MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Reinforced Neural Processes: Memory-Efficient Time-Series Forecasting with a World-Feedback-Trained Memory Policy

Khan, Nibraas; Wichern, Gordon; Laughman, Christopher R.

TR2026-095 June 30, 2026

Abstract

Neural Processes (NPs) provide a lightweight framework for uncertainty-aware regression by conditioning predictions on a compact context set of observed input-output examples in set- tings such as meta-regression, Bayesian optimization, and spatiotemporal prediction. In continuous learning settings, however, context selec- tion becomes an online memory problem: as new observations arrive, which examples should be retained? Since retaining every observation is intractable, bounded-memory implementations rely on fixed heuristics such as sliding windows, reservoir sampling, or surprise thresholds, each encoding a static memory prior. We introduce Reinforced Neural Processes (RNP), a backbone-agnostic memory framework that pairs a tiered context buffer with a gated two-branch encoder and learns an insertion/eviction policy from world feedback: the downstream predictive log-likelihood induced by each memory action relative to its counterfactual alternative. We instantiate RNP on attention (R-ANP) and convolutional (R-ConvCNP) backbones and evaluate on four streaming benchmarks (delay-differential sys- tems, regime-switching streams, abrupt-MNIST, and a wearable energy-expenditure dataset) across varying memory budgets. The best RNP variant attains the highest likelihood on 27 of 32 streams

ICML Workshop on Reinforcement Learning from World Feedback (RLxF) 2026

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Reinforced Neural Processes: Memory-Efficient Time-Series Forecasting with a World-Feedback-Trained Memory Policy

Nibraas Khan [1 2] **Gordon Wichern** [1] **Christopher Laughman** [1]

Abstract

Neural Processes (NPs) provide a lightweight framework for uncertainty-aware regression by conditioning predictions on a compact context set of observed input-output examples in settings such as meta-regression, Bayesian optimization, and spatiotemporal prediction. In continuous learning settings, however, context selection becomes an online memory problem: as new observations arrive, which examples should be retained? Since retaining every observation is intractable, bounded-memory implementations rely on fixed heuristics such as sliding windows, reservoir sampling, or surprise thresholds, each encoding a static memory prior. We introduce **Reinforced Neural Processes (RNP)**, a backbone-agnostic memory framework that pairs a tiered context buffer with a gated two-branch encoder and learns an insertion/eviction policy from world feedback: the downstream predictive log-likelihood induced by each memory action relative to its counterfactual alternative. We instantiate RNP on attention (R-ANP) and convolutional (R-ConvCNP) backbones and evaluate on four streaming benchmarks (delay-differential systems, regime-switching streams, abrupt-MNIST, and a wearable energy-expenditure dataset) across varying memory budgets. The best RNP variant attains the highest likelihood on 27 of 32 streams.

1. Introduction

Neural Processes (NPs) are a family of conditional models that learn distributions over functions from examples (Garnelo et al., 2018a;b). Given a small context set of observed input-output pairs, an NP predicts outputs at new query inputs by producing a distribution rather than a single point estimate. This makes NPs useful in settings where predictions from limited observations guide downstream decisions, including Bayesian optimization, active learning, spatiotemporal prediction, and risk-aware control. They are also a natural fit for time-series forecasting when uncertainty estimation matters: forecasts may inform monitoring, control, or intervention decisions, and the model must indicate when limited or shifting observations make its predictions unreliable. Unlike models that must be fine-tuned for each new task, an NP can adapt in a single forward pass by changing its context set. The combination of fast adaptation and calibrated uncertainty makes NPs attractive for continuous learning problems, where observations arrive sequentially as a stream and predictions must be updated as conditions change.

A vanilla NP applied to a stream needs only one extra ingredient: a way of choosing which past observations its context should currently contain. Offline meta-learning never has to ask this, because every episode is handed a freshly sampled context. In principle one could simply retain every past observation, but for high-dimensional streams sampled at high rates, such as wearable sensor channels, image pixels, and multi-channel control telemetry, the memory and per-step compute of an unbounded context become intractable within the deployment cycle. Online, the buffer is therefore bounded, so each admission of a new point forces the eviction of an old one, and those choices compound along the rest of the stream. The implementation default is a fixed heuristic (sliding window, reservoir sampling (Vitter, 1985), surprise threshold), each encoding a single static prior. The right answer generally depends on the system, the prediction horizon, and where in the stream we are. It is a moving target that calls for a moving strategy.

The NP is already equipped to grade memory states for itself. After every step the network produces a calibrated predictive distribution over the next several observations; the log-likelihood of those observations as they arrive is a real-valued, noisy, naturally delayed signal that scores how well the current memory served the prediction. We view this

Published at RLxF: Reinforcement Learning from World Feedback. This work was performed while N. Khan was an intern at MERL. [1]Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, USA [2]Vanderbilt University, Nashville, TN, USA. Correspondence to: Gordon Wichern <wichern@merl.com>.

Proceedings of the 43^{rd} International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026.

signal as world feedback: measurable, consequential, automatically produced by the deployed stream, and grounded in the process the model is trying to learn. Unlike human preference labels, simulator-provided rewards, or static supervised targets sampled independently of the action, this feedback is observed after a memory decision has changed the model's state and shaped its future predictions. Treating the signal as a reward turns online context curation into a reinforcement-learning problem, with the underlying NP serving as both the policy's environment and the source of its training signal.

Reinforced Neural Processes (RNP) implements this idea in three backbone-agnostic pieces. (i) A tiered memory buffer split into a recent FIFO queue and a curated exemplar bank. (ii) A gated two-branch encoder that runs the underlying NP in parallel over the two tiers and fuses them per query through a learned gate; this is the only architectural change we make to the backbone. (iii) A Proximal Policy Optimization (PPO)-trained insertion/eviction policy on the action-relative lookahead predictive log-likelihood, defined as the difference between how the future observations would have been predicted under the chosen memory action and under the counterfactual alternative, summed across a set of lookahead offsets so that the reward reflects both short- and long-range consequences of a memory decision. Subtracting the counterfactual isolates the memory-decision credit from the backbone's overall ability and from stream noise. We instantiate RNP on attention-based ANP (Kim et al., 2019) and translation-equivariant ConvCNP (Gordon et al., 2019); the same algorithm, state features, and reward train both.

Contributions.

- We frame online context curation in NPs as a reinforcement learning problem with the action-relative lookahead predictive log-likelihood as the world-feedback reward, and identify the counterfactual subtraction as what isolates memory credit from backbone ability and stream noise.
- We introduce RNP, a backbone-agnostic algorithm pairing a tiered memory buffer, a gated two-branch encoder, and a PPO-trained insertion/eviction policy that uses the backbone's own embeddings.
- We evaluate RNP across four streaming benchmarks at memory budgets $K \in \{16, 32, 64, 128\}$, including delay-differential and chaotic streams, abrupt regime-switching streams, abrupt-MNIST image completion, and wearable energy-expenditure data, and show that the best RNP variant attains the lowest mean Negative Log-Likelihood (NLL) on 27 of 32 streams across the memory-budget sweep.

2. Related Work

Neural Processes. Conditional NPs (Garnelo et al., 2018a) and stochastic NPs (Garnelo et al., 2018b) introduced amortized meta-learning with explicit context aggregation. Subsequent work has improved expressivity: attention (Kim et al., 2019); convolutions and translation equivariance (Gordon et al., 2019; Foong et al., 2020); equivariance to other group actions (Ashman et al., 2024); Gaussian NPs (Bruinsma et al., 2021); bootstrap NPs (Lee et al., 2020); training objectives (Le et al., 2018); transformer-style sequence-modeling alternatives (Nguyen & Grover, 2022; Müller et al., 2023); practical and autoregressive variants (Bruinsma et al., 2023); spectral convolutional variants (Mohseni & Duffield, 2026); and task-specific applications such as planning with attentive NPs (Jain et al., 2025). All of these works assume that the context set is given.

Streaming and online deployment. Prediction in streams whose distribution shifts during deployment is studied under concept drift (Gama et al., 2014; Hoi et al., 2021) and test-time adaptation (Wang et al., 2020). The Neural Process literature has overwhelmingly considered the offline meta-learning regime, in which an episode is a context-plus-target set drawn i.i.d. from a task distribution, rather than the streaming regime in which the context itself evolves under deployment. Our setting sits in this gap, and the natural question becomes how to construct that evolving context set from a bounded slice of the past.

Memory and context selection. Practical baselines for online context selection are simple heuristics: sliding windows, reservoir sampling (Vitter, 1985), and surprise- or uncertainty-thresholded retention. Each encodes a fixed prior and does not couple the retention rule to the downstream task. The same question is asked in continual learning (what to keep in a bounded rehearsal buffer) and addressed by gradient-conflict criteria (Lopez-Paz & Ranzato, 2017), prototype rehearsal (Rebuffi et al., 2017), maximally-interfered retrieval (Aljundi et al., 2019), prioritised experience replay (Schaul et al., 2015; Mnih et al., 2015), and external memory networks (Santoro et al., 2016). Closer to our setting, Meta-Continual Learning with Neural Process (MCLNP) (Wang et al., 2022) pairs a latent ConvCNP encoder with a coreset-style KL term against past observations. Our framework differs in that the "buffer" *is* the model's predictive context, so curation directly changes the distribution against which we train the selector, and the selector itself is learned. We include reservoir-sampling, surprise-threshold, and NP variants as direct baselines.

Reinforcement learning from world feedback. Where Reinforcement Learning from Human Feedback (RLHF) shapes rewards from human preferences (Christiano et al.,

2017; Casper et al., 2023) and Reinforcement Learning from AI Feedback (RLAIF) replaces the human labeller with another model (Li et al., 2025), a parallel line asks what to do when neither party labels the data and the reward must come from the world the system is deployed into (Dulac-Arnold et al., 2021). Our reward is this kind of world feedback: the next observations of the stream score each memory decision through the model's own predictive log-likelihood, and the counterfactual subtraction isolates the credit due to the memory choice from backbone ability and stream noise. The state and reward reuse the backbone's existing embeddings and likelihood; we optimise with PPO (Schulman et al., 2017) and Generalized Advantage Estimation (GAE) (Schulman et al., 2015).

3. Method

3.1. Neural Process preliminaries

We first fix notation for a standard NP. An NP receives a context set

$$\mathcal{C} = \{(x_i, y_i)\}_{i=1}^{|\mathcal{C}|}, \qquad x_i \in \mathbb{R}^{D_x},\ y_i \in \mathbb{R}^{D_y},$$

of observed input-output pairs and predicts the output y_* at a target input x_*. Unlike a point predictor, an NP returns a predictive distribution conditioned on the context set.

Conceptually, an NP with learnable parameters ϕ has three components: an encoder, an aggregator, and a decoder. The *encoder* maps each observed input-output pair into a feature vector,

$$h_i = e_\phi(x_i, y_i), \tag{1}$$

where h_i is the representation of the individual context point (x_i, y_i). Let

$$\mathcal{H}_\mathcal{C} = \{h_i : (x_i, y_i) \in \mathcal{C}\} \tag{2}$$

denote the set of encoded context features. The *aggregator* combines these features into a query-level context representation,

$$z_* = A_\phi(\mathcal{H}_\mathcal{C}, x_*). \tag{3}$$

For simple NPs, A_ϕ may ignore x_* and reduce to a permutation-invariant pooling operation over $\mathcal{H}_\mathcal{C}$. For an attentive NP, A_ϕ applies self-attention over the context points followed by cross-attention from the query to the context. For a convolutional NP, A_ϕ uses SetConv and convolution to construct a representation over the input domain and evaluates it at the query.

The *decoder* maps the target input and the context representation to the parameters of the predictive distribution,

$$(\mu_*, \sigma_*) = d_\phi(x_*, z_*). \tag{4}$$

Here μ is the predicted mean vector and σ is the predicted standard-deviation vector. We use Gaussian predictive heads with diagonal covariance,

$$p_\phi(y_* \mid x_*; \mathcal{C}) = \mathcal{N}\big(y_* \mid \mu_*, \mathrm{diag}\big(\sigma_*^2\big)\big)\,. \tag{5}$$

We write f_ϕ for the full NP backbone: given a context set and a target input, it returns the predictive Gaussian parameters.

3.2. Problem setup

We consider an online stream of observations

$$\mathcal{S} = \{(x_t, y_t)\}_{t \geq 1}.$$

The subscript t denotes arrival time: at step t, the pair (x_t, y_t) is revealed, with $x_t \in \mathbb{R}^{D_x}$ and $y_t \in \mathbb{R}^{D_y}$. The stream may be non-stationary, so the predictive relationship between inputs and outputs can change over time. At step t, the model has observed the prefix $\{(x_i, y_i)\}_{i \leq t}$, but cannot retain all past observations as NP context.

We denote the stored context at time t by

$$\mathcal{M}_t = \{(x_i, y_i) : i \in I_t\}. \tag{6}$$

Here $I_t \subseteq \{1, \ldots, t\}$ indexes the observations currently stored in memory. The memory has capacity K, meaning that $|\mathcal{M}_t| \leq K$; this is the maximum number of observations that can be supplied to the NP backbone as context.

The current memory $\mathcal{M}_t$ is supplied to the NP backbone f_ϕ as its context. For any future offset $\delta > 0$, let

$$(\mu_{t,\delta}, \sigma_{t,\delta}) = f_\phi(\mathcal{M}_t, x_{t+\delta}).$$

The corresponding predictive distribution is

$$p_\phi(y_{t+\delta} \mid x_{t+\delta}; \mathcal{M}_t) = \mathcal{N}\big(y_{t+\delta} \mid \mu_{t,\delta}, \mathrm{diag}(\sigma_{t,\delta}^2)\big)\,. \tag{7}$$

The learning problem is to update $\mathcal{M}_t$ online. If memory were unbounded, the NP could condition on every past observation. In deployment, however, both storage and per-step computation grow with the context size, so the memory must remain bounded by K. Context selection therefore becomes a sequential decision problem: each update changes the context used by the NP, and the quality of that update is revealed only through future observations.

RNP casts memory management as a sequential decision problem. At each time step, the stream reveals a new observation, but the NP can only condition on a bounded memory of at most K context points. Therefore, when the memory is full, adding a new point requires deciding whether that point is useful enough to keep and, if so, which older point should be removed. We treat this keep/insert-and-evict decision as the action of a policy. The policy observes features computed from the current memory, the candidate point, and the NP backbone, and is trained according to whether its memory choices improve predictive likelihood on subsequent observations. Figure 1 shows this loop.

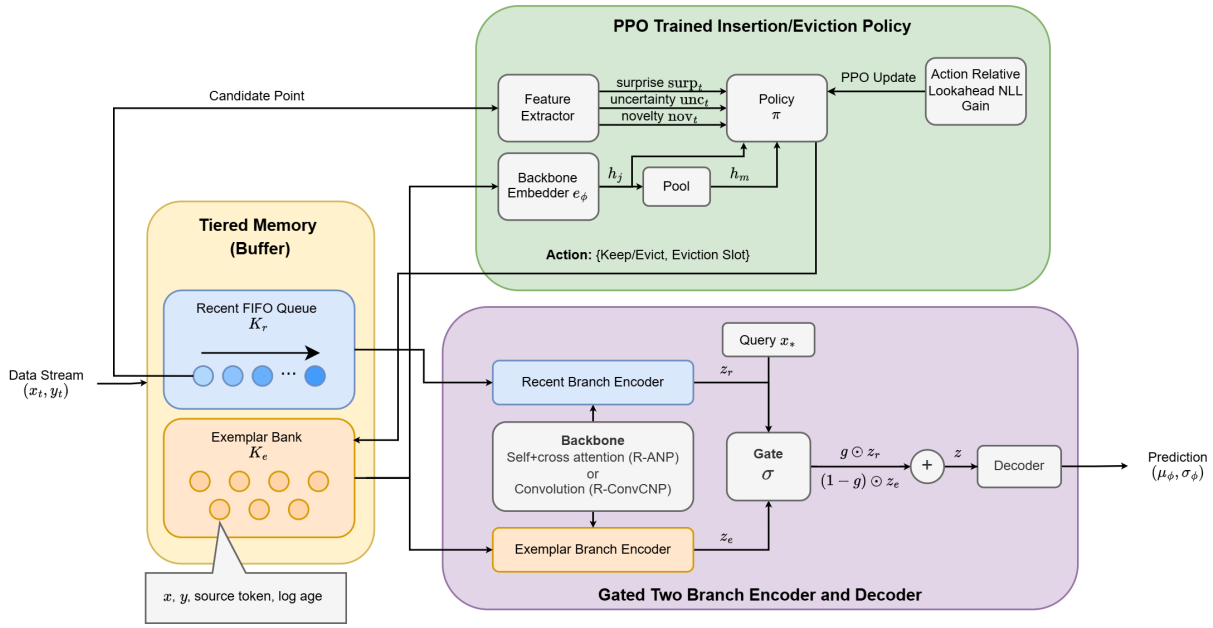

Figure 1. **Reinforced Neural Process.** A streaming context buffer is split into a recent FIFO queue and a learned-curated exemplar bank. The underlying NP backbone runs two parallel branches over the two memory tiers and fuses them through a learned per-feature sigmoid gate before decoding. A separate policy network reads the candidate embedding, exemplar slot embeddings with their log-ages, and three scalar features (surprise, uncertainty, novelty) and decides whether to keep or insert the new point and which slot to evict. The policy is trained with PPO, with a reward equal to the action-relative lookahead predictive log-likelihood gain of the chosen action over the alternative, summed across a configurable set of lookahead offsets. The recipe is backbone-agnostic: "Branch" is attention for R-ANP and 1-D convolution for R-ConvCNP.

3.3. Tiered memory buffer

The stored context $\mathcal{M}_t$ from Eq. 6 is the context set supplied to the NP backbone at time t. RNP organizes this context as a two-tier buffer,

$$\mathcal{M}_t = \mathcal{R}_t \cup \mathcal{E}_t. \tag{8}$$

The recent tier $\mathcal{R}_t$ has capacity K_r, the exemplar tier $\mathcal{E}_t$ has capacity K_e, and the total memory budget is $K_r + K_e = K$. The recent tier $\mathcal{R}_t$ is a FIFO queue that stores the freshest observations, while the exemplar tier $\mathcal{E}_t$ stores older observations selected by the policy.

New observations first enter the recent queue $\mathcal{R}_t$. When $\mathcal{R}_t$ exceeds its capacity K_r, the oldest point in the queue is removed from $\mathcal{R}_t$ and becomes a candidate c_t. The policy then chooses whether to discard this candidate, leaving the exemplar bank unchanged, or to insert it into $\mathcal{E}_t$. If $\mathcal{E}_t$ is already full, inserting c_t requires evicting one exemplar slot. We refer to these two operations as KEEP and INSERT: KEEP keeps the current memory state and discards the candidate, while INSERT promotes the candidate into the exemplar bank.

Each stored point carries two annotations in addition to its input-output values: a learned source token indicating whether the point belongs to $\mathcal{R}_t$ or $\mathcal{E}_t$, and a learned embedding of $\log(1 + \text{age})$, where age is the number of stream steps since the point was observed. These annotations are used by the encoder and policy; the underlying NP context remains the set of stored input-output pairs in $\mathcal{M}_t$. Together, the source and age annotations let the model distinguish fresh observations from older curated exemplars without having to infer recency from x alone.

3.4. Gated two-branch encoder

Each tier is encoded by its own backbone branch and the two outputs are fused per query. Each point in either tier is first encoded by the backbone's point encoder $e(\cdot)$ – the MLP that maps (x, y) into the backbone's representation space – then augmented with a learned source token (recent vs exemplar) and a learned MLP projection of $\log(1 + \text{age})$. For R-ANP the two branches are independent self-attention $\rightarrow$ cross-attention stacks on top of the ANP backbone of Kim et al. (2019); for R-ConvCNP they are independent 1-D convolutional stacks on top of the SetConv operator of Gordon et al. (2019). The two branches share architecture but not weights, so the encoder can specialize its treatment of recent versus exemplar context.

The branch outputs have the same form as the query-level representation in Eq. 3, but are computed separately on the recent and exemplar tiers. Let z_r and z_e be these per-query representations from the recent and exemplar branches at

query x_*, and let q be the backbone's encoding of x_*. A learned per feature sigmoid gate fuses them:

$$g = \mathrm{sigm}\big(\mathrm{MLP}_g([\,q\,;\,z_r\,;\,z_e\,])\big),$$
$$z = g \odot z_r + (\mathbf{1} - g) \odot z_e, \tag{9}$$

where $\odot$ is element-wise product and $g \in [0,1]^d$ is applied per feature. The fused z is concatenated with x_* and decoded as in the underlying NP backbone.

3.5. Insertion/eviction policy

State. At each decision, the policy decides whether the candidate should enter the exemplar bank and, if so, which stored exemplar to replace. Its state summarizes the candidate, the current memory, and their predictive relationship under the backbone. Writing the current candidate as $c_t = (x_t^c, y_t^c)$, where c_t is the point leaving the recent FIFO queue, and using the backbone point encoder e_ϕ from Eq. 1, we use three scalar diagnostics:

$$\mathrm{surp}_t = -\log p_\phi\big(y_t^c \mid x_t^c; \mathcal{M}_t\big), \tag{10}$$
$$\mathrm{unc}_t = \sigma_\phi\big(x_t^c; \mathcal{M}_t\big), \tag{11}$$
$$\mathrm{nov}_t = \min_{m \in \mathcal{M}_t} \big\| e_\phi(c_t) - e_\phi(m) \big\|_2. \tag{12}$$

Surprise (surp_t) is label-aware: it is large when the current memory poorly explains y_c. Uncertainty (unc_t) is label-agnostic: it is large where the backbone is unsure at x_c. Novelty (nov_t) measures coverage: it is large when the candidate is far from stored points in the backbone embedding space. These signals are complementary. A point can be surprising but not novel, novel but unsurprising, or both, so the learned policy can combine them in a stream-dependent way rather than committing to a fixed heuristic.

The global feature for the KEEP/INSERT head is

$$s_t^{\mathrm{global}} = \big[h_c; h_m; \mathrm{surp}_t; \mathrm{unc}_t; \mathrm{nov}_t\big], \tag{13}$$

where $h_c = e_\phi(c_t)$ and h_m is a mean-pooled embedding of the current memory. Each exemplar slot j is scored with a slot-specific feature

$$s_{t,j}^{\mathrm{slot}} = \big[s_t^{\mathrm{global}}; h_j; \log(1 + \mathrm{age}_j)\big], \tag{14}$$

where $h_j = e_\phi(m_j)$ is the embedding of exemplar slot j.

Action and policy network. At each decision, the policy chooses whether to discard the candidate or insert it into the exemplar bank. We write the action as $a_t = (o_t, j_t)$, where $o_t \in \{\mathrm{KEEP}, \mathrm{INSERT}\}$ and j_t is an eviction slot used only when insertion requires replacing a full exemplar bank. The policy factorizes into an operation head and, when needed, a slot head. The operation head reads the global feature s_t^{global} from Eq. 13 and outputs $\pi_\theta(o_t \mid s_t)$.

For eviction, the slot head applies the same MLP to each slot feature $s_{t,j}^{\mathrm{slot}}$ from Eq. 14, producing logits $\ell_{t,j}$ and a distribution over slots,

$$\pi_\theta(j_t = j \mid s_t, o_t = \mathrm{INSERT}) = \frac{\exp(\ell_{t,j})}{\sum_{k=1}^{|\mathcal{E}_t|} \exp(\ell_{t,k})}. \tag{15}$$

Sharing the slot MLP across exemplar slots makes the eviction distribution permutation-equivariant. When no eviction is required, the slot head is masked out and the action probability reduces to the operation probability.

Reward. For an action a that yields memory $\mathcal{M}_t^{(a)}$ and a finite set of positive lookahead *offsets* $\Delta \subset \mathbb{N}_{>0}$, define the lookahead negative log-likelihood

$$\mathrm{NLL}_\Delta\big(\mathcal{M}_t^{(a)}\big) \;=\; -\sum_{\delta \in \Delta} \log p_\phi\big(y_{t+\delta} \,\big|\, x_{t+\delta};\, \mathcal{M}_t^{(a)}\big), \tag{16}$$

evaluated against the same backbone f_ϕ and the same observed future $\{(x_{t+\delta}, y_{t+\delta})\}_{\delta \in \Delta}$. The single reward signal we use is the *action-relative* lookahead gain

$$r_t \;=\; \mathrm{NLL}_\Delta\big(\mathcal{M}_t^{(\bar{a})}\big) \;-\; \mathrm{NLL}_\Delta\big(\mathcal{M}_t^{(a)}\big), \tag{17}$$

where $\bar{a}$ is the action not taken: an action is rewarded to the extent that it improves the predictive likelihood at the chosen offsets over what the alternative would have given. Figure 2 illustrates the construction. We treat Δ as a tunable parameter; multi-horizon offsets let short- and long-range consequences both contribute, which matters because the time scale on which a curated exemplar pays off varies across streams. Because both NLLs are evaluated against the same backbone on the same future, terms attributable to backbone quality and stream stochasticity cancel in expectation, leaving credit due to the memory choice itself.

optimization. We train the policy with proximal policy optimization (Schulman et al., 2017) on the action-relative reward (Eq. 17), using GAE (Schulman et al., 2015) and a standard PPO objective with a squared-error value loss and entropy bonus. We use $\gamma = 0.995$, $\lambda = 0.95$, value-loss coefficient 0.5, entropy coefficient $5{\times}10^{-3}$, and learning rate 10^{-4}. The backbone f_ϕ is not frozen during this phase: it is jointly adapted alongside the policy, so backbone and memory policy co-evolve on the stream rather than the policy chasing a fixed encoder.

3.6. Algorithm

Algorithm 1 summarises the full online training loop.

4. Experiments

We evaluate RNP across four streaming benchmarks at memory budgets $K \in \{16, 32, 64, 128\}$, with $K_r = K_e =$

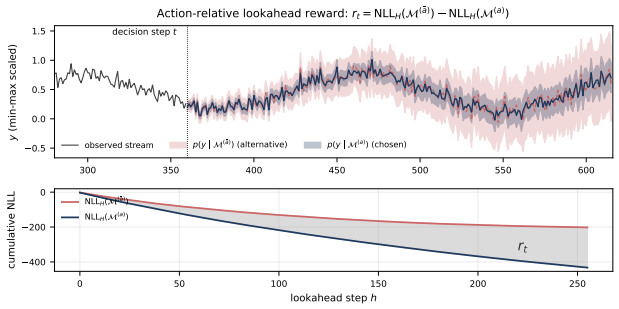

Figure 2. **Action-relative lookahead reward.** *Top:* at the decision step t, we compare the predictive band under the chosen memory (blue) with the band under the counterfactual alternative memory (red), where each band shows the predictive mean plus/minus one standard deviation; the case shown is illustrative, with the chosen memory predicting upcoming observations more tightly. *Bottom:* cumulative NLL of each memory state over the lookahead window. The reward r_t is the signed difference between the two cumulative NLL curves at the chosen offsets: positive when the chosen memory predicts better, negative when the alternative would have. Either way, memory decisions are graded by their actual predictive consequences on the observed stream.

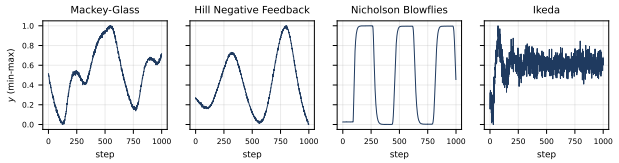

Figure 3. **Time-delayed task gallery.** A representative 1000-step segment of each of the four 1-D streams used in Section 4.1. The four systems span quasi-periodic (Mackey-Glass), slow saturating (Hill negative feedback), sharply peaked density-dependent oscillation (Nicholson), and chaotic discrete-time (Ikeda) dynamics.

$K/2$. The benchmarks span qualitatively different memory regimes: time-delayed streams reward retaining structurally specific past observations; abrupt streams reward forgetting old regimes quickly; Abrupt-MNIST is a 2-D pixel stream with abrupt switches between digit families; WEEE is a real wearable stream that rewards retaining longer-term physiological biomarkers from earlier activity stages in memory. Tables 1 and 2 report tail-20% NLL for every (model, stream, K). Here, tail-20% NLL means the mean negative log-likelihood evaluated over the final 20% of each stream. The subsections below walk through each benchmark family.

Models. We compare nine models throughout. Five are vanilla NP-family baselines: NP (Garnelo et al., 2018b), CNP (Garnelo et al., 2018a), ANP (Kim et al., 2019), ConvCNP (Gordon et al., 2019), and the latent ConvCNP variant MCLNP (Wang et al., 2022). Two are heuristic memory baselines pairing the ANP backbone with a fixed selector at the same context budget K. At step t with the bank full: ANP-RESERVOIR (Vitter, 1985) runs Vitter's reservoir sampling, inserting with probability K/t and evicting a uniformly random slot,

$$\Pr(\text{INSERT}) = K/t, \quad j^* \sim \mathrm{Uniform}(\{1, \dots, K\}); \tag{18}$$

ANP-SURPRISE greedily retains the top-K points by surprise, inserting when surp_t exceeds the smallest stored surprise and evicting that slot,

$$\text{INSERT if } \mathrm{surp}_t > \min_j s_j, \quad j^* = \arg\min_j s_j, \tag{19}$$

where s_j is the stored surprise of slot j. The remaining two models are our **R-ANP** and **R-ConvCNP**. All methods share the same context budget K and backbone update budget. Each experiment uses 10 random seeds. We report seed-mean NLL $\pm$ std; lower is better.

Algorithm 1 Reinforced Neural Process: training

Input: stream $(x_t, y_t)_{t \geq 1}$, NP backbone f_ϕ, policy π_θ with value head V_θ, rollout length L, lookahead offsets Δ.
Initialise bounded tiered memory $\mathcal{M}_0$ as in Eq. 8.
for update $u = 1, \dots, U$ **do**
  Collect a rollout of L memory decisions.
  for decision time t in the rollout **do**
    Obtain candidate c_t from the recent FIFO queue.
    Compute candidate embedding $h_c = e_\phi(c_t)$ and memory summary h_m using Eq. 1.
    Predict under the current memory using Eq. 5.
    Compute surprise, uncertainty, and novelty using Eqs. 10–12.
    Build the global and slot policy states using Eqs. 13–14.
    Sample action $a_t = (o_t, j_t)$ from π_θ, using Eq. 15 for eviction when needed.
    Apply a_t to obtain chosen memory $\mathcal{M}_t^{(a)}$.
    For each $\delta \in \Delta$, evaluate the future point $(x_{t+\delta}, y_{t+\delta})$ under chosen and alternative memories using Eq. 16.
    Compute reward $r_t \leftarrow \mathrm{NLL}_{\text{alt}} - \mathrm{NLL}_{\text{chosen}}$ using Eq. 17.
  end for
  Compute GAE advantages and returns using γ and λ.
  for E PPO epochs over minibatches **do**
    Update θ with the clipped PPO surrogate, value loss, and entropy bonus.
  end for
end for
Return π_θ, V_θ, and the adapted backbone f_ϕ.

Table 1. **NLL (mean ± std) on the time-delayed streams across the memory-budget sweep.** Lower is better. Bold marks the winner across all nine models.

	Time-Delayed							
	Mackey-Glass				Hill			
Model	16	32	64	128	16	32	64	128
NP	-0.86 ± 0.07	-0.38 ± 0.16	0.00 ± 0.08	0.05 ± 0.07	-0.96 ± 0.14	-0.35 ± 0.14	0.06 ± 0.06	0.30 ± 0.04
CNP	-0.89 ± 0.06	-0.44 ± 0.08	-0.01 ± 0.05	-0.06 ± 0.08	-0.92 ± 0.14	-0.42 ± 0.10	0.06 ± 0.11	0.33 ± 0.08
ANP	-1.24 ± 0.09	-0.99 ± 0.06	-0.71 ± 0.14	$\mathbf{-1.46} \pm 0.21$	-1.57 ± 0.14	-1.14 ± 0.06	-0.66 ± 0.04	$\mathbf{-1.67} \pm 0.21$
ConvCNP	-1.05 ± 0.08	-0.57 ± 0.08	-0.09 ± 0.05	-0.24 ± 0.04	-1.25 ± 0.09	-0.73 ± 0.08	-0.17 ± 0.04	0.26 ± 0.04
MCLNP	0.24 ± 0.31	0.14 ± 0.18	0.04 ± 0.08	0.36 ± 0.17	0.88 ± 0.21	0.83 ± 0.28	1.30 ± 0.42	0.92 ± 0.24
ANP-Reservoir	0.71 ± 0.06	0.24 ± 0.06	-0.34 ± 0.07	0.92 ± 0.13	1.11 ± 0.08	0.75 ± 0.06	-0.07 ± 0.10	-0.85 ± 0.09
ANP-Surprise	0.01 ± 0.12	-0.41 ± 0.05	-0.78 ± 0.09	1.78 ± 0.08	0.90 ± 0.07	0.48 ± 0.09	-0.29 ± 0.13	-1.25 ± 0.21
R-ANP	$\mathbf{-1.32} \pm 0.10$	$\mathbf{-1.31} \pm 0.13$	$\mathbf{-0.92} \pm 0.12$	-1.04 ± 0.19	$\mathbf{-1.58} \pm 0.17$	$\mathbf{-1.77} \pm 0.20$	$\mathbf{-1.51} \pm 0.17$	0.15 ± 0.17
R-ConvCNP	-0.90 ± 0.11	-0.70 ± 0.15	-0.33 ± 0.09	-0.02 ± 0.05	-1.13 ± 0.58	-0.84 ± 0.41	-0.58 ± 0.19	-0.16 ± 0.14

	Time-Delayed							
	Nicholson				Ikeda			
Model	16	32	64	128	16	32	64	128
NP	-0.53 ± 0.20	0.01 ± 0.14	0.53 ± 0.04	0.34 ± 0.04	-1.88 ± 0.16	-1.82 ± 0.15	-1.87 ± 0.15	-1.83 ± 0.30
CNP	-0.44 ± 0.06	0.11 ± 0.03	0.50 ± 0.04	0.32 ± 0.02	-1.95 ± 0.13	-1.95 ± 0.18	-1.95 ± 0.12	-1.95 ± 0.13
ANP	-1.07 ± 0.12	-0.71 ± 0.03	$\mathbf{-1.16} \pm 0.18$	$\mathbf{-1.45} \pm 0.32$	-1.89 ± 0.22	-1.92 ± 0.13	-1.91 ± 0.18	-1.95 ± 0.13
ConvCNP	-0.86 ± 0.06	-0.23 ± 0.05	0.38 ± 0.03	0.28 ± 0.05	$\mathbf{-1.97} \pm 0.12$	-1.94 ± 0.14	-1.95 ± 0.17	-1.92 ± 0.18
MCLNP	0.29 ± 0.14	0.42 ± 0.12	0.57 ± 0.06	1.74 ± 0.54	-0.57 ± 0.01	-0.57 ± 0.01	-0.57 ± 0.01	-0.57 ± 0.01
ANP-Reservoir	1.43 ± 0.05	0.79 ± 0.08	-0.13 ± 0.06	1.38 ± 0.26	-1.91 ± 0.19	-1.93 ± 0.12	-1.92 ± 0.18	-1.95 ± 0.12
ANP-Surprise	1.21 ± 0.13	0.33 ± 0.12	-0.90 ± 0.12	0.85 ± 0.16	-1.95 ± 0.13	-1.94 ± 0.11	-1.93 ± 0.14	-1.95 ± 0.12
R-ANP	$\mathbf{-1.28} \pm 0.59$	$\mathbf{-1.55} \pm 0.21$	-1.08 ± 0.16	-1.09 ± 0.22	-1.95 ± 0.11	$\mathbf{-1.96} \pm 0.12$	$\mathbf{-1.96} \pm 0.12$	$\mathbf{-1.96} \pm 0.12$
R-ConvCNP	-0.65 ± 0.40	-0.46 ± 0.35	-0.16 ± 0.17	0.31 ± 0.19	-1.62 ± 0.33	-1.74 ± 0.24	-1.83 ± 0.18	-1.84 ± 0.17

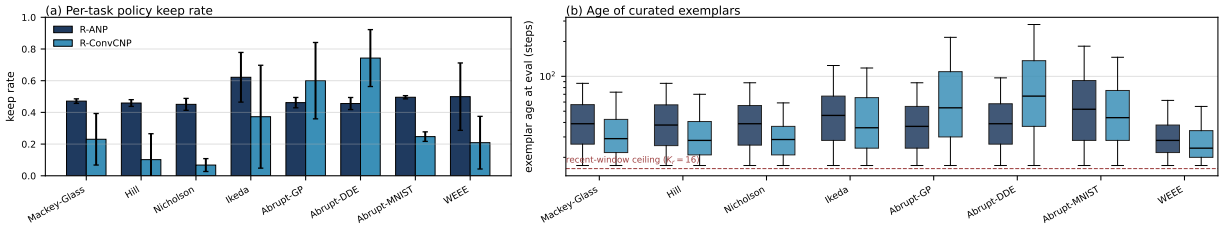

Figure 4. **Policy behavior,** $K = 32$**.** *(a)* Per-task keep rate (fraction of decisions that retain the current memory rather than insert+evict), with seed std error bars. R-ANP and R-ConvCNP converge to different strategies, and both adapt to the task. *(b)* Age distribution of the curated half of the buffer (slots outside the recent FIFO window) at evaluation time, pooled across seeds (and across participants for WEEE). The dashed line marks the recent-window ceiling $K_r = 16$; medians and upper whiskers sit well above it on every task.

4.1. Time-delayed functions

Streams. We evaluate on four canonical nonlinear delay-differential and discrete-time chaotic systems: Mackey-Glass (Mackey & Glass, 1977), Nicholson's blowflies (Gurney et al., 1980), the Hill/Goodwin negative-feedback oscillator (Goodwin, 1965), and the Ikeda map (Ikeda, 1979). Each stream is split into an online training prefix and a held-out evaluation tail (the final 20% of the stream). Parameter settings, integration schemes, and observation-noise levels follow the canonical literature values. Figure 3 shows a representative slice of each stream.

Results. R-ANP attains the lowest NLL on most of the time-delayed streams in the sweep (Table 1). Notably, the heuristic memory baselines (ANP-RESERVOIR, ANP-SURPRISE) *underperform* plain ANP at smaller K, showing that fixed selection rules can actively hurt when the right exemplars are structurally specific to the system's delay; under the same compute budget, only the learned policy improves on the unconstrained recent window.

What does the policy learn? Figure 4(a) shows that the two backbones converge to different curation strategies and that both adapt their keep rate across tasks. Panel (b) shows that the curated half of the buffer holds points well above the recent-window ceiling $K_r = 16$ on every task we report.

4.2. Abrupt regime-switching streams

To test what happens when the dynamics themselves change, we run a second experiment on streams that switch regimes abruptly through the course of the stream, so the recent FIFO queue often crosses one or more regime boundaries

Table 2. **NLL (mean ± std) on the regime-switching, Abrupt-MNIST, and WEEE benchmarks across the memory-budget sweep.** Lower is better. Bold marks the winner across all nine models.

	Abrupt							
	GP				DDE			
Model	16	32	64	128	16	32	64	128
NP	-1.83 ± 0.06	-1.49 ± 0.22	-1.10 ± 0.14	-0.82 ± 0.19	0.29 ± 0.02	0.29 ± 0.01	0.29 ± 0.02	0.30 ± 0.03
CNP	-1.84 ± 0.15	-1.53 ± 0.16	-1.17 ± 0.24	-0.83 ± 0.22	0.30 ± 0.05	0.31 ± 0.04	0.31 ± 0.04	0.33 ± 0.04
ANP	-1.86 ± 0.08	-1.55 ± 0.13	-1.14 ± 0.14	-0.85 ± 0.16	-0.05 ± 0.04	0.26 ± 0.01	0.27 ± 0.03	0.26 ± 0.01
ConvCNP	-1.54 ± 0.45	-1.22 ± 0.34	-1.05 ± 0.19	-0.70 ± 0.18	0.24 ± 0.02	0.25 ± 0.01	0.26 ± 0.03	0.26 ± 0.01
MCLNP	0.05 ± 0.61	0.03 ± 0.46	-0.03 ± 0.24	-0.00 ± 0.33	0.27 ± 0.02	0.27 ± 0.03	0.27 ± 0.02	0.29 ± 0.02
ANP-Reservoir	0.08 ± 0.45	0.19 ± 0.52	-0.09 ± 0.36	-0.30 ± 0.31	0.41 ± 0.05	0.26 ± 0.01	0.26 ± 0.02	0.26 ± 0.01
ANP-Surprise	-0.58 ± 0.52	-0.43 ± 0.53	-0.43 ± 0.43	-0.44 ± 0.38	0.41 ± 0.06	0.26 ± 0.01	0.26 ± 0.01	0.26 ± 0.01
R-ANP	$\mathbf{-1.90} \pm 0.08$	$\mathbf{-1.89} \pm 0.08$	$\mathbf{-1.82} \pm 0.10$	$\mathbf{-1.75} \pm 0.07$	$\mathbf{-0.23} \pm 0.07$	$\mathbf{-0.24} \pm 0.07$	$\mathbf{-0.14} \pm 0.19$	$\mathbf{0.04} \pm 0.24$
R-ConvCNP	-0.39 ± 0.27	-0.39 ± 0.25	-0.39 ± 0.26	-0.39 ± 0.24	0.24 ± 0.01	0.26 ± 0.01	0.27 ± 0.02	0.28 ± 0.05

	MNIST				WEEE			
Model	16	32	64	128	16	32	64	128
NP	0.06 ± 0.04	0.06 ± 0.04	0.06 ± 0.04	0.06 ± 0.04	3.34 ± 0.08	3.40 ± 0.08	3.45 ± 0.08	3.48 ± 0.08
CNP	0.03 ± 0.07	0.04 ± 0.07	0.04 ± 0.07	0.04 ± 0.07	3.37 ± 0.04	3.40 ± 0.04	3.43 ± 0.03	3.40 ± 0.04
ANP	0.10 ± 0.08	0.10 ± 0.08	0.10 ± 0.08	0.10 ± 0.08	3.37 ± 0.02	3.40 ± 0.01	3.44 ± 0.01	3.45 ± 0.02
ConvCNP	0.14 ± 0.07	0.14 ± 0.07	0.14 ± 0.07	0.14 ± 0.07	3.34 ± 0.04	3.38 ± 0.04	3.42 ± 0.05	3.37 ± 0.07
MCLNP	0.01 ± 0.17	0.01 ± 0.17	0.01 ± 0.17	0.01 ± 0.17	4.22 ± 0.00	4.22 ± 0.00	4.29 ± 0.00	4.30 ± 0.00
ANP-Reservoir	0.09 ± 0.08	0.09 ± 0.08	0.09 ± 0.08	0.09 ± 0.08	1.66 ± 0.02	1.66 ± 0.01	1.69 ± 0.01	1.70 ± 0.01
ANP-Surprise	0.18 ± 0.05	0.15 ± 0.09	0.15 ± 0.08	0.15 ± 0.08	1.62 ± 0.02	1.62 ± 0.01	1.62 ± 0.02	1.62 ± 0.01
R-ANP	0.03 ± 0.04	0.01 ± 0.05	0.01 ± 0.05	0.00 ± 0.05	$\mathbf{1.55} \pm 0.02$	$\mathbf{1.55} \pm 0.01$	$\mathbf{1.56} \pm 0.04$	$\mathbf{1.56} \pm 0.02$
R-ConvCNP	$\mathbf{-0.21} \pm 0.10$	$\mathbf{-0.25} \pm 0.12$	$\mathbf{-0.24} \pm 0.12$	$\mathbf{-0.17} \pm 0.23$	1.56 ± 0.02	1.57 ± 0.01	1.58 ± 0.02	1.59 ± 0.02

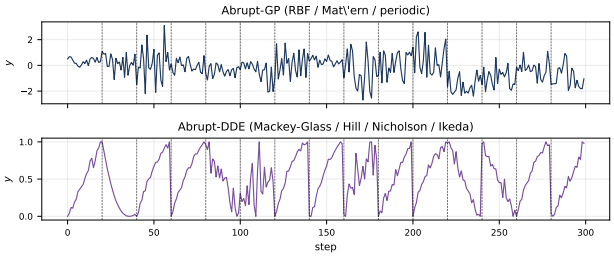

Figure 5. **Abrupt-stream gallery.** Representative slices of the two regime-switching streams used in Section 4.2. Vertical dotted lines mark regime boundaries.

within a single buffer turnover. **Abrupt-GP** concatenates Gaussian-process segments drawn from three kernels (RBF, Matérn-$\frac{3}{2}$, periodic) with hyperparameters resampled per segment. **Abrupt-DDE** concatenates trajectory segments drawn from the four systems of Section 4.1. Both streams use the same length, train/eval split, and protocol as the time-delayed experiment. Figure 5 shows representative slices.

Results. R-ANP is the lowest-NLL model on both streams at every K (Table 2). The gap is largest on Abrupt-DDE, where R-ANP is the only model to achieve a clearly negative tail NLL while every other model clusters near or above zero. The two settings ask different things of the policy: in Section 4.1 the useful exemplars sit far in the past and the policy's job is to keep them; here the useful exemplars have only just arrived and the policy's job is to discard the previous regime quickly. The action-relative lookahead reward is informative about both because both show up in upcoming observations.

4.3. Abrupt-MNIST: streaming image completion

The third benchmark switches modality from real-valued 1-D series to 2-D image completion. **Abrupt-MNIST** emits a stream of full MNIST images. The stream alternates abruptly between five digit *families* (FM1–FM5, pairing digits $\{0,1\}, \ldots, \{8,9\}$) in groups of same-family images (Figure 6). For each image, the model observes only a masked subset of K pixels and must reconstruct the remaining pixels of that same 28×28 image. Memory is maintained separately from the current-image mask: each memory slot stores a compact representation of a prior image, allowing the model to transfer information across recent family shifts while keeping the reconstruction context for the current image local to that image.

Results. **R-ConvCNP** attains the lowest NLL on Abrupt-MNIST at every K (Table 2). Where 1-D streams favor attention, image data favors the convolutional backbone: a learned memory policy on top of a translation-equivariant SetConv encoder extracts more signal than fixed selectors. This is the only benchmark where R-ConvCNP rather than R-ANP is the best performer, direct evidence that the recipe is backbone-agnostic.

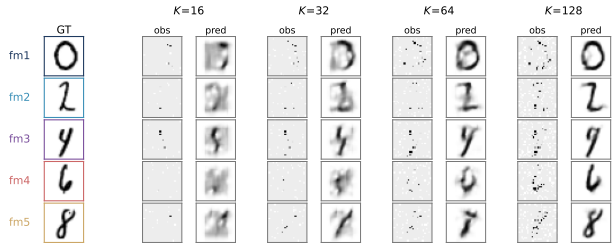

Figure 6. **Abrupt-MNIST per-budget reconstructions.** Held-out reconstructions for one example per digit family at each of the four memory sizes used in the sweep, $K \in \{16, 32, 64, 128\}$. For each $(family, K)$ the left tile shows the observed pixels and the right tile shows the predicted mean; the GT column on the far left is the same image across all four budgets. Recon fidelity grows visibly with K, mirroring the streaming-NLL progression in Tables 1–2.

4.4. WEEE: wearable energy expenditure

The fourth benchmark moves to real, sensor-derived streams. **WEEE** comprises 17 participants wearing a Zephyr chest strap during a structured exercise protocol of seated rest, standing, two cycling stages, and two running stages. Each participant's stream pairs continuous physiological features (heart rate, breathing rate, skin temperature, posture, accelerometer-derived activity, and related signals) with their concurrent energy expenditure (kcal/min) derived from VO_2 and body mass. We treat each participant as an independent stream: train on that participant's recording (interleaved with the policy's online updates) and evaluate predictive NLL on the held-out 20% tail. Numbers are mean over participants and over 10 seeds.

Results. WEEE is the hardest benchmark in absolute terms (Tables 1–2): vanilla NP-family backbones attain tail-20% NLL in the 3.4–4.3 range versus roughly -1 on the 1-D streams. The first jump comes from *any* memory curation: ANP-RESERVOIR and ANP-SURPRISE both drop NLL to the 1.62–1.66 range, a $\approx 52\%$ reduction. Unlike on the time-delayed and abrupt streams, where fixed heuristics underperform plain ANP, here the right exemplars are structurally diffuse and a fixed rule already captures most of the gain.

The second jump comes from learning the policy. R-ANP attains the lowest NLL on WEEE at every K. At the participant level, R-ANP beats ANP-SURPRISE on **15 of 17 participants** (one tie, one 0.001 NLL loss) and ANP-RESERVOIR on 13 of 17 (mean NLL gain 0.107). The median per-participant gain (0.063) tracks the mean (0.064), so the win is not outlier-driven, and the gap is preserved across the memory-budget sweep. R-ConvCNP attains comparable NLL on WEEE, suggesting the gain comes from the curation policy and not the encoder family.

4.5. Memory-budget sweep

We evaluate every model on every benchmark at $K \in \{16, 32, 64, 128\}$ (Tables 1–2). The best RNP variant wins on **27 of 32** streams. R-ConvCNP retains its Abrupt-MNIST lead at every K and R-ANP retains its abrupt GP, abrupt DDE, and WEEE lead at every K. The gain is thus present across the range of context capacities we test, with the largest absolute margins at moderate budgets where the learned curation matters most.

5. Conclusions and Future Work

We posed online context curation in Neural Processes as an RL from real world feedback problem: the model's predictive log-likelihood on the observations following each memory decision, minus the same log-likelihood under the counterfactual, is a clean, ungamed memory teacher. Implemented as RNP (tiered memory, gated two-branch encoder, PPO over the action-relative reward), this consistently beats vanilla NP backbones and fixed-policy memory heuristics on delay-differential streams, abrupt regime-switching streams, an abrupt-MNIST completion task, and a real wearable energy-expenditure dataset, across a memory-budget sweep and on both attention-based and convolutional backbones.

Several directions follow. Replacing the hard counterfactual with a calibrated value function over the memory state would remove the second forward pass and amortise the counterfactual through bootstrapped returns. Richer state features may unlock gains on benchmarks where the three scalars are insufficiently discriminating. Finally, an attention-based slot head over the exemplar bank would let the policy reason about pairwise relations between exemplars rather than scoring each in isolation.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Aljundi, R., Belilovsky, E., Tuytelaars, T., Charlin, L., Caccia, M., Lin, M., and Page-Caccia, L. Online continual learning with maximal interfered retrieval. *Advances in neural information processing systems*, 32, 2019.

Ashman, M., Diaconu, C., Weller, A., Bruinsma, W., and Turner, R. E. Approximately equivariant neural processes. *Advances in neural information processing systems*, 37: 97088–97123, 2024.

Bruinsma, W. P., Requeima, J., Foong, A. Y., Gordon, J.,

and Turner, R. E. The gaussian neural process. *arXiv preprint arXiv:2101.03606*, 2021.

Bruinsma, W. P., Markou, S., Requiema, J., Foong, A. Y., Andersson, T. R., Vaughan, A., Buonomo, A., Hosking, J. S., and Turner, R. E. Autoregressive conditional neural processes. *arXiv preprint arXiv:2303.14468*, 2023.

Casper, S., Davies, X., Shi, C., Gilbert, T. K., Scheurer, J., Rando, J., Freedman, R., Korbak, T., Lindner, D., Freire, P., et al. Open problems and fundamental limitations of reinforcement learning from human feedback. *arXiv preprint arXiv:2307.15217*, 2023.

Christiano, P. F., Leike, J., Brown, T., Martic, M., Legg, S., and Amodei, D. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

Dulac-Arnold, G., Levine, N., Mankowitz, D. J., Li, J., Paduraru, C., Gowal, S., and Hester, T. Challenges of real-world reinforcement learning: definitions, benchmarks and analysis. *Machine Learning*, 110(9):2419–2468, 2021.

Foong, A., Bruinsma, W., Gordon, J., Dubois, Y., Requeima, J., and Turner, R. Meta-learning stationary stochastic process prediction with convolutional neural processes. *Advances in Neural Information Processing Systems*, 33: 8284–8295, 2020.

Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., and Bouchachia, A. A survey on concept drift adaptation. *ACM computing surveys (CSUR)*, 46(4):1–37, 2014.

Garnelo, M., Rosenbaum, D., Maddison, C., Ramalho, T., Saxton, D., Shanahan, M., Teh, Y. W., Rezende, D., and Eslami, S. A. Conditional neural processes. In *International conference on machine learning*, pp. 1704–1713. PMLR, 2018a.

Garnelo, M., Schwarz, J., Rosenbaum, D., Viola, F., Rezende, D. J., Eslami, S., and Teh, Y. W. Neural processes. *arXiv preprint arXiv:1807.01622*, 2018b.

Goodwin, B. C. Oscillatory behavior in enzymatic control processes. *Advances in enzyme regulation*, 3:425–437, 1965.

Gordon, J., Bruinsma, W. P., Foong, A. Y., Requeima, J., Dubois, Y., and Turner, R. E. Convolutional conditional neural processes. *arXiv preprint arXiv:1910.13556*, 2019.

Gurney, W., Blythe, S., and Nisbet, R. Nicholson's blowflies revisited. *Nature*, 287(5777):17–21, 1980.

Hoi, S. C., Sahoo, D., Lu, J., and Zhao, P. Online learning: A comprehensive survey. *Neurocomputing*, 459:249–289, 2021.

Ikeda, K. Multiple-valued stationary state and its instability of the transmitted light by a ring cavity system. *Optics communications*, 30(2):257–261, 1979.

Jain, A., Shaw, S., and Roy, N. Learning attentive neural processes for planning with pushing actions. *arXiv preprint arXiv:2504.17924*, 2025.

Kim, H., Mnih, A., Schwarz, J., Garnelo, M., Eslami, A., Rosenbaum, D., Vinyals, O., and Teh, Y. W. Attentive neural processes. *arXiv preprint arXiv:1901.05761*, 2019.

Le, T. A., Kim, H., Garnelo, M., Rosenbaum, D., Schwarz, J., and Teh, Y. W. Empirical evaluation of neural process objectives. In *NeurIPS workshop on Bayesian Deep Learning*, volume 4, 2018.

Lee, J., Lee, Y., Kim, J., Yang, E., Hwang, S. J., and Teh, Y. W. Bootstrapping neural processes. *Advances in neural information processing systems*, 33:6606–6615, 2020.

Li, M., Lin, J., Zhao, X., Lu, W., Zhao, P., Wermter, S., and Wang, D. Curriculum-rlaif: Curriculum alignment with reinforcement learning from ai feedback. *arXiv preprint arXiv:2505.20075*, 2025.

Lopez-Paz, D. and Ranzato, M. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017.

Mackey, M. C. and Glass, L. Oscillation and chaos in physiological control systems. *Science*, 197(4300):287–289, 1977.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533, 2015.

Mohseni, P. and Duffield, N. Spectral convolutional conditional neural processes. *Advances in Neural Information Processing Systems*, 38:40416–40446, 2026.

Müller, S., Feurer, M., Hollmann, N., and Hutter, F. Pfns4bo: In-context learning for bayesian optimization. In *International Conference on Machine Learning*, pp. 25444–25470. PMLR, 2023.

Nguyen, T. and Grover, A. Transformer neural processes: Uncertainty-aware meta learning via sequence modeling. *arXiv preprint arXiv:2207.04179*, 2022.

Rebuffi, S.-A., Kolesnikov, A., Sperl, G., and Lampert, C. H. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010, 2017.

Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., and Lillicrap, T. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pp. 1842–1850. PMLR, 2016.

Schaul, T., Quan, J., Antonoglou, I., and Silver, D. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Vitter, J. S. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*, 11(1): 37–57, 1985.

Wang, D., Shelhamer, E., Liu, S., Olshausen, B., and Darrell, T. Tent: Fully test-time adaptation by entropy minimization. *arXiv preprint arXiv:2006.10726*, 2020.

Wang, X., Yao, L., Wang, X., Paik, H.-Y., and Wang, S. Uncertainty estimation with neural processes for meta-continual learning. *IEEE Transactions on Neural Networks and Learning Systems*, 34(10):6887–6897, 2022.