Demonstration Selection	
Base	$28.8_{\pm 1.1}$
Random	$36.0_{\pm 1.2}$
Category-Aware	$\mathbf{39.2}_{\pm 1.1}$

Table 4: Effect of teacher conditioning context and category-aware demonstration selection in FSSD. Mean ± SEM.

One explanation is that, when a pretrained teacher policy is provided with the reference response in its context, it tends to reproduce that response

verbatim (Appendix D.1). As a consequence, reference-response conditioning may provide a weaker instruction-tuning signal than conditioning on a related, non-reference demonstration.

Category-Aware Demonstration Selection. We study whether selecting teacher demonstrations that are relevant to the current instruction improves FSSD. Using Qwen3-1.7B-Base, we compare our default category-aware selection, which samples demonstrations from the same task category as the current instruction, against random selection from the full training set.

FSSD remains effective under random selection (Table 4), improving Arena-Hard v2 performance from 28.8 for the base model to 36.0. Restricting demonstrations to the same task category further improves performance to 39.2, suggesting that task-relevant demonstrations provide a more effective conditioning signal for the teacher.

Supervised Fine-Tuning on Self-Generated Responses. Prior work suggests that fine-tuning on model-generated responses can reduce the distribution gap between the training data and the model being fine-tuned (Yang et al., 2024), thereby reducing the degree of distributional shift induced during fine-tuning (Zhang et al., 2026).

We therefore ask whether FSSD's gains can be explained by replacing target responses in the dataset with responses generated by the pretrained model conditioned on FSSD's few-shot demonstrations.

On Qwen3-4B-Base, training on self-generated responses improves open-ended assistant performance, increasing the AlpacaEval 2.0 score from 8.37 under standard SFT to 13.35 and Arena-Hard v2 score from 62.4 to 66.2, while reducing the KL divergence from the pretrained policy from 0.063 to 0.047 (Table 5).

However, this approach still underperforms FSSD, which achieves 16.39 on AlpacaEval 2.0 and 70.3 on Arena-Hard v2 with a substantially lower KL divergence of 0.020. These results suggest that replacing target responses with model-generated responses recovers part of the improvement in assistant performance and reduction in policy shift observed with FSSD.

On-Policy vs. Off-Policy FSSD. To isolate the effect of on-policy training, we compare FSSD with an off-policy variant in which responses are generated once from the pretrained base model and then kept fixed throughout training, while the teacher is conditioned on the same few-shot demonstrations as in FSSD. Although we refer to this setting as off-policy, it can more precisely be viewed as quasi-on-policy: the fixed responses are initially sampled from the student policy, but become increasingly off-policy as the student policy evolves during training.

Method	AlpacaEval 2.0	Arena-Hard v2	KL div
Base	$11.27_{\pm 1.05}$	$61.4_{\pm 1.3}$	$0.000_{\pm 0.000}$
SFT	$8.37_{\pm 0.99}$	$62.4_{\pm 1.0}$	$0.063_{\pm 0.002}$
Self-gen	$13.35_{\pm 1.13}$	$66.2_{\pm 1.2}$	$0.047_{\pm 0.001}$
Off-FSSD	$14.64_{\pm 1.15}$	$65.6_{\pm 1.1}$	$0.021_{\pm 0.001}$
FSSD	$\mathbf{16.39}_{\pm 1.27}$	$\mathbf{70.3}_{\pm 1.1}$	$0.020_{\pm 0.001}$

Table 5: Comparison with supervised fine-tuning on self-generated responses and off-policy FSSD. Mean ± SEM.

Off-FSSD achieves a KL divergence from the pretrained policy similar to that of FSSD, but yields weaker open-ended assistant performance, scoring 14.64 vs. 16.39 on AlpacaEval 2.0 and 65.6 vs. 70.3 on Arena-Hard v2 (Table 5).

These results suggest that training on trajectories sampled from the current student policy contributes to FSSD's performance gains beyond the effect of few-shot teacher conditioning alone.

6 Related Work

On-Policy Distillation. In on-policy distillation, a student model is trained using a more capable teacher model that provides token-level supervision on the student's own generations (Agarwal et al., 2024; Xu et al., 2025). This helps mitigate the exposure-bias problem that can affect off-policy distillation (Ross et al., 2011). Recent work builds on this line of research by using a copy of the same model as both the student and the teacher, while conditioning the teacher copy on privileged context (Zhao et al., 2026; Shenfeld et al., 2026; Ye et al., 2026a; Yang et al., 2026; Ye et al., 2026b).

Zhao et al. (2026) and Shenfeld et al. (2026) formulate the method by conditioning the teacher on ground-truth solutions and reasoning traces, and show that On-Policy Self-Distillation can better preserve capabilities acquired during previous post-training stages (Shenfeld et al., 2026). Ye et al. (2026b) use the same framework to distill self-generated experiential knowledge into the model, using solution traces obtained through interaction with an environment.

It has been observed that providing the ground-truth solutions and chains of thought to the teacher can result in a student that suppresses its epistemic verbalization (Kim et al., 2026) as it learns to imitate a confident reasoning style that suppresses information unavailable at inference time. Building on this observation, Pan et al. (2026) argue that privileged solutions induce a style drift favoring shorter and more assertive phrasing, and propose mitigating this drift by contrasting the teacher's dense supervision under privileged context with that obtained from an incorrect hint.

Hübotter et al. (2026) extend this framework by incorporating feedback from the environment into the teacher's context. Further work builds on this to enable continual personalization by aligning models through user interactions (Buening et al., 2026). Yang et al. (2026) integrate the self-distillation framework into GRPO (Shao et al., 2024), leading to more stable training. Agrawal et al. (2026) emphasize the importance of future credit assignment by propagating future expert-student disagreement back to earlier decisions. Liao et al. (2026) use teacher-generated interactions with the environment as prefixes for the student during the distillation process.

Instruction Tuning. Zhou et al. (2023) introduce the superficial alignment hypothesis, arguing that a model's knowledge and capabilities are learned almost entirely during pretraining and that very small instruction-tuning datasets can be sufficient to teach models to produce high-quality outputs.

Building on this idea, Hewitt et al. (2024) argue that only small changes to a language model's distribution are required to elicit assistant behavior and show that training on responses without paired instructions can be sufficient. Lin et al. (2023) further build on the superficial alignment hypothesis, showing that carefully crafted few-shot demonstrations can be sufficient to elicit instruction-tuned behavior through in-context learning. Ebouky et al. (2025) demonstrate another form of inference-time guidance without parameter updates using cognitive tools.

Zhang et al. (2026) argue that the pretrained model distribution should be carefully preserved during fine-tuning. The authors observe that, in instruction-tuning datasets, instructions are paired with responses that are often out of distribution with respect to the model being fine-tuned, potentially increasing the risk of distributional distortion. They then introduce a distributional alignment framework that compares data from different sources and selects those that are closest to the target model's pretrained distribution.

Our work further supports this line of research by showing that large language models can be instruction-tuned directly on-policy and that conditioning the teacher on few-shot demonstrations, without access to the current example's reference response, can support instruction tuning of the student.

7 Discussion And Conclusion

In this work, we introduce Few-Shot Self-Distillation (FSSD), a method for performing on-policy instruction tuning of pretrained language models. Rather than providing the teacher with the reference response for the current instruction, FSSD conditions the teacher on few-shot demonstrations and distills the resulting assistant behavior on-policy into the student, showing that conditioning the teacher on few-shot demonstrations can provide sufficient supervision for instruction tuning.

Across Qwen3 models ranging from 1.7B to 8B parameters, FSSD consistently achieves stronger open-ended assistant performance than standard SFT while exhibiting fewer regressions on the evaluated benchmarks and substantially lower KL divergence from the pretrained policy. More broadly, FSSD provides an on-policy alternative to standard SFT that converts the few-shot learning capabilities of large language models into a training signal for assistant behavior while inducing less distributional shift from the pretrained policy.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. On-Policy Distillation of Language Models: Learning from Self-Generated Mistakes, January 2024. URL http://arxiv.org/abs/2306.13649. arXiv:2306.13649 [cs.LG].

Rishabh Agrawal, Jacob Fein-Ashley, and Paria Rashidinejad. Reinforcement Learning from Rich Feedback with Distributional DAgger, June 2026. URL https://arxiv.org/abs/2606.05152v2.

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, et al. SmolLM2: When Smol Goes Big – Data-Centric Training of a Small Language Model, February 2025. URL http://arxiv.org/abs/2502.02737. arXiv:2502.02737 [cs.CL].

Sihem Amer-Yahia, Jasmina Bogojeska, Roberta Facchinetti, Valeria Franceschi, Aristides Gionis, Katja Hose, Georgia Koutrika, Roger Kouyos, Matteo Lissandrini, Silviu Maniu, Katsiaryna Mirylenka, Davide Mottin, Themis Palpanas, Mattia Rigotti, and Yannis Velegrakis. Towards Reliable Conversational Data Analytics, 2025. URL https://openproceedings.org/2025/conf/edbt/paper-160.pdf.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, et al. Constitutional AI: Harmlessness from AI Feedback, December 2022. URL http://arxiv.org/abs/2212.08073. arXiv:2212.08073 [cs].

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, et al. Language Models are Few-Shot Learners, July 2020. URL http://arxiv.org/abs/2005.14165. arXiv:2005.14165 [cs.CL].

Thomas Kleine Buening, Jonas Hübotter, Barna Pásztor, Idan Shenfeld, Giorgia Ramponi, and Andreas Krause. Aligning Language Models from User Interactions, February 2026. URL https://arxiv.org/abs/2603.12273v1.

Howard Chen, Noam Razin, Karthik Narasimhan, and Danqi Chen. Retaining by Doing: The Role of On-Policy Data in Mitigating Forgetting, October 2025. URL https://arxiv.org/abs/2510.18874v2.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, et al. Evaluating Large Language Models Trained on Code, July 2021. URL http://arxiv.org/abs/2107.03374. arXiv:2107.03374 [cs.LG].

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge, March 2018. URL http://arxiv.org/abs/1803.05457. arXiv:1803.05457 [cs.AI].

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, et al. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning, January 2025a. URL http://arxiv.org/abs/2501.12948. arXiv:2501.12948 [cs].

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, et al. DeepSeek-V3 Technical Report, February 2025b. URL http://arxiv.org/abs/2412.19437. arXiv:2412.19437 [cs.CL].

Aladin Djuhera, Swanand Ravindra Kadhe, Syed Zawad, Farhan Ahmed, Heiko Ludwig, and Holger Boche. Fixing It in Post: A Comparative Study of LLM Post-Training Data Quality and Model Performance, December 2025. URL http://arxiv.org/abs/2506.06522. arXiv:2506.06522 [cs.CL].

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B. Hashimoto. Length-Controlled AlpacaEval: A Simple Way to Debias Automatic Evaluators, March 2025. URL http://arxiv.org/abs/2404.04475. arXiv:2404.04475 [cs.LG].

Brown Ebouky, Andrea Bartezzaghi, and Mattia Rigotti. Eliciting Reasoning in Language Models with Cognitive Tools, November 2025. URL http://arxiv.org/abs/2506.12115. arXiv:2506.12115 [cs].

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, et al. The language model evaluation harness, 2024. URL https://zenodo.org/records/12608602.

Jonas Gehring, Kunhao Zheng, Jade Copet, Vegard Mella, Quentin Carbonneaux, Taco Cohen, and Gabriel Synnaeve. RLEF: Grounding Code LLMs in Execution Feedback with Reinforcement Learning, February 2025. URL http://arxiv.org/abs/2410.02089. arXiv:2410.02089 [cs.CL].

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, et al. The Llama 3 Herd of Models, November 2024. URL http://arxiv.org/abs/2407.21783. arXiv:2407.21783 [cs.AI].

Wes Gurnee and Max Tegmark. Language Models Represent Space and Time, March 2024. URL http://arxiv.org/abs/2310.02207. arXiv:2310.02207 [cs.LG].

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring Mathematical Problem Solving With the MATH Dataset, November 2021. URL http://arxiv.org/abs/2103.03874. arXiv:2103.03874 [cs].

John Hewitt, Nelson F. Liu, Percy Liang, and Christopher D. Manning. Instruction Following without Instruction Tuning, September 2024. URL http://arxiv.org/abs/2409.14254. arXiv:2409.14254 [cs.CL].

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the Knowledge in a Neural Network, March 2015. URL http://arxiv.org/abs/1503.02531. arXiv:1503.02531 [stat.ML].

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models, October 2021. URL http://arxiv.org/abs/2106.09685. arXiv:2106.09685 [cs].

Jonas Hübotter, Frederike Lübeck, Lejs Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, and Andreas Krause. Reinforcement Learning via Self-Distillation, January 2026. URL http://arxiv.org/abs/2601.20802. arXiv:2601.20802 [cs].

Jeonghye Kim, Xufang Luo, Minbeom Kim, Sangmook Lee, Dohyung Kim, Jiwon Jeon, Dongsheng Li, and Yuqing Yang. Why Does Self-Distillation (Sometimes) Degrade the Reasoning Capability of LLMs?, May 2026. URL http://arxiv.org/abs/2603.24472. arXiv:2603.24472 [cs.CL].

Yoon Kim and Alexander M. Rush. Sequence-Level Knowledge Distillation, September 2016. URL http://arxiv.org/abs/1606.07947. arXiv:1606.07947 [cs.CL].

Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. Fine-Tuning can Distort Pretrained Features and Underperform Out-of-Distribution, February 2022. URL http://arxiv.org/abs/2202.10054. arXiv:2202.10054 [cs.LG].

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, et al. Tulu 3: Pushing Frontiers in Open Language Model Post-Training, April 2025. URL http://arxiv.org/abs/2411.15124. arXiv:2411.15124 [cs.CL].

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E. Gonzalez, and Ion Stoica. From Crowdsourced Data to High-Quality Benchmarks: Arena-Hard and BenchBuilder Pipeline, October 2024. URL http://arxiv.org/abs/2406.11939. arXiv:2406.11939 [cs.LG].

Baohao Liao, Hanze Dong, Christof Monz, Xinxing Xu, Li Dong, and Furu Wei. Multi-Turn On-Policy Distillation with Prefix Replay, July 2026. URL http://arxiv.org/abs/2607.04763. arXiv:2607.04763 [cs.LG].

Bill Yuchen Lin, Abhilasha Ravichander, Ximing Lu, Nouha Dziri, Melanie Sclar, Khyathi Chandu, Chandra Bhagavatula, and Yejin Choi. The Unlocking Spell on Base LLMs: Rethinking Alignment via In-Context Learning, December 2023. URL http://arxiv.org/abs/2312.01552. arXiv:2312.01552 [cs.CL].

Alexander H. Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, Alexandre Sablayrolles, Amélie Héliou, Amos You, Andy Ehrenberg, Andy Lo, et al. Ministral 3, January 2026. URL http://arxiv.org/abs/2601.08584. arXiv:2601.08584 [cs.CL].

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. What Makes Good In-Context Examples for GPT-3?, January 2021. URL http://arxiv.org/abs/2101.06804. arXiv:2101.06804 [cs.CL].

Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization, January 2019. URL http://arxiv.org/abs/1711.05101. arXiv:1711.05101 [cs.LG].

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, et al. Training language models to follow instructions with human feedback, March 2022. URL http://arxiv.org/abs/2203.02155. arXiv:2203.02155 [cs].

Leyi Pan, Shuchang Tao, Yunpeng Zhai, Lingzhe Zhang, Zhaoyang Liu, Bolin Ding, Aiwei Liu, and Lijie Wen. RLCSD: Reinforcement Learning with Contrastive On-Policy Self-Distillation, June 2026. URL http://arxiv.org/abs/2606.11709. arXiv:2606.11709 [cs.LG].

Dean A. Pomerleau. Efficient Training of Artificial Neural Networks for Autonomous Navigation. *Neural Computation*, 3(1):88–97, March 1991. ISSN 0899-7667. doi: 10.1162/neco.1991.3.1.88. URL https://ieeexplore.ieee.org/document/6796843.

Qwen Team. Qwen3.5-Omni Technical Report, April 2026. URL http://arxiv.org/abs/2604.15804. arXiv:2604.15804 [cs.CL].

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A Graduate-Level Google-Proof Q&A Benchmark, November 2023. URL http://arxiv.org/abs/2311.12022. arXiv:2311.12022 [cs].

Stephane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning, March 2011. URL http://arxiv.org/abs/1011.0686. arXiv:1011.0686 [cs.LG].

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An Adversarial Winograd Schema Challenge at Scale, November 2019. URL http://arxiv.org/abs/1907.10641. arXiv:1907.10641 [cs.CL].

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter, March 2020. URL http://arxiv.org/abs/1910.01108. arXiv:1910.01108 [cs.CL].

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models, April 2024. URL http://arxiv.org/abs/2402.03300. arXiv:2402.03300 [cs].

Idan Shenfeld, Jyothish Pari, and Pulkit Agrawal. RL's Razor: Why Online Reinforcement Learning Forgets Less, September 2025. URL https://arxiv.org/abs/2509.04259v1.

Idan Shenfeld, Mehul Damani, Jonas Hübotter, and Pulkit Agrawal. Self-Distillation Enables Continual Learning, January 2026. URL http://arxiv.org/abs/2601.19897. arXiv:2601.19897 [cs.LG].

David Silver and Richard S Sutton. Welcome to the Era of Experience. In George Konidaris (ed.), *Upcoming chapter of Designing an intelligence*. MIT Press, 2025.

Zayne Sprague, Xi Ye, Kaj Bostrom, Swarat Chaudhuri, and Greg Durrett. MuSR: Testing the Limits of Chain-of-thought with Multistep Soft Reasoning, March 2024. URL http://arxiv.org/abs/2310.16049. arXiv:2310.16049 [cs.CL].

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. Challenging BIG-Bench Tasks and Whether Chain-of-Thought Can Solve Them, October 2022. URL http://arxiv.org/abs/2210.09261. arXiv:2210.09261 [cs.CL].

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, M. C, Jianfeng Cai, Xinyuan Cai, Peizhou Cao, Yuxuan Cao, Ziwei Chai, Y. Charles, H. S. Che, Guanduo Chen, Guangyu Chen, Guanzheng Chen, et al. Kimi K3: Open Frontier Intelligence, August 2026a. URL http://arxiv.org/abs/2607.24653. arXiv:2607.24653 [cs.CL].

Kimi Team, Yifan Bai, Yiping Bao, Y. Charles, Cheng Chen, Guanduo Chen, Haiting Chen, Huarong Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, Zhuofu Chen, et al. Kimi K2: Open Agentic Intelligence, February 2026b. URL http://arxiv.org/abs/2507.20534. arXiv:2507.20534 [cs.LG].

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Gary Lai, Ishan Purohit, Ishani Mondal, et al. Super-NaturalInstructions: Generalization via Declarative Instructions on 1600+ NLP Tasks, October 2022. URL http://arxiv.org/abs/2204.07705. arXiv:2204.07705 [cs.CL].

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, et al. MMLU-Pro: A More Robust and Challenging Multi-Task Language Understanding Benchmark, November 2024. URL http://arxiv.org/abs/2406.01574. arXiv:2406.01574 [cs.CL].

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned Language Models Are Zero-Shot Learners, February 2022. URL http://arxiv.org/abs/2109.01652. arXiv:2109.01652 [cs.CL].

Wenda Xu, Rujun Han, Zifeng Wang, Long T. Le, Dhruv Madeka, Lei Li, William Yang Wang, Rishabh Agarwal, Chen-Yu Lee, and Tomas Pfister. Speculative Knowledge Distillation: Bridging the Teacher-Student Gap Through Interleaved Sampling, April 2025. URL http://arxiv.org/abs/2410.11325. arXiv:2410.11325 [cs.CL].

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. Magpie: Alignment Data Synthesis from Scratch by Prompting Aligned LLMs with Nothing, October 2024. URL http://arxiv.org/abs/2406.08464. arXiv:2406.08464 [cs.CL].

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, et al. Qwen3 Technical Report, May 2025. URL http://arxiv.org/abs/2505.09388. arXiv:2505.09388 [cs].

Chenxu Yang, Chuanyu Qin, Qingyi Si, Minghui Chen, Naibin Gu, Dingyu Yao, Zheng Lin, Weiping Wang, Jiaqi Wang, and Nan Duan. Self-Distilled RLVR, April 2026. URL http://arxiv.org/abs/2604.03128. arXiv:2604.03128 [cs.LG].

Zhaorui Yang, Tianyu Pang, Haozhe Feng, Han Wang, Wei Chen, Minfeng Zhu, and Qian Liu. Self-Distillation Bridges Distribution Gap in Language Model Fine-Tuning, May 2024. URL http://arxiv.org/abs/2402.13669. arXiv:2402.13669 [cs.CL].

Tianzhu Ye, Li Dong, Qingxiu Dong, Xun Wu, Shaohan Huang, and Furu Wei. Online Experiential Learning for Language Models, March 2026a. URL http://arxiv.org/abs/2603.16856. arXiv:2603.16856 [cs.CL].

Tianzhu Ye, Li Dong, Xun Wu, Shaohan Huang, and Furu Wei. On-Policy Context Distillation for Language Models, February 2026b. URL https://arxiv.org/abs/2602.12275v2.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does Reinforcement Learning Really Incentivize Reasoning Capacity in LLMs Beyond the Base Model?, November 2025. URL http://arxiv.org/abs/2504.13837. arXiv:2504.13837 [cs].

Dylan Zhang, Qirun Dai, and Hao Peng. The Best Instruction-Tuning Data are Those That Fit, January 2026. URL http://arxiv.org/abs/2502.04194. arXiv:2502.04194 [cs.CL].

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. Instruction Tuning for Large Language Models: A Survey, October 2025. URL http://arxiv.org/abs/2308.10792. arXiv:2308.10792 [cs.CL].

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-Distilled Reasoner: On-Policy Self-Distillation for Large Language Models, March 2026. URL http://arxiv.org/abs/2601.18734. arXiv:2601.18734 [cs].

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. LIMA: Less Is More for Alignment, May 2023. URL http://arxiv.org/abs/2305.11206. arXiv:2305.11206 [cs.CL].

A Future Work and Limitations

Due to computational constraints, we restrict our experiments to three Qwen3 model sizes, with up to 8B parameters. Future work should investigate how our findings generalize to larger model scales.

Our method also relies on pre-annotated task categories to select relevant demonstrations. Automatic retrieval methods, such as embedding-based similarity search, could eliminate this requirement and broaden the applicability of the approach.

Finally, the on-policy generation required by our method requires approximately 2.5$\times$ as many FLOPs as SFT. Future work could explore more computationally efficient training strategies as well as adaptive teacher policies.

B Dataset Curation

To construct our training dataset, we start from a Magpie-annotated (Xu et al., 2024; Djuhera et al., 2025) version of the Tulu3-sft-mixture (Lambert et al., 2025) and follow the data curation recipe outlined by Djuhera et al. (2025). In particular, we retain examples whose input quality is rated as

'excellent' and that receive a high reward score. To preserve a task-category distribution close to that of the original dataset, we use examples rated as 'good' as a fallback for categories that would otherwise become underrepresented after filtering. We also deduplicate the dataset and remove examples with empty responses.

C JUDGE PROMPTS

In this section, we report the judge prompts used for AlpacaEval 2.0 and Arena-Hard v2.

AlpacaEval 2.0

```
[SYSTEM]

You are a highly efficient assistant, who evaluates and selects the best large language
     model (LLMs) based on the quality of their responses to a given instruction. This
     process will be used to create a leaderboard reflecting the most accurate and human-
     preferred answers.

[USER]

I require a leaderboard for various large language models. I'll provide you with prompts
     given to these models and their corresponding outputs. Your task is to assess these
     responses, and select the model that produces the best output from a human
     perspective.

## Instruction

{
    "instruction": """{instruction}""",
}

## Model Outputs

Here are the unordered outputs from the models. Each output is associated with a specific
     model, identified by a unique model identifier.

{
    {
        "model_identifier": "m",
        "output": """{output_1}"""
    },
    {
        "model_identifier": "M",
        "output": """{output_2}"""
    }
}

## Task

Evaluate the models based on the quality and relevance of their outputs, and select the
     model that generated the best output. Answer by providing the model identifier of the
      best model. We will use your output as the name of the best model, so make sure your
      output only contains one of the following model identifiers and nothing else (no
     quotes, no spaces, no new lines, ...): m or M.

## Best Model Identifier
```

Arena-Hard v2: Hard-Prompt

```
[SYSTEM]

Please act as an impartial judge and evaluate the quality of the responses provided by two
      AI assistants to the user prompt displayed below. You will be given assistant A's
     answer and assistant B's answer. Your job is to evaluate which assistant's answer is
     better.

Begin your evaluation by generating your own answer to the prompt. You must provide your
     answers before judging any answers.

When evaluating the assistants' answers, compare both assistants' answers with your answer
     . You must identify and correct any mistakes or inaccurate information.
```

```
Then consider if the assistant's answers are helpful, relevant, and concise. Helpful means
     the answer correctly responds to the prompt or follows the instructions. Note when
    user prompt has any ambiguity or more than one interpretation, it is more helpful and
     appropriate to ask for clarifications or more information from the user than
    providing an answer based on assumptions. Relevant means all parts of the response
    closely connect or are appropriate to what is being asked. Concise means the response
     is clear and not verbose or excessive.

Then consider the creativity and novelty of the assistant's answers when needed. Finally,
    identify any missing important information in the assistants' answers that would be
    beneficial to include when responding to the user prompt.

After providing your explanation, you must output only one of the following choices as
    your final verdict with a label:

1. Assistant A is significantly better: [[A>>B]]
2. Assistant A is slightly better: [[A>B]]
3. Tie, relatively the same: [[A=B]]
4. Assistant B is slightly better: [[B>A]]
5. Assistant B is significantly better: [[B>>A]]

Example output: "My final verdict is tie: [[A=B]]".

[USER]

<|User Prompt|>
{QUESTION}

<|The Start of Assistant A's Answer|>
{ANSWER_A}
<|The End of Assistant A's Answer|>

<|The Start of Assistant B's Answer|>
{ANSWER_B}
<|The End of Assistant B's Answer|>
```

D TEACHER OUTPUT EXAMPLE

In this section, we show teacher-model outputs under the few-shot prompting used in FSSD and the reference-response conditioning used in OPSD. Under reference-response conditioning, the base model reproduces the reference response verbatim. The outputs are generated using Qwen3-4B-Base with a temperature of 1.0, top-p=0.95, and top-k=100.

Few-Shot Teacher Prompt

```
### SYSTEM:
You are a helpful assistant.
### END_SYSTEM

### QUERY:
Alexandria is planning to buy a new desk and four office chairs for her home office. She
    expected the desk to cost $80 and each office chair to cost $30. However, when she
    arrived at the store, she discovered that the desk was on sale for $10 less, and the
    price of an office chair had increased by $4. Calculate the total amount of money
    Alexandria spent at the store.
### END_QUERY

### ANSWER:
Let's break down the problem step by step.

The desk was originally priced at $80 but was on sale for $10 less, so the new price of
    the desk is $80 - $10 = $70.

The office chairs were originally priced at $30 each, but the price had increased by $4,
    so the new price of each office chair is $30 + $4 = $34.

Alexandria buys one desk and four office chairs, so the total cost is the cost of the desk
     plus the cost of the four chairs: $70 + 4 x $34.

The total cost of the four chairs is 4 x $34 = $136.
```

```
Adding this to the cost of the desk, the total amount Alexandria spent is $70 + $136 =
     $206.

Thus, the total amount of money Alexandria spent at the store is \boxed{206} dollars.
### END_ANSWER

### QUERY:
Maria is a government official working in the US Department of Transportation. She is
     tasked with distributing new road safety kits to various cities in her state. There
     are 5 major cities that need the kits, and each city requires a different number of
     kits: City A needs 120 kits, City B needs 150 kits, City C needs 175 kits, City D
     needs 200 kits, and City E needs 155 kits.

Maria has a budget to purchase 1,000 road safety kits. After distributing the kits to
     these 5 cities, how many kits will Maria have left over?
### END_QUERY

### ANSWER:
To solve the problem, we need to determine how many road safety kits Maria will distribute
      to the five cities and then subtract this total from the number of kits she
     initially has.

**Step 1: Calculate the total number of kits needed by all cities.**

- City A needs 120 kits.
- City B needs 150 kits.
- City C needs 175 kits.
- City D needs 200 kits.
- City E needs 155 kits.

Add these numbers together to find the total number of kits required:

\[
120 + 150 + 175 + 200 + 155 = 800
\]

**Step 2: Calculate the number of kits Maria will have left over.**

Maria started with 1,000 kits. After distributing the kits to all five cities, she will
     have:

\[
1000 - 800 = 200
\]

Maria will have 200 kits left over after distributing them to the cities.

#### 200
### END_ANSWER

### QUERY:
A sports editor, known for providing valuable feedback on composition and storytelling in
     photographs, is curating a special photo book. The book will include a series of
     photographs, each telling a unique story through its composition. The editor is
     particularly focused on the geometric layout and the mathematical balance within each
      photograph.

1. For one of the photographs, the editor wants to ensure that the main subject is placed
     according to the rule of thirds, which divides the photograph into a 3x3 grid of
     equal rectangles. If the photograph is a rectangle with dimensions \( 18 \text{ cm} \
     times 12 \text{ cm} \), calculate the coordinates of the points where the main
     subject can be placed to follow the rule of thirds. Provide your answer in terms of
     centimeters from the bottom-left corner of the photograph.

2. The editor also wants to analyze the storytelling aspect by considering the golden
     ratio. If the same photograph (18 cm x 12 cm) is divided into sections according to
     the golden ratio, determine the dimensions of the smaller rectangle that would be
     formed within the original photograph. Use the approximation \( \phi \approx 1.618 \)
      for the golden ratio.
### END_QUERY

### ANSWER:
Let's solve the problem step-by-step.

### Part 1: Rule of Thirds
The rule of thirds divides the photograph into nine equal rectangles by drawing two
     vertical and two horizontal lines.

1. **Dimensions of the photograph**: \( 18 \text{ cm} \times 12 \text{ cm} \)
```

```
2. **Dividing by thirds**:
   - Vertically: \( 18 \text{ cm} / 3 = 6 \text{ cm} \)
   - Horizontally: \( 12 \text{ cm} / 3 = 4 \text{ cm} \)

So, the vertical lines will be at \( x = 6 \text{ cm} \) and \( x = 12 \text{ cm} \), and
     the horizontal lines will be at \( y = 4 \text{ cm} \) and \( y = 8 \text{ cm} \).

**Coordinates of the intersection points (from the bottom-left corner)**:
- \((6, 4)\)
- \((6, 8)\)
- \((12, 4)\)
- \((12, 8)\)

### Part 2: Golden Ratio
The golden ratio \( \phi \) divides a line into two parts such that the ratio of the whole
      line to the larger part is the same as the ratio of the larger part to the smaller
     part.

1. **Golden ratio approximation**: \( \phi \approx 1.618 \)
2. **Dimensions of the photograph**: \( 18 \text{ cm} \times 12 \text{ cm} \)

For the golden ratio, we need to find the length \( L \) such that:
\[
L = \frac{18}{\phi} \approx \frac{18}{1.618} \approx 11.13 \text{ cm}
\]

So, the length of the smaller section along the width is approximately \( 18 - 11.13 =
     6.87 \text{ cm} \).

Similarly, for the height:
\[
L = \frac{12}{\phi} \approx \frac{12}{1.618} \approx 7.42 \text{ cm}
\]

So, the height of the smaller section along the height is approximately \( 12 - 7.42 =
     4.58 \text{ cm} \).

**Dimensions of the smaller rectangle**:
- Width: \( 6.87 \text{ cm} \)
- Height: \( 4.58 \text{ cm} \)

### Final Answer:
Final Answer: The coordinates for the rule of thirds are \((6, 4)\), \((6, 8)\), \((12, 4)
     \), and \((12, 8)\). The dimensions of the smaller rectangle following the golden
     ratio are approximately \( 6.87 \text{ cm} \times 4.58 \text{ cm} \). I hope it is
     correct.
### END_ANSWER

### QUERY:
A tailor purchased buttons for the clothes. She bought 90 green buttons, 10 more yellow
     buttons than the green buttons, and 5 fewer blue buttons than the green buttons. How
     many buttons did the tailor purchase?
### END_QUERY

### ANSWER:
```

Few-Shot Teacher Answer

```
Let's break down the problem step by step to find out the total number of buttons the
     tailor purchased.

1. **Green buttons**: The tailor bought 90 green buttons.
2. **Yellow buttons**: She bought 10 more yellow buttons than green buttons.
   \[
   \text{Yellow buttons} = 90 + 10 = 100
   \]
3. **Blue buttons**: She bought 5 fewer blue buttons than green buttons.
   \[
   \text{Blue buttons} = 90 - 5 = 85
   \]

Now, we can calculate the total number of buttons:

\[
\text{Total buttons} = \text{Green buttons} + \text{Yellow buttons} + \text{Blue buttons}
\]
```

````
\[
\text{Total buttons} = 90 + 100 + 85 = 275
\]

Thus, the tailor purchased \(\boxed{275}\) buttons.
### END_ANSWER
````

OPSD Teacher Prompt

````
### SYSTEM:
You are a helpful assistant. You are given a question and a reference response. Answer
     with a response of your own.
### END_SYSTEM

### QUERY:
A tailor purchased buttons for the clothes. She bought 90 green buttons, 10 more yellow
     buttons than the green buttons, and 5 fewer blue buttons than the green buttons. How
     many buttons did the tailor purchase?
### END_QUERY

### ANSWER:
Let's break down the problem step-by-step and use Python to ensure the accuracy of our
     calculations.

First, define the given quantities:
- Green buttons (G): 90
- Yellow buttons (Y): 10 more than the green buttons
- Blue buttons (B): 5 fewer than the green buttons

Using this information, we can determine the quantities of yellow and blue buttons:
- Y = G + 10
- B = G - 5

Next, sum up all the buttons to get the total quantity.

Let's calculate these values using Python.
```python
# Given data
green_buttons = 90
yellow_buttons = green_buttons + 10
blue_buttons = green_buttons - 5

# Total buttons
total_buttons = green_buttons + yellow_buttons + blue_buttons
print(total_buttons)
```
```output
275
```
The total number of buttons that the tailor purchased is \(\boxed{275}\).
### END_ANSWER

### QUERY:
A tailor purchased buttons for the clothes. She bought 90 green buttons, 10 more yellow
     buttons than the green buttons, and 5 fewer blue buttons than the green buttons. How
     many buttons did the tailor purchase?
### END_QUERY

### ANSWER:
````

OPSD Teacher Answer

````
Let's break down the problem step-by-step and use Python to ensure the accuracy of our
     calculations.

First, define the given quantities:
- Green buttons (G): 90
- Yellow buttons (Y): 10 more than the green buttons
- Blue buttons (B): 5 fewer than the green buttons

Using this information, we can determine the quantities of yellow and blue buttons:
- Y = G + 10
- B = G - 5
````

````
Next, sum up all the buttons to get the total quantity.

Let's calculate these values using Python.
```python
# Given data
green_buttons = 90
yellow_buttons = green_buttons + 10
blue_buttons = green_buttons - 5

# Total buttons
total_buttons = green_buttons + yellow_buttons + blue_buttons
print(total_buttons)
```
```output
275
```
The total number of buttons that the tailor purchased is \(\boxed{275}\).
````

D.1 Verbatim Reference-Response Reproduction Frequency

To quantify the tendency of the teacher to reproduce the reference response under reference-response conditioning, we compute the exact-match rate, after normalization, between 100 reference responses and teacher-generated responses for each conditioning strategy. This directly measures the behavior illustrated in Appendix D.

For normalization, we apply Unicode NFC normalization, lowercase the text, replace each contiguous sequence of whitespace (including newlines and tabs) with a single space, and strip leading and trailing whitespace.

Reference-response conditioning results in verbatim reproduction in 75% of generations, whereas no exact matches are observed under few-shot conditioning (Table 6). This shows that few-shot conditioning avoids the severe reference-copying behavior observed when a base-model teacher is conditioned directly on the target response.

Teacher conditioning	Exact match
Few-shot demonstrations	0/100 (0%)
Reference response	75/100 (75%)

Table 6: Frequency with which the teacher output exactly matches the reference response after normalization.

E KL Divergence During Training

We examine how the KL divergence from the pretrained policy evolves over training, reporting measurements at 500, 1000, and 1500 training steps (Figure 5). SFT exhibits a higher KL divergence throughout training and continues to diverge from the base model as training progresses. In contrast, OPSD and FSSD remain considerably closer to the pretrained policy, with their KL divergences beginning to plateau after approximately 500 steps. These results further illustrate that on-policy training induces smaller deviations from the pretrained policy than standard SFT.

F Hyperparameters

All experiments are conducted using two H100 or A100 GPUs, with gradient checkpointing and Flash Attention 2 enabled to improve training efficiency. We use bfloat16 precision for all runs and AdamW (Loshchilov & Hutter, 2019) as the optimizer. Full training hyperparameters are reported in Table 7.

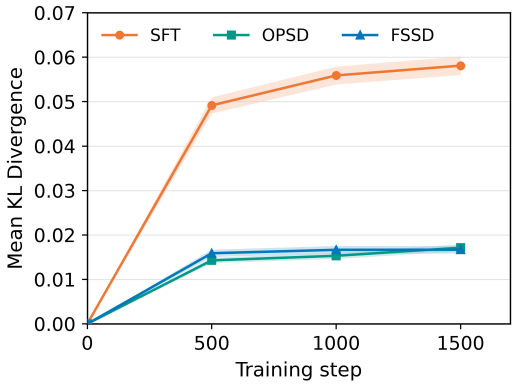


Figure 5: **Divergence from the pretrained policy over training**. Mean KL divergence between the Qwen3-1.7B-Base policy and the policies obtained with SFT, OPSD, or FSSD, evaluated on a held-out set of 500 examples at 500, 1,000, and 1,500 training steps. The on-policy methods remain closer to the pretrained policy than SFT throughout training. Shaded regions indicate ± SEM.

Parameter	**SFT**	**OPSD**	**FSSD**
Learning Rate	{1e-6, 5e-6, 1e-5}	{1e-6, 5e-6, 1e-5}	{1e-6, 5e-6, 1e-5}
Effective Batch Size	32	32	32
LoRA Rank (r)	64	64	64
LoRA Alpha (α)	32	32	32
LoRA Target Modules	q_proj, k_proj, v_proj, o_proj, gate_proj, up_proj, down_proj		
Max Sequence Length	20000		
Max Completion Length		2048	2048
Sampling Temperature		1.0	1.0
Distillation Divergence		Forward KL	Forward KL
Warmup Ratio	0.02	0.02	0.02
Epochs	1	1	1
Top-k Distillation		256	256
LR Scheduler	Linear	Linear	Linear

Table 7: **Training hyperparameters for SFT, OPSD, and FSSD**. Curly braces {} indicate a sweep over the specified values.

G AI USE STATEMENT

In this work, we used generative AI tools to assist with software implementation and code editing, and to improve the clarity, grammar, and readability of the manuscript. All AI-assisted code was reviewed, tested, and verified by the authors, and all AI-assisted edits to the manuscript were reviewed by the authors. The authors take full responsibility for the final content of this work, including all text, claims, code, and other artifacts produced with the aid of generative AI.