Trust Region Continual Learning as an Implicit Meta-Learner

Zekun Wang* **Anant Gupta*** **Christopher J. MacLellan**
Georgia Institute of Technology
{zekun,agupta886,cmaclell}@gatech.edu

Abstract

Continual learning aims to acquire tasks sequentially without catastrophic forgetting, yet standard strategies face a core tradeoff: regularization-based methods (e.g., EWC) can overconstrain updates when task optima are weakly overlapping, while replay-based methods can retain performance but drift due to imperfect replay. We study a hybrid perspective: *trust region continual learning* that combines generative replay with a Fisher-metric trust region constraint. We show that, under local approximations, the resulting update admits a Model-Agnostic Meta-Learning (MAML) interpretation with a single implicit inner step: replay supplies an old-task gradient signal (query-like), while the Fisher-weighted penalty provides efficient offline curvature shaping (support-like). This yields an emergent meta-learning property in continual learning: the model becomes an initialization that rapidly *re-converges* to prior task optima after each task transition, without explicitly optimizing a bilevel objective that couples inner adaptation with outer meta-updates. Empirically, on task-incremental diffusion image generation and continual diffusion-policy control, trust region continual learning achieves the best final performance and retention, and accelerates early-task recovery compared with EWC, replay, and continual meta-learning baselines.

1 Introduction

Learning is rarely stationary. Models face shifting data streams and evolving environments. Continual learning studies how to acquire new tasks sequentially while remaining robust on previously learned tasks without repeatedly retraining from scratch. However, neural networks trained with standard gradient-based updates can suffer severe catastrophic forgetting, motivating methods that explicitly stabilize old knowledge while learning new skills. Two families of methods dominate continual learning. Regularization-based approaches preserve earlier tasks by penalizing changes to important parameters, often using Fisher-like quadratic constraints [22, 47, 1]. Yet, they rely critically on both the existence of shared parameters across tasks and the fidelity of their curvature approximations, conditions that often fail under distribution shift or crude Fisher estimates. Replay-based approaches retain performance by interleaving current training with stored or generated past samples [33, 36], but are limited by memory policies and can drift when generations are imperfect and bootstrapped [48].

A natural extension is to combine replay and regularization so that replay encourages parameter sharing while regularization constrains replay-induced drift. Recent diffusion-based generative models [17] make this combination particularly effective. Diffusion models can produce high-quality replay samples, and their gradient geometry can yield substantially improved Fisher approximations: [40] shows that diffusion models admit an approximately rank-1 empirical Fisher, enabling a cheap yet informative Fisher estimate and making elastic weight consolidation (EWC) [22] a strong complement to replay. Inspired by this line of work, we adopt a trust region view of EWC and replay synergy:

*Equal contribution.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

EWC as a Fisher-metric trust region while replay supplies the gradient signal that drives parameter updates, analogous in spirit to trust region methods that limit steps under local Kullback–Leibler divergence [35].

In this paper, we analyze the learning dynamics induced by such updates on diffusion models. We show that, under local approximations, the resulting update takes the form of Model-Agnostic Meta-Learning (MAML) with one inner adaptation step [7]: replay supplies a query gradient on past data, while the EWC Fisher forms an efficient offline approximation to the second-order Hessian matrix on the past support set. This view links continual learning and meta-learning at the level of optimization dynamics. Even without an explicit bilevel procedure that couples inner adaptation with outer meta-updates [7], the continual learning update can yield an initialization-like property that supports faster re-adaptation on prior tasks. Unlike continual meta-learning methods that begin with an explicit bi-level objective and then adapt it to sequential tasks [8, 32, 19, 14, 42], we start from this continual learning objective and analyze what meta-learning structure it induces. Empirically, we test whether trust region continual learning yields *faster re-convergence* under sequential training in two regimes: (i) low-heterogeneity task-incremental diffusion image generation on ImageNet-500, and (ii) high-heterogeneity diffusion policy control on Continual-World-10. We compare against continual-learning baselines (generative replay [26], EWC with rank-1 Fisher [22, 40], and naive fine-tuning) and continual meta-learning baselines (follow the meta leader (FTML) [8], variance reduced meta-continual learning (VR-MCL) [44]). Across both domains, trust region continual learning re-converges on earlier tasks faster and improves retention, supporting our claim that Fisher-metric trust region constraints can induce emergent meta-learning behavior.

Our contributions are: (i) We frame the combination of EWC and replay as trust region continual learning, where replay supplies old-task gradients and the EWC pulls updates to be near old-task optima [22, 35]; (ii) Under local approximations, we show the trust region continual learning takes a one-step MAML form, exposing an implicit meta-objective over past tasks [7]; (iii) On diffusion image generation and diffusion policy control, we benchmark against EWC-only, replay-only, and continual meta-learning [8, 44], showing faster early re-convergence with competitive retention. More broadly, we propose a robust, scalable, and efficient approach that bridges continual learning and meta-learning.

2 Related Work

2.1 Continual Learning

Continual learning studies training on a stream of tasks while maintaining performance on earlier tasks, where catastrophic forgetting is a central challenge [28, 9]. A major family of methods mitigates forgetting by constraining parameter drift, e.g., EWC via a Fisher-weighted quadratic penalty [22] and synaptic intelligence via online importance estimates accumulated over training trajectories [47]. Another family relies on replay, interleaving stored examples from prior tasks with current-task updates [31, 2]. Hybrid combinations of replay and regularization [15, 40] are also consistent with consolidation and replay perspectives from cognitive science [27].

Generative replay replaces explicit storage of past data with a learned generator that synthesizes pseudo-samples from previous tasks, reducing memory demands and enabling flexible replay schedules [36, 39]. Diffusion models are particularly attractive generators due to their high-fidelity synthesis; diffusion-based generative replay has been explored in class-incremental generation and dense prediction [10, 3, 20]. Since the generator itself is continually updated, stabilizing the reverse denoising dynamics becomes important; generative distillation addresses this by distilling the reverse chain across timesteps [26]. Building on this line, we study hybrid continual diffusion training that combines diffusion replay with stronger curvature approximations derived from diffusion gradients [40]. Online Curvature-Aware Replay (OCAR) [38] uses online Fisher preconditioning, while we combine replay with rank-1 Fisher constraints around past task optima and analyze the resulting connection to MAML.

2.2 Meta-Learning and Continual Meta-Learning

Meta-learning ("learning to learn") aims to acquire inductive biases from a distribution of tasks so that a model can adapt rapidly to a new task using only a few examples [7, 18]. A common formulation is

bilevel optimization: for task $\mathcal{T} \sim p(\mathcal{T})$, an inner loop adapts parameters using task training data, and an outer loop updates meta-parameters to minimize post-adaptation loss. With one inner step, the objective is:

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{\mathcal{T}\sim p(\mathcal{T})}\Big[\mathcal{L}_{\mathcal{T}}\big(\theta'; \mathcal{D}^{\mathrm{te}}_{\mathcal{T}}\big)\Big] \quad \text{s.t.} \quad \theta' \;=\; \theta - \alpha\nabla_{\theta}\mathcal{L}_{\mathcal{T}}(\theta; \mathcal{D}^{\mathrm{tr}}_{\mathcal{T}}). \tag{1}$$

where $\mathcal{D}^{\mathrm{tr}}_{\mathcal{T}}$ and $\mathcal{D}^{\mathrm{te}}_{\mathcal{T}}$ are the task's training and test splits, θ is the meta-initialization, and θ' is obtained by taking a training gradient step of size α. The outer objective minimizes the expected test loss at θ' [7]. Follow-up work improves scalability by avoiding higher-order derivatives [29] or by learning the update rule itself [24].

Continual meta-learning extends this paradigm to settings where tasks arrive sequentially (often with distributional shift), requiring the meta-learner to accumulate reusable knowledge while remaining adaptable. Most prior continual meta-learning methods start from an explicit meta-learning objective and adapt it to sequential task streams, often using replay, online meta-updates, or meta-regularization [8, 32, 19, 14]. [44] further bridges meta-learning and continual learning approaches through Hessian approximation. Regularization-based continual learning typically uses fixed (often diagonal) curvature surrogates for past tasks, whereas meta-gradient updates implicitly maintain an online Hessian estimate whose variance can be reduced via improved memory sampling [44]. In our analysis, EWC provides an efficient offline Fisher surrogate for this curvature, especially for diffusion models where the empirical Fisher closely tracks local Hessian structure [40]. Unlike prior bi-level formulations, we start from a practical continual learning objective and analyze the meta-learning structure it already induces.

3 Continual Learning in Trust Regions Induces Meta-Learning

In this section, we first define our continual learning setup and where the meta-learning behavior could emerge in section 3.1, then recast continual learning as learning within a trust region around task optima in section 3.2, and finally reinterpret the resulting updates as a MAML-style optimization procedure in continual learning in Section 3.3.

3.1 Continual Learning Problem Set-Up

In continual learning, the learner observes a stream of tasks $\{\mathcal{T}_t\}_{t=1}^{T}$ sequentially. Each task $\mathcal{T}_t$ comes with a training set $\mathcal{D}_t^{\mathrm{tr}}$ and a disjoint evaluation set $\mathcal{D}_t^{\mathrm{te}}$. Let f_θ denote the model and let $\mathcal{L}_t(\theta; \mathcal{D})$ be the empirical loss on task t evaluated on dataset $\mathcal{D}$. The standard continual learning objective focuses on retention: after finishing task t, we want good performance on all tasks seen so far, e.g., low $\frac{1}{t}\sum_{i=1}^{t}\mathcal{L}_i(\theta_t; \mathcal{D}_i^{\mathrm{te}})$, while learning the new task.

Beyond final retained performance, we study a re-convergence phenomenon: during the learning of new tasks, performance on an old task $\mathcal{T}_i$ may drop; yet a strong continual learning solution should recover its earlier good performance quickly. We say θ_t exhibits fast recovery on task $\mathcal{T}_i$ if the evaluation performance returns near the task's previously achieved level using fewer update steps.

Fast recovery also means that θ_t functions as an initialization from which few gradient steps rapidly (re-)attain good performance on old tasks. This is the same structural property optimized in gradient-based meta-learning, which finds parameters that enable fast adaptation under a fixed inner-loop rule [7]. Our setting studies an analogous fast adaptation problem, but under the continual learning constraint that tasks arrive sequentially and must not be forgotten. The key difference is that traditional few-shot evaluation of meta-learning typically targets fast adaptation to new tasks drawn i.i.d. from a task distribution. In contrast, our "tasks of interest" are the previously encountered tasks, and θ_t must simultaneously acquire new tasks and preserve the ability to rapidly re-converge on old tasks.

3.2 Continual Learning in Trust Regions

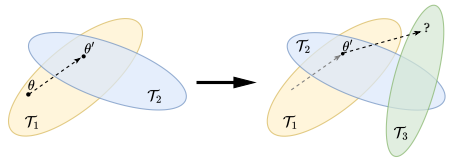
We first revisit two canonical continual learning strategies: EWC and generative replay through the lens of trust region optimization in parameter space. Throughout, let $\theta_i^\star$ denote the end-of-task parameters after learning $\mathcal{T}_i$ in the continual run. For replay-based methods, this training uses the mixture of current-task and replay data. Let $F^{(i)}_{\theta_i^*}$ be the Fisher information (evaluated at $\theta_i^\star$) used to quantify how sensitive task $\mathcal{T}_i$ is to perturbations of θ.

EWC as a quadratic trust region around past optima. From a Bayesian update perspective, EWC can be derived by applying a Laplace approximation to the previous posterior around the past optimum $\theta_i^\star$. The resulting objective for learning task $\mathcal{T}_t$ takes the form $\min_\theta \ \mathcal{L}_{\mathcal{T}_t}(\theta; \mathcal{D}_t) \ + \ \frac{\lambda}{2}\sum_{i=1}^{t-1}(\theta-\theta_i^\star)^\top F_{\theta_i^*}^{(i)}(\theta-\theta_i^\star)$, where $\lambda > 0$ trades off plasticity and stability. This is the Lagrangian relaxation of an explicit trust region constraint:

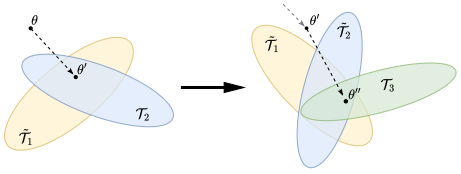
$$\min_\theta \ \mathcal{L}_{\mathcal{T}_t}(\theta; \mathcal{D}_t) \quad \text{s.t.} \quad \sum_{i=1}^{t-1}(\theta-\theta_i^\star)^\top F_{\theta_i^*}^{(i)}(\theta-\theta_i^\star) \le \delta, \tag{2}$$

for some radius $\delta > 0$. Intuitively, EWC pulls updates toward an ellipsoidal neighborhood around previous task optima, penalizing movement against the directions that are deemed important by the Fisher.

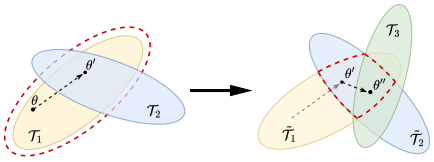
In practice, forming the full $F_{\theta_i^*}^{(i)}$ is often infeasible, and EWC typically uses a diagonal approximation $F_{\theta_i^*}^{(i)} \approx \mathrm{diag}(F_{\theta_i^*}^{(i)})$. This simplification can be especially limiting in diffusion models where curvature may concentrate in off-diagonal structure, making diagonal constraints weak or even uninformative [40]. A second limitation is geometric: if the new task optimum lies outside (or far from) the intersection of these trust regions, then the feasible set in Equation (2) can be effectively empty, yielding the failure mode illustrated in Figure 1a.



(a) Elastic Weight Consolidation-based approach



(b) Generative replay-based approach



(c) Trust region continual learning

Figure 1: Illustrations of parameter updates when learning a new task $\mathcal{T}_3$ under different continual-learning strategies. Colored ellipses denote low-loss regions for each task. Dark arrows show the current update direction from θ, gray arrows indicate prior trajectory, and the red dashed region indicates the trust region constraint.

Generative replay as optimizing on a union of task data. Generative replay-based continual learning instead trains on a mixture of current-task data and replayed samples from past tasks. Abstractly, let $\tilde{\mathcal{D}}_k$ denote replay data approximating task $\mathcal{T}_k$. A generic replay objective is:

$$\min_\theta \ \mathcal{L}_{\mathcal{T}_t}(\theta; \mathcal{D}_t) \ + \ \beta \sum_{i=1}^{t-1} \mathcal{L}_{\mathcal{T}_i}(\theta; \tilde{\mathcal{D}}_i), \tag{3}$$

with $\beta \ge 0$ controlling the replay ratio.

Replay addresses EWC's "disjoint optima" issue by explicitly encouraging solutions that perform well on the *union* of (current + replayed) data. This effectively explores shared low-dimensional data manifolds supported by all task datasets, often enabling convergence to a basin that overlaps across tasks (Figure 1b). However, replay can drift due to imperfect generators or compounding approximation error: even if the model continues to fit replayed samples, small replay mismatches can accumulate and move parameters toward a different-but-valid basin that is far from earlier optima in overparameterized networks, requiring more updates to converge to new optima.

Trust region replay and the role of a stronger Fisher. One can combine the complementary strengths of replay and EWC by enforcing trust region updates while replay promotes cross-task overlap:

$$\min_\theta \ \mathcal{L}_{\mathcal{T}_t}(\theta; \mathcal{D}_t) \ + \ \beta\, \mathcal{L}_{\text{Replay}}(\theta) \ + \ \lambda \mathcal{L}_{\text{EWC}}. \tag{4}$$

This objective matches the geometry in Figure 1c: replay pulls optimization toward regions that remain good for previous tasks (making the trust region non-vacuous), while the Fisher-weighted penalty anchors the updates to be around past task optima.

The effectiveness of Equation (4) depends critically on the quality of $F^{(i)}_{\theta^*_i}$. Following [40], diffusion models admit a cheap but informative Fisher approximation: in the later diffusion timesteps, per-sample gradients become strongly collinear with their mean, so the empirical Fisher is effectively rank-1. As a result, replay makes a shared basin available, while the rank-1 trust region makes movement within that basin stable by explicitly constraining the dominant curvature direction. This trust region view will be the key bridge in Section 3.3, where we reinterpret the resulting continual updates as a MAML-style optimization.

3.3 Reinterpretation as a MAML-style meta-learning

We now ask *what* our continual objective in Equation (4) is implicitly learning. Taking one gradient step on Equation (4) yields:

$$\theta \;\leftarrow\; \theta \;-\; \eta\Big(\underbrace{\nabla_\theta \mathcal{L}_{\mathcal{T}_t}(\theta;\mathcal{D}_t)}_{\textbf{(A) current-task fit}} \;+\; \sum_{i=1}^{t-1} \underbrace{\beta \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(\theta;\tilde{\mathcal{D}}_i)}_{\textbf{(B) old-task replay}} \;+\; \sum_{i=1}^{t-1} \underbrace{\lambda F^{(i)}_{\theta^*_i}(\theta-\theta^*_i)}_{\textbf{(C) old-task EWC}}\Big), \tag{5}$$

where η is the step size, $\tilde{\mathcal{D}}_i$ denotes generative replayed data for old task $\mathcal{T}_i$, and $F^{(i)}_{\theta^*_i}$ is the empirical Fisher estimated from task $\mathcal{T}_i$.

For standard MAML with a single inner step ($k = 1$), the adapted parameters for task $\mathcal{T}_i$ are $\theta'_i = \theta - \alpha\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(\theta;\mathcal{D}^{\mathrm{tr}}_{\mathcal{T}_i})$. The outer update minimizes the post-adaptation (query) loss:

$$\theta \leftarrow \theta \;-\; \eta \sum_{i=1}^{t-1} \left(I - \alpha H^{\mathrm{tr}}_{\theta_i}\right) \nabla_{\theta'} \mathcal{L}_{\mathcal{T}_i}\left(\theta'_i;\mathcal{D}^{\mathrm{te}}_{\mathcal{T}_i}\right),$$

where $H^{\mathrm{tr}}_{\theta_i} = \nabla^2_\theta \mathcal{L}_{\mathcal{T}_i}(\theta;\mathcal{D}^{\mathrm{tr}}_{\mathcal{T}_i})$. Expanding the product:

$$\theta \leftarrow \theta \;-\; \eta\Big(\sum_{i=1}^{t-1} \underbrace{\nabla_{\theta'} \mathcal{L}_{\mathcal{T}_i}\left(\theta'_i;\mathcal{D}^{\mathrm{te}}_{\mathcal{T}_i}\right)}_{\textbf{(I) \textit{query} gradient at adapted params}} \;-\; \sum_{i=1}^{t-1} \underbrace{\alpha H^{\mathrm{tr}}_{\theta_i} \nabla_{\theta'} \mathcal{L}_{\mathcal{T}_i}\left(\theta'_i;\mathcal{D}^{\mathrm{te}}_{\mathcal{T}_i}\right)}_{\textbf{(II) \textit{support} curvature correction}}\Big). \tag{6}$$

We compare the **old-task** part of the trust region update directly with the MAML meta-gradient. The connection to MAML relies on a trust region-style observation: for many old tasks $i < t$, the continual update keeps θ in a low-loss neighborhood around θ^*_i (the low-error region in Figure 1c), so evaluating old-task gradients at θ'_i versus at θ makes little difference to first order [12]. Moreover, replayed samples provide a practical proxy for held-out performance *query/test* on old tasks. With these, evaluating the old-task query gradient at θ'_i is well-approximated by evaluating it at θ:

$$\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(\theta;\tilde{\mathcal{D}}_i) \;\approx \nabla_{\theta'} \mathcal{L}_{\mathcal{T}_i}(\theta'_i;\mathcal{D}^{\mathrm{te}}_{\mathcal{T}_i}) \;= \mathbf{(I)}, \tag{7}$$

Define the old-task displacement $\delta_i := \theta - \theta^*_i$. A local Taylor expansion of the old-task query gradient around θ^*_i gives:

$$\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(\theta;\mathcal{D}^{\mathrm{te}}_{\mathcal{T}_i}) \;\approx\; H^{\mathrm{te}}_{\theta^*_i}\,\delta_i. \tag{8}$$

For negative log-likelihood objectives, the Fisher information equals the expected Hessian under the model distribution. Let $\ell(\theta;x) = -\log p_\theta(x)$ and $g(\theta;x) = \nabla_\theta \ell(\theta;x)$. Taking expectations over $x \sim p_\theta$ gives:

$$F_\theta = \mathbb{E}_x\left[g(\theta;x)g(\theta;x)^\top\right] = \mathbb{E}_x\left[\nabla^2_\theta \ell(\theta;x)\right] = \mathbb{E}_x\left[H_\theta\right], \tag{9}$$

with a full derivation in Appendix A. This identity motivates using Fisher curvature for diffusion training [37, 17]. In practice, MAML computes $H^{\mathrm{tr}}_{\theta_i}$ from support-set mini-batches. Under locality, curvature varies slowly within the neighborhood, so $H^{\mathrm{tr}}_{\theta_i} \approx H^{\mathrm{tr}}_{\theta^*_i}$, consistent with the observed stability of dominant Hessian directions near converged optima [12]. Moreover, we assume $H^{\mathrm{tr}}_{\theta^*_i} \approx H^{\mathrm{te}}_{\theta^*_i}$ following the insight that the Fisher/Hessian characterizes the intrinsic geometric properties of the model's learned predictive distribution [25]. Consequently, the local curvature remains similar across any samples (replayed or real) that align with this underlying manifold. Therefore, by Equation (9), both can be approximated by a common Fisher curvature proxy evaluated at the old optimum:

$$H^{\mathrm{tr}}_{\theta_i} \;\approx\; H^{\mathrm{tr}}_{\theta^*_i} \;\approx\; H^{\mathrm{te}}_{\theta^*_i} \;\approx\; F^{(i)}_{\theta^*_i}. \tag{10}$$

Substituting Equations (8) and (10) into Equation (5), the combined replay+EWC old-task update becomes:

$$\textbf{(B)} + \textbf{(C)} \;\approx\; \beta F^{(i)}_{\theta^*_i}\delta_i + \lambda F^{(i)}_{\theta^*_i}\delta_i \;=\; (\beta+\lambda)F^{(i)}_{\theta^*_i}\delta_i. \tag{11}$$

We now compare this with the MAML meta-gradient. Using the same local query-gradient and curvature approximations from Equations (8) and (10) gives:

$$\textbf{(I)} - \textbf{(II)} \;\approx\; F^{(i)}_{\theta^*_i}\delta_i - \alpha\left(F^{(i)}_{\theta^*_i}\right)^2\delta_i \;=\; \left(F^{(i)}_{\theta^*_i} - \alpha\left(F^{(i)}_{\theta^*_i}\right)^2\right)\delta_i. \tag{12}$$

When $F^{(i)}_{\theta^*_i}$ is approximately rank-1 in diffusion models [40], $F^{(i)}_{\theta^*_i} \approx \rho_i u_i u_i^\top$ with $\|u_i\| = 1$, so $\left(F^{(i)}_{\theta^*_i}\right)^2 \approx \rho_i F^{(i)}_{\theta^*_i}$; see Appendix B. Therefore, the MAML old-task meta-gradient under the rank-1 approximation is:

$$\textbf{(I)} - \textbf{(II)} \;\approx\; (1-\alpha\rho_i)F^{(i)}_{\theta^*_i}\delta_i. \tag{13}$$

Equations (11) and (13) show that the combined replay+EWC old-task update and the MAML meta-gradient share the same Fisher-shaped direction, up to task-dependent scalar factors. The directions agree when the one-step inner update is locally stable along the dominant Fisher direction, $1-\alpha\rho_i > 0$, and more generally whenever the scalar multiplying $F^{(i)}_{\theta^*_i}\delta_i$ is positive. Under the rank-1 form, ρ_i is also the Fisher scale, since $\rho_i = \|F^{(i)}_{\theta^*_i}\|_F$. Thus, a task-dependent EWC weight can be interpreted as $\lambda_i \propto \rho_i$, making the EWC contribution match the MAML descent direction up to scale. In practice, we use a benchmark-wise constant λ set by the observed Fisher/gradient scale, since task Fisher scales are comparable within each benchmark.

Finally, under local approximations, **(B)** + **(C)** can be read as a MAML-style old-task meta-gradient **(I)** − **(II)**. Substituting this Fisher-shaped old-task meta-gradient approximation into Equation (5), the trust-region replay update can be read as:

$$\theta \;\leftarrow\; \theta \;-\; \eta\Big(\nabla_\theta \mathcal{L}_{\mathcal{T}_t}(\theta;\mathcal{D}_t) + \nabla_\theta \mathcal{L}_{\mathrm{MAML}}(\theta_{i<t})\Big).$$

We refer to Appendix D.1 for empirical analysis of these local approximations.

4 Experimental Evaluations

We evaluate whether our continual learning approach yields *faster convergence* when training sequentially across tasks in two settings with different heterogeneity: (i) low-heterogeneity task-incremental diffusion image generation on ImageNet-500, and (ii) high-heterogeneity continual diffusion-policy control on Continual-World-10.

Baselines. We compare our method against three continual-learning baselines: (i) **Generative replay**, implemented via generative distillation, which has been shown effective for diffusion models [26]; (ii) **EWC** [22] using the rank-1 approximation [40]; and (iii) **Naïve continual fine-tuning**, which sequentially trains on each task. To benchmark against continual meta-learning, we include **FTML**, an online extension of MAML [8], and **VR-MCL** [44]. For scalability, we implement these meta-learning baselines with the standard *first-order* approximation, which typically matches second-order performance while avoiding the prohibitive cost of higher-order differentiation at our scale [29], as our model and batch sizes are substantially larger than those in prior VR-MCL experiment setups; we provide a detailed size/computation comparison and implementation notes in the Appendix C.3. Importantly, VR-MCL's control-variate update still reduces variance of the (first-order) stochastic meta-gradient. For meta-learning baselines, we use generative replay to keep the memory footprint and data-access assumptions consistent with our continual learning setting.

4.1 Task-Incremental Image Generation Set-up

Datasets and Metrics. We use a 500-class ImageNet subset as a scaled up Tiny-ImageNet while keeping the compute footprint low [43, 34, 5]. We follow a task-incremental protocol by splitting the 500 classes into 10 disjoint tasks of 50 classes each. We evaluate with Fréchet Inception Distance

(FID) [16] on a held-out split, reporting average FID across tasks and forgetting, defined as the change in a task's FID from when it is first learned to the end of training. We refer to Appendix C.1 for additional details.

Implementation Details. We train conditional diffusion models with a standard denoising diffusion objective [17, 30] that condition on class labels. For meta-learning baselines, we split each training batch evenly into a support/query pair. We perform a single inner adaptation step so that the total number of parameter updates and the total number of training samples processed match our continual-learning setting. We refer to Appendix C.1 for model details and training configurations.

4.2 Continual Robotic Manipulation Set-up

Datasets and Metrics. We evaluate continual robotic manipulation on Continual-World-10 (CW10) [41], a 10-task sequence from Meta-World [46] (e.g., *push-wall*, *close-window*). Unlike ImageNet, CW10 spans distinct manipulation skills with different dynamics and reward conditions, and thus exhibits substantially higher task heterogeneity. Following standard practice in Continual-World, we report the success rate for each task, defined as the fraction of evaluation rollouts that satisfy the environment's success condition. After each task, we evaluate on all tasks seen so far using 100 trajectories per task and report average success. Forgetting is the drop in a task's success rate from when it is first learned to the end of training. See Appendix C.2 for CW10 details.

Implementation Details. We instantiate the control policy as a Diffusion Policy that generates action chunks conditioned on recent observations [4]. To collect expert demonstrations, we roll out scripted experts provided by the benchmark for 2500 trajectories per task, yielding an offline trajectory dataset for each task. Each training sample uses an observation horizon of 6 steps and predicts an action chunk of 2 steps, forming a sequence length of 8; we selected these values via a grid search over horizon/chunk combinations, with full results reported in Appendix C.2. For meta-learning baselines, we similarly split each training batch evenly into a support/query pair and perform a single inner adaptation step to match our continual-learning setting. We refer to Appendix C.2 for model details and training configurations.

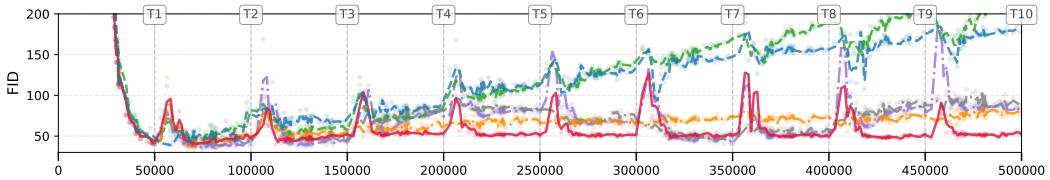
4.3 Results and Discussions

Trust region continual learning reduces catastrophic forgetting. We study whether Trust Region combines the strengths of EWC and replay by comparing it to its components and finetuning across two datasets with different task heterogeneity. High task heterogeneity, such as CW10, contains meaningfully diverse manipulation tasks with varying degrees of transfer and interference, and prior analyses explicitly use transfer matrices to quantify when tasks reuse versus overwrite features [41]. More broadly, feature reuse and forgetting depend on task similarity: dissimilar tasks tend to transfer less and interfere more, making local constraints around old optima harder to satisfy [45, 23, 13].

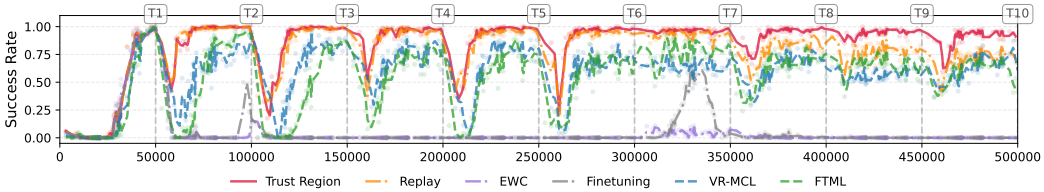
Table 1 shows that Trust Region achieves the best final performance and the lowest forgetting on both datasets. On high-heterogeneity CW10, Trust Region attains the highest average success rate ($88.3\%_{\pm 0.4}$) while also exhibiting the smallest magnitude of forgetting ($4.4_{\pm 0.9}$), improving over replay ($85.3\%_{\pm 2.0}$, $8.2_{\pm 2.0}$) and outperforming regularization-only baselines. EWC performs similarly to finetuning on CW10, consistent with the view that a local quadratic surrogate can be a weak guide when tasks have limited overlap [22]. Replay is competitive on CW10 because mixing past and current samples approximates multitask training over the union of tasks and thus directly supplies gradients for retention. However, generative replay can suffer from error accu-

Table 1: Final performance and forgetting on ImageNet-500 and CW10 with standard errors. For ImageNet, we report average FID across tasks ($\mathcal{A}$FID, lower is better) and average forgetting $\mathcal{F}$ (lower is better). For CW10, we report average Success rate (SR, higher is better) and average forgetting $\mathcal{F}$ (lower is better).

Methods	ImageNet-500		CW10	
	$\mathcal{A}$FID ↓	$\mathcal{F}$ ↓	SR ↑	$\mathcal{F}$ ↓
Finetuning	$86.2_{\pm 7.3}$	$50.4_{\pm 7.2}$	$17.9\%_{\pm 2.4}$	$73.0_{\pm 2.0}$
EWC	$77.2_{\pm 1.2}$	$41.2_{\pm 2.3}$	$17.6\%_{\pm 1.8}$	$73.7_{\pm 1.3}$
Replay	$53.4_{\pm 6.0}$	$18.2_{\pm 4.6}$	$85.3\%_{\pm 2.0}$	$8.2_{\pm 2.0}$
FTML	$172.5_{\pm 9.1}$	$128.5_{\pm 9.0}$	$78.5\%_{\pm 3.8}$	$13.3_{\pm 3.3}$
VRMCL	$142.2_{\pm 8.5}$	$96.6_{\pm 8.4}$	$77.9\%_{\pm 1.5}$	$11.7_{\pm 2.0}$
Trust Region	$\mathbf{44.5}_{\pm 2.3}$	$\mathbf{10.6}_{\pm 3.0}$	$\mathbf{88.3\%}_{\pm 0.4}$	$\mathbf{4.4}_{\pm 0.9}$



(a) FID (lower is better) of task 1 while continual learning other tasks on ImageNet-500. Averaged over 3 random seeds.



(b) Success rate (higher is better) of task 1 while continual learning other tasks on CW10. Averaged over 4 random seeds.

Figure 2: Task 1 performance over the course of continual training on 10 tasks, averaged over random seeds, on two datasets. X-axis: gradient update steps.

mulation and distribution drift when earlier generators are imperfect, which compounds over the task sequence and can worsen forgetting [10, 26]. Trust Region mitigates this by anchoring updates within a neighborhood that preserves past-task performance, improving stability relative to replay alone.

On low-heterogeneity ImageNet-500, Trust Region outperforms replay, achieving the best average FID ($44.5_{\pm 2.3}$) and the lowest forgetting ($10.6_{\pm 3.0}$), improving over replay ($53.4_{\pm 6.0}$, $18.2_{\pm 4.6}$) and outperforming finetuning and EWC. This suggests that, even when tasks share more structure, the additional local constraint effectively helps prevent "mode hopping" across many equivalent low-loss solutions during long-horizon training in overparameterized generative models [11, 6].

Surprisingly, continual meta-learning baselines underperform on ImageNet-500, falling well below finetuning: FTML reaches $172.5_{\pm 9.1}$ average FID with $128.5_{\pm 9.0}$ forgetting, and VRMCL reaches $142.2_{\pm 8.5}$ with $96.6_{\pm 8.4}$ forgetting. Yet on CW10 their performance remains good, though still below Trust Region. Intuitively, lower task heterogeneity implies greater overlap among task optima, so replay-only training already tracks a shared low-loss region. Bilevel meta-learning then becomes more error-amplifying where imperfections in replay bias the outer meta-gradient objective and simultaneously corrupt the inner adaptation step taken from that slightly off initialization, pushing updates away from the shared basin (see Figure 2a). In CW10, optima overlap less and inner adaptation is less informative, so replay supplies most of the retention signal. Performance is therefore dominated by the outer meta-gradient objective, making the adaptation step comparatively less sensitive. We refer to Appendices D.2 and D.3 for hyperparameter and task-order experiments.

Trust region continual learning recovers early-task performance faster. To quantify how quickly early-task performance returns after learning new tasks, we measure re-convergence using the step-to-threshold metric in Table 2: after each task transition in Figure 2, we count the number of gradient updates until Task 1 returns to a target level relative to its *initial* optimum. For ImageNet-500, $\mathrm{FID}_1 \leq (1+\tau)\mathrm{FID}_1^{(0)}$ with $\tau \in \{10\%, 20\%, 30\%\}$, and for CW10, $\mathrm{SR}_1 \geq \alpha\,\mathrm{SR}_1^{(0)}$ with $\alpha \in \{99\%, 90\%, 80\%\}$. Across tasks, these step counts typically range from single-digit to tens of updates for Trust Region methods (e.g., often recovers within ≤ 55 steps on ImageNet and ≤ 45 steps for CW10 at the $90\%/80\%$ thresholds), while unstable baselines can exhibit orders-of-magnitude longer recovery tails (up to 10^4 steps) or fail to re-converge ("–").

This behavior is informative about *where* updates live in parameter space. Replay provides gradients that keep old-task behavior "reachable," but without an explicit locality constraint it can drift among multiple solutions that fit the replay mixture yet are less aligned with the original Task 1 basin, requiring more steps to re-attain the previous Task 1 optimum. Trust Region sharpens this by constraining each update to remain in a neighborhood that preserves low error on past tasks, so the

Table 2: **Steps to re-converge Task 1 under continual learning.** Each entry reports the number of gradient updates for Task 1 performance to return to the target degradation threshold after each task transition. Smaller is better; "–" indicates that the target was not reached during training on that task.

Method	ImageNet-500																								
	+10%									+20%									+30%						
	T2	T3	T4	T5	T6	T7	T8	T9	T10	T2	T3	T4	T5	T6	T7	T8	T9	T10	T2	T3	T4	T5	T6	T7	T8	T9	T10
Finetune	25	45	–	–	–	–	–	–	–	5	35	–	–	–	35	50	–	–	2	25	–	–	–	25	50	–	–
EWC	20	50	–	–	–	650	–	–	–	15	50	–	–	–	85	55	–	–	8	30	55	–	–	55	45	–	–
Replay	30	90	–	–	–	–	–	–	–	30	75	–	–	–	–	–	–	–	2	2	5	–	–	–	–	–	–
FTML	45	–	–	–	–	–	–	–	–	10	80	–	–	–	–	–	–	–	5	55	–	–	–	–	–	–	–
VRMCL	4	–	–	–	–	–	–	–	–	4	–	–	–	–	–	–	–	–	4	15	–	–	–	–	–	–	–
Trust Region	35	15	–	–	–	–	–	–	–	20	15	–	–	–	50	–	350	2	20	15	2	35	35	25	55	40	2

Method	CW10																								
	99%									90%									80%						
	T2	T3	T4	T5	T6	T7	T8	T9	T10	T2	T3	T4	T5	T6	T7	T8	T9	T10	T2	T3	T4	T5	T6	T7	T8	T9	T10
Finetune	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
EWC	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Replay	60	95	30	80	60	30	10000	–	–	10	50	2	20	30	2	35	2	8000	10	35	2	20	15	2	4	2	45
FTML	–	–	–	–	–	–	–	–	–	550	20000	–	3000	–	30	–	–	–	250	3000	700	800	300	2	150	900	–
VRMCL	–	850	10000	–	–	–	–	–	–	750	650	500	250	–	–	–	–	–	400	350	100	250	85	40	550	–	30000
Trust Region	60	80	45	70	55	50	2000	–	100	10	45	2	30	30	2	25	2	2	10	35	2	15	4	2	4	2	2

model does not need to "re-learn" Task 1 after each transition—it only needs a small correction to re-enter the original low-loss region.

Table 2 also helps explain an otherwise surprising pattern on ImageNet-500: finetuning can occasionally look competitive, and sometimes even stronger than EWC or Replay, at intermediate task transitions. This does not indicate better forgetting robustness. Rather, it reflects incidental positive transfer when the current task happens to be semantically or visually similar to Task 1. In our default ImageNet-500 order, Task 1 is dominated by animal categories, and Tasks 2–3 are likewise animal-centric; later, Tasks 7–8 again contain animal-related classes. In such cases, naive finetuning can partially preserve or recover Task 1 performance through shared visual structure, even without explicitly protecting old knowledge. However, this advantage is fragile: once later tasks become less related to Task 1, finetuning quickly deteriorates and fails to re-converge under stricter thresholds. Accordingly, these temporary gains should be interpreted as accidental cross-task similarity rather than genuine robustness to forgetting.

The contrast is even clearer on CW10. In Figure 2b, finetuning collapses to near-zero Task 1 success on later tasks, and EWC provides little improvement, consistent with stronger task heterogeneity where local regularization alone cannot maintain competence. We do observe occasional transfer in finetuning (e.g., a transient improvement around task 6), suggesting some shared sub-skills, but this effect is sporadic and does not prevent eventual collapse. In comparison, FTML and VR-MCL retain good Task 1 performance on CW10 but recover more slowly than Trust Region after each transition, indicating that an explicit "fast-adaptation" objective does not automatically translate to fast *re-convergence* under long-horizon continual updates when replay and initialization drift accumulate.

5 Conclusion

We presented a trust region view of hybrid continual learning that unifies generative replay and EWC: replay keeps past-task behavior reachable by providing direct training signal on previous tasks, while the Fisher-metric penalty constrains each update to remain within a neighborhood that preserves low error on earlier optima. Our analysis shows that this simple continual learning objective induces a MAML-style optimization form under local approximations, explaining why it can exhibit meta-learning-like behavior (fast re-adaptation) without an explicit bilevel procedure. Across low-heterogeneity ImageNet-500 and high-heterogeneity CW10, trust region continual learning improves

retention and accelerates old-task recovery, achieving the strongest final performance and the least forgetting. More broadly, our results suggest a scalable continual and meta-learning paradigm for large models and demanding tasks like robotic control.

Limitations One limitation is our implicit assumption that the task 1 optimum lies close to a shared optimum that also works well for later tasks (see fig. 1c). When tasks are highly heterogeneous, this shared optimum may be far from the task-1 optimum, weakening the benefits of our approach. We refer to Appendices D.1 and D.2.1 for additional analysis of replay fidelity, Fisher estimation, and the rank-1 approximation. Future work could study how to identify or construct candidate tasks that are more likely to share a common parameter region for future tasks.

References

[1] R. Aljundi, F. Babiloni, M. Elhoseiny, M. Rohrbach, and T. Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 139–154, 2018.

[2] N. Barari, X. Lian, and C. J. MacLellan. Robust incremental learning of visual concepts without catastrophic forgetting. *Cognitive Systems Research*, page 101447, 2026.

[3] J. Chen, Y. Wang, P. Wang, X. Chen, Z. Zhang, Z. Lei, and Q. Li. Diffusepast: Diffusion-based generative replay for class incremental semantic segmentation. *arXiv preprint arXiv:2308.01127*, 2023.

[4] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems*, 2023.

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009.

[6] F. Draxler, K. Veschgini, M. Salmhofer, and F. Hamprecht. Essentially no barriers in neural network energy landscape. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1309–1318. PMLR, 10–15 Jul 2018.

[7] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1126–1135. JMLR.org, 2017.

[8] C. Finn, A. Rajeswaran, S. Kakade, and S. Levine. Online meta-learning. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1920–1930. PMLR, 09–15 Jun 2019.

[9] R. M. French. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4):128–135, 1999.

[10] R. Gao and W. Liu. DDGR: Continual learning with deep diffusion-based generative replay. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 10744–10763. PMLR, 23–29 Jul 2023.

[11] T. Garipov, P. Izmailov, D. Podoprikhin, D. Vetrov, and A. G. Wilson. Loss surfaces, mode connectivity, and fast ensembling of deep neural networks. In *Advances in Neural Information Processing Systems 31 (NeurIPS 2018)*, pages 8803–8812, 2018.

[12] B. Ghorbani, S. Krishnan, and Y. Xiao. An investigation into neural net optimization via hessian eigenvalue density, 2019.

[13] D. Goldfarb, I. Evron, N. Weinberger, D. Soudry, and P. Hand. The joint effect of task similarity and overparameterization on catastrophic forgetting: An analytical model. In *International Conference on Learning Representations (ICLR)*, 2024.

[14] G. Gupta, K. Yadav, and L. Paull. La-maml: Look-ahead meta learning for continual learning, 2020.

[15] A. Heng and H. Soh. Selective amnesia: A continual learning approach to forgetting in deep generative models. *Advances in Neural Information Processing Systems*, 36:17170–17194, 2023.

[16] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[17] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.

[18] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey. Meta-learning in neural networks: A survey. *arXiv preprint arXiv:2004.05439*, 2020.

[19] K. Javed and M. White. Meta-learning representations for continual learning. In *Advances in Neural Information Processing Systems*, pages 1818–1828, 2019.

[20] J. Kim, H. Cho, J. Kim, Y. Y. Tiruneh, and S. Baek. Sddgr: Stable diffusion-based deep generative replay for class incremental object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[21] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

[22] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, Mar. 2017.

[23] S. Lee, S. Goldt, and A. Saxe. Continual learning in the teacher-student setup: Impact of task similarity. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 6109–6119. PMLR, 18–24 Jul 2021.

[24] Z. Li, F. Zhou, F. Chen, and H. Li. Meta-sgd: Learning to learn quickly for few-shot learning. *arXiv preprint arXiv:1707.09835*, 2017.

[25] J. Martens. New insights and perspectives on the natural gradient method. *Journal of Machine Learning Research*, 21(146):1–76, 2020.

[26] S. Masip, P. Rodriguez, T. Tuytelaars, and G. M. v. d. Ven. Continual learning of diffusion models with generative distillation. In V. Lomonaco, S. Melacci, T. Tuytelaars, S. Chandar, and R. Pascanu, editors, *Proceedings of The 3rd Conference on Lifelong Learning Agents*, volume 274 of *Proceedings of Machine Learning Research*, pages 431–456. PMLR, 29 Jul–01 Aug 2025.

[27] J. L. McClelland, B. L. McNaughton, and R. C. O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models. *Psychological Review*, 102(3):419–457, 1995.

[28] M. McCloskey and N. J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of Learning and Motivation*, volume 24, pages 109–165. Academic Press, 1989.

[29] A. Nichol, J. Achiam, and J. Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[30] A. Nichol and P. Dhariwal. Improved denoising diffusion probabilistic models. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, 2021.

[31] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[32] M. Riemer, I. Cases, R. Ajemian, M. Liu, I. Rish, Y. Tu, and G. Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *International Conference on Learning Representations*, 2019. Poster.

[33] D. Rolnick, A. Ahuja, J. Schwarz, T. P. Lillicrap, and G. Wayne. Experience replay for continual learning, 2019.

[34] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[35] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel. Trust region policy optimization, 2017.

[36] H. Shin, J. K. Lee, J. Kim, and J. Kim. Continual learning with deep generative replay. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 2994–3003, Red Hook, NY, USA, 2017. Curran Associates Inc.

[37] J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2256–2265. PMLR, 2015.

[38] E. Urettini and A. Carta. Online curvature-aware replay: Leveraging 2nd order information for online continual learning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 60590–60609. PMLR, 2025.

[39] G. M. van de Ven and A. S. Tolias. Generative replay with feedback connections as a general strategy for continual learning, 2019.

[40] Z. Wang, A. Gupta, Z. Dong, and C. J. MacLellan. Avoid catastrophic forgetting with rank-1 fisher from diffusion models. In *The Fourteenth International Conference on Learning Representations*, 2026.

[41] M. Wołczyk, M. Zając, R. Pascanu, Ł. Kuciński, and P. Miłoś. Continual world: A robotic benchmark for continual reinforcement learning. In *Advances in Neural Information Processing Systems*, 2021.

[42] B. Wu, J. Fang, X. Zeng, S. Liang, and Q. Zhang. Adaptive compositional continual meta-learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 37358–37378. PMLR, 2023.

[43] J. Wu, Q. Zhang, and G. Xu. Tiny imagenet challenge. Technical report, Stanford University, 2017. CS231N course report / challenge description.

[44] Y. Wu, L.-K. Huang, R. Wang, D. Meng, and Y. Wei. Meta continual learning revisited: Implicitly enhancing online hessian approximation via variance reduction. In *The Twelfth International Conference on Learning Representations*, 2024.

[45] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems*, 2014.

[46] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Proceedings of The Conference on Robot Learning (CoRL)*, volume 100 of *Proceedings of Machine Learning Research*, pages 1094–1100. PMLR, 2020.

[47] F. Zenke, B. Poole, and S. Ganguli. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 3987–3995. JMLR.org, 2017.

[48] D. Zverev, A. S. Koepke, and J. F. Henriques. On the dangers of bootstrapping generation for continual learning and beyond, 2025.

A Full derivation of Equation (9).

We now derive the identity linking Fisher and Hessian using the same notation. Let $p_\theta(x)$ be the model distribution and define the per-example negative log-likelihood $\ell(\theta; x) := -\log p_\theta(x)$. Then

$$g(\theta; x) = \nabla_\theta \ell(\theta; x) = -\nabla_\theta \log p_\theta(x).$$

Consider the Hessian:

$$\begin{aligned}
\nabla^2_\theta \ell(\theta; x) &= -\nabla^2_\theta \log p_\theta(x) \\
&= -\nabla_\theta \left(\frac{\nabla_\theta p_\theta(x)}{p_\theta(x)} \right) \\
&= -\left(\frac{\nabla^2_\theta p_\theta(x)}{p_\theta(x)} - \frac{\nabla_\theta p_\theta(x) \nabla_\theta p_\theta(x)^\top}{p_\theta(x)^2} \right) \\
&= \frac{\nabla_\theta p_\theta(x) \nabla_\theta p_\theta(x)^\top}{p_\theta(x)^2} \; - \; \frac{\nabla^2_\theta p_\theta(x)}{p_\theta(x)} \\
&= \big(\nabla_\theta \log p_\theta(x)\big)\big(\nabla_\theta \log p_\theta(x)\big)^\top \; - \; \frac{\nabla^2_\theta p_\theta(x)}{p_\theta(x)}.
\end{aligned} \tag{14}$$

Taking expectation over $x \sim p_\theta$,

$$\begin{aligned}
\mathbb{E}_{x\sim p_\theta}\big[\nabla^2_\theta \ell(\theta; x)\big] &= \mathbb{E}_{x\sim p_\theta}\Big[\big(\nabla_\theta \log p_\theta(x)\big)\big(\nabla_\theta \log p_\theta(x)\big)^\top\Big] - \mathbb{E}_{x\sim p_\theta}\left[\frac{\nabla^2_\theta p_\theta(x)}{p_\theta(x)}\right] \\
&= \mathbb{E}_{x\sim p_\theta}\Big[\big(\nabla_\theta \log p_\theta(x)\big)\big(\nabla_\theta \log p_\theta(x)\big)^\top\Big] - \int \nabla^2_\theta p_\theta(x)\, dx
\end{aligned} \tag{15}$$

where we used $\mathbb{E}_{x\sim p_\theta}\left[\frac{\nabla^2_\theta p_\theta(x)}{p_\theta(x)}\right] = \int p_\theta(x) \frac{\nabla^2_\theta p_\theta(x)}{p_\theta(x)} dx = \int \nabla^2_\theta p_\theta(x)\, dx$. Finally, because $\int p_\theta(x)\, dx = 1$,

$$\int \nabla^2_\theta p_\theta(x)\, dx = \nabla^2_\theta \int p_\theta(x)\, dx = \nabla^2_\theta 1 = 0. \tag{16}$$

Substituting Equation (16) into Equation (15) and using $g(\theta; x) = -\nabla_\theta \log p_\theta(x)$ yields

$$\mathbb{E}_{x\sim p_\theta}\big[\nabla^2_\theta \ell(\theta; x)\big] = \mathbb{E}_{x\sim p_\theta}\big[g(\theta; x) g(\theta; x)^\top\big],$$

which is exactly Equation (9).

B Full derivation of Equation (13)

Let $\delta_i = \theta - \theta^*_i$. If θ remains in a local neighborhood of θ^*_i (trust-region), then a first-order Taylor expansion gives

$$\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(\theta; \tilde{\mathcal{D}}_i) = \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(\theta^*_i; \tilde{\mathcal{D}}_i) + H^{\text{te}}_i(\theta^*_i)\, \delta_i + \mathcal{O}(\|\delta_i\|^2). \tag{17}$$

Assuming θ^*_i is an (approximate) optimum for task $\mathcal{T}_i$, the first term is near zero, yielding $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(\theta; \tilde{\mathcal{D}}_i) \approx H^{\text{te}}_i(\theta^*_i)\delta_i$.

The MAML correction term is $H^{\text{tr}}_i(\theta) \nabla_{\theta'} \mathcal{L}_{\mathcal{T}_i}(\theta'; \mathcal{D}^{\text{te}}_{\mathcal{T}_i})$. Under the trust-region assumption, $H^{\text{tr}}_i(\theta) \approx H^{\text{tr}}_i(\theta^*_i)$. Under replay, $\tilde{\mathcal{D}}_i$ is generated to match the old-task data distribution, so the query curvature $H^{\text{te}}_i(\theta^*_i)$ is close to the support curvature $H^{\text{tr}}_i(\theta^*_i)$. Thus $H^{\text{tr}}_i(\theta) \approx H^{\text{te}}_i(\theta^*_i)$.

For likelihood-based objectives, the empirical Fisher is commonly used as an approximation to the Hessian in expectation, motivating $H^{\text{tr}}_i(\theta^*_i) \approx H^{\text{te}}_i(\theta^*_i) \approx F^{(i)}(\theta^*_i)$. Therefore,

$$H^{\text{tr}}_i(\theta)\, H^{\text{te}}_i(\theta^*_i)\, \delta_i \;\approx\; F^{(i)}(\theta^*_i)\, F^{(i)}(\theta^*_i)\, \delta_i = \big(F^{(i)}(\theta^*_i)\big)^2 \delta_i. \tag{18}$$

If $F^{(i)}(\theta_i^*)$ is approximately rank-1, write $F^{(i)}(\theta_i^*) = \rho_i u_i u_i^\top$ with $\rho_i \geq 0$ and $\|u_i\| = 1$. Then

$$\left(F^{(i)}(\theta_i^*)\right)^2 = (\rho_i u_i u_i^\top)^2 = \rho_i^2 u_i (u_i^\top u_i) u_i^\top = \rho_i^2 u_i u_i^\top = \rho_i (\rho_i u_i u_i^\top) = \rho_i F^{(i)}(\theta_i^*), \tag{19}$$

so $\left(F^{(i)}(\theta_i^*)\right)^2 \delta_i \approx \rho_i F^{(i)}(\theta_i^*)\delta_i$. The scalar ρ_i is the unique nonzero eigenvalue of $F^{(i)}(\theta_i^*)$. Absorbing ρ_i into the regularizer weight λ (and/or the MAML inner-step α) yields an EWC-form term.

C Additional Experimental details

C.1 Task-Incremental Image Generation

C.1.1 Additional Implementation Details

We adopt a label-conditioned UNet from the *HuggingFace* diffusion library as the denoising model, using default hyperparameters unless otherwise specified. The architecture consists of four ResNet blocks, with 128 output channels in the first downsampling block and 256 output channels in each subsequent block. For sampling, we use a DDIM scheduler with 50 inference steps and 1000 diffusion timesteps. Training hyperparameters are summarized in Table 3. Unless stated otherwise, the same configuration is used across all datasets and tasks.

Table 3: Training configurations used across methods for ImageNet.

Setting	**Value**
General	
Batch Size	128 (bilevel: 64 support + 64 query)
Optimizer	Adam [21] (outer); SGD (inner for bilevel)
Learning Rate	2×10^{-4}
Per-task Gradient Update Steps	100 epochs/task
Seeds Averaged	3
EWC	
Fisher Representation	Rank-1 [40], estimated over 10,000 samples/task
EWC λ	15,000
Replay	
Buffer Size per Task	1,300
Replay Ratio	1.0 (bilevel: 1.0)
Meta-learning (bilevel) additional hyperparameters	
Adaptation Steps	1
Inner-loop LR	1×10^{-4}
Outer-loop LR	2×10^{-4}

C.1.2 Additional Dataset Details

We use the downsampled **ImageNet-1k** dataset (32×32) for computational efficiency while preserving sufficient visual diversity and semantic complexity. Each sample is normalized to $(-1, 1)$ for diffusion training. We restrict training to the first 500 classes, partitioned into 10 tasks of 50 classes each. Each task contains approximately 64,000 training images. Task splits: $T_1 = \{0, \ldots, 49\},\ T_2 = \{50, \ldots, 99\},\ \ldots\ T_{10} = \{450, \ldots, 499\}$.

C.2 Continual Robotic Manipulation

C.2.1 Diffusion Policy Implementation Details

We implement a 1D conditional diffusion model over action sequences: given an observation history $o_{1:H}$, the policy generates an action sequence $a_{1:T}$. The denoiser ϵ_θ is a 1D conditional U-Net that operates on the action tensor in channel-first form (B, D_a, T) and is FiLM-conditioned on a global context vector formed by concatenating (i) a diffusion-timestep embedding and (ii) the flattened observation history of size HD_o. The timestep embedding has dimension 512 and is produced by a

sinusoidal positional encoding followed by a 2-layer MLP (512→2048→512) with Mish activations. The U-Net uses channel widths $[256, 512, 1024]$ with kernel size 5 and GroupNorm (8 groups): the encoder has 3 resolution stages, each with two FiLM-conditioned residual Conv1D blocks plus a strided downsampling layer (except the last stage); the bottleneck has 2 additional residual blocks; and the decoder has 2 upsampling stages with skip connections, each with two residual blocks and transposed-convolution upsampling, followed by a final Conv1D block and a 1×1 projection back to D_a. Overall, the noise-prediction network contains 12 FiLM-conditioned residual blocks, and we add multi-head self-attention (8 heads) at U-Net levels 2 and 3 (the 512- and 1024-channel resolutions). For the diffusion process we use a DDIM scheduler with 1000 training timesteps and run 50 denoising steps at inference. Training hyperparameters are summarized in Table 4

Table 4: Training configurations used across methods for Meta-World CW10.

Setting	**Value**
General	
Batch Size	256 (bilevel: 128 support + 128 query; effective 256)
Optimizer	Adam [21] (outer); SGD (inner for bilevel)
Learning Rate	2×10^{-4}
Per-task Update Steps	50,000 (gradient / meta-updates)
Seeds Averaged	4
EWC	
Fisher Representation	Rank-1 [40], estimated over 20,000 training samples/task
EWC λ	12
Replay	
Buffer Size per Task	2,500 rollouts/task
Replay Ratio	1.0 (bilevel: 1.0)
Meta-learning (bilevel) additional hyperparameters	
Adaptation Steps	1
Inner-loop LR	1×10^{-4}
Outer-loop LR	2×10^{-4}

C.2.2 Additional Task and Dataset Details

We evaluate on a 10-task continual manipulation sequence (CW10) derived from Meta-World. All tasks share a common low-dimensional state representation and continuous control interface, enabling transfer across tasks.

We use low-dimensional, goal-conditioned state observations $o_t \in \mathbb{R}^{39}$ and continuous actions $a_t \in \mathbb{R}^4$. Following the standard Meta-World state interface, the 4D action consists of a desired end-effector displacement in $\mathbb{R}^3$ and a 1D gripper control. The 39D observation includes end-effector state, gripper state, object pose information, and goal information (shared across tasks). For diffusion policy training, we linearly rescale each action dimension to $[-1, 1]$ (and clip to the same range at training and sampling time) to keep the action domain bounded and consistent across tasks.

Episodes use a fixed horizon of maximum 200 environment steps. We report success rate, defined as the fraction of evaluation rollouts that satisfy the task-specific success condition.

Task suite. Each task shares the same observation/action interface but requires a distinct manipulation skill:

- **hammer-v1** — The agent uses a hammer to drive a nail into a wall.
- **push-wall-v1** — The agent pushes a block toward a designated target position on a wall.
- **faucet-close-v1** — The agent rotates a faucet handle clockwise to close it.
- **push-back-v1** — The agent pushes an object backward to a specified target location.
- **stick-pull-v1** — The agent grasps a stick tool and uses it to pull an object toward a target region.
- **handle-press-side-v1** — The agent presses a handle on an appliance from the side.

- **push-v1** — The agent pushes an object to a specified target position.
- **shelf-place-v1** — The agent picks up an object and places it onto a shelf.
- **window-close-v1** — The agent closes a window by manipulating the window or its handle.
- **peg-unplug-side-v1** — The agent unplugs a peg from a socket by pulling it out from the side.

C.2.3 Diffusion Policy Data Representation

We conduct a small grid search over three sequence-encoding hyperparameters for the CW10 diffusion policy: sequence length S, observation horizon O, and action chunk size A. To isolate the effect of these design choices, we use the *same model architecture and training configuration* as in our continual learning experiments, except that we train *one model per task from scratch* (i.e., no task sequence and no continual updates), and then report the average success rate across the 10 CW10 tasks.

As shown in Table 5, the configuration $(S, O, A) = (8, 6, 2)$ achieves the highest mean success rate (0.859), improving over the next best setting $(8, 8, 1)$ (0.808). We therefore adopt S=8, O=6, and A=2 as the default CW10 diffusion policy configuration in all main continual learning experiments.

Table 5: CW10 diffusion policy configuration grid search results. We report the mean success rate across 10 tasks. Higher is better. Each configuration is trained from scratch separately for each task, using the same model and training setup as in our continual learning experiments.

Config	S	O	A	Mean success ↑
S8_O6_A2	8	6	2	**0.859**
S8_O8_A1	8	8	1	0.808
S8_O4_A4	8	4	4	0.744
S8_O4_A2	8	4	2	0.622
S16_O8_A8	16	8	8	0.511
S16_O8_A4	16	8	4	0.486
S16_O12_A4	16	12	4	0.449
S16_O16_A1	16	16	1	0.326
S32_O16_A4	32	16	4	0.184
S32_O24_A4	32	24	4	0.177
S32_O16_A8	32	16	8	0.121
S32_O16_A16	32	16	16	0.081
S32_O24_A8	32	24	8	0.034

C.3 Compute Efficacy

Our models are 34–$76\times$ larger and use substantially larger batches than the original VR-MCL setting [44], making full second-order hypergradients impractical at our target scale (second-order reduces throughput from $\sim$5 it/s to $\sim$1 it/s, i.e., $\approx 5\times$ slower) (See Table 6). Nevertheless, the first-order meta-learning baselines remain scalable and allow us to evaluate variance-reduction and continual-learning behavior under a matched compute budget. Table 7 reports the average runtime for each dataset and method (using first order), including evaluation.

Table 6: Model scale and training throughput. Our diffusion backbones are substantially larger than the reduced ResNet used in VR-MCL, and we report the resulting training speed under our first-order implementation versus a second-order (higher-order autodiff) variant.

Setting	Backbone	Params (M)	× vs. 1.09M	Batch size	it/s (FO)	it/s (2nd)
ImageNet diffusion	UNet	37.45	34.4×	128	∼5	∼1
CW10 diffusion policy	UNet	83.27	76.4×	256	∼5	∼1
VR-MCL from [44]	reduced ResNet	1.09	1.0×	32	–	–

Table 7: Average training runtime (hours) and GPU used per dataset/method.

Evaluation	Method	Hours	GPU
CW10	Continual Finetuning	~ 21	NVIDIA A40
CW10	EWC	~ 29	NVIDIA A40
CW10	Replay	~ 22	NVIDIA A40
CW10	Trust Region	~ 31	NVIDIA A40
CW10	FTML	~ 29	NVIDIA A40
CW10	VRMCL	~ 41	NVIDIA A40
ImageNet	Continual Finetuning	~ 12	NVIDIA H200
ImageNet	EWC	~ 15	NVIDIA H200
ImageNet	Replay	~ 22	NVIDIA H200
ImageNet	Trust Region	~ 31	NVIDIA H200
ImageNet	FTML	~ 29	NVIDIA H200
ImageNet	VRMCL	~ 44	NVIDIA H200

D Additional Results

D.1 Empirical analysis of the local approximations

The MAML-style interpretation in Section 3.3 relies on locality around old-task optima, agreement between replay and query gradients, and a common Fisher approximation to the support and query Hessians. We examine these properties on the 83.27M-parameter CW10 diffusion-policy models from our main experiments, then compare the resulting old-task update directly with an exact second-order MAML meta-gradient.

Displacement along protected directions. We first examine how the Fisher penalty controls displacement from the task-1 anchor $\theta_1^\star$. We track $\delta_1 = \theta - \theta_1^\star$ at the end of tasks 2 through 10 and compare against a replay-only ablation with the same seed and identical task-1 parameters. Here, $\theta_i^\star$ denotes the end-of-task parameters of the continual run after training on the mixture of current-task and replay data. For the rank-1 approximation $F_{\theta_1^\star}^{(1)} \approx \rho_1 u_1 u_1^\top$, with $\|u_1\| = 1$, we measure the protected displacement $|u_1^\top \delta_1|$ and the quadratic trust-region quantity $\delta_1^\top F_{\theta_1^\star}^{(1)} \delta_1$. The latter corresponds to $c^\star(\mu^\top \delta_1)^2$ in Equation (2) when $F_{\theta_1^\star}^{(1)} \approx c^\star \mu\mu^\top$.

As shown in Table 8, Trust Region reduces the protected displacement by a factor of approximately 7.5 and the quadratic trust-region quantity by a factor of 55 relative to replay alone. The protected displacement remains stable across all nine subsequent tasks, showing that the penalty maintains locality around the task-1 anchor in the Fisher metric. The nine per-task protected directions have a mean absolute cosine of 0.11, indicating that the per-task penalties act along largely different directions in parameter space. Together with the stable protected displacement, this supports the robustness of the Fisher constraint as tasks accumulate.

Gradient and curvature alignment. We next examine the gradient and curvature approximations used to connect the two update rules. Within the replay stream, the replay gradient and the query gradient evaluated at adapted parameters have a cosine similarity of 0.92, supporting the approximation in Equation (7). To evaluate the curvature terms, we compute the top five Hessian eigenpairs at each task optimum on both training and held-out data using Lanczos iterations with exact Hessian-vector

Table 8: CW10 displacement relative to the task-1 anchor, measured at the ends of tasks 2–10. Trust Region and the replay-only ablation share the same seed and task-1 parameters. Values are rounded.

Measurement	Trust Region	Replay only
Protected displacement $\lvert u_1^\top \delta_1 \rvert$	4.3×10^{-3}	3.3×10^{-2}
Trust-region quantity $\delta_1^\top F_{\theta_1^\star}^{(1)} \delta_1$	3.2×10^{-5}	1.8×10^{-3}

products. The Fisher direction has a cosine similarity of 0.74 with the measured Hessian directions, compared with approximately 10^{-7} for the random reference. Training and held-out Hessians also agree closely, with a top-direction cosine similarity of 0.90 and a leading-eigenvalue ratio of 1.10 (Table 9). This agreement in direction and curvature scale supports the common curvature approximation for the support and query terms in Equation (10).

Table 9: Gradient and curvature measurements on CW10. Cosine similarities measure directional agreement. The eigenvalue ratio measures agreement in curvature scale, with 1 indicating equal scale. The final row compares the old-task update with an exact second-order one-step MAML meta-gradient using support and query data from the same stream.

Measurement	Value
Replay gradient vs. adapted-query gradient: cosine	0.92
Fisher vs. Hessian direction: cosine	0.74
Training vs. held-out Hessian top direction: cosine	0.90
Training vs. held-out leading Hessian eigenvalue: ratio	1.10
Old-task update vs. exact MAML meta-gradient: mean cosine	0.83

Alignment with the exact MAML meta-gradient. Under the rank-1 Fisher approximation for diffusion models, the curvature reduction $F^2 \approx \rho F$ in Equation (13) predicts that our old-task update and the MAML meta-gradient share a Fisher-shaped direction. This is the model setting used in both our image-generation and policy-control experiments. To test the prediction directly on CW10, we compare the old-task component of our update with an exact second-order one-step MAML meta-gradient at five checkpoints within each task. Both updates are computed at the same parameters with support and query samples from the same data stream. The exact MAML calculation agrees with the $(\mathrm{I}) - (\mathrm{II})$ expression in Equation (6) to within 2×10^{-7}. Our old-task update has a mean cosine similarity of 0.83 with this meta-gradient (Table 9), providing direct empirical support for the predicted alignment.

Replay quality over longer task sequences. To examine how the replay-based alignment relates to the original tasks, we repeat the gradient comparison in Equation (7) with queries drawn from the true old-task distribution. The cosine similarity is 0.92 initially, reaches 0.55 over the first three tasks, and is 0.28 for tasks 8–10. These measurements show how replay quality affects agreement with old-task query gradients over longer task sequences. Together with the displacement results, they illustrate the complementary roles of the two components in the local MAML interpretation. The Fisher penalty maintains locality along protected directions, while replay supplies the old-task learning signal whose agreement with the original data depends on replay fidelity. The FMNIST experiments in Section D.2.2 further examine the role of replay in inner-loop adaptation.

D.2 Hyperparameter sensitivity and additional baseline experiments

We complement the analysis above with parameter sweeps on CW10 and additional continual meta-learning baseline experiments on Fashion-MNIST.

D.2.1 CW10 sensitivity analysis

We choose the order of magnitude of the EWC coefficient using the Fisher scales $\rho_i = \|F_{\theta_i^\star}^{(i)}\|_{\mathrm{F}}$, following Section 3.3. These scales range from 3 to 20 in measurements taken at already observed task boundaries, so the search uses information available during continual learning. As in the main experiments, we use a constant coefficient within each benchmark.

At a replay ratio of 1.0, we sweep $\lambda \in \{0, 5, 12, 20\}$, where $\lambda = 0$ corresponds to replay alone. Table 10a shows that the default value $\lambda = 12$ achieves the highest success rate and lowest forgetting, at 88.3% and 4.4, respectively. This setting improves on replay alone, which achieves a success rate of 85.3% and forgetting of 8.2.

We next vary the amount of replay to examine whether the benefit of the Fisher penalty persists across replay budgets. Table 10b compares $\lambda = 12$ with $\lambda = 0$ at four replay ratios from 0.1 to 1.0. Trust Region improves success rate at every tested ratio, showing that the Fisher penalty complements replay across these settings.

Table 10: CW10 hyperparameter sweeps. Higher success rate (SR, %) and lower forgetting indicate better performance. The coefficient sweep reports standard errors for $\lambda = 0$ and $\lambda = 12$.

(a) EWC coefficient at replay ratio 1.0.

λ	SR (%)	Forgetting
0	$85.3_{\pm 2.0}$	$8.2_{\pm 2.0}$
5	81.0	7.3
12 (default)	$\mathbf{88.3}_{\pm 0.4}$	$\mathbf{4.4}_{\pm 0.9}$
20	84.8	8.3

(b) Success rate (%) across replay ratios.

Replay ratio	$\lambda = 0$	$\lambda = 12$
0.1	69.9	74.7
0.5	75.3	85.2
0.75	82.8	85.5
1.0	85.3	88.3

Fisher estimation sample size. Because the penalty is defined by the Fisher direction, we also examine how that direction changes with the number of samples used to estimate it. Estimates from 2,000 and 20,000 samples have a cosine similarity of 0.96, showing that the penalty direction remains consistent across a tenfold increase in sample size. This complements the Fisher–Hessian comparison in Section D.1 by assessing the stability of the estimator itself.

D.2.2 Additional FTML and VR-MCL experiments on Fashion-MNIST

To examine the role of hyperparameter choice and adaptation depth in the continual meta-learning baselines, we run additional FTML and VR-MCL experiments on Fashion-MNIST (FMNIST), where broader sweeps are computationally feasible. We retain the ImageNet-500 training settings and first-order implementations described in Section 4, changing the dataset and the parameters explored in each sweep. We report final average FID, for which lower values indicate better image quality.

FTML: inner learning rate and adaptation steps. We jointly vary the inner learning rate and adaptation depth, comparing one and five inner steps at three learning rates (Table 11a). The default learning rate of 10^{-4} gives the lowest FID for both adaptation depths. A single inner step also achieves a lower FID than five steps at all three learning rates. At the default rate, the corresponding FIDs are 40.8 and 48.0.

VR-MCL: regularization coefficient. The VR-MCL sweep varies its regularization coefficient α across four values (Table 11b). The default coefficient $\alpha = 10^{-4}$ gives the lowest FID of 70.2. Thus, the default hyperparameters used in the ImageNet-500 experiments are also the best settings in the corresponding FMNIST sweeps for both baselines.

Table 11: Final average FID of continual meta-learning baselines on FMNIST. Lower values indicate better image quality. The default hyperparameters are those used in the ImageNet-500 experiments. Bold entries mark the best result for each FTML adaptation depth and for VR-MCL.

(a) FTML inner adaptation.

Inner learning rate	1 step	5 steps
5×10^{-5}	46.1	55.7
10^{-4} (default)	**40.8**	**48.0**
2×10^{-4}	57.3	70.0

(b) VR-MCL regularization.

Coefficient α	FID
10^{-5}	83.4
5×10^{-5}	72.4
10^{-4} (default)	**70.2**
2×10^{-4}	90.4

Trust Region achieves a final average FID of 15.4 on FMNIST, compared with 19.1 for Replay, 40.8 for the best FTML configuration, and 70.2 for the best VR-MCL configuration. Its advantage therefore persists across the baseline configurations examined here.

The adaptation-depth results also support the replay-based explanation in Section 4. Since each inner step uses replay samples, the higher FID with five steps is consistent with replay errors accumulating during inner optimization. This complements the CW10 query-gradient comparison in Section D.1 by showing how additional adaptation to replay samples affects generation performance.

D.3 Permuting Task Order

We conduct an additional experiment by randomly permuting the task order in both CW10 and Imagenet. We see that our methodology is robust to task permutations in Section D.3.

CW10. Our experiments in Section 4 were conducted using the standard ordering

$$\texttt{hammer-v1} \to \texttt{push-wall-v1} \to \texttt{faucet-close-v1} \to \texttt{push-back-v1} \to \texttt{stick-pull-v1} \to \texttt{handle-press-side-v1} \to \texttt{push-v1} \to \texttt{shelf-place-v1} \to \texttt{window-close-v1} \to \texttt{peg-unplug-side-v1}$$

whereas we now rerun the same experiments with a new task ordering

$$\texttt{stick-pull-v1} \to \texttt{push-v1} \to \texttt{hammer-v1} \to \texttt{push-back-v1} \to \texttt{window-close-v1} \to \texttt{push-wall-v1} \to \texttt{faucet-close-v1} \to \texttt{handle-press-side-v1} \to \texttt{shelf-place-v1} \to \texttt{peg-unplug-side-v1}$$

Imagenet-500. Our experiments in Section 4 were conducted using the standard ordering

$$T_1 \to T_2 \to T_3 \to T_4 \to T_5 \to T_6 \to T_7 \to T_8 \to T_9 \to T_{10}$$

whereas we now rerun the same experiments with a new task ordering

$$T_9 \to T_8 \to T_6 \to T_{10} \to T_3 \to T_4 \to T_7 \to T_2 \to T_5 \to T_1$$

Note: These orderings were randomly sampled using `random.shuffle`.

Methods	ImageNet-500		CW10	
	$\mathcal{A}$FID↓	$\mathcal{F}$↓	SR↑	$\mathcal{F}$↓
Finetuning	91.4	55.3	26.1%	68.3
EWC	77.8	42.8	17.01%	70.57
Replay	57.5	21.5	85.8%	6.1
FTML	155.6	111.5	81.3%	6.6
VRMCL	146.3	104.1	74.3%	12.8
Trust Region	**42.6**	**8.2**	**86.8%**	**3.4**

Table 12: Final performance and forgetting on permuted ImageNet-500 and CW10.

D.4 Imagenet Per Task Re-Convergence Tables

We present the re-convergence tables for all Imagenet tasks Tables 13 to 16, 17a, 17b, 18a and 18b.

D.5 CW10 Per Task Re-Convergence Tables

We present the re-convergence tables for all CW10 tasks Tables 19 to 22, 23a, 23b, 24a and 24b

E Broader Impacts.

This work studies continual learning methods that improve retention and rapid re-adaptation in diffusion models and diffusion-based control policies. Positively, such methods could reduce retraining

Imagenet-500																								
Method	+10%								+20%								+30%							
	T3	T4	T5	T6	T7	T8	T9	T10	T3	T4	T5	T6	T7	T8	T9	T10	T3	T4	T5	T6	T7	T8	T9	T10
Finetune	76	–	–	–	–	–	–	–	26	–	–	–	41	–	–	–	16	–	–	–	41	–	–	–
EWC	–	–	–	–	–	–	–	–	66	–	–	–	9001	–	–	–	56	–	–	–	7001	–	–	–
Replay	–	–	–	–	–	–	–	–	91	–	–	–	–	–	–	–	71	3001	–	–	–	–	–	–
FTML	–	–	–	–	–	–	–	–	41	36	–	–	–	–	–	–	21	21	–	–	–	–	–	–
VRMCL	–	–	–	–	–	–	–	–	26	–	–	–	–	–	–	–	26	–	–	–	–	–	–	–
Trust Region	56	–	–	–	–	–	–	–	16	56	–	–	–	–	–	–	16	51	51	–	76	–	–	–

Table 13: Steps to re-converge Task 2 under continual learning for Imagenet.

Imagenet-500																								
Method	+10%								+20%								+30%							
	T4	T5	T6	T7	T8	T9	T10		T4	T5	T6	T7	T8	T9	T10		T4	T5	T6	T7	T8	T9	T10	
Finetune	–	–	–	–	–	–	–		–	–	–	–	–	–	–		–	–	–	–	–	–	–	
EWC	–	–	–	–	–	–	–		–	–	–	–	–	–	–		–	–	–	–	–	–	–	
Replay	701	–	–	–	–	–	–		351	–	–	–	–	–	–		96	–	–	–	–	–	–	
FTML	–	–	–	–	–	–	–		–	–	–	–	–	–	–		26	–	–	–	–	–	–	
VRMCL	–	–	–	–	–	–	–		451	–	–	–	–	–	–		71	–	–	–	–	–	–	
Trust Region	31	51	56	76	–	–	–		31	26	41	46	–	–	–		11	26	36	31	66	41	–	

Table 14: Steps to re-converge Task 3 under continual learning for Imagenet.

costs, improve sample efficiency, and make adaptive perception and robotics systems more practical in non-stationary settings. However, stronger continual adaptation may also increase misuse risk in generative systems, including more persistent content generation capabilities, and in robotics, where failures under distribution shift could create safety concerns. In addition, results may inherit biases and access restrictions from the underlying datasets and benchmarks, especially ImageNet and robotic simulation environments, so real-world deployment should include dataset auditing, safety constraints, and domain-specific evaluation.

F Licenses

We provide details regarding the licenses and access terms of external assets used in this work in Table 25.

Imagenet-500																		
Method	+10%						+20%						+30%					
	T5	T6	T7	T8	T9	T10	T5	T6	T7	T8	T9	T10	T5	T6	T7	T8	T9	T10
Finetune	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
EWC	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Replay	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
FTML	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
VRMCL	40001	–	–	–	–	–	9001	–	–	–	–	–	6	46	–	–	–	–
Trust Region	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Table 15: Steps to re-converge Task 4 under continual learning for Imagenet.

Imagenet-500																
Method	+10%					+20%					+30%					
	T6	T7	T8	T9	T10	T6	T7	T8	T9	T10	T6	T7	T8	T9	T10	
Finetune	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	
EWC	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	
Replay	46	–	–	–	–	2	201	–	–	–	2	71	–	–	–	
FTML	–	–	–	–	–	–	–	–	–	–	8	–	–	–	–	
VRMCL	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	
Trust Region	701	–	–	–	–	51	951	–	–	–	41	551	–	–	–	

Table 16: Steps to re-converge Task 5 under continual learning for Imagenet.

Imagenet-500						
Method	+10%		+20%		+30%	
	T9	T10	T9	T10	T9	T10
Finetune	–	–	–	–	4	–
EWC	–	–	–	–	–	–
Replay	851	–	501	401	151	66
FTML	–	–	–	–	7001	–
VRMCL	–	–	–	–	–	–
Trust Region	2001	8001	851	551	41	51

(a) Task 8

Imagenet-500									
Method	+10%			+20%			+30%		
	T8	T9	T10	T8	T9	T10	T8	T9	T10
Finetune	–	–	–	–	–	–	2	–	–
EWC	–	–	–	56	–	–	46	–	–
Replay	31	–	–	26	551	–	8	151	56
FTML	–	–	–	46	–	–	36	–	–
VRMCL	–	–	–	61	–	–	46	–	–
Trust Region	66	2001	8001	56	61	51	56	41	26

(b) Task 7

Table 17: Steps to re-converge across tasks 7, 8 under continual learning for Imagenet-500.

Imagenet-500			
Method	+10%	+20%	+30%
	T10	T10	T10
Finetune	16	2	2
EWC	66	51	16
Replay	4	2	2
FTML	11	8	8
VRMCL	–	201	81
Trust Region	26	21	16

(a) Task 9

Imagenet-500												
Method	+10%				+20%				+30%			
	T7	T8	T9	T10	T7	T8	T9	T10	T7	T8	T9	T10
Finetune	–	–	–	–	–	–	–	–	–	–	–	–
EWC	–	–	–	–	–	–	–	–	–	–	–	–
Replay	–	–	–	–	–	–	–	–	–	–	–	–
FTML	–	–	–	–	–	–	–	–	–	–	–	–
VRMCL	–	–	–	–	–	–	–	–	–	–	–	–
Trust Region	–	–	–	–	–	–	–	–	–	–	–	–

(b) Task 6

Table 18: Steps to re-converge across tasks 6, 9 under continual learning for Imagenet-500.

CW10																								
Method	99%								90%								80%							
	T3	T4	T5	T6	T7	T8	T9	T10	T3	T4	T5	T6	T7	T8	T9	T10	T3	T4	T5	T6	T7	T8	T9	T10
Finetune	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
EWC	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Replay	76	46	41	31	5001	751	9001	10001	21	21	21	16	4	46	4	16	10	4	6	4	4	21	4	4
FTML	4000	–	550	–	–	–	–	–	3000	250	95	60	4	–	40	–	3000	70	60	55	4	70	4	6
VRMCL	40000	30000	–	–	–	–	–	–	200	60	150	80	4	200	8	35	4	30	150	6	4	75	6	8
Trust Region	151	61	46	4	5001	41	4	6	10	26	16	4	11	21	4	6	8	4	16	4	4	16	4	4

Table 19: Steps to re-converge Task 2 under continual learning for CW10.

	CW10																			
Method	99%							80%							70%						
	T4	T5	T6	T7	T8	T9	T10	T4	T5	T6	T7	T8	T9	T10	T4	T5	T6	T7	T8	T9	T10
Finetune	–	–	–	5000	–	10000	–	–	–	–	4000	–	7000	–	4000	–	–	3000	–	6000	–
EWC	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Replay	21	4	4	4	6	4	26	4	4	4	4	4	4	8	4	4	4	4	4	4	6
FTML	35	65	4	4	35	15	35	4	4	4	4	10	4	8	4	4	4	4	4	4	6
VRMCL	95	400	40	4	40	500	150	20	4	4	4	4	8	45	20	4	4	4	4	8	8
Trust Region	16	4	4	4	6	4	16	4	4	4	4	4	4	6	4	4	4	4	4	4	6

Table 20: Steps to re-converge Task 3 under continual learning for CW10.

	CW10																			
Method	99%						90%						80%							
	T5	T6	T7	T8	T9	T10	T5	T6	T7	T8	T9	T10	T5	T6	T7	T8	T9	T10		
Finetune	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–		
EWC	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–		
Replay	36	21	4	8	4	21	10	16	4	8	4	8	8	11	4	4	4	4		
FTML	300	55	20	80	–	600	80	50	20	80	4	20	55	50	10	20	4	20		
VRMCL	55	40	40	50	4	–	6	40	8	50	4	–	6	30	8	15	4	75		
Trust Region	21	16	4	61	4	16	21	16	4	10	4	6	8	10	4	4	4	6		

Table 21: Steps to re-converge Task 4 under continual learning for CW10.

	CW10															
Method	99%					90%					80%					
	T6	T7	T8	T9	T10	T6	T7	T8	T9	T10	T6	T7	T8	T9	T10	
Finetune	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	
EWC	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	
Replay	31	6	451	71	76	31	6	31	51	41	26	6	31	4	4	
FTML	–	–	–	–	–	9000	50	–	–	–	550	40	–	–	–	
VRMCL	–	–	–	–	–	7000	10	–	–	–	2000	10	2000	–	350	
Trust Region	96	91	6001	76	301	86	10	51	16	4	51	6	31	4	4	

Table 22: Steps to re-converge Task 5 under continual learning for CW10.

	CW10					
Method	99%		90%		80%	
	T9	T10	T9	T10	T9	T10
Finetune	–	–	–	–	–	–
EWC	–	–	–	–	–	–
Replay	36	2001	4	86	4	4
FTML	65	–	45	–	6	500
VRMCL	–	–	8000	550	4	350
Trust Region	96	–	4	71	4	4

(a) Task 8

	CW10								
Method	99%			90%			80%		
	T8	T9	T10	T8	T9	T10	T8	T9	T10
Finetune	–	–	–	–	–	–	–	–	–
EWC	–	–	–	–	–	–	–	–	–
Replay	51	21	36	10	4	4	10	4	4
FTML	550	550	–	60	6	30	40	4	6
VRMCL	2000	75	–	55	6	–	30	4	45
Trust Region	41	4	36	21	4	4	10	4	4

(b) Task 7

Table 23: Steps to re-converge across tasks 7, 8 under continual learning for CW10.

	CW10		
Method	99%	90%	80%
	T10	T10	T10
Finetune	–	–	30
EWC	30	30	30
Replay	4	4	4
FTML	4	4	4
VRMCL	550	75	10
Trust Region	4	4	4

(a) Task 9

	CW10											
Method	99%				90%				80%			
	T7	T8	T9	T10	T7	T8	T9	T10	T7	T8	T9	T10
Finetune	35	750	–	–	4	4	–	55	4	4	–	55
EWC	40	–	–	–	4	–	–	60	4	4	–	40
Replay	4	4	4	6	4	4	4	4	4	4	4	4
FTML	4	4	8	8	4	4	4	6	4	4	4	6
VRMCL	4	4	4	25	4	4	4	6	4	4	4	4
Trust Region	4	4	6	6	4	4	4	4	4	4	4	4

(b) Task 6

Table 24: Steps to re-converge across tasks 7, 8 under continual learning for CW10.

Asset	URL	License
Datasets / Benchmarks		
ImageNet-500 subset of ImageNet [5, 34]	Link	ImageNet Terms of Access
Meta-World / CW10 source benchmark [46, 41]	Link	MIT License
Code / Libraries		
Hugging Face `diffusers`	Link	Apache License 2.0

Table 25: Table of licenses.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: See Section 3 and 4.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See Section 5.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: See Section 3.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: See Section 4.1, 4.2, and Appendix C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: See supplementary materials.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: See Section 4.1, 4.2, and Appendix C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: See Section 4.3.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See Appendix C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We comply with NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: See Section E.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [No]

    Justification: The work proposed in this paper does not pose these specific risks that may warrant safeguards.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: See Table 25.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We will release our models to github upon acceptance.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core work of this paper does not use LLMs, nor are LLMs used as any important, original, or non-standard components.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.