BLAST: Bayesian Learning for Adaptive Selection of Transitions for Continual Reinforcement Learning

Mason Nakamura, Saaduddin Mahmud, Shlomo Zilberstein

Manning College of Information and Computer Sciences
University of Massachusetts Amherst, USA
{smahmud,mnakamura,shlomo}@umass.edu

Abstract

An important step toward improving the efficiency and practicality of autonomous systems is enabling the ability to continuously learn and adapt to new environments while maintaining performance on past environments. Traditional deep reinforcement learning methods utilize a complete replay buffer for training, however, not all transitions are useful for learning a policy that minimizes catastrophic forgetting over multiple tasks. We introduce an offline replay-based approach that employs a diverse collection of heuristics as a form of knowledge distillation, reducing the replay buffer to a buffer of critical transitions. Using these heuristics, we introduce a Bayesian experience-selection strategy that actively selects heuristics for transition selection to mitigate catastrophic forgetting over a sequence of tasks. We applied our method to a continual Mujoco environment, showing that Bayesian experience selection reduces catastrophic forgetting and effectively distills knowledge from replay buffers.

Introduction

Deep reinforcement learning has proven to be an efficient method for training a policy on complex environments such as Atari (Mnih 2013) and the game of Go (Silver et al. 2016). However, to produce effective lifelong deep reinforcement learning agents, they must learn efficiently on new tasks while retaining performance on previously learned tasks.

Continual reinforcement learning involves endlessly training an agent on continually changing environments to obtain a policy that maintains plasticity and stability (Rolnick et al. 2019; Abel et al. 2024). More specifically, the two primary challenges of continual learning are (1) avoiding plasticity loss—attaining optimal single-task performance—and (2) mitigating catastrophic forgetting—preventing degradation of performance on previously learned tasks. In this setting, a model is utilized across environments, and the agent is not notified of an environment change a priori. Previous work has proven that lifelong learning agents can operate optimally with an unlimited storage or perfect memory of past experiences (Isele and Cosgun 2018; Knoblauch, Husain, and Diethe 2020). However, in practice, this method is intractable given memory constraints and the limitation of current algorithms' ability to fully utilize the information in perfect memory. Therefore, replay-based selection methods that summarize perfect memory are most promising for efficient CL algorithms (Knoblauch, Husain, and Diethe 2020). As such, we design memory distilled replay buffers that utilize weighted heuristics for pruning transitions, best approximating the replay buffers across tasks. This reduces memory usage in continual reinforcement learning settings and, more importantly, reduces catastrophic forgetting by distilling critical transitions from the replay buffer for effective continual learning.

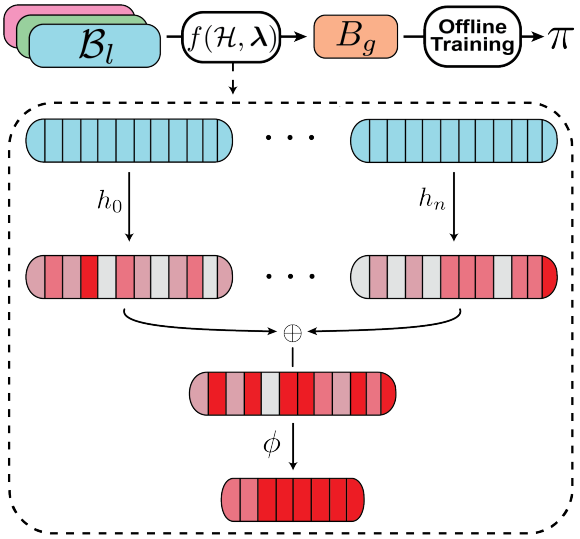

Figure 1: BLAST Framework. BLAST takes a set of local replay buffers $\mathcal{B}_l$ from various tasks and applies a transition selection policy f to each local replay buffer given a set of heuristics H and a vector of learnable parameters $\boldsymbol{\lambda}$ that determines the weight of each heuristic. Every transition is given an aggregated score according to the weighted heuristics. The aggregated buffer is then sorted and pruned according to ϕ, and the remaining transitions are concatenated with other processed buffers and combine into a global replay buffer B_g that is used for offline policy optimization.

We propose a Bayesian experience replay selection framework (BLAST) illustrated in Figure 1. BLAST utilizes Bayesian optimization to reduce catastrophic forgetting over a set of tasks by distilling knowledge from their replay buffers and learning a policy on this distilled knowledge. Unlike previous experience selection strategies that use a

single, static heuristic across every task (Rolnick et al. 2019; Isele and Cosgun 2018), our architecture adapts by utilizing a Bayesian framework for prioritizing heuristics that maximize memory retention among all tasks, minimizing catastrophic forgetting.

Replay-based methods (Rolnick et al. 2019; Isele and Cosgun 2018) have shown promise to mitigate catastrophic forgetting in continuously changing environments with non-stationary data distributions. However, prior work has not explored applying Bayesian learning for dynamically selecting heuristics in replay buffer pruning. To the best of our knowledge, this is the first work to apply Bayesian optimization on heuristic weighted selection for continual reinforcement learning, utilizing multiple heuristics for transition selection.

The **key advantages** of BLAST are:

1. **Composability** – Our framework can leverage arbitrary heuristics for selecting critical transitions.
2. **Modularity** – BLAST, a replay-based approach, can be combined with regularization techniques that minimize catastrophic forgetting to potentially improve memory retention.
3. **Scalability** – Utilizing distributed computing, our method scales with respect to the number of tasks.

Related Work

Continual Supervised Learning Catastrophic forgetting, a known challenge for connectionist networks since the 1990s (French 1999), has been been prominent for classification tasks. Numerous studies have proposed network architecture modifications to combat this issue, such as Local Winner-Take-All (LWTA) blocks that simulate locally competing neurons (Srivastava et al. 2013). Other studies focused on regularization-based techniques such as Elastic Weight Consolidation (EWC) where they slowed updates to parameters that are important for previously learned tasks (Kirkpatrick et al. 2016), reducing the risk of forgetting.

Multi-Task Reinforcement Learning Traditional single-task reinforcement learning methodologies learn a policy for a specified task and discard the policy when migrating to a new task. However, in multi-task reinforcement learning, an agent will learn simultaneously across many tasks (Calandriello, Lazaric, and Restelli 2014), using knowledge learned on other tasks to learn policies that generalizes well across all tasks. However, the agent is typically given a set of tasks or the number of tasks prior to learning; whereas, continual reinforcement learning agents are task-agnostic, making them more practical and scalable.

Neurobiology Prior work on mappings of brains show that hippocampal replay within mammalian brains has been found to assist in decision-making via simulation based on past trajectories (Ólafsdóttir, Bush, and Barry 2018; Carr, Jadhav, and Frank 2011; Wu et al. 2017). Further findings suggest that the replay function is dynamic and dependent on the current task, retrieving specific memories that is needed for efficient task completion (Ólafsdóttir, Bush, and Barry 2018). In other instances, it has been shown that animals replay harmful memories prior to decision-making, invoking a fear memory retrieval that guides animals to avoid averse regions (Wu et al. 2017). Inspired by this, we adopted heuristics that simulate this type of memory retrieval by prioritizing critical transitions that may be useful for distilling knowledge from perfect memory.

Continual Reinforcement Learning A challenge of continual reinforcement learning is the ability to extract important information from a large stream of data in the form of an agent's history (Khetarpal et al. 2022). Typical continual reinforcement learning algorithms are categorized as either replay-based approaches or regularization techniques with the goal of minimizing catastrophic forgetting. In the continual reinforcement learning formulation, agents are generally memory-bounded, making replay-based approaches critical for distilling information from agents' histories. However, regularization techniques are also crucial for enhancing memory retention given an agent's bounded memory. For example, PackNet (Mallya and Lazebnik 2018), a parameter pruning and packing technique, mitigated catastrophic forgetting on both supervised and sequential decision-making domains such as ContinualWorld (Wołczyk et al. 2021). Thus, combining replay-based approaches and regularization techniques will lead to efficient and practical continual learning algorithms.

Background

Heuristics have been developed to select transitions from local replay buffers across tasks, demonstrating promising results in mitigating catastrophic forgetting (Isele and Cosgun 2018). A *local replay buffer*, B_l^i, associated with task i, aligns with the traditional concept of an experience replay buffer in single-task reinforcement learning, containing transitions exclusive to that task. To improve memory efficiency, the local replay buffer is typically discarded once a task has been learned. In contrast, the *global replay buffer*, $B_g^{\mathcal{T}}$, aggregates a subset of transitions from all local replay buffers, $\{B_l^T\}_{T\in\mathcal{T}}$, where $\mathcal{T} = \{T_0, \ldots, T_N\}$ represents the set of all tasks. Transitions $e \in B_g^{\mathcal{T}}$ are referred to as *critical transitions* due to their essential role in consolidating key memories across tasks $\mathcal{T}$.

Tasks and environments can be formally represented as a *Markov decision process* (MDP) represented by the tuple $M = \langle S, A, T, S_0, R, \gamma\rangle$. Here, S is a set of states, A is the set of possible actions for each state, $T : S \times A \to [0, 1]$ is the transition function such that $\sum_{a\in A} T(s, a, s')$, S_0 is the initial state distribution, and $R : S \times A \to \mathbb{R}$ is the reward function. Let $\pi : S \times A \to [0, 1]$ be a policy, mapping from states and actions to a conditional probability distribution over actions conditioned on states. A value function $V^\pi : S \to \mathbb{R}$ induced by policy π is the expected cumulative return for a state $s \in S$ while following policy π. An optimal policy π^* maximizes the expected cumulative return $V^*(s)$ for all $s \in S$.

In the following subsections, we introduce heuristics $h : B \times \pi_\theta \to \mathbb{R}$, which assign a real-valued score to transitions in the replay buffer, $e \in B$. These scores are used within the transition selection function f to identify crit-

ical transitions for inclusion in the global replay buffer. The BLAST framework incorporates three distinct heuristics, each leveraging different aspects of the model or MDP: value-functions (fatality), model parameters (quantization), or observations (coverage). Critical states, a localized view of critical transitions, have been explored in the context of ensuring safety for autonomous agents, where taking appropriate actions in safety-critical states is essential to maintain operational safety (Huang et al. 2018). Beyond safety, critical states also play a role in inverse reinforcement learning tasks with sparse rewards, where they serve as key indicators for identifying important agent decision points in video data (Liu et al. 2023).

Fatality

Agents are said to encounter *fatal transitions* when selecting an incorrect action results in a significant reward difference compared to the optimal action or the average q-value action. Following the definition proposed by (Huang et al. 2018), a transition is classified as fatal if there is a substantial disparity between the maximum and the average q-value, $\hat{Q}$, over all actions for a given observation s_i in transition e_i. This is expressed as:

$$h_{\text{fatal}}(e_i) = \left| \max_{a'} \hat{Q}(s_i, a') - \frac{1}{|A|} \sum_{a'} \hat{Q}(s_i, a') \right|.$$

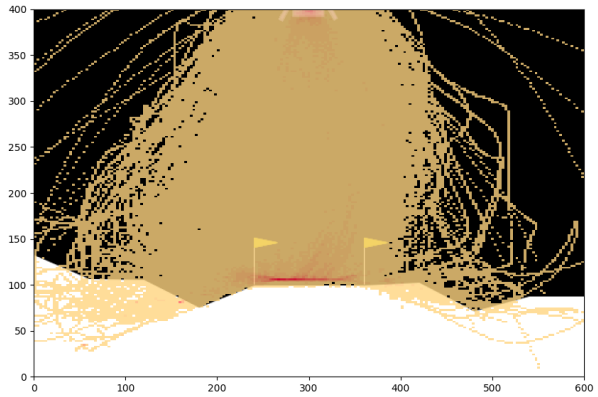

Figure 2: Quantization heuristic applied to the LunarLander Environment (Towers et al. 2024) over the (x, y) position features using DQN (Mnih 2013) on 300k timesteps. Bright red regions represent higher criticality while unmasked regions were not traversed during training. Intuitively, critical regions represent states where quantization from FP32 to FP16 on the Q-network results in large differences in the Q-value predictions.

Quantization

Building on regularization techniques for robust networks post-training (AskariHemmat et al. 2022), we define a *quantized critical transition* as a transition exhibiting significant differences in q-values when the weights of a Q-network are perturbed or pruned. Transitions that produce a high score based on the quantization heuristic indicate that the states in those transitions are distributed across a broader set of neurons compared to other states. The distance function is denoted as d, $\hat{Q}^*$ represents the quantized or pruned Q-function, and $\boldsymbol{a} \in A^n$ is a vector of actions of size n. The heuristic is defined as:

$$h_{\text{quantization}}(e_i) = \text{dist}(\hat{Q}(s_i, \boldsymbol{a}) - \hat{Q}^*(s_i, \boldsymbol{a})).$$

Coverage

Experience selection through coverage maximization has shown promising results in various continual learning domains (Isele and Cosgun 2018). This approach evaluates critical states by aiming to maximize coverage of the state space within a given environment, ensuring a diverse set of transitions. Let $d \in \mathbb{R}^+$ denote the threshold parameter and dist() represent a distance function. The coverage score for a transition $e_i \in B_l^T$ is defined as:

$$h_{\text{coverage}}(e_i) = -\left| \{e_j \mid \text{dist}(e_i - e_j) < d, \forall e_j \in B_l^T\} \right|.$$

Catastrophic Forgetting

An agent exhibits catastrophic forgetting if performance on at least one previously learned task is sufficiently lost after training on a new task. As seen in Figure 3, when transferring learning to a new environment, the agent's policy on the previously learned environment is nearly forgotten with issues learning on the new task due to plasticity loss. More formally, let $R_i^{\pi_j}(s) \in [0, 1]$ be the max-min normalized return with respect to their best and worst-case returns starting from state s on previous task i induced by the policy π_j from current task j. Then, catastrophic forgetting loss between two sequential tasks induced by a single policy can be defined as

$$\chi_{i,j} = R_i^{\pi_j}(s) - R_j^{\pi_j}(s)$$

and the catastrophic forgetting loss of a policy π_j on all previous tasks is

$$\chi_j = \sum_{i=0}^{j-1} \chi_{i,j}.$$

Intuitively, if $\chi_j = 0$ then π_j exhibits memory retention, and if $\chi_j > 0$ then π_j exhibits memory generalization. Whereas, if $\chi_j < 0$ then π_j exhibits memory degradation or catastrophic forgetting if χ_j is sufficiently negative.

In the BLAST framework, a policy is trained on a concatenated set of environments, effectively removing the sequential nature of catastrophic forgetting as previously defined. To address this, we introduce the *forgetting loss*, defined as the difference between the optimal return and the averaged normalized return over tasks:

$$\chi_{\mathcal{T}} = 1 - \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} R_t^{\pi}(s), \quad s \sim s_0^t,$$

where s_0^t represents the initial state distribution for task $t \in \mathcal{T}$. When training policy π on a global replay buffer

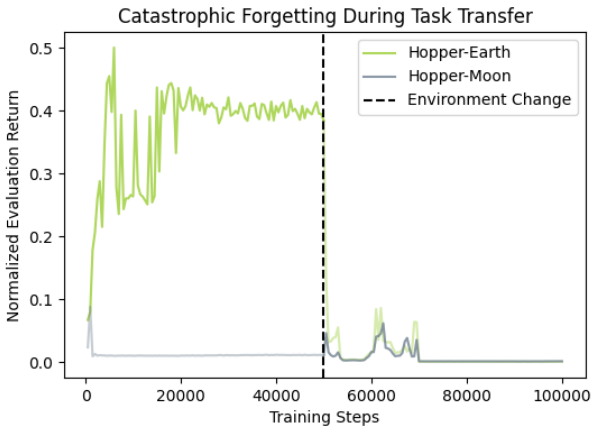

Figure 3: During task transfer, the agent's ability to retain a near-optimal policy on the previous environment is lost. The darker colored line represents the return on the task currently being learned. During task transfer, the gravitational transition dynamics are changed.

constructed using BLAST with weighting coefficients $\boldsymbol{\lambda}$, we denote the catastrophic forgetting loss over the task set $\mathcal{T}$ as $\chi_{\mathcal{T}}(\pi^{\boldsymbol{\lambda}})$, where $\pi^{\boldsymbol{\lambda}}$ is the policy induced by the weights $\boldsymbol{\lambda}$ as depicted in Algorithm 1.

Transition Selection Policy

Let $\mathcal{H}$ be the function space of heuristic functions, $\mathbb{H} = \mathcal{P}(\mathcal{H})$ be the power set of $\mathcal{H}$, Λ be the unit-n simplex $\triangle^n$, $\mathcal{B}$ be the space of possible replay buffers, and Φ be the function space of pruning functions. A transition selection policy $f : \mathbb{H} \times \Lambda \times \mathcal{B} \times \Phi \to \mathcal{B}$ operates on a local replay buffer $B_l^T \in \mathcal{B}$. Using a set of heuristics $H \in \mathbb{H}$ and a weight vector $\boldsymbol{\lambda} \in \Lambda$ a real-valued score is assigned to each transition $e \in B_l^T$. The scored transitions are subsequently processed and pruned to extract critical local memories into a global replay buffer. Let $\boldsymbol{\lambda} = \langle \lambda_0, \ldots, \lambda_n \rangle$ be a vector of learnable parameters, and let $\mathbf{h}(e) = \langle h_0(e), \ldots, h_n(e) \rangle$ be the vector of scores from n heuristics. Then, the score for transition $e \in B_l^T$ is defined as

$$\begin{aligned} h(e) &= \lambda_0 \cdot h_0(e) + \ldots \lambda_n \cdot h_n(e) \\ &= \boldsymbol{\lambda}^T \mathbf{h}(e). \end{aligned}$$

These scores are sorted then pruned with proportion $\rho \in [0, 1]$ using a pruning function $\phi : [0, 1] \times \mathcal{B} \to \mathcal{B}$. The resulting pruned buffer is concatenated with the global buffer $B_g^{\mathcal{T}}$ where we repeat transition selection for all local replay buffers in $\{B_l^i\}_{i=0}^{|\mathcal{T}|}$.

Problem Formulation

Continual Reinforcement Learning (CRL) introduces a setting where agents are required to continuously adapt to new tasks without forgetting how to perform previous ones. Formally, a CRL agent is trained on a set of N tasks, $\mathcal{T} = \{T_0, \ldots, T_N\}$, each of which is represented as a MDP,

Algorithm 1: Policy Optimization, ψ

Require: Set of heuristics $\mathcal{H}$, set of local replay buffers $\mathcal{B}_l$, set of tasks $\mathcal{T}$, and a vector of weights $\boldsymbol{\lambda}$.

```
B_g = {}
for all T ∈ 𝒯 do
    B = f(λ, ℋ, B_l^T)
    B_g := B_g ∪ B
end for
Train offline DRL algorithm on B_g, obtaining π^λ
return π^λ
```

$M_i = \langle S_i, A_i, T_i, R_i, \gamma_i \rangle$ for all tasks $i \in \mathcal{T}$. An optimal policy π^* over all tasks $\mathcal{T}$, given task-specific optimal policies π_i^* for all $i \in \mathcal{T}$, can be defined as

$$\pi^* = \arg\max_{\pi \in \Pi} \sum_{i=0}^{j-1} R_i^{\pi}(s) - R_{i+1}^{\pi}(s),\ s \sim s_0^t$$

which aims to maximize the difference in the return induced by π^* and the individual returns induced by their task-specific optimal policies π_t^* where s_0 is the initial state distribution. Intuitively, π^* minimizes catastrophic forgetting across a set of tasks $\mathcal{T}$. However, finding π^* is intractable from costly evaluation and large policy parameter spaces, making search infeasible. This, instead, directs attention to approaches that reduce the policy search space Π by adapting the network architecture or selectively choosing transitions for training π^*.

In this paper, we aim to address the challenge of catastrophic forgetting by introducing a novel experience selection framework that employs Bayesian optimization to actively prioritize experiences based on their contribution to catastrophic forgetting across tasks.

Proposed Method

BLAST (Algorithm 1) utilizes a transition selection function f that requires predefined heuristics $\mathcal{H}$, parameters $\boldsymbol{\lambda}$ that are learned via Bayesian optimization, and a pruning function ϕ. BLAST minimizes catastrophic forgetting across all tasks $\mathcal{T}$ by selecting transitions using $f(\boldsymbol{\lambda}, \mathcal{H}, \phi, B_l^T)$ from the local replay buffers, $\mathcal{B}_l = \{B_l^T \mid T \in \mathcal{T}\}$, and inserting them into the global replay buffer $B_g^{\mathcal{T}}$. Next, we train an offline deep reinforcement algorithm (e.g., CQL) on the global buffer $B_g^{\mathcal{T}}$ to compute $\chi_{\mathcal{T}}$ for each Bayesian update. During Bayesian optimization we model the posterior of $\chi_{\mathcal{T}}$ on a set of learnable weights $\boldsymbol{\lambda}$. Then, we sample for an optimal set of weights $\boldsymbol{\lambda}^*$ from the posterior, and select transitions using $f(\boldsymbol{\lambda}^*, \mathcal{H}, \mathcal{B}_l) = B_g^{\mathcal{T}}$.

By adapting multiple heuristics for experience selection, we increase the space of possible selection strategies, permitting a larger search space for finding a memory stabilizing set of heuristics compared to prior work that employed single heuristics across tasks (Isele and Cosgun 2018).

Bayesian Learning

Our objective is to maximize memory retention $\chi_{\mathcal{T}}(\pi)$ on all tasks $\mathcal{T}$ by optimally selecting a vector of weights $\boldsymbol{\lambda} \in \Lambda$

Algorithm 2: Bayesian Optimization

Require: Bayesian updates m, number of initial samples n, set of heuristics $\mathcal{H}$, set of local replay buffers $\mathcal{B}_l$, set of tasks $\mathcal{T}$.

$\mathcal{D} = \{\}$
while $j < n$ **do**
  $\boldsymbol{\lambda} \sim \mathcal{N}(\mu, \Sigma)$
  $\pi^{\boldsymbol{\lambda}} = \psi(\mathcal{H}, \mathcal{B}_l, \mathcal{T}, \boldsymbol{\lambda})$
  $y = \chi_{\mathcal{T}}(\pi^{\boldsymbol{\lambda}})$
  $\mathcal{D} := \mathcal{D} \cup \{(\boldsymbol{\lambda}, y)\}$
  $j := j + 1$
end while
while $i \leq m$ **do**
  Update posterior distribution for $\chi_{\mathcal{T}}$ using $\mathcal{D}$
  Compute maximizer of the acquisition function, $\boldsymbol{\lambda}^*$
  $\pi^{\boldsymbol{\lambda}^*} = \psi(\mathcal{H}, \mathcal{B}_l, \mathcal{T}, \boldsymbol{\lambda}^*)$
  $y = \chi_{\mathcal{T}}(\pi^{\boldsymbol{\lambda}^*})$
  $\mathcal{D} := \mathcal{D} \cup \{(\boldsymbol{\lambda^*}, y)\}$
  $i := i + 1$
end while
return $\pi^{\boldsymbol{\lambda}^*}$

that will affect policy optimization on $\pi^{\boldsymbol{\lambda}}$,

$$\max_{\boldsymbol{\lambda} \in \Lambda} \chi_{\mathcal{T}}(\pi^{\boldsymbol{\lambda}}). \tag{1}$$

Here, $\chi_{\mathcal{T}}(\pi^{\boldsymbol{\lambda}})$ is difficult to compute since a deep reinforcement learning policy needs to be trained on a global replay buffer which is the concatenation of pruned local replay buffers that need to be selected using a transition selection function f across all tasks $\mathcal{T}$. Moreover, after training, the learned policy is evaluated on N tasks where the test-time returns can then compute $\chi_{\mathcal{T}}(\boldsymbol{\lambda})$. In the proceeding section, we discuss how we scale the computation of $\chi_{\mathcal{T}}(\boldsymbol{\lambda})$ with respect to N, improving computational efficiency.

Now, we define the Bayesian optimization setup in Algorithm 2 to approximately solve 1, given a dataset $\mathcal{D} = \{(\boldsymbol{\lambda}_i, \chi_{\mathcal{T}}(\boldsymbol{\lambda_i}))_{i=0}^n\}$ where $\chi_{\mathcal{T}}(\boldsymbol{\lambda_i}) \in [0, 1]$, we define the Bayesian update rule as $P(\boldsymbol{\lambda} \mid \mathcal{D}) \propto P(\mathcal{D} \mid \boldsymbol{\lambda}) P(\boldsymbol{\lambda})$. The likelihood is defined as $P(\mathcal{D} \mid \boldsymbol{\lambda}) = e^{\beta(\chi_{\mathcal{T}}(\boldsymbol{\lambda_i})-1)}$, where $\beta \in [0, \infty)$ is a static parameter and N is the total number of tasks. This likelihood function applies higher likelihood to instances of $\boldsymbol{\lambda}$ that preserve memory and promote memory generalization. Additionally, we represent the prior distribution, $P(\boldsymbol{\lambda})$, as a multivariate Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ where $\boldsymbol{\mu}$ is the mean vector and $\boldsymbol{\Sigma}$ is the covariance matrix of the prior distribution.

We model the acquisition function using the Gaussian Process Upper Confidence Bound (GP-UCB) algorithm (Srinivas et al. 2009) that leverages a hyperparameter $\beta_t \in [0, \infty)$ to balance between exploration and exploitation within the search space:

$$\boldsymbol{\lambda}_t = \arg\max_{\boldsymbol{\lambda} \in \Lambda} \mu_{t-1}(\boldsymbol{\lambda}) + \sqrt{\beta_t} \sigma_{t-1}(\boldsymbol{\lambda})$$

where $\mu_{t-1}(\boldsymbol{\lambda})$ and $\sigma_{t-1}(\boldsymbol{\lambda})$ represent the predicted mean and uncertainty of the model, respectively. When updating the posterior distribution, $P(\boldsymbol{\lambda} \mid \mathcal{D})$, we append $(\boldsymbol{\lambda}_t, \chi_{\mathcal{T}}(\pi^{\boldsymbol{\lambda}}))$ to the dataset $\mathcal{D}$, allowing for continuous improvement of the model over m iterations.

Given the significant computation for evaluating the forgetting loss, $\chi_{\mathcal{T}}(\pi^{\boldsymbol{\lambda}})$, leveraging multiple machines for parallel objective function evaluation can accelerate Bayesian optimization. In addition, computing $\mathbf{h}(e)$ can be precomputed for all transitions in each local replay buffer, requiring only a dot product calculation and the application of ϕ for computing transition selection function f.

Online Reinforcement Learning

In online reinforcement learning, a stream of tasks is given to an agent sequentially. In this setting, the agent will learn a local policy π_l for each task, resetting the local policy after each task change. For on-policy algorithms that tend to have small replay buffers, we use the trained policy π_l to generate an offline dataset of trajectories, forming a local replay buffer B_l^T for task T. In the off-policy setting that tends to have larger replay buffers, this buffer used for training can directly be used as the local replay-buffer for pruning. Next, BLAST can be applied to prune the local replay buffer and concatenate the pruned buffer to the global replay buffer. Furthermore, during online reinforcement learning, the memory-preserving policy trained on the global buffer can be learned on-demand, when needed, to minimize computation or as tasks change to maintain a read-to-use policy.

Experiments

We analyze our approach on the Mujoco (Todorov, Erez, and Tassa 2012) Hopper environment with comparisons to a random uniform transition selector. In the following experiment, we aim to answer the the following question: what are the effects of weighted, heuristic-based transition selection on forgetting loss?

Environments

The Mujoco suite of physics-based environments (Todorov, Erez, and Tassa 2012) utilizes continuous observations and continuous actions with constant reward throughout each episode. In our experiments, we used the D4RL (Fu et al. 2020) dataset for Hopper-medium-v0 on 100k observations which we name Hopper-Earth. In addition, we created an alternate Hopper environment, Hopper-Moon, where we changed the transitional dynamics by changing the gravitational constant to that of the Moon.

Offline Reinforcement Learning

In these experiments, we utilize an offline dataset of 100k trajectories produced by medium-level[1] agents for Hopper-Earth and Hopper-Moon. To enable the computation of scores for each transition required by the heuristics Quantization and Fatality, we first trained a CQL agent and extracted its Q-functions. We then applied transition selection to efficiently prune 50% to 99% of the buffer for each

[1]The policy generated an approximate average return of 2260 on each environment

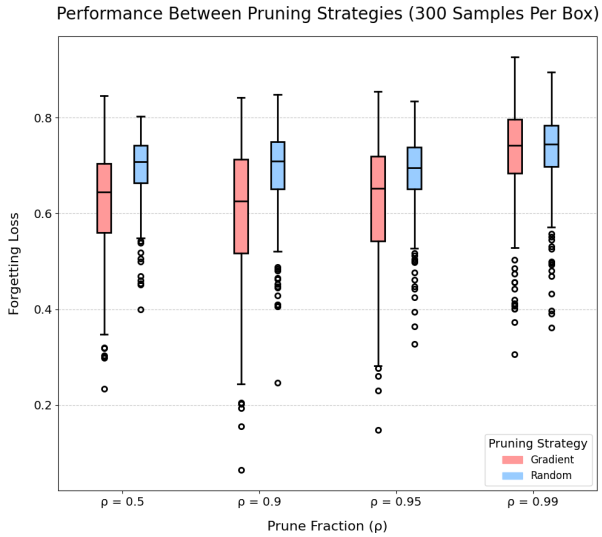

Figure 4: BLAST minimizes catastrophic forgetting on the continual Hopper environment using the last 300 samples during Bayesian optimization for the gradient-based pruning technique compared to uniform random sampling.

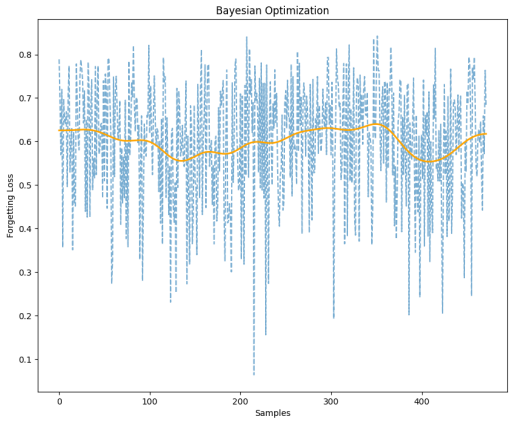

Figure 5: Bayesian Optimization over Λ with pruning fraction of $\rho = .9$ and gradient-based pruning.

Bayesian sample using fatality, quantization, and coverage as heuristics. We used a gradient-based transition selection strategy that removes transitions based on the distribution of the scores, $h(e)$. More specifically, we pruned transitions with smaller gradients, prioritizing the removal of plateaus in the score distribution which maintains a diverse collection of transitions.

We utilized the Conservative Q-Learning (CQL) algorithm (Kumar et al. 2020) for training on the global buffer, however, other offline algorithms such as Implicit Q-Learning (IQL) (Kostrikov, Nair, and Levine 2021) or Decision Transformers (Chen et al. 2021) may also be employed, depending on the problem setting. During training, we utilized a batch size of 512 and a maximum step size of 40k with 2k step checkpoints, using the model checkpoint with highest online return for evaluation on 100 episodes.

During Bayesian optimization over Λ to minimize forgetting loss as seen in Figure 5, we observed instability in CQL training, leading to significant variance in forgetting loss for each $\lambda \in \Lambda$. However, increasing the number of samples is expected to reduce this variance, enabling the optimization process to more accurately reflect the true relationship between forgetting loss and the sampled hyperparameters. Despite this, we found that collections of heuristics were able to reduce forgetting without the potential improvement from Bayesian optimization.

In Figure 4, we deploy BLAST on both Hopper-Earth and Hopper-Moon, showing its potential to distill knowledge from the local replay buffers and reducing catastrophic forgetting when compared to the baseline, uniform random pruner. However, when pruning 99% of the local buffer, we found their to be no difference in performance between random and gradient-based pruning. We also found the pruning function ϕ to significantly effect forgetting loss, motivating directions in finding efficient pruning functions.

Conclusion

We present a novel Bayesian optimization framework for reducing catastrophic forgetting by actively selecting heuristics to build pruned replay buffers of critical transitions for specific tasks, then concatenating these critical transitions into a global replay buffer for training a multi-task policy. To scale with respect to the number of tasks, BLAST utilizes distributed computing across task instances and evaluations, making computation practical in real-world scenarios. Beyond continual reinforcement learning, heuristics may reduce memory requirements for offline reinforcement learning datasets by distilling knowledge from local replay buffers. For future work, we aim to extend the application of BLAST to other continual learning environments, such as HomeGrid (Lin et al. 2023) and ContinualWorld (Wołczyk et al. 2021), to further evaluate its generalizability. Additionally, we plan to integrate both offline and online reinforcement learning evaluations, demonstrating its applicability across diverse learning paradigms. To address challenges arising from noisy evaluation functions, we intend to enhance the Bayesian optimization framework by incorporating techniques from noise-aware Bayesian optimization and leveraging Bayesian neural networks for more robust and reliable optimization.

Acknowledgments

This research was supported in part by the U.S. Army DEVCOM Analysis Center (DAC) under contract number W911QX23D0009, and by the National Science Foundation under grants #2321786, #2326054, and #2416459.

References

Abel, D.; Barreto, A.; Van Roy, B.; Precup, D.; van Hasselt, H. P.; and Singh, S. 2024. A definition of continual reinforcement learning. *Advances in Neural Information Processing Systems*, 36.

AskariHemmat, M.; Hemmat, R. A.; Hoffman, A.; Lazarevich, I.; Saboori, E.; Mastropietro, O.; Sah, S.; Savaria, Y.; and David, J.-P. 2022. QReg: On regularization effects of quantization. *arXiv preprint arXiv:2206.12372*.

Calandriello, D.; Lazaric, A.; and Restelli, M. 2014. Sparse Multi-Task Reinforcement Learning. In Ghahramani, Z.; Welling, M.; Cortes, C.; Lawrence, N.; and Weinberger, K., eds., *Advances in Neural Information Processing Systems*, volume 27, 819–827.

Carr, M.; Jadhav, S.; and Frank, L. 2011. Hippocampal replay in the awake state: A potential substrate for memory consolidation and retrieval. *Nature Neuroscience*, 14: 147–53.

Chen, L.; Lu, K.; Rajeswaran, A.; Lee, K.; Grover, A.; Laskin, M.; Abbeel, P.; Srinivas, A.; and Mordatch, I. 2021. Decision Transformer: Reinforcement Learning via Sequence Modeling. arXiv:2106.01345.

French, R. 1999. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3: 128–135.

Fu, J.; Kumar, A.; Nachum, O.; Tucker, G.; and Levine, S. 2020. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*.

Huang, S. H.; Bhatia, K.; Abbeel, P.; and Dragan, A. D. 2018. Establishing appropriate trust via critical states. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 3929–3936.

Isele, D.; and Cosgun, A. 2018. Selective experience replay for lifelong learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 3302–3309.

Khetarpal, K.; Riemer, M.; Rish, I.; and Precup, D. 2022. Towards continual reinforcement learning: A review and perspectives. *Journal of Artificial Intelligence Research*, 75: 1401–1476.

Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N. C.; Veness, J.; Desjardins, G.; Rusu, A. A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; Hassabis, D.; Clopath, C.; Kumaran, D.; and Hadsell, R. 2016. Overcoming catastrophic forgetting in neural networks. *CoRR*, abs/1612.00796.

Knoblauch, J.; Husain, H.; and Diethe, T. 2020. Optimal continual learning has perfect memory and is np-hard. In *International Conference on Machine Learning*, 5327–5337.

Kostrikov, I.; Nair, A.; and Levine, S. 2021. Offline Reinforcement Learning with Implicit Q-Learning. arXiv:2110.06169.

Kumar, A.; Zhou, A.; Tucker, G.; and Levine, S. 2020. Conservative Q-Learning for Offline Reinforcement Learning. arXiv:2006.04779.

Lin, J.; Du, Y.; Watkins, O.; Hafner, D.; Abbeel, P.; Klein, D.; and Dragan, A. 2023. Learning to model the world with language. *arXiv preprint arXiv:2308.01399*.

Liu, H.; Zhuge, M.; Li, B.; Wang, Y.; Faccio, F.; Ghanem, B.; and Schmidhuber, J. 2023. Learning to identify critical states for reinforcement learning from videos. In *IEEE/CVF International Conference on Computer Vision*, 1955–1965.

Mallya, A.; and Lazebnik, S. 2018. Packnet: Adding multiple tasks to a single network by iterative pruning. In *IEEE conference on Computer Vision and Pattern Recognition*, 7765–7773.

Mnih, V. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Ólafsdóttir, H. F.; Bush, D.; and Barry, C. 2018. The role of hippocampal replay in memory and planning. *Current Biology*, 28: R37–R50.

Rolnick, D.; Ahuja, A.; Schwarz, J.; Lillicrap, T.; and Wayne, G. 2019. Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32.

Silver, D.; Huang, A.; Maddison, C.; Guez, A.; Sifre, L.; Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; Dieleman, S.; Grewe, D.; Nham, J.; Kalchbrenner, N.; Sutskever, I.; Lillicrap, T.; Leach, M.; Kavukcuoglu, K.; Graepel, T.; and Hassabis, D. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529: 484–489.

Srinivas, N.; Krause, A.; Kakade, S. M.; and Seeger, M. 2009. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*.

Srivastava, R. K.; Masci, J.; Kazerounian, S.; Gomez, F. J.; and Schmidhuber, J. 2013. Compete to compute. *Advances in Neural Information Processing Systems*, 2310–2318.

Todorov, E.; Erez, T.; and Tassa, Y. 2012. MuJoCo: A physics engine for model-based control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 5026–5033.

Towers, M.; Kwiatkowski, A.; Terry, J.; Balis, J. U.; De Cola, G.; Deleu, T.; Goulão, M.; Kallinteris, A.; Krimmel, M.; KG, A.; et al. 2024. Gymnasium: A Standard Interface for Reinforcement Learning Environments. *arXiv preprint arXiv:2407.17032*.

Wołczyk, M.; Zajac, M.; Pascanu, R.; Kuciński, Ł.; and Miłoś, P. 2021. Continual world: A robotic benchmark for continual reinforcement learning. *Advances in Neural Information Processing Systems*, 34: 28496–28510.

Wu, C.-T.; Haggerty, D.; Kemere, C.; and Ji, D. 2017. Hippocampal awake replay in fear memory retrieval. *Nature Neuroscience*, 20.