STATISTICAL CONTEXT DETECTION FOR DEEP LIFELONG REINFORCEMENT LEARNING

Jeffery Dick[1], Saptarshi Nath[1], Christos Peridis[1], Eseoghene Benjamin[1,2], Soheil Kolouri[3], Andrea Soltoggio[1]

[1] Loughborough University, United Kingdom

[2] The Alan Turing Institute, United Kingdom

[3] Vanderbilt University, Tennessee

j.dick@4open.science, {s.nath, c.peridis, a.soltoggio}@lboro.ac.uk

ABSTRACT

Context detection involves labeling segments of an online stream of data as belonging to different tasks. Task labels are used in lifelong learning algorithms to perform consolidation or other procedures that prevent catastrophic forgetting. Inferring task labels from online experiences remains a challenging problem. Most approaches assume finite and low-dimension observation spaces or a preliminary training phase during which task labels are learned. Moreover, changes in the transition or reward functions can be detected only in combination with a policy, and therefore are more difficult to detect than changes in the input distribution. This paper presents an approach to learning both policies and labels in an online deep reinforcement learning setting. The key idea is to use distance metrics, obtained via optimal transport methods, i.e., Wasserstein distance, on suitable latent action-reward spaces to measure distances between sets of data points from past and current streams. Such distances can then be used for statistical tests based on an adapted Kolmogorov-Smirnov calculation to assign labels to sequences of experiences. A rollback procedure is introduced to learn multiple policies by ensuring that only the appropriate data is used to train the corresponding policy. The combination of task detection and policy deployment allows for the optimization of lifelong reinforcement learning agents without an oracle that provides task labels. The approach is tested using two benchmarks and the results show promising performance when compared with related context detection algorithms. The results suggest that optimal transport statistical methods provide an explainable and justifiable procedure for online context detection and reward optimization in lifelong reinforcement learning.

1 INTRODUCTION

Deep reinforcement learning algorithms (Sutton & Barto, 2018; Mnih et al., 2015a) for learning single tasks have become effective in many domains (Arulkumaran et al., 2017). The extension to learning multiple sequential tasks is a focus of recent research in lifelong reinforcement learning (LRL) (Khetarpal et al., 2022). Such effort is motivated by the observation that real-life scenarios could entail many sequential tasks. LRL borrows from the related field of lifelong learning (LL) (Thrun, 1998; Silver et al., 2013; Kudithipudi et al., 2022; Soltoggio et al., 2024) that more generally research machine learning (ML) algorithms that perform well with different ML paradigms when data distributions change over time. To aid the LL algorithm, ML research to detect, categorize, or label different data distributions is becoming increasingly more relevant (Kolouri et al., 2019a; Kim et al., 2022)

Different approaches to LL (Parisi et al., 2019; Lange et al., 2022) have proven effective in a variety of different domains. Such approaches can be grouped into weight plasticity and regularization methods, replay methods, and structural adaptation methods. Weight plasticity and regularization approaches include EWC (Kirkpatrick et al., 2017), SCP (Kolouri et al., 2019a), SI (Zenke et al., 2017), MAS (Aljundi et al., 2018), and ANCL (Kim et al., 2023). These methods work by anchoring learning around the important parameters of previous tasks. Therefore, these approaches require task labels, or at least the task change time, to consolidate the parameters when required. One exception is the task-free continual learning protocol by Aljundi et al. (2019) that performs consolidation when the agent's performance is stable, effectively implementing an implicit detection of good performance on a task (Section 2.3).

Examples of memory and replay approaches include REMIND (Hayes et al., 2020), DGR (Shin et al., 2017), and LwF (Li & Hoiem, 2017). While most of these methods focus on classification, experience replay has been used in the

reinforcement learning setting (Mnih et al., 2015b) with a particular focus on continual learning in CLEAR (Rolnick et al., 2019). Under certain conditions, these methods may not require task labels, but in such a case, replay buffers require unbounded size to store experiences from all tasks.

Methods that adapt the structure of the network to learn multiple tasks include multi-head approaches (Yosinski et al., 2014), Progressive NNs (Rusu et al., 2016), XdG (Masse et al., 2018), PathNet (Fernando et al., 2017), mask based approaches (Ben-Iwhiwhu et al., 2023; Wortsman et al., 2020), and HAT (Serra et al., 2018). In these approaches, weights are either added to the network to scale to new tasks or different weights are activated for dealing with different tasks. Task labels are particularly useful with structural adaptation methods to associate particular structures, e.g., network masks, with tasks.

Task changes in RL can be categorized as changes in the input distribution, changes in the transition function, or changes in the reward function. Methods to detect changes in input distributions have been developed in the field of novelty and out-of-distribution detection with applications in LL (Geisa et al., 2021; van de Ven et al., 2022; Aljundi et al., 2022; Liu et al., 2022). One interesting property of LRL is that changes in the reward function cannot be detected only by looking at the distribution of the observations under a random policy. In such cases, the policy can determine or change the distribution, effectively making it more difficult to determine the current task. In this paper, we address specifically this case.

The algorithm proposed in this paper aims to detect changes in deep RL tasks and match the best policy for each task. It is designed to operate online by measuring distances between experiences in the data stream, represented by suitable latent spaces in the network architecture. Using statistical methods, the proposed algorithm determines whether the task has changed. To do so, mechanisms are introduced to achieve a desired low level of false positives, to account for distribution changes under evolving policies, and to re-detect changes seen in the past. Task similarities are measured via the Sliced Wasserstein Distance (SWD) which is subsequently used with the Kolmogorov-Smirnov (KS) statistical test to determine task changes. The resulting method is named *Sliced Wasserstein Online Kolmogorov-Smirnov (SWOKS)*. Simulations are conducted on the CT-graph (Soltoggio et al., 2023), Minigrid (Chevalier-Boisvert et al., 2021) and Half-Cheetah (Todorov et al., 2012) benchmarks, and compared with the baselines Task-Free Continual Learning (TFCL) (Aljundi et al., 2019), Model-Based RL Context Detection (MBCD) (Alegre et al., 2021) and Replay-based Recurrent Reinforcement Learning (3RL) (Caccia et al., 2023).

2 Related Work

The effort to detect novel distributions and changing tasks has seen developments in a number of research areas that are reviewed in this section. The following approaches provide the foundations and motivate the novel method proposed later in section 4.1.

2.1 Novelty and Out Of Distribution Detection

In the absence of labeled data, out-of-distribution (OOD) detection methods such as Lee et al. (2018); Haroush et al. (2021); de Faria et al. (2016) have been developed to detect novel distributions in classification. These methods classify individual samples as in or out of distribution. Similarly, novelty detection (Sedlmeier et al., 2019) aims to determine when data no longer fits an expected distribution. Such approaches may prove useful in LL (Aljundi et al., 2022) by providing confidence values on distributions of incoming data. One example is the inductive conformal prediction (ICP) framework (Papadopoulos, 2008) that provides a statistical estimation, or confidence, of the correctness of a predictive function. ICP learns such estimates via training, calibration, and validation phases, which limits its applicability to LL. The max-Simes-Fisher (MaSF) algorithm (Haroush et al., 2021) frames OOD detection in DNNs as a statistical hypothesis testing problem. It uses evidence from the entire network to generate p-values for each sample.

2.2 Context Detection

Context Detection methods are designed to detect changes in context or task in reinforcement learning (RL). Early examples include approaches such as Da Silva et al. (2006a;b) that can learn and label different tasks for tabular RL algorithms by tracking prediction error rates of partial models. For more challenging approximate methods, Kessler et al. (2022) introduced OWL that learns task labels during a preliminary training phase.

2.2.1 MODEL BASED RL CONTEXT DETECTION

In the field of model-based RL, Alegre et al. (2021) introduced an algorithm (MBCD) for online context detection with approximate methods that do not require pre-training. By utilizing an ensemble model of neural networks, MBCD estimates a model of the environment, including the probability of given state-action-reward-state (SARS) tuples appearing. The SARS tuples are associated with an estimated probability, emulating the tabular methods applied in discrete state (PO)MDP environments. MCUSUM is used to determine which task is most likely: the current task, a previous, a seen task, or a brand new task. One objective of MBCD is task detection with few samples. This objective is achieved by training simultaneously a Bayesian Neural Network (BNN) and a policy network. The BNN is trained on a rolling window of the agent's history, and at each time step predicts what the environment's output should be. One disadvantage of the method is that the training of the BNN can be computationally expensive.

2.3 OTHER APPROACHES TO TASK DETECTION IN LIFELONG LEARNING

In an early approach to LRL, Brunskill & Li (2013) introduced a multi-task learning algorithm for use in tabular methods (i.e., prior to the introduction of deep approximate RL). The algorithm, however, requires a pre-training phase to learn the parameters of different MDPs and applies clustering to determine which knowledge is beneficial in new tasks.

Task-free LL approaches such as Aljundi et al. (2019); Ye & Bors (2022) learn from a non-stationary data stream without an explicit task boundary given. The approach in Aljundi et al. (2019) consolidates knowledge when the agent's performance is stable, thus avoiding the problem of task detection. One drawback to many task-free LL approaches is the reliance on a single policy network to learn multiple tasks, which is unable to deal with interfering tasks if labels are not provided to the input (Kessler et al., 2022).

Promising novel approaches are based on metrics that measure distances between datasets. In Liu et al. (2022), Wasserstein embeddings (WE) are used that map a set of high-dimensional data points to a lower-dimensional Euclidean space. The distance between embeddings of different sets approximates the Wasserstein distance (WD) between the sets and can be used to measure the similarity between sets. These types of approaches are an essential step to perform statistical tests and assess whether an incoming stream of data belongs to the same task or to a different task.

3 PRELIMINARIES

3.1 TASK AND ENVIRONMENT

In RL, at each time step $t \in \mathbb{N}$, an agent is exposed to an observation $o_t \in \Gamma$ and reward $r_t \in \mathbb{R}$. The observation is a function of the state, $o_t = O(s_t) \in \Gamma$. The agent can influence the next state encountered ($s_{t+1} \in S$) by taking actions at each time step, $a_t \in A$, with probability given by the transition function, $T : (S \times A) \to (S)$. The transition is associated with a reward given by the reward function, $r_t = R(s_t)$. In LRL environments, tasks are added to the environment definition. Each task $k \in K$ may be associated with its own $S_k, A_k, T_k, R_k, \Gamma_k$ and O_k.

3.2 SLICED WASSERSTEIN DISTANCE

The Wasserstein distance (WD) is a measure of the distance between two probability distributions over a given metric space. It originates from the field of optimal transport, where the goal is to find the most efficient way to transform one distribution into another. It has gained increased popularity recently for measuring distances in datasets, for use in RL, classification learning, and generative image models (Arjovsky et al., 2017).

There are many notions of dissimilarity for probability measures, among which are ϕ-divergences, integral probability metrics, and optimal transport-based distances (e.g., Wasserstein distances). In our application, we have specific desiderata for such a dissimilarity measure: (i) it should be able to measure dissimilarities between probability measures with disjoint supports; (ii) it should have a small sample complexity; (iii) it should be computationally efficient. The sliced-Wasserstein distance is one such measure that satisfies all these conditions, whereas alternative distances do not. For instance, Wasserstein distances exhibit poor sample complexity and are not computationally efficient, while ϕ-divergences are unsuitable for comparing distributions with non-overlapping supports.

Given two datasets D_1, D_2 of equal size, assume each data point has equal significance. Then, the WD between those two datasets is given by:

$$W(D_1, D_2) = \inf_{\sigma} \sqrt{\sum_{i=1}^{n} \|D_{1,i} - D_{2,\sigma(i)}\|^2} \quad , \tag{1}$$

where $\sigma : [1, n] \cap \mathbb{N} \rightarrow [1, n] \cap \mathbb{N}$ is a permutation [1]. Solving Eq. 1 can be computationally expensive for large data sets. For this reason, the sliced Wasserstein distance (SWD) (Rabin et al., 2012; Bonneel et al., 2015; Kolouri et al., 2019b) is often used to reduce the computational requirements whilst still providing a useful approximation to the true WD. The SWD provides an estimation of the true WD via radon transforms:

$$SW(D_1, D_2) = \int_{\mathcal{S}} W(D_1^{\theta}, D_2^{\theta})\, d\theta \quad , \tag{2}$$

where $\mathcal{S}$ is the unit sphere, and D^{θ} is the radon transform of D at the angle of θ. Due to the one-dimensional nature of D_i^{θ}, and given that the optimal transport map is cyclically monotonous (i.e., in one dimension, it is monotonically increasing), the optimal permutation π can be obtained by sorting the sliced samples, $\{D_{k,i}^{\theta}\}_{i=1}^{N}$. By choosing a high number of sample projections, a high accuracy can be attained.

3.3 Kolmogorov-Smirnov statistical test

The Kolmogorov-Smirnov (KS) statistical test is designed for use on non-parametric distributions. It compares two empirical cumulative distribution functions (ECDFs) and determines whether these samples are likely to be drawn from different distributions (alternative hypothesis). The null hypothesis is that the samples are drawn from the same distribution. We specifically use the one-sided KS test. For two ECDFs X_1 and X_2, the one-sided KS statistic is defined as

$$KS(X_1, X_2) = \sup_{x} \left(\mathrm{P}[X_1 < x] - \mathrm{P}[X_2 < x]\right) \quad . \tag{3}$$

The null hypothesis that the sample distributions are drawn from the same underlying function is rejected at confidence level α if

$$KS(X_1, X_2) > \sqrt{\frac{-0.5\left(|X_1| + |X_2|\right) \times \ln(0.5\alpha)}{|X_1| \times |X_2|}} \quad . \tag{4}$$

Enhancements to the K-S have been developed to overcome limitations such as the applicability to continuous distributions only, e.g., the Anderson-Darling (Anderson & Darling, 1952) or the Cramer Von-Mises (Anderson, 1962) tests, which could be considered a refinement of the proposed method.

3.4 Network Masks

Network masks are structural adaptation methods for lifelong learning. Their use in supervised learning is demonstrated in Wortsman et al. (2020); Kim et al. (2022). Masks in reinforcement learning have also shown promise due to their ability to combine knowledge from multiple tasks (Ben-Iwhiwhu et al., 2023; Nath et al., 2023). Masking approaches work by combining a network with constant parameters θ with a mask that identifies a subset of θ. Multiple such masks $\{m_1, m_2, \ldots, m_n\}$ with parameters ϕ_i can be used to learn multiple policies. Due to the separation of masks, interfering tasks can be learned by different masks. When a particular task i is identified as the current task, the scores for mask m_i are applied and trained using edge-popup (Bengio et al., 2013). Details of the implementation of the masking approach follow those in Ben-Iwhiwhu et al. (2023) and are further expanded in the Appendix.

4 Sliced Wasserstein Online Kolmogorov-Smirnov (SWOKS)

Let $K = \{k_1, k_2, \ldots, k_n\}$ be the set of tasks available in the environment in which each task has a corresponding T_k, R_k, and O_k function. Let k_t be the task governing the environment at time step t, and $z' \in Z$ be the label predicted by an agent at that time. The algorithm attempts to minimize the sum of differences between predicted labels and true tasks (where δ is the *Dirac delta* function):

$$E = \min_{F} \left(\sum_{t \in T} \delta\left(F(z') - k_t\right) \right) \quad . \tag{5}$$

[1] The WD can use a non-bijective mapping: we use one in this case due to both datasets containing the same number of equally weighted points.

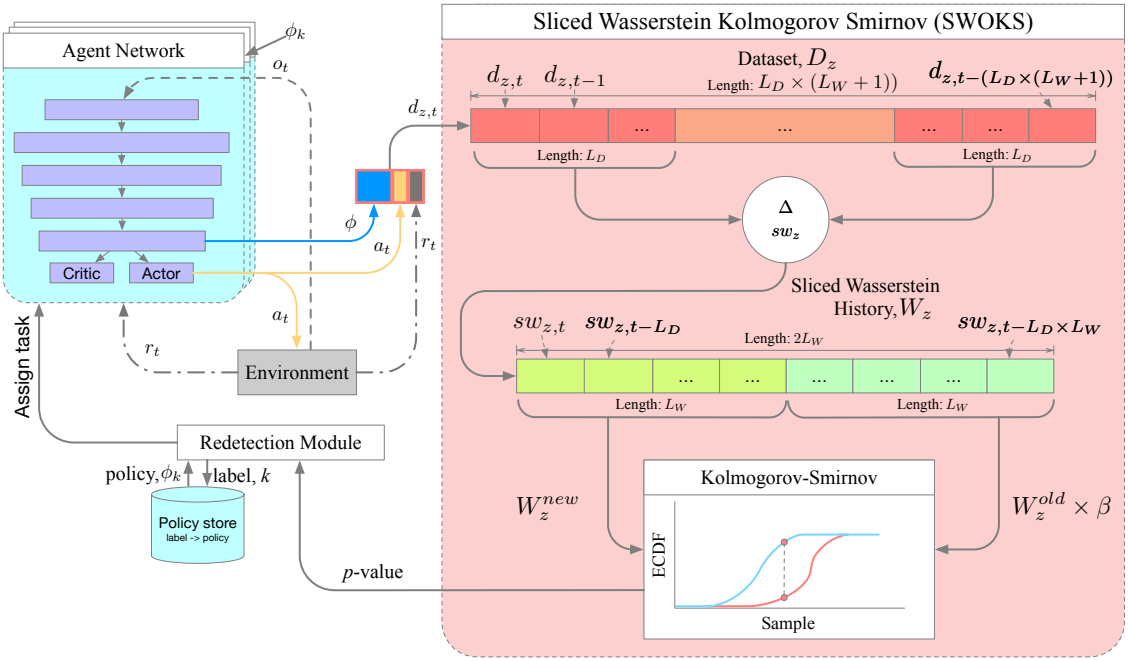

Figure 1: Graphical representation of the SWOKS architecture.

The correct identification of the present context allows a LL system to train multiple policies $\Pi = \{\pi_1, \pi_2, ..., \pi_n\}$ with different subsets of the data stream with the aim of maximizing the infinite horizon return at each timestep.

Maximizing the expected return relies upon both correct task labels and the successful training of each policy. The key aspects of the algorithm are (1) the tuning of the false positive and false negative rates for the specific problem domain (Section 4.1); (2) the ability to re-detect a previously seen task (Section 4.2); (3) the ability to match a subset of the data stream to the correct policy for each task (Section 4.2).

SWOKS uses the most recent L_D data points in the stream as reference set (D_0) to compute a distance with an older set D_z^{old}, $L_D \times L_W$ time steps in the past. L_W is the number of sets, corresponding to distance measurements, that can be taken over a sliding window buffer of length $L_D \times (L_W + 1)$. Thus, $D_{z'} \in \mathbb{R}^{(L_D \times (L_W+1)),(|\phi|+2)}$ where $d_{0,t} \in \mathbb{R}^{(|\phi|+2)}$ is a single data point in D_0, and contains the concatenation of $\sqrt{len(\phi|t)} \times r_t, a_t$, and ϕ_t (Fig. 1). In summary,

$$sw_{z,t} = SW(D_0, D_z^{old}) \quad , \tag{6}$$

is computed for each new set D_0, thus at a frequency of L_D time steps.

4.1 Kolmogorov-Smirnov calculation and adjustments

The Kolmogorov-Smirnov test is performed on the set of data points derived from Eq. 6 at every L_W time steps, comparing the most recent distances sw from indices $[t, \ldots, t - L_W]$ and an older set of distances sw at indices $[t - L_W - 1, t - (2 \times L_W)]$. However, the application of the standard test does not lead to useful results due to the multiple testing problem (Weisstein, 2004; Benjamini & Hochberg, 1995) that increases the probability of making at least one Type I error when performing numerous tests. This corresponds in our case to a false detection of task change. Various methods for adjustments to reduce the rate of Type I errors have been proposed in literature (Benjamini & Hochberg, 1995). Bonferroni correction alone is not useful for SWOKS as the α adjustment must scale to the number of tests performed, which in our case is unbounded. Preliminary tests (not shown) suggested a correction procedure that works well in SWOKS: a multiplicative increase (β) of the reference SWD (W_z^{old}) to reduce the frequency at which small variances in SWD hold statistical significance. The standard setup has a significance level $\alpha = 0.001$ and a $\beta = 1.1$ upward adjustment for the reference SWD W_z^{old}. Future experimental tests could establish the exact relationship between such settings and the precise rate of Type I and Type II errors. However, in the set of experiments we ran across all proposed benchmarks, we observed that such adjustments are robust enough to reliably detect task changes. One strength of the algorithm is that, if experiments on one particular benchmark fail due to excessive Type I or II errors, the β parameter can be adjusted accordingly.

Algorithm 1 SWOKS main loop

$z' \leftarrow 1$ // current task
$t \leftarrow 1$ // time step
$Z \leftarrow [1]$
$last\_z\_change \leftarrow 1$
while true **do**
  $t+ = 1$
  $D_{z'}$.FIFOupdate($[\phi_t,\ a_t,\ r_t]$)
  if $L_D \% t == 0$ **then**
    $W_{z'}$.FIFOupdate($sw(D_{z'}^{new})$
    $K_{z'} \leftarrow KS(W_{z'}^{new}, W_{z'}^{old} \times \beta)$
    if $K_{z'} < \alpha$ **then**
      Re-detection Algorithm
    end if
  end if
end while

Algorithm 2 Re-detection

if $t - last\_z\_change < stablePhase$ **then**
  Abort testing procedure
end if
for $z \in Z \setminus z'$ **do**
  D_0^* = test(policy(z))
  if $p\text{-val}_{task} > \alpha$ **then**
    $D_0 \leftarrow D_0^*$
    $z' \leftarrow z$
    Exit()
  end if
end for
Z.append($z_{|Z|+1}$)
$z' \leftarrow z_{|Z|+1}$
$last\_z\_change \leftarrow t$

4.2 Task change detection and re-detection challenges

When the p-value generated by the KS calculation drops below the significance threshold parameter α, the most recent data is assumed to be produced by a new, different distribution. If this is the first task change detected by SWOKS, a new task label is created to track the new detected task. Otherwise, SWOKS must decide whether a previously seen task label fits the current data, or whether a new task label should be created (Algorithm 2).

Task re-detection poses the following challenges.

1. The policy under which a task was experienced in the past is an integral part of the data distribution.
2. A consequence of the previous point, a policy that undergoes learning will also lead to a shifting distribution of the observations, the latent spaces, and the rewards.
3. The KS test reveals when two distributions are different for detecting new tasks, but no statistical method offers a test to rematch previously seen distributions.
4. The exact data point at which a distribution change might not be known because each test performed corresponds to a set of observations rather than a single data point.

To address challenge 1 when attempting to re-detect a given task, the agent must deploy the policy that was used when originally experiencing that task. The solution to the swapping policy adopted here is based on recent advances in the use of modulating masks in LRL (Ben-Iwhiwhu et al., 2023). Masks are compact representations of policies that can be applied to network backbones to learn multiple tasks. SWOKS reproduces the LRL implementation proposed in Ben-Iwhiwhu et al. (2023).

To address challenge 2, two solutions are implemented. The first is to enable task detection only after an initial learning phase when engaging with a new task. We set a value *stablePhase* $= 50,000$ time steps. A second solution is to use RL algorithms that perform gradual policy updates, e.g., TRPO (Schulman et al., 2015) or PPO (Schulman et al., 2017).

To address challenge 3, we consider the KS test results after each policy assessment in the loop through policies devised to address challenge 1. If the p-value is greater than the significance threshold, it means the data is not different enough to assume it came from a different task. Whilst this test does not guarantee the tasks are the same, if the tasks appear indistinguishable it is reasonable to treat them as the same task.

To address challenge 4, SWOKS implements a roll-back mechanism. The current policy, encoded as a network mask, and trained via PPO, is backed up every 50 iterations. Upon task detection, the current mask is considered corrupted with data from the new task, and the previously backed-up network is stored in association with the previous task. In addition, a SAC implementation is used to benchmark SWOKS in continuous environments. In this case, we used the same setup as in MBCD Alegre et al. (2021). All the steps outlined in this section are summarized in Algorithms 1 and 2. The source code for reproducing the simulations is available at https://github.com/JupiLogy/swoks.

5 EXPERIMENTS

Simulations are conducted on three benchmarks: CT-graph (Soltoggio et al., 2023), Minigrid (Chevalier-Boisvert et al., 2021), and Mujoco Half-Cheetah. The CT-graph and Half-Cheetah environments have task-independent observations, i.e., different tasks have the same observation space. The Minigrid environment has observations that may vary slightly across tasks. SWOKS is implemented in combination with modulating masks and PPO for the environments CT-graph and Minigrid, and with SAC for the continuous environment Half-Cheetah.

The Mujoco Half-Cheetah environment is set up with four different tasks, performed in sequence, as in Alegre et al. (2021). Observations given by the environment are a vector of seventeen real numbers, giving information about the location, rotation, and velocity of different sections of the simulated robot. The first task is the default "normal" Half-Cheetah environment. The other three tasks are "joint-malfunction", "wind", and "velocity" tasks respectively. These task changes affect the physics of the environment, i.e., the transition function, but not the rewarding conditions or observations made by the agent. SWOKS, MBCD, and 3RL are evaluated in the Half-Cheetah environment. Further details are provided in the Appendix.

The Configurable Tree graph (CT-graph) environment is a configurable tree graph designed for testing LRL algorithms due to its feature of automatically creating multiple tasks with measurable levels of complexity, sparsity of reward, and similarities among tasks. In the setup for this study, we used four tasks with the same observations and transition function, but different reward functions. This environment is used to evaluate SWOKS and the TFCL algorithm (Aljundi et al., 2019).

The Minigrid environment is a lightweight grid-world environment designed to test the generalizability and adaptability of LRL algorithms due to its scalability to generate various permutations of tasks, enabling varying complexities and similarities. The environment employs sparse rewards. In this study, we used three seed variations of the SimpleCrossingS9N2-v0 environment, maintaining certain similarities between tasks. This environment is used to evaluate SWOKS and the TFCL algorithm.

5.1 SWOKS ON INTERFERING TASKS AND OBSERVATION SPACE CHANGES

To test the resilience of SWOKS to task interference, we compare its performance to TFCL. Both algorithms are equipped with the same network structure, though TFCL does not make use of the modulating masks due to lacking a task change detection mechanism. TFCL uses a "prioritized keeping" replay buffer in order to retain knowledge on old tasks. When compared with SWOKS on the 4-task CT-graph environment, we observe that the TFCL is unable to maintain performance on all tasks despite the continual learning mechanism. This inability is caused by the same observation and transition functions, and different-reward properties of the four tasks in this benchmark creating interference between tasks. SWOKS, on the contrary, appears to learn all tasks. It is important to note that the modulating masks method requires task labels or in this case the automated SWOKS mechanism, to enable the algorithm to select the correct mask for each task.

The reward achieved by SWOKS and TFCL in the CT-graph is shown in Figure 2(a). The p-values and detected task labels are found in Figure 3.

SWOKS is designed to detect tasks in difficult cases when changes occur in the reward or transition functions. However, changes in the observation space may also be detected. In the following simulations, SWOKS is tested on three tasks in the Minigrid environment specifically to assess the ability of SWOKS to deal with variations in the input distribution. Fig. 4(a) shows that tasks 1 and 2 are learned but task 3 fails to learn. Nevertheless, SWOKS identifies the third task as an additional one as shown in Fig. 4(b).

5.2 SWOKS ON CONTINUOUS ACTION SPACES

SWOKS is compared to Alegre et al. (2021)'s MBCD algorithm and Caccia et al. (2023)'s 3RL algorithm. This comparison is made in the Half-Cheetah environment with the same task setup as used in Alegre et al. (2021). MBCD is used without simulated replay to work under the same assumptions on input data as SWOKS and 3RL. The average reward over 5 seeds for each algorithm is plotted in Figure 2(b). While 3RL reaches a plateau with a policy that works fairly well for all tasks, MBCD and SWOKS each apply new network masks when new tasks are detected, allowing for more robustness against interfering tasks. MBCD is known to assign the same task label for different tasks that may be similar. This allows for knowledge transfer between similar tasks, but in these experiments resulted in lower performance.

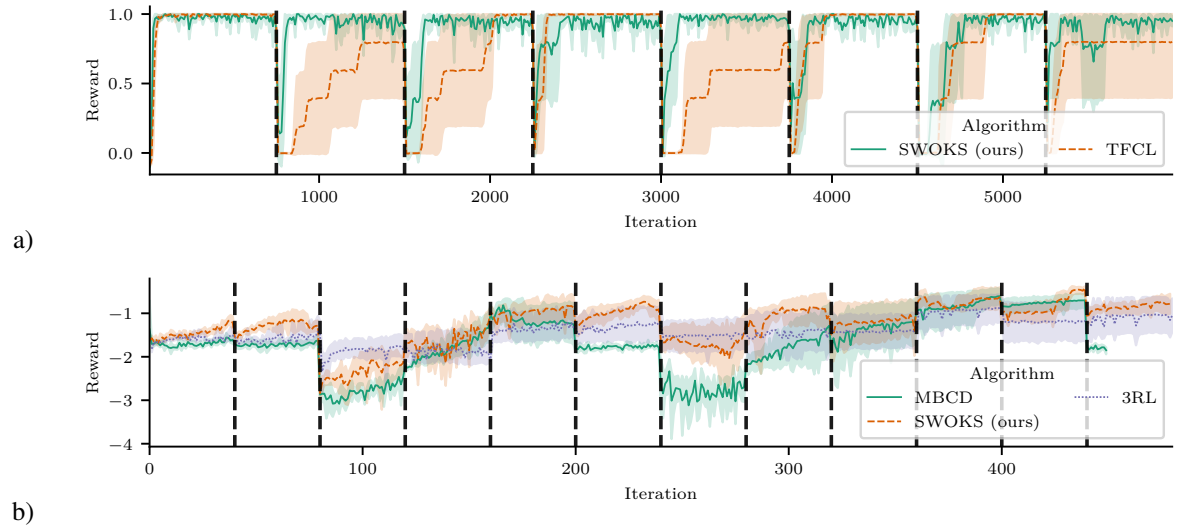

Figure 2: Reward achieved over sequences of tasks. a) Average reward over 5 seeds for SWOKS and TFCL in the CT-graph environment. The sequence of tasks from 1 to 4 is seen twice in the order 1-2-3-4-1-2-3-4. The rolling average over 10 iterations is plotted for easier viewing. b) SWOKS, MBCD and 3RL are tested on the Half-Cheetah environment. Task changes occur every 40000 timesteps (40 iterations).

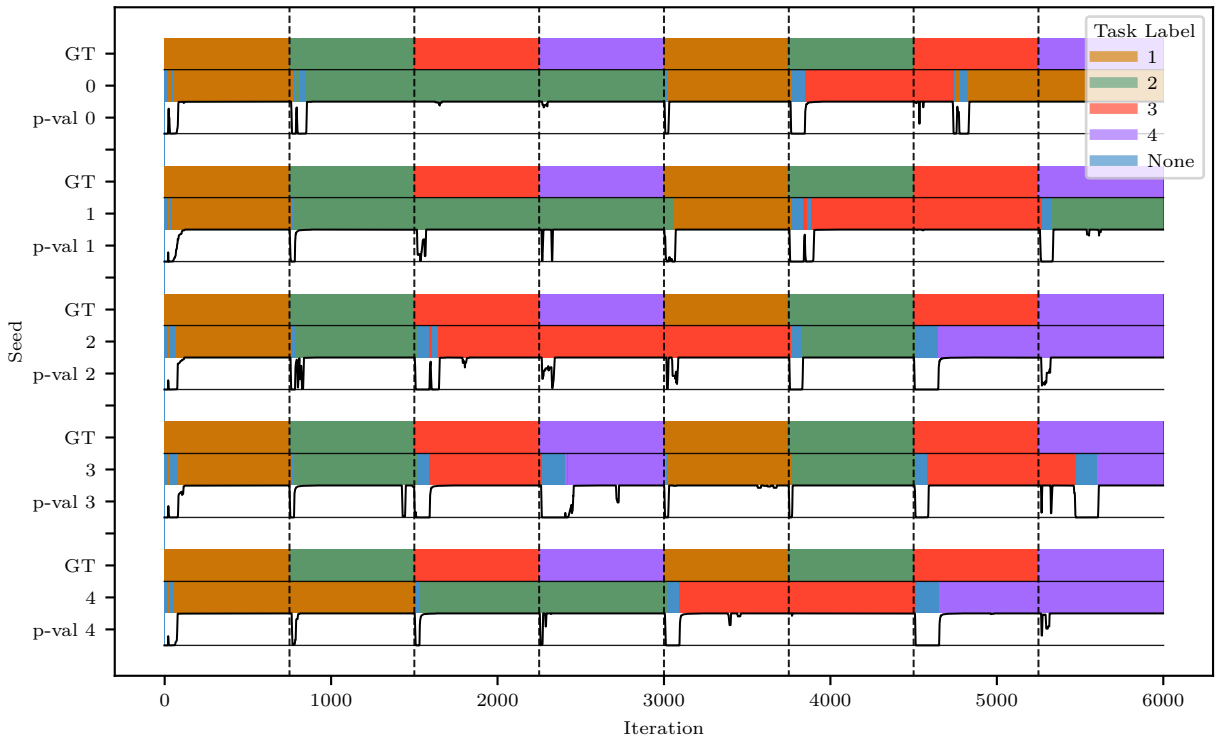

Figure 3: Detected task over time for 5 seeds of the SWOKS agent performing in the CT-graph environment. The ground truth (GT) task is compared with the task label generated by each agent. Task labels given in this graph are from the same run as Figure 2. The label "None" corresponds to the agent having a low p-value for all previously seen tasks.

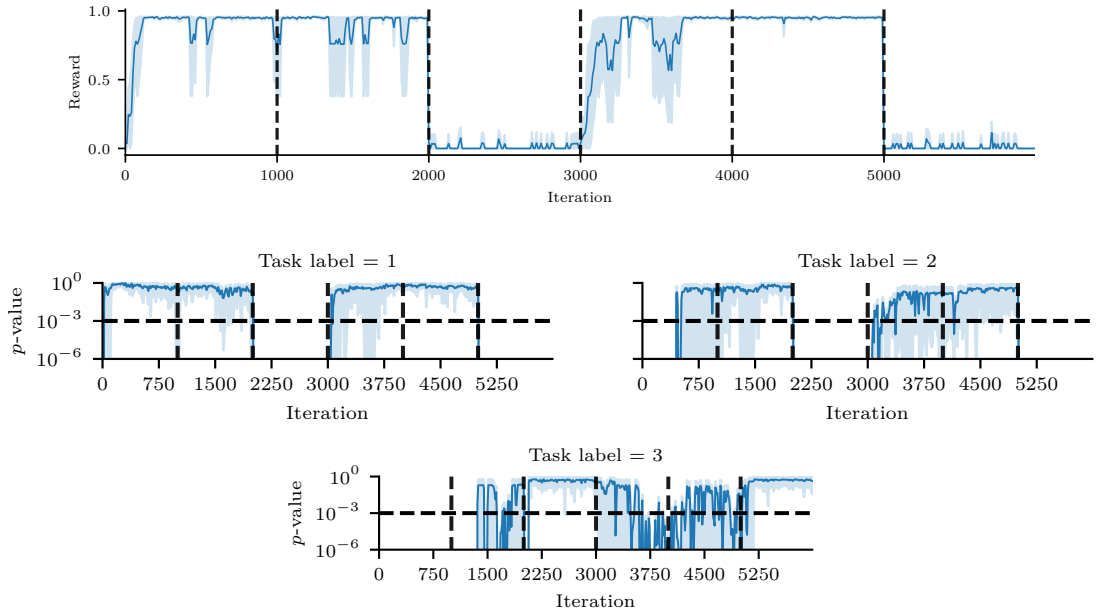

Figure 4: Analysis of learning dynamics on the Minigrid environment for two seed runs. Task changes occur every 1000 iterations, between 3 tasks (details in appendix). The three tasks are each seen twice in the order 1-2-3-1-2-3. (a) Average reward: the first two tasks are learned, but the third one fails. (b) The p-values are high (above $0.001 = 10^{-3}$) when a task is detected. We observe that task 2 is mistakenly identified as task 1 (top left graph) but without affecting the performance. Task 3 is also correctly identified despite the agent failing to learn it.

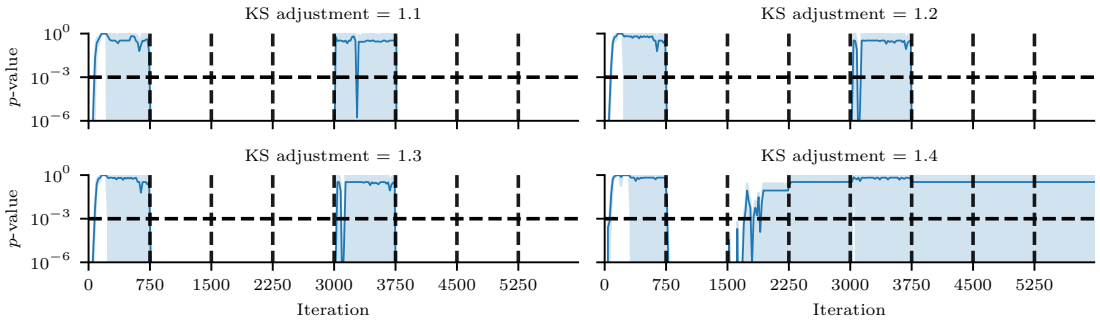

Figure 5: Plot over time of p-values for task label 1 and for different values of β in the CT-graph benchmark. The curriculum is formed of tasks 1-2-3-4-1-2-3-4. We can observe that task 1 is correctly identified for all settings, but with $\beta = 1.4$, SWOKS also believes tasks 2, 3, and 4 to be tasks 1. A rolling mean is taken over a sliding window of size 20 for readability.

5.3 Analysis of β and p-values

The KS p-values generated by SWOKS are essential to its task detection capabilities. The plots in Figure 4 show that the p-values are high for the matching task, and lower for the other tasks, when the task changes are correctly identified. The KS adjustment, β, is a parameter introduced to reduce the number of false positives detected by SWOKS. An analysis of different values of β on the task classification of SWOKS is seen in Figure 5.

6 DISCUSSION

The proposed SWOKS algorithm introduces hyper-parameters that address fundamental assumptions on the dynamics of context detection and task change. Namely, how frequently tasks change over time, how many SAR data points are required to draw statistically significant conclusions, and what types of variations can be observed in the online data stream. Such aspects are highly domain-dependent, and it makes sense that tunable parameters should be used. SWOKS can be tuned with different L_D and L_W parameters and KS adjustment β.

Environments with many different states require larger L_D and L_W because different tasks can be differentiated only after sampling many observations over a prolonged period of time. L_W determines the size of SWD samples, which are highly compressed representations of differences between sets of data. Thus, increasing L_W may provide a computationally efficient way to increase the statistical power of the test provided that L_D is sufficient to capture a minimum set of observations.

The KS adjustment β parameter is an important feature of the algorithm to tune the type I error rate. As β depends on the distribution of the data within SWOKS, using an adaptive β, dependent on the standard deviation of the data, could be investigated in future research in order to reduce the number of hyper-parameters of SWOKS.

Future work could extend SWOKS with a hyper-parameter analysis and its sensitivity to variation of the adjustments. Although such hyper-parameter tuning is a limitation of the approach, we note that, by means of such a tuning, SWOKS can be set to target a specific accuracy in the detection, providing a statistically motivated decision-making mechanism that can be set to address domain and problem-specific targets.

6.1 SCALABILITY TO MANY TASKS

One apparent limitation in same-observation-different-reward tasks is the requirement for policies to be tested in sequence to determine whether the new task is already known. It is important to note that this is not a limitation of SWOKS, but an intrinsic property of the problem. Assume a 4-task problem in which an agent in a square arena (e.g. Minigrid) needs to reach four different corners. After learning all four tasks, if the task changes again to one unknown corner, the only way to discover where the reward is located is to try all policies, i.e., explore all corners. However, if the distribution of observations helps in detecting tasks (i.e. via cues that lead to different distributions), SWOKS could be augmented with a clustering solution (e.g., like in Brunskill & Li (2013) but still using the SWD (Liu et al. (2022) for high dimensional observations), to narrow the search to only specific policies. Such an approach could find the set of nearest policies prior to executing them.

7 CONCLUSION

A novel algorithm, SWOKS, is introduced to detect tasks in deep RL where subtle changes involve transition functions or reward distributions. SWOKS exploits SWD to extract compressed representations of task changes and performs KS tests to provide task change detection and re-detection based on statistically meaningful decisions. The simulations suggest that the approach is a promising tool that can be used in combination with a policy gradient method (PPO) to optimize an overall measure of performance for an RL agent learning a sequence of tasks. Further tests are required to assess the flexibility of the method on a larger variety of benchmarks, including, e.g., continuous action spaces. While this study provides only a proof-of-concept, it also encourages further studies on the key idea of targeting desired error rates in task detection by means of statistical tools.

REFERENCES

Alegre, L. N., Bazzan, A. L., and da Silva, B. C. Minimum-delay adaptation in non-stationary reinforcement learning via online high-confidence change-point detection. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 97–105, 2021.

Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., and Tuytelaars, T. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 139–154, 2018.

Aljundi, R., Kelchtermans, K., and Tuytelaars, T. Task-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11254–11263, 2019.

Aljundi, R., Reino, D. O., Chumerin, N., and Turner, R. E. Continual novelty detection. In *Conference on Lifelong Learning Agents*, pp. 1004–1025. PMLR, 2022.

Anderson, T. W. On the distribution of the two-sample cramer-von mises criterion. *The Annals of Mathematical Statistics*, pp. 1148–1159, 1962.

Anderson, T. W. and Darling, D. A. Asymptotic Theory of Certain "Goodness of Fit" Criteria Based on Stochastic Processes. *The Annals of Mathematical Statistics*, 23(2):193 – 212, 1952. doi: 10.1214/aoms/1177729437. URL https://doi.org/10.1214/aoms/1177729437.

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein generative adversarial networks. In *International conference on machine learning*, pp. 214–223. PMLR, 2017.

Arulkumaran, K., Deisenroth, M. P., Brundage, M., and Bharath, A. A. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6):26–38, 2017.

Ben-Iwhiwhu, E., Nath, S., Pilly, P. K., Kolouri, S., and Soltoggio, A. Lifelong reinforcement learning with modulating masks. *Transactions on Machine Learning Research*, 2023.

Bengio, Y., Léonard, N., and Courville, A. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Benjamini, Y. and Hochberg, Y. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)*, 57(1):289–300, 1995.

Bonneel, N., Rabin, J., Peyré, G., and Pfister, H. Sliced and radon wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision*, 51:22–45, 2015.

Brunskill, E. and Li, L. Sample complexity of multi-task reinforcement learning. *arXiv preprint arXiv:1309.6821*, 2013.

Caccia, M., Mueller, J., Kim, T., Charlin, L., and Fakoor, R. Task-agnostic continual reinforcement learning: Gaining insights and overcoming challenges. In *Conference on Lifelong Learning Agents*, pp. 89–119. PMLR, 2023.

Chevalier-Boisvert, M., Willems, L., and Pal, S. Minimalistic gridworld environment for openai gym (2018). *URL https://github. com/maximecb/gym-minigrid*, 6, 2021.

Da Silva, B. C., Basso, E. W., Bazzan, A. L., and Engel, P. M. Dealing with non-stationary environments using context detection. In *Proceedings of the 23rd international conference on Machine learning*, pp. 217–224, 2006a.

Da Silva, B. C., Basso, E. W., Perotto, F. S., C. Bazzan, A. L., and Engel, P. M. Improving reinforcement learning with context detection. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pp. 810–812, 2006b.

de Faria, E. R., de Leon Ferreira, A. C. P., Gama, J., et al. Minas: multiclass learning algorithm for novelty detection in data streams. *Data mining and knowledge discovery*, 30(3):640–680, 2016.

Fernando, C., Banarse, D., Blundell, C., Zwols, Y., Ha, D., Rusu, A. A., Pritzel, A., and Wierstra, D. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*, 2017.

Flamary, R., Courty, N., Gramfort, A., Alaya, M. Z., Boisbunon, A., Chambon, S., Chapel, L., Corenflos, A., Fatras, K., Fournier, N., Gautheron, L., Gayraud, N. T., Janati, H., Rakotomamonjy, A., Redko, I., Rolet, A., Schutz, A., Seguy, V., Sutherland, D. J., Tavenard, R., Tong, A., and Vayer, T. Pot: Python optimal transport. *Journal of Machine Learning Research*, 22(78):1–8, 2021. URL http://jmlr.org/papers/v22/20-451.html.

Geisa, A., Mehta, R., Helm, H. S., Dey, J., Eaton, E., Dick, J., Priebe, C. E., and Vogelstein, J. T. Towards a theory of out-of-distribution learning. *arXiv preprint arXiv:2109.14501*, 2021.

Haroush, M., Frostig, T., Heller, R., and Soudry, D. A statistical framework for efficient out of distribution detection in deep neural networks. *arXiv preprint arXiv:2102.12967*, 2021.

Hayes, T. L., Kafle, K., Shrestha, R., Acharya, M., and Kanan, C. Remind your neural network to prevent catastrophic forgetting. In *European Conference on Computer Vision*, pp. 466–483. Springer, 2020.

Kessler, S., Parker-Holder, J., Ball, P., Zohren, S., and Roberts, S. J. Same state, different task: Continual reinforcement learning without interference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 7143–7151, 2022.

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

Kim, G., Esmaeilpour, S., Xiao, C., and Liu, B. Continual learning based on ood detection and task masking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3856–3866, 2022.

Kim, S., Noci, L., Orvieto, A., and Hofmann, T. Achieving a better stability-plasticity trade-off via auxiliary networks in continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11930–11939, 2023.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Kolouri, S., Ketz, N. A., Soltoggio, A., and Pilly, P. K. Sliced Cramer synaptic consolidation for preserving deeply learned representations. In *International Conference on Learning Representations*, 2019a.

Kolouri, S., Nadjahi, K., Simsekli, U., Badeau, R., and Rohde, G. Generalized sliced Wasserstein distances. *Advances in neural information processing systems*, 32, 2019b.

Kudithipudi, D., Aguilar-Simon, M., Babb, J., Bazhenov, M., Blackiston, D., Bongard, J., Brna, A. P., Chakravarthi Raja, S., Cheney, N., Clune, J., Daram, A., Fusi, S., Helfer, P., Kay, L., Ketz, N., Kira, Z., Kolouri, S., Krichmar, J. L., Kriegman, S., Levin, M., Madireddy, S., Manicka, S., Marjaninejad, A., McNaughton, B., Miikkulainen, R., Navratilova, Z., Pandit, T., Parker, A., Pilly, P. K., Risi, S., Sejnowski, T. J., Soltoggio, A., Soures, N., Tolias, A. S., Urbina-Meléndez, D., Valero-Cuevas, F. J., Van de Ven, G. M., Vogelstein, J. T., Wang, F., Weiss, R., Yanguas-Gil, A., Zou, X., and Siegelmann, H. Biological underpinnings for lifelong learning machines. *Nature Machine Intelligence*, 4(3):196–210, 2022.

Lange, M. D., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., and Tuytelaars, T. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44:3366–3385, 7 2022. ISSN 19393539. doi: 10.1109/TPAMI.2021.3057446.

Lee, K., Lee, K., Lee, H., and Shin, J. A simple unified framework for detecting out-of-distribution samples and adversarial attacks, 2018.

Li, Z. and Hoiem, D. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

Liu, X., Bai, Y., Lu, Y., Soltoggio, A., and Kolouri, S. Wasserstein task embedding for measuring task similarities. *arXiv preprint arXiv:2208.11726*, 2022.

Masse, N. Y., Grant, G. D., and Freedman, D. J. Alleviating catastrophic forgetting using context-dependent gating and synaptic stabilization. *Proceedings of the National Academy of Sciences*, 115(44):E10467–E10475, 2018.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015a. doi: 10.1038/nature14236. URL https://doi.org/10.1038/nature14236.

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, 2015b.

Nath, S., Peridis, C., Ben-Iwhiwhu, E., Liu, X., Dora, S., Liu, C., Kolouri, S., and Soltoggio, A. Sharing lifelong reinforcement learning knowledge via modulating masks. In *Second Conference on Lifelong Learning Agents (CoLLAs) 2023*, 2023.

Papadopoulos, H. Inductive conformal prediction: Theory and application to neural networks. In *Tools in artificial intelligence*. Citeseer, 2008.

Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., and Wermter, S. Continual lifelong learning with neural networks: A review. *Neural networks*, 113:54–71, 2019.

Rabin, J., Peyré, G., Delon, J., and Bernot, M. Wasserstein barycenter and its application to texture mixing. In *Scale Space and Variational Methods in Computer Vision: Third International Conference, SSVM 2011, Ein-Gedi, Israel, May 29–June 2, 2011, Revised Selected Papers 3*, pp. 435–446. Springer, 2012.

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

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., and Hadsell, R. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015.

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

Sedlmeier, A., Gabor, T., Phan, T., Belzner, L., and Linnhoff-Popien, C. Uncertainty-based out-of-distribution detection in deep reinforcement learning. *arXiv preprint arXiv:1901.02219*, 2019.

Serra, J., Suris, D., Miron, M., and Karatzoglou, A. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, pp. 4548–4557. PMLR, 2018.

Shin, H., Lee, J. K., Kim, J., and Kim, J. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30, 2017.

Silver, D. L., Yang, Q., and Li, L. Lifelong machine learning systems: Beyond learning algorithms. In *2013 AAAI spring symposium series*, 2013.

Soltoggio, A., Ben-Iwhiwhu, E., Peridis, C., Ladosz, P., Dick, J., Pilly, P. K., and Kolouri, S. The configurable tree graph (ct-graph): measurable problems in partially observable and distal reward environments for lifelong reinforcement learning, 2023.

Soltoggio, A., Ben-Iwhiwhu, E., Braverman, V., Eaton, E., Epstein, B., Ge, Y., Halperin, L., How, J., Itti, L., Jacobs, M. A., et al. A collective ai via lifelong learning and sharing at the edge. *Nature Machine Intelligence*, 6(3): 251–264, 2024.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

Thrun, S. Lifelong learning algorithms. In *Learning to learn*, pp. 181–209. Springer, 1998.

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 5026–5033. IEEE, 2012.

van de Ven, G. M., Tuytelaars, T., and Tolias, A. S. Three types of incremental learning. *Nature Machine Intelligence*, 4(12):1185–1197, 2022.

Weisstein, E. W. Bonferroni correction. *MathWorld–A Wolfram Web Resource.*, 2004. URL https://mathworld.wolfram.com/.

Wortsman, M., Ramanujan, V., Liu, R., Kembhavi, A., Rastegari, M., Yosinski, J., and Farhadi, A. Supermasks in superposition. *Advances in Neural Information Processing Systems*, 33:15173–15184, 2020.

Ye, F. and Bors, A. G. Task-free continual learning via online discrepancy distance learning. *Advances in Neural Information Processing Systems*, 35:23675–23688, 2022.

Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. How transferable are features in deep neural networks? *CoRR*, abs/1411.1792, 2014. URL http://arxiv.org/abs/1411.1792.

Zenke, F., Poole, B., and Ganguli, S. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pp. 3987–3995. PMLR, 2017.

A Implementation details

A.1 SWOKS setup and hyper-parameters

Table 1 provides a breakdown of the specific SWOKS hyper-parameters used in each of the experiments provided in this paper. We provide a brief explanation of each of these hyper-parameters and their effects on the operation of SWOKS.

- **History length, L_D:** Determines the length of the data taken from the beginning and end of the Dataset, D_z. A higher value provides more statistical power (i.e., the p-value is more likely to reach 0) and better stability, at the cost of slower detection speed.
- **Wasserstein history length, L_W:** Similar to the history length, this determines the length of the data taken from the beginning and end of the Sliced Wasserstein history, W_z. A higher value also increases the statistical power (with more effectiveness), at the cost of slower detection speed.
- **Significance threshold α:** The value of α determines the base sensitivity of the p-value. A higher value triggers a task change at a higher p-value with a very minor effect. Very low values can cause SWOKS to be unable to detect task changes if the statistical power of the observations provided are not high enough to reach such small values.
- **KS adjustment β:** An adjustment value is used to stabilize the sensitivity of the task change detection. A higher value improves robustness to small oscillations in the p-value.
- **Stable phase duration:** A duration of n-steps to determine how long to wait before detecting task changes. This is necessary due to p-values dropping early on in the task-learning process as changing policies leads to exploration.
- **Model backup frequency:** Two consecutive backups of the model history are saved at specified intervals to ensure that if a task change detection is late, then the model of the previous task is not affected by the new task.

A.2 Network setup and hyper-parameters

SWOKS and TFCL are used with PPO for the CT-graph and Minigrid experiments. Table 2 shows the hyper-parameters used for the PPO network for each of these experiments. Table 3 describes the network structure for these experiments in Minigrid, which is the same structure used in Kessler et al. (2022).

A.2.1 Modulating masks

Continuous masks are used as in Ben-Iwhiwhu et al. (2023), and are not discretized at any point. Upon new task detection, for the CT-graph and Minigrid environments, the new masks are initialized randomly. For the Half-Cheetah environment, each newly created mask is created as a duplicate of the previously seen mask, allowing for better forward transfer of task knowledge. Combinations of modulating masks may provide more optimized mask initiation policies, however, this is not applied to the experiments in this paper.

Hyper-Parameter	Value
History length, L_D	240
SWD history length, L_W	125
Significance threshold, α	0.001
KS adjustment, β	1.1
Stable phase duration	50000 (100000 for MiniGrid)
Model backup freq. (iterations)	50

Table 1: Lists the SWOKS-specific hyper-parameters used in the experiments conducted in this paper.

Layer	Channel	Kernel	Stride	Padding
Input	3	-	-	-
Conv1	16	(2x2)	1	0
ReLU	-	-	-	-
Max Pool 2-d	16	(2x2)	2	0
Conv2	32	(2x2)	1	0
ReLU	-	-	-	-
Conv3	64	(2x2)	1	0
ReLU	-	-	-	-
Flatten	-	-	-	-
Linear	200	-	-	-
ReLU	-	-	-	-

Table 3: Network model in PPO for Minigrid experiments.

Hyper-Parameter	CT-graph	Minigrid
Learning rate	0.007	0.007
CL preservation	supermasks	supermasks
Number of workers	4	4
Optimizer function	RMSprop	RMSprop
Discount rate	0.99	0.99
Entropy weight	0.01	0.01
Rollout length	128	128
Optimization epochs	5	5
Number of mini-batches	64	64
PPO ratio clip	0.1	0.1
Iteration log interval	1	1
Gradient clip	5	5
Max iterations (per task)	750	1000

Table 2: PPO agent hyper-parameters used for CT-graph and Minigrid experiments across SWOKS and TFCL.

SWOKS, MBCD and 3RL are employed with an SAC network, with parameters as follows. MBCD is used without simulated replay to work under the same assumptions on input data as SWOKS and 3RL. The hyperparameters used for the SAC network is given in Table 4.

Hyper-Parameter	Half-Cheetah
Learning rate	0.00015
CL preservation	masks
Number of workers	4
Optimizer function	RMSprop
Discount rate	0.99
Entropy weight	0.00015
Rollout length	128
Optimization epochs	8
Number of mini-batches	64
PPO ratio clip	0.1
Iteration log interval	1
Gradient clip	5
Max steps (per task)	12800
Evaluation episodes	25
Require task label	True
Backbone network seed	9157

Table 4: SAC hyper-parameters used for Half-Cheetah experiments across SWOKS, MBCD, and 3RL.

Hyper-Parameter	Minigrid SC (3 tasks)
Tasks	MiniGrid-SimpleCrossingS9N2-v0
Seeds	129, 112, 111

Table 6: Minigrid environment parameters. We use Minigrid to compare SWOKS to TFCL.

A.3 CT-GRAPH ENVIRONMENT HYPER-PARAMETERS

Hyper-Parameter	Depth 2 CT-graph (4 tasks)
General seed	1
Tree depth	2
Branching factor	2
Wait probability	0.0
High reward value	1.0
Fail reward value	-0.1
Stochastic sampling	false
Reward standard deviation	0
Min static reward episodes	0
Max static reward episodes	0
Reward distribution	needle in haystack
MDP decision states	true
MDP wait states	true
Wait states	$[2, 8]$
Decision states	$[9, 11]$
Graph ends	$[12, 15]$
Image dataset seed	1
1D format	false
Number of images	16
Noise on images on read	0
Small rotation on read	1

Table 5: CT-graph environment parameters. We use CT-graph to compare SWOKS to TFCL.

A.4 MINIGRID ENVIRONMENT HYPER-PARAMETERS

Table 6 contains the environment hyper-parameters used for experiments in the Minigrid environment. Three tasks are used within the environment, with different walls placed in different locations for different tasks (Figure 6). As the environment is partially observable, some states will look the same between tasks, making different tasks appear very similar. Seeds used are the same as in Kessler et al. (2022).

A.5 HALF-CHEETAH ENVIRONMENT HYPERPARAMETERS

Tasks experienced by Half-Cheetah are: Normal, Joint-Malfunction, Velocity, and Wind. These tasks are dramatically visualized in Figure 6, and summarized in Table 7.

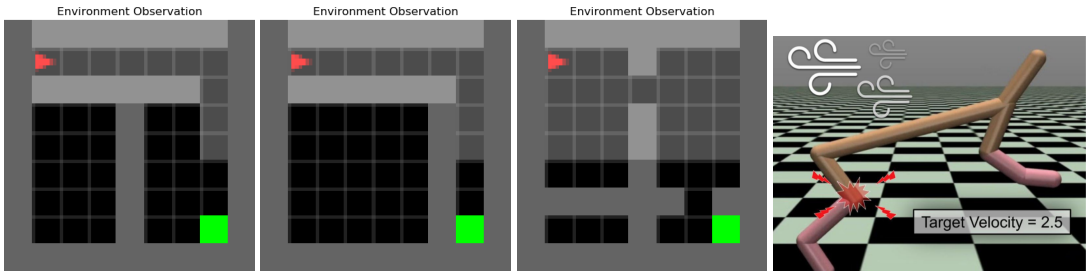

Figure 6: Left: Illustrations of the Minigrid SimpleCrossingS9N2-v0 task variations used in experiments. Right: Dramatized illustration of task changes in the Half-Cheetah experiments (Half-Cheetah figure from Alegre et al. (2021)).

Hyper-Parameter	Half-Cheetah (4-tasks)
Tasks	normal, joint-malfunction, wind, velocity
Change freq.	40000
Default target vel.	1.5
Velocity target vel.	2.0
Joint malfunction mask	[-1, -1, 0, 0, 0, 0]

Table 7: Half-cheetah environment parameters. We use Half-Cheetah to compare SWOKS to MBCD and 3RL.

A.6 Implementation Libraries

Implementation libraries are different between experiments due to the differing underlying networks implemented. Half-Cheetah experiments utilize TensorFlow, while the CT-graph and Minigrid experiments use pytorch. Package libraries for the different experiments conducted are in Table 8.

Package	CT-graph & Minigrid	Half-Cheetah
python	3.7.0	3.7.16
pytorch	1.12.1	n/a
gym	0.26.2	0.19.0
gymnasium	0.28.1	0.28.1
minigrid	2.3.0	n/a
gym-ctgraph	1.0	n/a
cuda toolkit	11.6.0	10.0.130
cudnn	n/a	7.6.5
cython	0.29.36	0.29.36
pot	0.9.1	0.9.1
tensorflow	n/a	1.15.0
mujoco-py	n/a	2.1.2.14

Table 8: Packages used in the SWOKS implementation with Minigrid and CT-graph. The system is based on a PyTorch RL implementation using CUDA and Gym as the environment interface. Calculations of the sliced Wasserstein distances are performed using the Python Optimal Transport library (Flamary et al., 2021). We use Gymnasium for CT-graph and Minigrid and Gym for Half-Cheetah.