OPR: ON-POLICY REPLAY FOR CONTINUAL LEARNING IN LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models can continually learn new tasks through sequential supervised fine-tuning (SFT), but remain susceptible to catastrophic forgetting. We propose **O**n-**P**olicy **R**eplay (**OPR**), a generative replay method that uses selected responses from the current model as supervision for previously learned tasks. After each task, the current checkpoint generates responses to historical prompts. High-scoring prompt–response pairs are then mixed with the next task's labeled data for SFT using the standard cross-entropy objective. We introduce two response selection strategies: OPR-RU uses task rewards, whereas OPR-SC uses model confidence without requiring historical labels for replay construction. By regenerating replay responses at each task boundary, OPR updates the supervision for past tasks to reflect the model's current output distribution. On the TRACE benchmark, OPR-RU achieves higher average task accuracy and better backward transfer (BWT) than vanilla Replay and LAMOL at a matched 1% replay budget on both Qwen3-4B and Qwen3-8B. It also achieves higher average task accuracy than SDFT, although SDFT has less negative BWT. Increasing the replay budget to 10% yields a BWT of -0.19 on Qwen3-4B, indicating little average degradation relative to task scores immediately after learning. These results demonstrate that OPR preserves previously learned task capabilities, providing an effective approach to continual learning in large language models.

1 INTRODUCTION

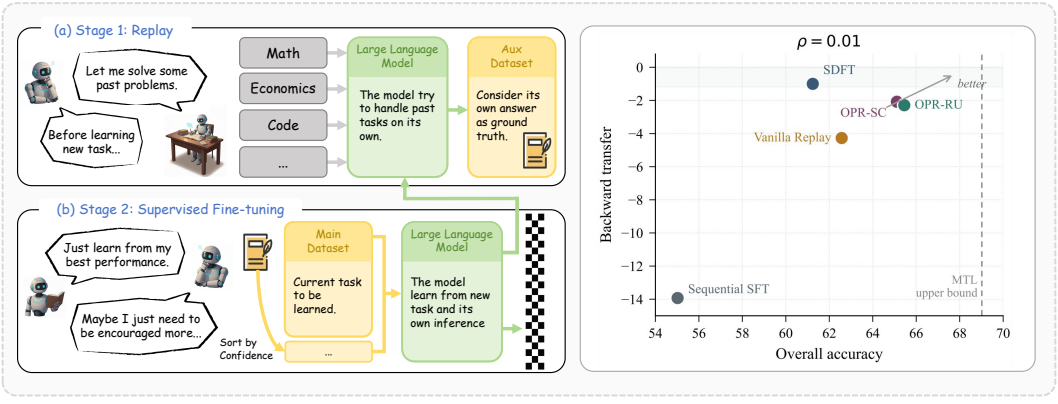


Figure 1: On-Policy Replay (OPR) mitigates forgetting by learning from the model's own responses to historical tasks. *Left:* OPR builds a replay buffer from high-scoring responses and combines the buffer with new-task data for continual learning. *Right:* OPR extends the accuracy–retention Pareto frontier, offering higher accuracy while maintaining low forgetting. Higher is better on both axes.

Large language models continue to learn after deployment as new tasks, domains, and labeled data emerge (Chung et al., 2022; Ouyang et al., 2022; Wu et al., 2024). Supervised fine-tuning (SFT) is a common approach to updating these models, but adapting to new tasks can degrade performance on previously learned ones (Luo et al., 2025; Wang et al., 2023b). Offline replay mitigates this

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.

forgetting by mixing historical prompt–gold-answer pairs stored in a replay buffer with the training data for each new task. As tasks accumulate, however, the buffer must retain an expanding training history within a fixed memory budget.

Earlier work such as LAMOL explored the use of model-generated data to mitigate forgetting by generating pseudo-samples of previous tasks and replaying them alongside new-task data (Sun et al., 2019). More recent work shows that learning from a model's own responses can help retain its existing capabilities, linking this effect to smaller changes in its output distribution (Shenfeld et al., 2025; Chen et al., 2026; Chu et al., 2025; Zhu et al., 2026). On-policy distillation methods, including SDFT (Shenfeld et al., 2026), OPSD (Zhao et al., 2026), and GKD (Agarwal et al., 2024), train on student-generated responses using teacher distributions as supervision. Research on SFT with correct self-generated answers further shows that selected responses can serve as supervised targets for learning the current task (Shenfeld et al., 2025).

We introduce On-Policy Replay, which incorporates on-policy supervision into continual supervised fine-tuning through historical replay data. After training on each task, the current model generates responses to historical prompts. A scorer then selects high-scoring prompt–response pairs to populate a task-balanced replay buffer. For the next task, standard SFT is performed on labeled task data mixed with the buffered samples. OPR thus integrates directly into existing SFT pipelines.

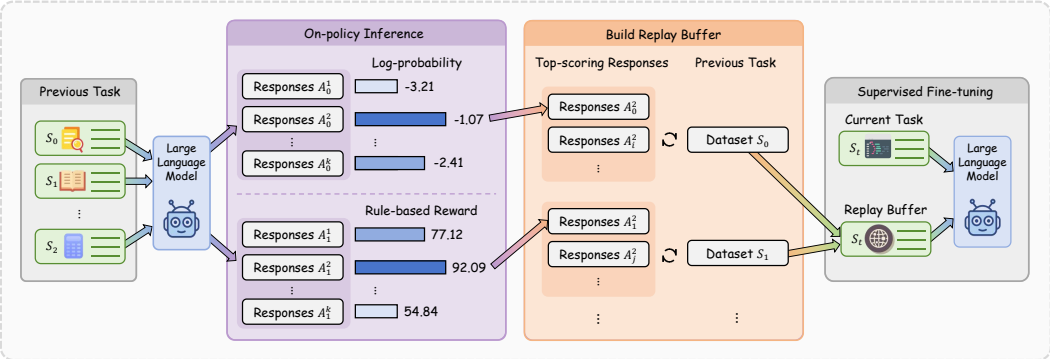


Figure 2: Overview of the On-Policy Replay pipeline. Before each new task, the latest checkpoint generates responses to historical prompts. These responses are ranked using task-specific rewards (OPR-RU) or length-normalized log-probabilities (OPR-SC). The highest-scoring responses are paired with their original prompts to construct a replay buffer, which is combined with new-task data to train the model.

Our contributions are threefold: (i) OPR, a method for constructing on-policy replay data for continual SFT; (ii) a confidence-based variant that requires no historical labels; and (iii) an experimental evaluation of response selection and replay targets. On Qwen3-4B at a 1% replay budget, high-scoring replay responses yield higher average task accuracy and better backward transfer (BWT) than low-scoring ones. A prompt-aligned gold-answer control slightly improves BWT over vanilla Replay but lowers average task accuracy. With the selected prompts held fixed, using generated responses instead of gold answers improves both metrics.

2 Related Work

LAMOL mitigates forgetting through generative replay, jointly learning to solve tasks and generate pseudo-samples of previous tasks for training alongside new-task data (Sun et al., 2019). On-policy distillation provides a related approach to learning from model-generated responses (Shenfeld et al., 2025; Chen et al., 2026; Chu et al., 2025; Zhu et al., 2026). GKD addresses the distribution mismatch between training and inference by applying teacher supervision to student-generated trajectories (Agarwal et al., 2024). SDFT uses a demonstration-conditioned teacher and reverse-KL training to support new-task learning while reducing forgetting (Shenfeld et al., 2026), whereas OPSD uses privileged reference information to provide token-level supervision on student rollouts (Zhao et al., 2026). OPR incorporates on-policy generation into historical replay construction: the current checkpoint generates responses to retained historical prompts at task boundaries, and selected responses

form a fixed buffer for subsequent standard SFT. Our local KL analysis characterizes the resulting replay updates.

Self-generated responses can also serve as targets for ordinary supervised fine-tuning. S3FT selects correct model-generated answers and uses gold answers or their paraphrases for the remaining examples, aiming to improve target-task performance while preserving generalization (Gupta et al., 2025). In this setting, response selection primarily supports learning the current task. OPR-RU applies filtered self-generation to historical retention, using reference-based task rewards to refresh the replay buffer while preserving gold targets for current-task training. OPR-SC instead ranks responses by length-normalized model log-probability, enabling replay construction without historical labels. Both variants focus on constructing historical supervision under a limited replay budget.

Continual-learning approaches more broadly include regularization (Kirkpatrick et al., 2017), parameter isolation (Wang et al., 2023a; Ren et al., 2024), and rehearsal (Lopez-Paz & Ranzato, 2022; Wu et al., 2024). Within this landscape, GeRe uses general pretraining samples to preserve capabilities (Zhang et al., 2025), I-LoRA combines replay with adapter interpolation to balance adaptation and retention (Ren et al., 2024), and SuRe prioritizes surprising sequences with high negative log-likelihood (Hazard et al., 2025). These methods address different aspects of memory selection and knowledge consolidation, while OPR focuses on constructing historical replay targets. TRACE provides a shared eight-task evaluation benchmark (Wang et al., 2023b), and research on spurious forgetting highlights that performance declines can reflect disrupted task alignment rather than underlying knowledge loss (Zheng et al., 2025). We compare OPR with Replay, LAMOL, SDFT, and other methods, and use a prompt-aligned gold-answer control to assess the contribution of generated replay targets while holding prompt selection fixed.

3 METHOD

4 METHOD

4.1 CONTINUAL SFT AND REPLAY CONSTRUCTION

A learner observes tasks $T_1, \ldots, T_N$ with datasets $\mathcal{D}_i = \{(x_{ij}, y_{ij})\}_{j=1}^{n_i}$. At stage $i+1$, it trains checkpoint π_{θ_i} on $\mathcal{D}_{i+1} \cup \mathcal{B}_{i+1}$, where the total replay budget is $b = \lfloor \rho n_{i+1} \rfloor$. The budget is shared across historical tasks and scales with the current dataset size. With $n_i = 5000$, $\rho = 0.01$ retains 50 historical examples in total. The historical share of the concatenated dataset is $\rho/(1+\rho)$, up to rounding.

At the end of stage i, OPR generates one response per eligible historical training prompt using the latest checkpoint, then ranks the responses separately within each historical task. It distributes the budget as evenly as possible across tasks, respecting the number of eligible candidates. The selected generated responses replace gold answers only in the replay portion. Algorithm 1 describes one stage. The buffer is generated once per stage and held fixed during subsequent optimization, so "on-policy" refers to its construction checkpoint.

We rank responses within each historical task because task rewards need not share a common scale. This matters when replay combines classification and generation tasks evaluated by different metrics. Task quotas determine how much of the buffer each task occupies, while ranking determines which prompt–response pairs fill those slots. This separation maintains coverage across historical tasks without requiring cross-task score calibration. Regenerating responses at each task boundary also allows the same historical prompt to receive an updated replay target as the model changes.

4.2 CHOICE OF SCORER

Rule-based Scoring. We score historical generations with task-specific metrics: classification accuracy for C-STANCE, FOMC, and ScienceQA; final numerical answer extraction followed by exact matching for NumGLUE-cm/ds; edit similarity for Py150; ROUGE-L F1 for MeetingBank; and SARI for 20Minuten. NumGLUE uses regular expressions to extract the final numerical answer before comparison, allowing variation in the surrounding response text. Historical training references

Algorithm 1 On-Policy Replay at a stage boundary

Require: Checkpoint π_{θ_i}; historical training prompts $P_1, \ldots, P_i$; scorer r; budget b.
1: Generate one response per eligible prompt with the stage-i decoder.
2: Assign task quotas $b_1, \ldots, b_i$ as evenly as possible, with $\sum_j b_j \leq b$.
3: $\mathcal{B}_{i+1} \leftarrow \emptyset$
4: **for** $j = 1, \ldots, i$ **do**
5: Score task-j candidates and add its top b_j (prompt, generated response) pairs to $\mathcal{B}_{i+1}$.
6: **end for**
7: Train on $\mathcal{D}_{i+1} \cup \mathcal{B}_{i+1}$ with ordinary token-level cross-entropy.
8: **return** $\pi_{\theta_{i+1}}$

provide rewards, while the selected generated responses serve as replay targets. Aligning selection with the evaluation metric prioritizes task-relevant target quality.

Self-Confidence Scoring. This variant uses length-normalized log-probability under the construction checkpoint:

$$r_{\mathrm{SC}}(x, y) = \frac{1}{|y|} \sum_{k=1}^{|y|} \log \pi_{\theta_i}(y_k \mid x, y_{<k}). \tag{1}$$

Token log-probabilities are obtained during rollout, making the construction checkpoint its own scorer. Historical buffer construction is label-free and requires no external reward function; incoming tasks use labeled SFT data. Scores quantify model confidence, so confidently incorrect responses can also receive high scores.

4.3 A Local KL View of Replay

Prior work motivates output-distribution change as a useful lens on forgetting (Shenfeld et al., 2025; Chen et al., 2026). We use this lens to connect target selection to the replay-gradient direction and local distributional change under explicit smoothness and loss assumptions.

Let $s_i(y \mid x)$ be the actual stage-i sampling distribution, including decoding temperature, and let τ_j denote the selection threshold for historical task j. An idealized conditional selected-response distribution is

$$q_i(y \mid x) = \frac{s_i(y \mid x)\mathbf{1}[r(x, y) \geq \tau_j]}{Z_i(x)}, \tag{2}$$

for prompts with $Z_i(x) > 0$. Selection also changes which prompts enter the buffer. In the implementation, $\widehat{q}_i$ denotes the empirical joint distribution over the retained pairs, including this prompt selection. The sampling distribution s_i incorporates low-temperature decoding, while π_{θ_i} denotes the untempered training policy. Both q_i and $\widehat{q}_i$ are fixed during stage $i + 1$.

For sequence loss $\ell(\theta; x, y) = -\log \pi_\theta(y \mid x)$, the confidence filter gives the exact identity and bound at construction time:

$$\ell(\theta_i; x, y) = -|y| r_{\mathrm{SC}}(x, y) \leq -|y|\tau_j \leq \ell_{\max}, \tag{3}$$

where $\ell_{\max} = \max_{(x,y)\in\mathcal{B}_{i+1}}(-|y|\tau_{j(x)})$ is finite for the bounded-length buffer. For length-normalized loss, the corresponding item-wise bound is $-\tau_j$. The condition for Eq. (3) is that Eq. (1) uses untempered training-policy log-probabilities, rather than temperature-adjusted sampling log-probabilities.

For OPR-RU, applying the subsequent norm bound requires an independent bound on the selected-response losses. Reward–likelihood alignment can motivate an empirical low-loss interpretation; the

uniform bound remains an additional assumption because selection thresholds task reward. For either variant, extending the construction-time bound to later within-stage iterates requires the selected losses to remain bounded as parameters change.

For any fixed conditional replay target distribution, its entropy is independent of θ, yielding

$$\mathbb{E}_{(x,y)\sim\widehat{q}_i}[\nabla_\theta \ell] = \nabla_\theta \mathbb{E}_{x\sim\widehat{q}_i}[\mathrm{KL}(\widehat{q}_i(\cdot \mid x) \,\|\, \pi_\theta(\cdot \mid x))]. \tag{4}$$

Thus a sufficiently small replay-only gradient step descends KL toward the selected target distribution. The identity also holds for gold replay, locating OPR's distinguishing mechanism in its generated targets and score-based selection. Token weighting gives an analogous cross-entropy identity over replay prefixes. The following bound uses sequence loss, with constants determined by this normalization.

Assume each selected loss is nonnegative and L-smooth on a domain containing its gradient descent segment, so the standard self-bounding inequality applies. At an iterate θ where selected losses are bounded by $\ell_{\max}$, Jensen's inequality gives

$$\|g_{\mathrm{rep}}\|^2 \le \mathbb{E}_{\widehat{q}_i}\|\nabla \ell\|^2 \le 2L\mathbb{E}_{\widehat{q}_i}\ell \le 2L\ell_{\max}. \tag{5}$$

Let $F_{\mathrm{hist}}(\theta)$ be the Fisher information averaged over historical prompts, with spectral norm at most $\sigma_{\max}$. Under the regularity needed for a local second-order KL expansion, the replay-only SGD step $\theta' = \theta - \eta g_{\mathrm{rep}}$ obeys

$$\begin{aligned}\mathbb{E}_{p_{\mathrm{hist}}}\mathrm{KL}(\pi_{\theta'} \,\|\, \pi_\theta) &= \tfrac{1}{2}\eta^2 g_{\mathrm{rep}}^\top F_{\mathrm{hist}}(\theta) g_{\mathrm{rep}} + o(\eta^2) \\ &\le L\sigma_{\max}\eta^2\ell_{\max} + o(\eta^2).\end{aligned} \tag{6}$$

The proof follows by substituting Eq. (5) into the quadratic expansion; the constant and first-order terms vanish at $\theta' = \theta$.

Equation (6) isolates the contribution of a replay-only SGD step to local KL change. Actual training uses AdamW and mixes new-task and replay gradients. For a mixed SGD direction $g = \lambda g_{\mathrm{rep}} + (1-\lambda) g_{\mathrm{new}}$, the quadratic KL expression additionally contains new-task and cross terms, which require their own bounds. End-to-end retention is therefore evaluated empirically through BWT. The two methods address the stability–plasticity trade-off through distinct objectives: OPR applies cross-entropy to selected historical targets, while SDFT's demonstration-conditioned teacher supplies a reverse-KL objective on new-task trajectories at every step. SDFT implements this objective without a separately tuned KL multiplier β.

This analysis motivates selecting high-confidence responses to obtain a construction-time loss bound. For reward selection, a similar low-loss interpretation depends on reward–likelihood alignment. The low-score ablation measures the empirical contribution of filtering, while the prompt-aligned gold control tests target construction with selected prompts held fixed. Together, these controls distinguish the roles of scoring and generated targets in retention.

5 Experiments

5.1 Setup

Dataset. We use TRACE in its canonical order (Wang et al., 2023b). C-STANCE (abbreviated as C-STA) focuses on detecting the stance expressed in Chinese text toward a given target. FOMC focuses on classifying the financial stance expressed in monetary policy communications. MeetingBank (abbreviated as MeBa) requires generating summaries of meeting transcripts. Py150 evaluates Python code completion given the preceding code context. ScienceQA (abbreviated as SciQA) evaluates scientific knowledge and reasoning through question answering. NumGLUE-cm (abbreviated as NG-cm) evaluates numerical reasoning through mathematical computation. NumGLUE-ds (abbreviated as NG-ds) evaluates numerical reasoning in domain-specific contexts. 20Minuten (abbreviated as 20Min) requires simplifying German text while preserving its meaning. We subsample

5,000 training examples per task and use the official evaluation splits with the length filter. The sequence spans Chinese, English, German, and code. Backbones are Qwen3-4B and Qwen3-8B (Yang et al., 2025).

Model. All models are trained using AdamW with an initial learning rate of 10^{-5} and a cosine learning rate schedule without warmup. We set the weight decay to 0, the global batch size to 128, and the maximum sequence length to 2048, and use bfloat16 precision throughout training. The numbers of training epochs for the eight tasks, in the order listed above, are 5, 3, 7, 5, 3, 5, 5, and 7, respectively. For each model, we report performance on each task as the mean score over four evaluation runs (Avg@4).

Baselines. We compare OPR with four baselines. Sequential SFT fine-tunes the model on each task in sequence without replaying historical data. Vanilla replay (abbreviated as Replay) combines current-task data with gold input–output pairs from previous tasks, matching OPR in total replay buffer size, allocation across historical tasks, and the cross-entropy training objective. LAMOL is a generative replay method that uses a single language model for both task learning and the generation of pseudo-samples from previous tasks, which are replayed alongside current-task data (Sun et al., 2019). SDFT uses a demonstration-conditioned teacher and an on-policy reverse-KL objective (Shenfeld et al., 2026). We adapt SDFT to TRACE while retaining this formulation and using the shared optimizer and learning rate schedule.

Metrics. We evaluate the eight TRACE tasks using their corresponding task-specific metrics: accuracy for C-STANCE, FOMC, ScienceQA, NumGLUE-cm, and NumGLUE-ds; ROUGE-L for MeetingBank; edit similarity for Py150; and SARI for 20Minuten. To assess overall continual learning performance, we report average accuracy (ACC) and backward transfer (BWT):

$$\mathrm{ACC} = \frac{1}{N}\sum_{i=1}^{N} a_{i,N}, \qquad \mathrm{BWT} = \frac{1}{N-1}\sum_{i=1}^{N-1}\left(a_{i,N} - a_{i,i}\right),$$

where $N = 8$ is the number of tasks and $a_{i,j}$ denotes the score on task i after training on task j, with all task-specific scores expressed on a common scale. ACC measures the average performance across all tasks after completing the task sequence. BWT measures the average change in performance on earlier tasks relative to their scores immediately after they were learned. Negative BWT indicates forgetting, whereas positive BWT indicates beneficial backward transfer. Together, these metrics characterize final task performance and the retention of previously acquired knowledge.

5.2 Main result

OPR-RU achieves the highest ACC on both backbones, reaching 64.95 on Qwen3-4B and 65.99 on Qwen3-8B, followed closely by OPR-SC. Compared with sequential SFT, OPR-RU improves ACC by 7.62 and 5.28 points and increases BWT from -11.55 to -1.88 and from -8.85 to -1.70, respectively. These results demonstrate consistent improvements in final average performance and retention across both model scales.

With a replay buffer sized at 1% of the training set, and under matched historical task allocations and training objectives, OPR-RU outperforms vanilla Replay in both aggregate metrics. ACC improves by 0.49 and 1.08 points on Qwen3-4B and Qwen3-8B, while BWT improves by 0.82 and 2.57 points. OPR-RU also exceeds LAMOL by 1.40 and 1.97 ACC points, with corresponding BWT improvements of 1.24 and 1.27 points. These comparisons support the effectiveness of OPR's replay construction. The gains vary across tasks: improvements over Replay are concentrated in C-STANCE and FOMC on Qwen3-4B, whereas Qwen3-8B benefits most on NumGLUE-cm, MeetingBank, and Py150.

Since BWT measures changes relative to each method's own scores immediately after learning a task, smaller declines do not necessarily indicate stronger retention at comparable levels of task acquisition. SDFT's less negative BWT may partly reflect weaker initial acquisition of long-form generation tasks, such as MeetingBank, leaving less room for subsequent score degradation. The task-wise trajectories on Qwen3-4B are consistent with this interpretation: SDFT scores lower on MeetingBank immediately after learning it and remains below OPR-RU throughout subsequent

Table 1: Main results on the eight-task TRACE benchmark with Qwen3-4B and Qwen3-8B. Replay-based methods use a replay budget of $\rho = 0.01$, corresponding to 50 historical examples in total per stage. Task-wise scores and average accuracy (ACC) are measured after completing the task sequence; backward transfer (BWT) measures the average score change on previously learned tasks relative to their scores immediately after learning. Higher is better for all metrics, and negative BWT indicates forgetting. Task scores are averaged over four evaluation runs (Avg@4). OPR-RU achieves the highest ACC on both backbones.

Method	ACC	BWT	C-STA	FOMC	MeBa	Py150	SciQA	NG-cm	NG-ds	20Min
Qwen3-4B										
SFT	57.33	-11.55	48.93	59.07	52.97	63.75	58.45	60.49	73.54	41.40
Replay	64.46	-2.70	52.68	72.78	56.75	64.67	90.98	61.73	75.38	40.69
LAMOL	63.55	-3.12	51.18	73.20	54.65	63.87	89.73	62.08	73.70	39.99
SDFT	62.34	-1.34	50.36	73.06	39.25	51.26	88.24	79.50	78.20	38.85
OPR-RU	64.95	-1.88	54.95	75.20	57.13	64.35	90.40	61.73	75.08	40.79
OPR-SC	64.70	-1.92	54.47	75.43	56.27	63.98	89.75	62.04	75.32	40.34
Qwen3-8B										
SFT	60.71	-8.85	48.19	65.13	56.43	57.05	77.26	69.14	71.38	41.07
Replay	64.91	-4.27	51.78	71.52	57.47	64.49	92.20	62.96	77.54	41.33
LAMOL	64.02	-2.97	50.58	70.67	55.37	63.54	90.55	63.40	77.92	40.13
SDFT	62.79	-1.23	52.18	75.42	41.63	53.27	86.59	80.34	76.81	36.08
OPR-RU	65.99	-1.70	51.85	70.31	59.56	66.16	93.20	68.21	76.85	41.78
OPR-SC	65.92	-1.89	51.43	70.62	58.88	66.40	92.67	68.58	77.31	41.47

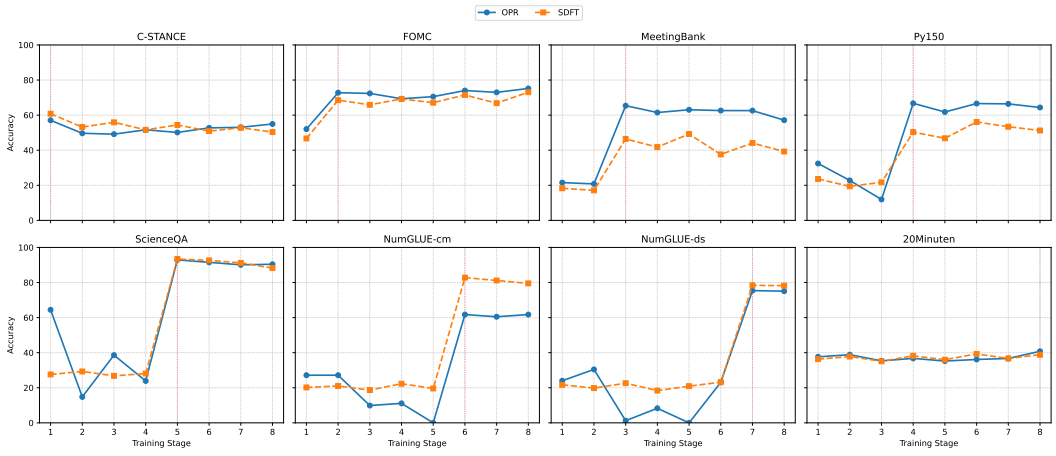


Figure 3: Task-wise performance trajectories of OPR-RU and SDFT during sequential training on the eight TRACE tasks with Qwen3-4B. OPR-RU uses a replay budget of $\rho = 0.01$. Each panel reports the corresponding task's evaluation score after each training stage; the red vertical dotted line marks the stage at which that task is learned. OPR-RU achieves higher final scores on six of the eight tasks, with particularly large gains on MeetingBank and Py150, while SDFT performs better on the two NumGLUE tasks.

training. Thus, SDFT's smaller average decline should be considered alongside its lower final task performance, whereas OPR-RU achieves the highest final ACC among the evaluated methods.

OPR-SC closely approaches OPR-RU without requiring historical labels for replay construction. Its ACC is only 0.25 and 0.07 points lower on the two backbones, with BWT gaps of 0.04 and 0.19 points. It also consistently improves both ACC and BWT over Replay and LAMOL. These results support confidence-based selection as an effective alternative when historical prompts remain available but their labels are unavailable.

5.3 Ablations

5.3.1 Replay Budget

We evaluate four replay budgets, $\rho \in \{0.01, 0.02, 0.05, 0.10\}$. OPR-RU consistently achieves less negative BWT than Vanilla Replay across all budgets, reducing the magnitude of negative BWT by 27.1–78.2%. As ρ increases from 0.01 to 0.10, accuracy improves from 64.95% to 67.05% for OPR-RU and from 64.46% to 66.57% for Vanilla Replay, corresponding to gains of 2.10 and 2.11 percentage points, respectively. OPR-RU achieves higher accuracy at three of the four budgets, with a slight decrease of 0.21 percentage points relative to Vanilla Replay at $\rho = 0.02$. These results indicate that both methods benefit from larger replay budgets, while OPR-RU consistently mitigates forgetting across the evaluated budget range.

Table 2: Effect of the replay budget on TRACE with Qwen3-4B. We compare Vanilla Replay and OPR-RU at replay ratios $\rho \in \{0.01, 0.02, 0.05, 0.10\}$, corresponding to total buffers of 50, 100, 250, and 500 historical examples per stage. ACC reports average task performance after the full task sequence, and BWT measures backward transfer; higher is better for both. OPR-RU achieves less negative BWT at every budget and higher ACC at three of the four budgets.

Method	ρ	ACC	BWT
Vanilla Replay	0.01	64.46	-2.70
	0.02	65.24	-1.81
	0.05	65.26	-1.21
	0.10	66.57	-0.87
OPR-RU	0.01	64.95	-1.88
	0.02	65.03	-1.32
	0.05	66.14	-0.54
	0.10	67.05	-0.19

5.3.2 Prompt Selection

Prompt-aligned replay retains exactly the historical prompts selected for the OPR-RU replay buffer but pairs them with their corresponding gold targets instead of the model-generated responses. This control holds prompt selection fixed to examine the effect of the replay target source. OPR-LOW uses the same candidate rollout pool and task-specific rule-based scorer as OPR-RU but selects low-scoring prompt–response pairs rather than high-scoring ones. It thus tests the importance of high-score filtering when replay responses are generated by the model itself.

OPR-RU achieves the highest accuracy and the least negative BWT among the four methods, indicating the best final performance and the least forgetting in this comparison. Relative to Prompt-Aligned Replay, OPR-RU improves accuracy by 0.86 points and BWT by 0.65 points. With prompt selection held fixed, these gains support the benefit of using the selected model-generated responses as replay targets. Prompt-Aligned Replay yields only a small BWT improvement over Vanilla Replay, alongside a 0.37-point decrease in accuracy, suggesting that prompt selection alone does not account for OPR-RU's advantage. Moreover, OPR-LOW underperforms OPR-RU by 0.63 accuracy points and 1.50 BWT points, and exhibits more forgetting than Vanilla Replay. Together, these results support the combination of model-generated replay targets and high-score filtering; using model-generated responses alone is insufficient.

5.4 Loss Dynamics

Figure 4 shows that OPR-SC continues to reduce its training loss after SDFT's loss largely plateaus, particularly on MeetingBank, Py150, and 20Minuten. Similar but less pronounced differences appear on C-STANCE, FOMC, and ScienceQA. Both methods substantially reduce their losses on NumGLUE, although OPR-SC fluctuates on NumGLUE-cm.

These curves describe optimization behavior rather than directly establish acquisition or retention: OPR-SC minimizes supervised cross-entropy, whereas SDFT minimizes reverse KL on student-generated responses, so their absolute loss values are not directly comparable. The contrast also

Table 3: Ablation of prompt selection and replay targets on TRACE with Qwen3-4B at $\rho = 0.01$. Prompt-Aligned Replay uses the same historical prompts as OPR-RU but replaces generated responses with gold targets. OPR-LOW uses the same candidate pool and reward scorer as OPR-RU but selects low-scoring responses. Accuracy denotes the average task score after the full task sequence, and BWT measures backward transfer; higher is better for both. OPR-RU achieves the highest accuracy and least negative BWT among these variants, supporting the use of high-scoring generated responses as replay targets.

Method	Accuracy	BWT
Vanilla Replay	64.46	-2.70
Prompt-Aligned Replay	64.09	-2.53
OPR-LOW	64.32	-3.38
OPR-RU	64.95	-1.88

appears on the first task, C-STANCE, before replay begins, and therefore cannot be attributed solely to replay. Downstream performance is assessed through task scores and BWT.

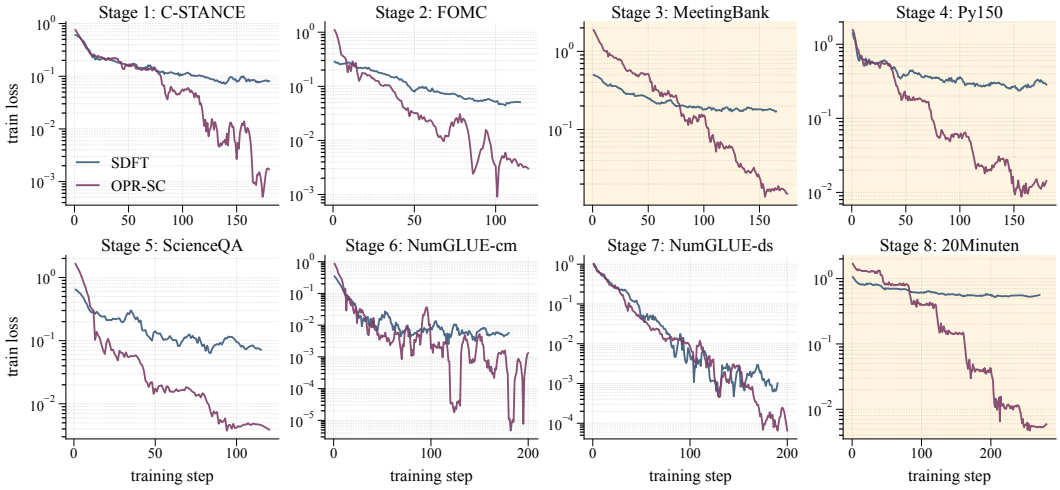


Figure 4: Stage-wise training loss of SDFT and OPR-SC across the eight TRACE tasks. The horizontal axis shows training steps within each stage, and the vertical axis uses a logarithmic scale. Shaded panels highlight MeetingBank, Py150, and 20Minuten, where SDFT shows relatively flat late-stage trajectories while OPR-SC continues to reduce its training objective. Each curve reports the corresponding method's own objective, so absolute loss values should not be compared directly across methods.

6 Conclusion

We introduced On-Policy Replay (OPR), which refreshes historical replay targets with selected responses from the current model and uses them in standard supervised fine-tuning. On TRACE with Qwen3-4B and Qwen3-8B, OPR-RU achieves the highest final average task score among the evaluated methods at a 1% replay budget, while improving backward transfer over vanilla Replay and LAMOL. OPR-SC closely matches this performance without requiring historical labels for replay construction. The prompt-aligned control and low-score selection ablation support the combined value of model-generated targets and high-score filtering. With a 10% replay budget, OPR-RU reaches a BWT of -0.19 on Qwen3-4B, indicating little average degradation relative to the scores obtained immediately after learning each task.

7 LIMITATION

Our evaluation is limited to one benchmark, its canonical task order, and two model scales from the same family. OPR also requires access to historical prompts and additional generation at task boundaries; confidence-based selection does not guarantee response correctness. Future work should examine alternative task orders, longer task sequences, and the trade-off between replay generation cost and retention. These results support refreshing historical supervision with selected model responses as a practical approach to continual SFT.

AI USE STATEMENT

We used generative AI tools to assist with manuscript writing and code generation, including OpenAI Codex for drafting the conclusion and accompanying statements. All AI-assisted text and generated code were manually reviewed and verified by the authors. Beyond these writing and coding tasks, we did not use generative AI tools as research assistants. The model-generated replay responses studied as part of OPR are described in the main text. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the assistance of generative AI.

ETHICS STATEMENT

This work studies continual learning using the existing TRACE benchmark and Qwen3 model backbones. Replaying model-generated responses may preserve or amplify errors and biases in the underlying models and data. Neither task-reward filtering nor confidence-based selection constitutes a safety or fairness guarantee, and benchmark retention scores do not establish suitability for deployment in sensitive settings. Although OPR-SC does not require historical labels for replay construction, it retains historical prompts and therefore does not eliminate privacy risks associated with storing or processing those prompts. Applications of OPR should respect the usage conditions of the underlying datasets and models and assess the privacy, bias, and safety implications of the intended setting.

REPRODUCIBILITY STATEMENT

Section 5.1 describes the benchmark tasks and their order, model backbones, training-set subsampling, optimizer settings, training schedule, sequence length, numerical precision, evaluation protocol, and aggregate metrics. Section 5.3.1 reports the evaluated replay budgets, and Section 5.3.2 describes the prompt-aligned and low-score selection controls. These sections document the experimental configuration and comparisons reported in this work.

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, 2024. URL https://arxiv.org/abs/2306.13649.

Howard Chen, Noam Razin, Karthik Narasimhan, and Danqi Chen. Retaining by doing: The role of on-policy data in mitigating forgetting, 2026. URL https://arxiv.org/abs/2510.18874.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V. Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training, 2025. URL https://arxiv.org/abs/2501.17161.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models, 2022. URL https://arxiv.org/abs/2210.11416.

Sonam Gupta, Yatin Nandwani, Asaf Yehudai, Dinesh Khandelwal, Dinesh Raghu, and Sachindra Joshi. Selective self-to-supervised fine-tuning for generalization in large language models, 2025. URL https://arxiv.org/abs/2502.08130.

Hugo Hazard, Zafeirios Fountas, Martin A. Benfeghoul, Adnan Oomerjee, Jun Wang, and Haitham Bou-Ammar. Sure: Surprise-driven prioritised replay for continual llm learning, 2025. URL https://arxiv.org/abs/2511.22367.

Yebowen Hu, Tim Ganter, Hanieh Deilamsalehy, Franck Dernoncourt, Hassan Foroosh, and Fei Liu. MeetingBank: A benchmark dataset for meeting summarization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023. URL https://arxiv.org/abs/2305.17529.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13): 3521–3526, March 2017. ISSN 1091-6490. doi: 10.1073/pnas.1611835114. URL http://dx.doi.org/10.1073/pnas.1611835114.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning, 2022. URL https://arxiv.org/abs/1706.08840.

Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. URL https://arxiv.org/abs/2209.09513.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. An empirical study of catastrophic forgetting in large language models during continual fine-tuning, 2025. URL https://arxiv.org/abs/2308.08747.

Swaroop Mishra, Arindam Mitra, Neeraj Varshney, Bhavdeep Sachdeva, Peter Clark, Chitta Baral, and Ashwin Kalyan. NumGLUE: A suite of fundamental yet challenging mathematical reasoning tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2022. URL https://arxiv.org/abs/2204.05660.

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, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

Veselin Raychev, Pavol Bielik, and Martin Vechev. Probabilistic model for code with decision trees. In *Proceedings of the 2016 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA)*, 2016. doi: 10.1145/2983990.2984041.

Weijieying Ren, Xinlong Li, Lei Wang, Tianxiang Zhao, and Wei Qin. Analyzing and reducing catastrophic forgetting in parameter efficient tuning, 2024. URL https://arxiv.org/abs/2402.18865.

Annette Rios, Nicolas Spring, Tannon Kew, Marek Kostrzewa, Andreas Säuberli, Mathias Müller, and Sarah Ebling. A new dataset and efficient baselines for document-level text simplification in German. In *Proceedings of the Third Workshop on New Frontiers in Summarization (NewSum@EMNLP)*, 2021. URL https://aclanthology.org/2021.newsum-1.16/.

Agam Shah, Suvan Paturi, and Sudheer Chava. Trillion dollar words: A new financial dataset, task & market analysis. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023. URL https://arxiv.org/abs/2305.07972.

Idan Shenfeld, Jyothish Pari, and Pulkit Agrawal. Rl's razor: Why online reinforcement learning forgets less, 2025. URL https://arxiv.org/abs/2509.04259.

Idan Shenfeld, Mehul Damani, Jonas Hübotter, and Pulkit Agrawal. Self-distillation enables continual learning, 2026. URL https://arxiv.org/abs/2601.19897.

Fan-Keng Sun, Cheng-Hao Ho, and Hung-Yi Lee. Lamol: Language modeling for lifelong language learning, 2019. URL https://arxiv.org/abs/1909.03329.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. Orthogonal subspace learning for language model continual learning, 2023a. URL https://arxiv.org/abs/2310.14152.

Xiao Wang, Yuansen Zhang, Tianze Chen, Songyang Gao, Senjie Jin, Xianjun Yang, Zhiheng Xi, Rui Zheng, Yicheng Zou, Tao Gui, Qi Zhang, and Xuanjing Huang. Trace: A comprehensive benchmark for continual learning in large language models, 2023b. URL https://arxiv.org/abs/2310.06762.

Tongtong Wu, Linhao Luo, Yuan-Fang Li, Shirui Pan, Thuy-Trang Vu, and Gholamreza Haffari. Continual learning for large language models: A survey, 2024. URL https://arxiv.org/abs/2402.01364.

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, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Yunan Zhang, Shuoran Jiang, Mengchen Zhao, Yuefeng Li, Yang Fan, Xiangping Wu, and Qingcai Chen. Gere: Towards efficient anti-forgetting in continual learning of llm via general samples replay, 2025. URL https://arxiv.org/abs/2508.04676.

Chenye Zhao, Yingjie Li, and Cornelia Caragea. C-STANCE: A large dataset for Chinese zero-shot stance detection. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023. URL https://aclanthology.org/2023.acl-long.747/.

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, 2026. URL https://arxiv.org/abs/2601.18734.

Junhao Zheng, Xidi Cai, Shengjie Qiu, and Qianli Ma. Spurious forgetting in continual learning of language models, 2025. URL https://arxiv.org/abs/2501.13453.

Taojie Zhu, Dongyang Xu, Ding Zou, Sen Zhao, Qiaobo Hao, Zhiguo Yang, and Yonghong He. Bridging sft and rl: Dynamic policy optimization for robust reasoning, 2026. URL https://arxiv.org/abs/2604.08926.

A TRACE TASKS, TEMPLATES, AND METRICS

This section describes the eight TRACE tasks, their input/output conventions, and evaluation metrics. It complements the experimental setup in Section 5.1. The original sources of the eight tasks are C-STANCE (Zhao et al., 2023), FOMC (Shah et al., 2023), MeetingBank (Hu et al., 2023), Py150 (Raychev et al., 2016), ScienceQA (Lu et al., 2022), NumGLUE (Mishra et al., 2022) (from which TRACE's NumGLUE-cm and NumGLUE-ds splits are taken), and 20Minuten (Rios et al., 2021).

A.1 TASK SUMMARY

Dataset	Source	Avg len	Metric	Language	Size
C-STANCE	Social media	109	Accuracy	Chinese	5000
FOMC	Finance	76	Accuracy	English	5000
MeetingBank	Meeting	3550	ROUGE-L	English	5000
Py150	GitHub	814	Edit similarity	Python	5000
ScienceQA	Science	285	Accuracy	English	5000
NumGLUE-cm	Math	50	Accuracy	English	5000
NumGLUE-ds	Math	39	Accuracy	English	5000
20Minuten	News	739	SARI	German	5000

Table 4: An overview of the dataset statistics in TRACE. "Source" denotes the origin of the context, "Avg len" indicates the average example length in tokens, and "Size" is the number of training examples sampled per task. SARI measures text simplification quality.

The tasks differ in both sequence length and evaluation metric. The reported average lengths range from 39 tokens for NumGLUE-ds to 3,550 tokens for MeetingBank, so a buffer budget measured in examples can correspond to different token counts across tasks. The replay budget is shared across historical tasks: at $\rho = 0.01$ and $n_i = 5{,}000$, the total buffer contains 50 examples. OPR-RU ranks candidates separately within each task using its corresponding metric. This task-wise ranking supports heterogeneous metrics without requiring their reward values to be calibrated or averaged across tasks. NumGLUE-cm and NumGLUE-ds are scored by extracting the final numerical answer and applying exact matching; MeetingBank uses ROUGE-L, Py150 uses edit similarity, and 20Minuten uses SARI.

A.2 INPUT/OUTPUT EXAMPLES

The examples below follow the task order in Table 4 and illustrate the instructions and target formats. C-STANCE and 20Minuten are shown in English translation for presentation; their training and evaluation use the original Chinese and German text, respectively. The Py150 entry shows the schematic code-context/completion format. The examples include task instructions, contextual inputs, and reference answers. The longer MeetingBank and 20Minuten passages illustrate the summarization and simplification formats.

Example for C-STANCE (English translation)

Query: Determine the stance of the following text toward the given target. Choose one of: A. Support, B. Oppose, C. Neutral. Output only A, B, or C.
Text: Starting bilingual language training for kids from an early age is actually feasible. They can at least handle daily conversations with ease. However, sufficient language input and output must be guaranteed. For example, one parent speaks one language while the other uses the second, and children should respond in the corresponding language. Otherwise, they will end up understanding the language but being unable to speak it.
Target: bilingual parenting for early language training
Answer: A

Example for FOMC

Query: What is the monetary policy stance for the following text? A. dovish, B. hawkish, C. neutral. Choose one from A, B and C.
Text: First, tightening will in all likelihood occur in the context of a more firmly established economic recovery in the United States so that any adverse effects on EME financial conditions should be buffered by the beneficial effects of higher external demand.
Answer: B

Example for MeetingBank

Query: Write a summary of the following meeting transcripts.
Meeting transcripts:Councilmember Clark, would you please put resolution 2a1 on the floor for adoption.As council president of the council, resolution 201 be adopted.Thank you. We've got a mover in a second. Comments and questions by members of Council on Resolution 201. Councilmember Cde-Baca.Thank you. I just want to go on record voting no for this tonight. Pepsi is not necessarily a group that needs $1,000,000 incentive to stay in Denver, and they are well on their way to completing their move with or without this incentive.So definitely just want to make sure that I'm clear in not supporting this tonight. Thank you.Thank you. And I'll go ahead. And since this is in my council district, District 11, we as community members, along with some of the original steering committee members from the far northeast neighborhood plan, reconvened and talked through all of the pros and possible challenges of welcoming a large scale manufacturer into District 11.And the community saw fit that this was a great use for this area and this will bring close to 500 jobs to the region as well. And I would ask my colleagues to support it tonight. Madam Secretary, roll call on Council Resolution 22, Dash 201, please.See tobacco? No. Clark.I.Swen.I. Herndon, I.Hi. All right. Cashman. Kenny Ortega. Sandoval.I.Sawyer. I. Torres. I. Black. Madam President, I.Madam Secretary, please close the voting and announce the results.One day, 11 eyes.11 Eyes. Council Resolution 22, dash 201 has been adopted. Madam Secretary, please put the next item up on our screens. Councilmember Flynn, go ahead with your comments on resolution 159, please.
Answer: A resolution approving a proposed Business Incentive Fund (BIF) contract with Bottling Group, LLC for an amount not to exceed $1,000,000 for 7 years and for performance based criteria that include establishment of a state-of-the-art manufacturing facility and associated significant investments, as well as job retention and creation and associated investments into upskilling and training of the manufacturing facility's Denver-based workforce.Approves a Business Incentive Fund (BIF) contract with Bottling Group, LLC for an amount not to exceed $1,000,000 and for 7 years through 12-31-2028 for performance based criteria that include establishment of a state-of-the-art manufacturing facility and associated significant investments, as well as job retention and creation and associated investments into upskilling and training of the manufacturing facility's Denver-based workforce (OEDEV-202160849-00). The last regularly scheduled Council meeting within the 30-day review period is on 4-11-2022. The Committee approved filing this item at its meeting on 3-9-2022.

Py150 input/output format

Query: *[preceding Python code context]*
Answer: *[reference code completion]*

Example for ScienceQA

Query: Choose an answer for the following question and give your reasons.
Question: What information supports the conclusion that Trent acquired this trait?
Hint: Read the description of a trait. Trent knows how to fly a kite.
Choices: A. Trent's neighbor taught him how to repair a kite. B. Trent likes to fly a kite with his younger brother. C. Trent's friend taught him how to fly a kite.
Answer: C. Organisms, including people, have both inherited and acquired traits.
Inherited and acquired traits are gained in different ways. Inherited traits are passed down from biological parents to their offspring through genes. Genes are pieces of hereditary material that contain the instructions that affect inherited traits. Offspring receive their genes, and therefore gain their inherited traits, from their biological parents. Inherited traits do not need to be learned.
Acquired traits are gained during a person's life. Some acquired traits, such as riding a bicycle, are gained by learning. Other acquired traits, such as scars, are caused by the environment. Parents do not pass acquired traits down to their offspring.

Example for NumGLUE-cm

Query: Solve the following math problem.
Question: In a car parking lot, Greyson observed that there are 48 car wheels. Find out the number of cars in the parking lot.
Answer: 12

Example for NumGLUE-ds

Query: Solve the following math problem.
Question: How many moles of Ammonium iodide are required to react with 3 moles of Potassium hydroxide to form 3 moles of Ammonia, 3 moles of Potassium iodide and 3 moles of Water.
Answer: 3

Example for 20Minuten (English translation)

Query: Provide a simplified version of the following paragraph in German.
Paragraph: In Lyss (Bern), police raided an illegal gambling den and arrested two women aged 30 and 41. Around 40 people were checked on Friday evening on suspicion of illegal gambling and betting. According to the Bernese Jura-Seeland Regional Public Prosecutor's Office and Bern Cantonal Police, the two women are believed to have worked in the premises without a valid work permit and will be charged with violations of the Foreign Nationals and Integration Act. Authorities confirmed that prohibited gambling and betting services were offered at the site. During the inspection and a subsequent house search, officers seized various electronic devices, a slot machine, tens of thousands of Swiss francs in cash and suspected gold items. The operation followed extensive prior investigations. Public prosecutors are still looking into the operators of the premises. Illegal gambling and sports betting raids have taken place in Lyss multiple times before. Back in 2018, 25 people were checked at another venue, with cash, betting computers and slot machines confiscated. Similar operations were also carried out in 2016 and 2015.
Answer: In Lyss, the Bern Cantonal Police conducted a targeted inspection. Cash was seized and two women were arrested during the operation. Similar inspections had already taken place in previous years.

B Training Details

All runs use the `swift sft` entry point of ms-swift with full-parameter fine-tuning, on 8×H100 (80 GB) GPUs under DeepSpeed ZeRO-2 and Liger Kernel, in bf16. We apply AdamW with learning rate 1×10^{-5}, cosine schedule, no warmup, weight decay 0, per-device batch size 16 (global 128), and max sequence length 2048. Per-task epochs follow $[5, 3, 7, 5, 3, 5, 5, 7]$. Only the final checkpoint of each stage is retained and used as the initialization of the next stage. The optimizer settings and training schedule are shared across methods, while each method retains its own data construction and training objective. The recorded training cost for a single model over the eight tasks is 32 GPU-hours. The Qwen3 task scores in the main tables are averages over four evaluation runs (Avg@4).

C Rollout for OPR

Between consecutive stages, the most recent checkpoint π_{θ_i} generates one response per historical prompt using **vLLM**. The tensor parallel size is 4, sampling temperature $\tau = 0.1$, and nucleus sampling parameter top-p at its default 1, with a maximum of 512 newly generated tokens. The rollout runs as a separate subprocess from the training step, which uses 8-way data parallelism, so the rollout GPUs need not hold the optimizer state.

For each prior task T_j, the prompts are identical to the $5{,}000$ training prompts used at stage j. OPR-RU retains the historical reference answers for reward computation, while OPR-SC scores responses without historical labels. Both variants use the selected generated responses as replay targets. Prompts whose chat-templated length exceeds 2048 tokens are removed by an overlong filter, and the same filter is consistently applied during both training and evaluation.

Given a global budget $b = \lfloor \rho n_{i+1} \rfloor = \lfloor \rho \cdot 5000 \rfloor$, the buffer at stage $i{+}1$ is divided among the i prior tasks as evenly as possible, subject to the constraint that the allocation for any single task does not

exceed the number of its eligible candidates. Within each task, the highest-scoring prompt–response pairs fill the assigned quota. The buffer is constructed once at the task boundary and remains fixed during the next training stage.