# Reconciling Meta learning and regularization in Lifelong Reinforcement Learning with Non-stationary Environments

**Fan Feng** [*1] **Qi Liu** [*1] **Bo Xiao** [*2]

## Abstract

Humans are capable of accumulating new knowledge without retaining learned information. Whereas artificial agents will suffer from *catastrophic forgetting* and concept drift under *lifelong* settings. The lifelong Reinforcement Learning (lifelong-RL) poses two major challenges for agents: 1) forgetting previous learnt policy; 2) hard to achieve transfer in future tasks with new environments. In this work, we show that by reconciling meta representation and regularization across tasks, the agents can realize lifelong learning with *minimal forgetting* and *maximal transfer*. Experimental results show that the proposed method can help agent achieve fast adaptation on new tasks and better remember previous knowledge in old tasks simultaneously in non-stationary environments. Code is released through *https://github.com/ffeng1996/metaContinualRL*.

## 1. Introduction

Our brains can learn new and retain previous knowledge at the same time from complex external environments with interaction with complex external environments. The capability is called *lifelong learning*, which enables human to fast adapt new knowledge without forgetting previous learned at the same time (Wixted, 2004). Recent breakthroughs in machine learning, especially deep learning field, proves that agents can also learn the external information and make prediction or action based on learning experiences in many fields, such as robotics, computer vision, online advertisement, gaming, intelligent financial engineering, etc (LeCun et al., 2015). However, most of these applications are set in stationary scenarios, which is not consistent with real-life deployment. The two major obstacles of designing lifelong learning algorithms are catastrophic forgetting and concept drift (French, 1999; McCloskey & Cohen, 1989).

[*]Equal contribution. Author list is in alphabetical order. [1]Department of Electrical Engineering, City University of Hong Kong [2]Department of Mathematics, City University of Hong Kong.

Course Project Report of SDSC8006 Reinforcement Learning.

Reinforcement learning (Sutton & Barto, 2011) is a machine learning paradigm that poses these challenges, where the changes to the data distribution can occur unpredictably during the training of one single or multiple tasks. Lifelong Reinforcement Learning is an online learning problem where agents face sequences of RL tasks with time-varying distributions.

In this paper, we explore a method for self-maintaining Deep Q-learning systems that can learn continually, as new task arrives.

## 2. Background

### 2.1. Lifelong Learning

Lifelong learning (a.k.a. continual/incremental/online learning) in machine learning context has been intensively studied in the past few years (Learning; Silver et al., 2013; Shalev-Shwartz et al., 2012). The general lifelong learning setting is that in a data steaming learning process, the agent can receive data $(x_t, y_t)$ at time step $t$. Note that data from each time step is non-stationary so there exists domain and distribution gaps among these data streams (Parisi et al., 2019). The objective of this problem is to learn an adaptive function $f$ with parameter set to minimize the designed loss function $L$, which contain loss functions of current, future and previous data streaming tasks (Parisi et al., 2017).

The two major obstacles of designing lifelong learning algorithms are catastrophic forgetting and concept drift (McCloskey & Cohen, 1989; French, 1999; Forman, 2006). Catastrophic forgetting means that agents who learn in lifelong context may forget the previous knowledge when absorbing new knowledge (Kemker et al., 2018). Concept drift occurs when different data streams have different data distributions. Traditional machine learning cannot address these two obstacles since the learning process is static in each time step and retrain the whole system when encountered with new data streams. The approaches to address these two problems will be discussed in later sections. Roughly, the approaches aiming to overcome catastrophic forgetting are more focused on optimizing backward tasks' losses, and approaches aiming to address concept drift is more related to maximizing the forward transfer, which is similar to the

objectives of transfer learning (Pan & Yang, 2009). The basic goals of lifelong learning are illustrated in Figure 1.


Figure 1: Lifelong Learning goals.

Lifelong learning also has a close relationship between other machine learning paradigms such as multi-task learning, transfer learning, and meta learning (Pan & Yang, 2009; Zhou et al., 2012; Schmidhuber, 1995; Finn et al., 2017). Here we give brief comparisons between lifelong learning and these machine learning paradigms:

- *Transfer learning:* transfer learning work mainly focuses on using current knowledge to adapt data from another task or domain. Thus transfer learning can somewhat address the problem of concept drift which is common in lifelong learning settings. However, lifelong learning also aims at overcoming catastrophic forgetting, which can be seen as backward transfer knowledge. Transfer learning does not involve this aspect.

- *Multi-task learning:* multi-task learning focuses the learning from different data distributions at the same time using sharing parameters. The memory size of multi-task learning networks grows with the size of training data. As opposed to lifelong learning, multi-task learning does not involve overcoming catastrophic forgetting since it retrains each new data distribution all the time.

- *Meta learning:* meta learning aims to learn the meta parameters (including network structure, hyper-parameters, etc.) from different data streams with variations in data distribution. The training data's streaming structure used in meta learning is similar to lifelong learning. Meta learning aims to learn a better set of representation for fast adaptation, which is also important in improving future transfer in lifelong learning setting. However, meta learning is trained as offline paradigm rather than online. Moreover, the objective of meta learning and lifelong learning is also different.

### 2.2. Reinforcement Learning

Reinforcement learning agent aims to perform actions based on environment to maximize cumulative reward. Reinforcement learning is learning to map from situations to actions so as to obtain maximal reward signal. Learning is the process during which the agent tries possible actions based on observations until gaining the maximal cumulative reward. There are many different settings in reinforcement learning for different problems, such as model-free reinforcement learning, model-based reinforcement learning, lifelong reinforcement learning and so on. To tackle these optimization problems and obtain optimal policy, researchers design various algorithms, among which Q-learning, deep Q-network and experience replay are most commonly used.

#### 2.2.1. Q-LEARNING

Q-learning (Watkins, 1989) is an off-line temporal difference (TD) control algorithm. Temporal different (TD) learning is the central and significant idea in reinforcement learning field, which incorporates Monte Carlo method and dynamic programming method. It can learn both from history experience and from current estimated value. Before introducing the procedures of Q-learning algorithm, we need set some notations in reinforcement learning.

We denote the state space as $\mathcal{S}$ and action space as $\mathcal{A}$. The agent observes state $s_t \in \mathcal{S}$ and takes action $a_t \in \mathcal{A}$ at time $t$, then obtains reward $r_t \in \mathbb{R}$. The cumulative reward $G_t$ in the future from time $t$ is the sum of discounted reward with discounted rate $\gamma$ at each step. $Q_t : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the action value function defined by

$$Q(s_t, a_t) = \mathbb{E}\left[G_t | s_t, a_t\right] = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t, a_t\right].$$

In Q-learning algorithm, learning target is $Q$ function. Given $Q$, the agent can select optimal action based current state. The update of $Q$ during iteration is from TD error as follows.

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a) - Q(s_t, a_t)],$$

where $\alpha$ is learning rate.

Then, Q-learning algorithm works as follows.

- Initialize $Q(s, a)$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$. In particular, $Q$ value for terminal state is always zero.

- For each episode, initialize $s_0$ at the beginning. At each time step $t$, sample action $a_t$ based on current state $s_t$ via $\epsilon$-greedy policy, then observe $r_t$ and $s_{t+1}$.

- Update $Q$: $Q(s_t, a_t) = Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a) - Q(s_t, a_t)]$.

- Take $s_{t+1}$ as current state and do same iteration until it is terminal state, in which agent moves to next episode.

After applying $Q$-learning algorithm, we obtain an optimal deterministic policy $\pi$ from $Q(\cdot,\cdot)$ tabular:

$$\pi(a|s) = \arg\max_{a\in\mathcal{A}} Q(a,s).$$

#### 2.2.2. Deep Q Networks

Deep Q network (Mnih et al., 2013; 2015) is a combination between Q-learning and artificial neural network. In Q-learning method, we should maintain a Q-value tabular among all possible states and actions. However, when the number of possible states and actions are greatly large like continuous state space, the scale of tabular is unbearable and computational cost is too expensive. And in some non-stationary environment, there may appear some unanticipated states, whose Q-value cannot be found, such that we don't know the scale of Q-tabular at the beginning. Therefore, deep Q network turns to approximate Q-value via artificial neural network. Given parameterized Q-value function with parameter $\theta$, we can utilize neural network to learn it by optimizing cost function below:

$$L(\theta) = \mathbb{E}_{(s,a,r,s')\sim D}\left[\left(r+\gamma\max_{a'} Q(s,a';\theta) - Q(s,a;\theta)\right)^2\right],$$

where $D$ is data set, $s, a, r$ is state, action and reward respectively at current step, $s'$ is next state and $\gamma$ is discounted rate.

#### 2.2.3. Experience Replay

Experience replay (Lin, 1992) is a method proposed to learn in non-stationary environment. It is usually leveraged in reinforcement learning. Reinforcement learning algorithms like AHC-learning and Q-learning are inefficient in experiences since they utilize the current experience only once and throw it away. Experience replay (ER) construct a technique which can reuse history data. The agent remembers history experience and can repeatedly utilize memory if it meets the identical situation. After combining experience replay method with AHC- and Q-learning algorithm, authors find it is capable to speed up the propagation and make the algorithm converge quickly.

Experience replay can also generate uncorrelated data without relying on multiple workers, which benefits many deep RL algorithms (Lillicrap et al., 2015; Andrychowicz et al., 2017). Many works based on experience replay don't take the size of buffer into account. Combining deep Q network with experience replay method, some works like (Mnih et al., 2015) success to improve the data efficiency with a default buffer capacity $10^6$. (Zhang & Sutton, 2017) discover that the capacity of buffer is relatively important during learning process although the details of applying ER are usually hidden by the learning algorithm. Due to the sensitivity to the size of buffer, they introduce a hyperparameter corresponding to its capacity.

The main idea of experience replay is to interleave current training data point with memory $M$ to stabilize learning. Before dealing with the coming stream, experience replay takes each example in the buffer as current training with same probability. Following the procedure as (Zhang & Sutton, 2017), we show the ER algorithm in 1.

**Algorithm 1** Experience Replay (ER) with sampling from buffer

**procedure** Train(D,$\theta$,$\alpha$,k)
M $\leftarrow \{\}$
**for** $t = 1, \cdots, T$ **do**
  **for** $(x, y) \in D_t$ **do**
    // sample batch from buffer:
    $B \leftarrow (x, y, k, M)$
    // update parameters via stochastic gradient descent
    $\theta \leftarrow \text{SGD}(B, \theta, \alpha)$
    //reservoir sampling memory update:
    $M \leftarrow M \cup \{(x, y)\}$
  **end for**
**end for**
**return** $\theta, M$
**end procedure**

## 3. Problem Statement

During lifelong learning process, the distribution $\mathcal{D}$ is a potentially infinite collection of unknown distributions $\mathcal{D} = \{D_1, \cdots, D_N\}$. For the distribution $D_n$, $Tr_n$ and $Te_n$ denote the training and testing sets, and $T_n$ is the task of this distribution. A performance matrix $R \in \mathcal{R}^{N\times N}$ contains in each entry $R_{ij}$ the testing performance of $Te_j$ after training the model over the $Tr_i$, which is shown in Table 1.

| $R$ | $Te_1$ | $Te_2$ | $\cdots$ | $Te_N$ |
|---|---|---|---|---|
| $Tr_1$ | $R_{11}$ | $R_{12}$ | $\cdots$ | $R_{1N}$ |
| $Tr_2$ | $R_{21}$ | $R_{22}$ | $\cdots$ | $R_{2N}$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $Tr_N$ | $R_{N1}$ | $R_{N2}$ | $\cdots$ | $R_{NN}$ |

Table 1: Performance matrix $R$, where $Tr$ = training data, $Te$ = testing data, and $R_{ij}$ = performance of the model training on $Tr_i$ and testing on $Te_j$. The number of tasks is $N$.

Our objectives for lifelong learning from stream are as follows.

- ***Maximize future transfer (fast adaptation):*** We aim to study whether previous learned knowledge can transfer to future tasks (See elements in light crayon area of Table 1 ) with good performances (-e.g.,high accuracy, fast adaption, few-samples learning).

- ***Minimize backward forgetting (non-forgetting):*** We aim to minimize the catastrophic forgetting on previous tasks (See elements in grey area of Table 1) during online learning setting.

These 2 objectives can be seen as *transfer-interference trade-off*, which consider knowledge sharing in both backward and forward directions.

### 3.1. Manifold Illustration

For better illustration of our objectives, suppose we have 3 clutters of data points sampled from different distributions $D_1, D_2, D_3$. We consider the manifold of solutions $\phi$ which can give solutions for each distribution (See Figure 2, here we only illustrate 2-dimensional examples, where $\phi \in \mathbb{R}^2$. Each clutter is denoted by different color). Our objective is to learn parameter matrix $\phi$ (In Figure 2, $\phi$ is one vector) which can generalize between parallel manifolds and avoid interference from orthogonal manifolds. We also illustrate naive training (sequential training and fine-tuning without any backward or forward adaptation) and multi-task training (joint-training on all distributions in one time) in Figure 2.

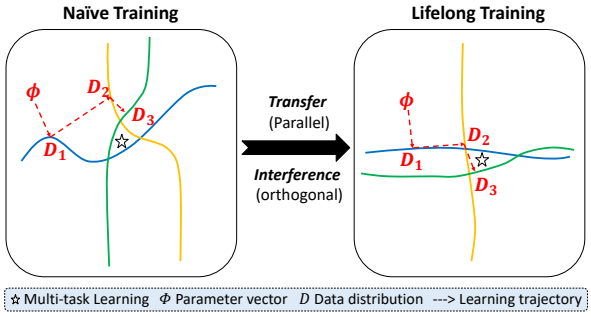



Figure 2: Manifold illustration of naive training, lifelong training and multi-task training. Each line denotes one distribution $D$. In naive training (left), the agent jumps between distant points on the manifolds and cannot generalize in all distributions.In lifelong training (right), with optimization on both transfer and interference, the agent can achieve both better transfer (parallel manifolds) and low interference (orthogonal manifolds).

## 4. Related Works

Lifelong reinforcement learning approaches can be divided into 1) Relearning the policy via regularizing the model parameters learned from previous environments, e.g., sublinear policy regret (Ammar et al., 2015), Elastic Weight Consolidation (EWC) (Kirkpatrick et al., 2017), Policy Distillation (Traoré et al., 2019), Policy Consolidation (Kaplanis et al., 2019), Learning without Forgetting (LwF) (Li & Hoiem, 2017), Complex Synapse (Kaplanis et al., 2018b) and Synaptic Intelligence (SI) (Zenke et al., 2017); 2) Dynamic architecture expansion, e.g., Progressive Neural Networks (PNN) (Rusu et al., 2016), Dynamic Expandable Network (DEN) (Yoon et al., 2017) and Context-dependent Gating (XdG) (Masse et al., 2018); 3) Replay mechanism (saving raw samples from old tasks to maintain knowledge about the past in the model and replay with samples drawn from the new task when training the model), e.g., Lifelong GAN (Zhai et al., 2019), Incremental Classifier and Representation Learning (ICaRL) (Rebuffi et al., 2017); and generative replay approaches train generative models on the data distribution (Wang et al., 2019), and they are able to afterward sample data from experience when learning new data, e.g., Deep Generative Replay (DGR) (Shin et al., 2017), DGR with dual memory (Kamra et al., 2017) and feedback (van de Ven & Tolias, 2018); 4) Meta Learning approaches to achieve fast adaptation on new tasks, e.g., meta RL (Clavera et al., 2019), online meta-learning (Finn et al., 2019) and Continual MAML (Caccia et al., 2020).

However, most prior works focus on either future transfer or backward interference and only a few works (Riemer et al., 2019; Kaplanis et al., 2018b) consider optimization on transfer-interference trade-off. Here we provide brief introduction on these two methods. Introductions on other methods can be found in Appendix A.

### 4.1. Meta Experience Replay

In real world, environment may change continually and anyone can predict the occurrence of special case, which leading to the need for learning to adapt a changing environment while maintaining history experiences. In the field of lifelong learning, some previous works tackle multi-tasks via sequence of experiences of each task from the same distribution, which means they are stationary. However, in non-stationary case, we meet many challenges in learning incrementally for every task after each experience. In the past, some papers utilize unexpected changes to formulate the dynamics of weight sharing based on current and past learning while they cannot find a consistent formulation. Meta-experience replay (MER) algorithm combines experience replay with optimization based meta-learning to obtain consistent optimal weight sharing dynamics, which can adapt environmental changes quickly while holding

history experiences.

Essentially, MER algorithm aims to look for the trade-off between transfer and interference. It leverages experience replay to tackle non-stationary stream and utilizes meta-learning to mitigate expensive computational cost during optimization. MER algorithm utilizes a buffer $M$ to hold memories. When a data point comes from data set $D_t$, it samples $s$ batches from buffer, of which each contains $k-1$ data. Then apply experience replay method onto each batch added with the current point and optimize loss function as well as update parameters $\theta$ via stochastic gradient descent (SGD) method. Finally, the buffer absorbs the current point into memory and algorithm run on next data point. We show the details of MER algorithm in 2.

**Algorithm 2** Meta-Experience Replay (MER)

**procedure** Train$(D, \theta, \alpha, \beta, \gamma, s, k)$
M $\leftarrow \{\}$
**for** $t = 1, \cdots, T$ **do**
  **for** $(x, y) \in D_t$ **do**
    // sample batches from buffer:
    $B_1, \cdots, B_s \leftarrow (x, y, s, k, M)$
    $\theta_0^A \leftarrow \theta$
    **for** $i = 1, \cdots, s$ **do**
      // experience replay for each batch:
      $\theta_{i,0}^W \leftarrow \theta$
      **for** $j = 1, \cdots, k$ **do**
        $x_c, y_c \leftarrow B_i[j]$
        $\theta_{i,j}^W \leftarrow \text{SGD}(x_c, y_c, \theta_{i,j-1}^W, \alpha)$
      **end for**
      //within batch meta-update:
      $\theta \leftarrow \theta_{i,0}^W + \beta(\theta_{i,k}^W - \theta_{i,0}^W)$
      $\theta_i^A \leftarrow \theta$
    **end for**
    //across batch meta-update:
    $\theta \leftarrow \theta_0^A + \gamma(\theta_s^A - \theta_0^A)$
    //reservoir sampling memory update:
    $M \leftarrow M \cup \{(x, y)\}$
  **end for**
**end for**
**return** $\theta, M$
**end procedure**

### 4.2. Complex Synapses

In lifelong reinforcement learning, one of the bottleneck is how to tackle catastrophic forgetting. Although artificial neural networks can maintain distributed memory, it cannot avoid catastrophic forgetting when meeting non-stationary environment. Our brain is able to learn during the whole life and in biological research, synapses are responsible for brain's ability to preserve history experience stably while adapting to new tasks quickly, which is called synaptic plasticity. With individual synapse participating in the storage of several memories, our brain can store history memories via those whose plasticity changes slowly and acquire new memories via fast changing ones. We try to imitate brain's capacity by equipping lifelong reinforcement learning agents with synaptic model named Benna-Fusi model (Benna & Fusi, 2016).

What we want to find is the weight of synapses at different time. Benna-Fusi model supposes that the weight of a synapse $w(t)$ at time $t$ is the summation of filtered history changes $\Delta w(t')$, where $t' < t$, which means:

$$w(t) = \sum_{t'<t} \Delta w(t') r(t - t'), \tag{1}$$

where $r(t)$ is kernel. Previous work proves that with the constraint of finite variance, the optimal case is reached when the kernel decays with a power law, i.e. $r(t) \sim t^{-\frac{1}{2}}$.

Since it's difficult to solve weight from Equation 1 directly, to implement this model under the power decaying kernel, they design a sequence of dynamic variables to approximate it. There are $N$ variables $u_1, u_2, \cdots, u_N$ in a chain, which means each variable only connects to two neighbors on its left and right respectively. The dynamics can be represented as follows.

$$C_k \frac{du_k}{dt} = g_{k-1,k}(u_{k-1} - u_k) + g_{k,k+1}(u_{k+1} - u_k),$$

where $k = 2, \cdots, N$.

And for $k = 1$, it's like:

$$C_1 \frac{du_1}{dt} = \frac{dw_{ext}}{dt} + g_{1,2}(u_2 - u_1),$$

where $\frac{dw_{ext}}{dt}$ estimate the change $\Delta w(t^{'})$ in 1. We set a dummy variable $u_{N+1} = 0$, which occurs in the dynamics of $u_N$. Therefore, we can find that the weight of synapses can be obtained, if the variable $u_1$ is known. All the other variables effecting $u_1$ act as history memories.

## 5. Proposed Model

Our proposed Meta Representation with Stable Weight Consolidation (MR-SWC) aims to minimize catastrophic forgetting and maximize future transfer at the same time (See Algorithm 3 and Figure 3).

### 5.1. Meta representation for maximizing transfer

Model Agnostic Meta Learning (MAML) is one of the most fundamental methods in meta learning proposed by (Finn et al., 2017). MAML aims to find generalized initialization among tasks. First order approximation is utilized to minimize computational burden in the optimization process.

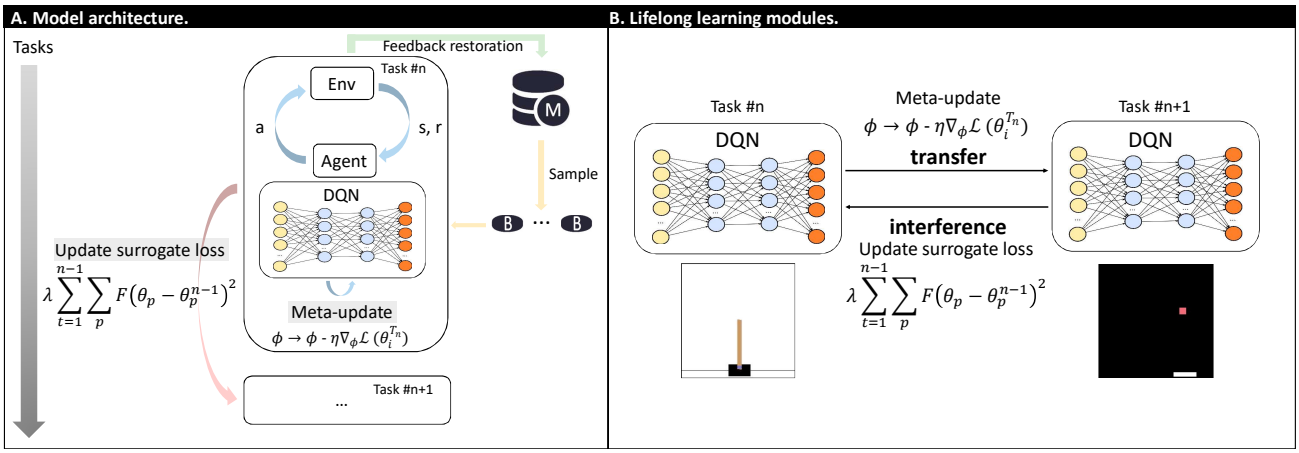



Figure 3: Structure diagram of proposed MR-SWC. The lifelong learning modules in Task $n$ and $n+1$ are illustrated.

Here we use First-order MAML (FMOMAL) to update initialization $\phi$ in DQN across batches for minimal computations (See Line 31-32 in Algorithm 3). MER also uses meta updating for lifelong DQN while they conducted both among and across updates with Reptile MAML, which is computational inefficient.

### 5.2. Stable weight consolidation for minimizing interference

To tackle the problem of catastrophic forgetting, we build a simple structural regularizer that could be computed and updated sequentially. We aim to measure the importance on parameters in previous tasks. To that end, we propose one variation of EWC (Kirkpatrick et al., 2017) called Stable Weight Consolidation (SWC). Same as EWC, the loss function for task $n$ is as follows.

$$\begin{aligned}\widetilde{\mathcal{L}} &= \mathcal{L}^n + \underbrace{\lambda \sum_{t=1}^{n-1} \sum_{p} F(\theta_p - \theta_p^{(n-1)})^2}_{\text{surrogate loss}} \\ &= \mathcal{L}^n + \lambda \sum_{p} (\theta_p^2 \sum_{t=1}^{n-1} F + \sum_{t=1}^{n-1} F(\theta_p^{n-1})^2 - 2\theta_p \sum_{t=1}^{n-1} \theta_p^{n-1}),\end{aligned} \tag{2}$$

where $\widetilde{\mathcal{L}}$, $\mathcal{L}^n$ denotes the whole loss function and task-specific loss respectively. In the surrogate loss, $\lambda$ denotes strength parameter, which can be seen as hyper-parameter to control the degree of remembering. $F$ here is the Fisher Information Matrix diagonal value corresponding to $p$. $\theta_p^{(n-1)}$ is the parameter $p$ after learning $(n-1)^{\text{th}}$ task. However, there is no upper-bound for inner product in Equation 2 when $n$ is large. This causes that SGD diverges in the landscape when facing a large amount of tasks. We utilize adaptive learning rate on learning rates to alleviate this problem. For each $p$, the corresponding learning rate is shown below.

$$\alpha_p = \frac{\alpha}{\lambda \sum_{t=1}^{n-1} F}.$$

The adaptive learning rate can prevent gradient divergence in our Lifelong DQN experiments. We call this combination of EWC and adaptive learning rate as Stable Weight Consolidation (SWC) and use it for each task during lifelong DQN training to minimize interference among time-varying distributions.

## 6. Experiments

In this section, we evaluate MR-SWC and make comparisons with the models we mention in section 4, including meta DQN, naive DQN, synapse DQN.

### 6.1. Experimental Setup

The DQN architecture is based on the deep, fully connected feedforward neural network, in which neurons between two adjacent layers are fully pairwise connected, but neurons within a single layer share no connection. This model is made up of an input layer, two adjacent hidden layers with 400 and 200 neurons, respectively, and an output layer. The network is trained with the soft Q-learning objective (Haarnoja et al., 2017), which helped to stabilise learning in each task, presumably by maintaining a more diverse set of experiences in the replay database. Furthermore, as in (Kirkpatrick et al., 2017) while the network weights are shared between tasks, each layer of the network is allowed to utilize task-specific gains and biases, such that computations at each layer are of the form:

$$y_i = g_i^{id}(g_i^{id} + \sum_{n=j} W_{ij} x_j)$$

**Algorithm 3** DQN with MR-SWC

**procedure** MR-SWC($env$, $sizelimit$, $\phi$, $\alpha$, $\eta$, $steps$, $k$, $p$)

// Initialize action-value function $Q$, $\hat{Q}$:
$Q \leftarrow Q(\phi)$
$\hat{Q} \leftarrow Q(\phi)$
// Initialize replay buffer:
$M \leftarrow \{\}$
$M.size = 0$
**while** $M.size \leq sizelimit$ **do**
  $env.reset()$
  **while** task $T_n$ not done **do**
    **while** episode not done **do**
      // Select action $a$:
      $a = \arg\max_{a'} Q(s_t, a'; \theta)$ ($\epsilon$-greedy)
      // Perform $a$ and store feedback:
      $s', r_t \leftarrow env.step(s, a)$
      $M \leftarrow M \cup (s, a, r, s')$
      $B_1, \cdots, B_b \leftarrow sample(s, a, r, s', b, k, M)$
      // Store current weights:
      $\theta_0^T \leftarrow \phi$
      **for** $i = 1, \cdots, b$ **do**
        $\theta_{i,0}^W \leftarrow \phi$
        **for** $j = 1, \cdots, k$ **do**
          // Sample one set of history from M:
          $s, a, r, s' = B_i[j]$

$$y = \begin{cases} r, & \text{terminate} \\ r + \eta \max_a \hat{Q}(s', a; \hat{\theta}), & \text{otherwise} \end{cases}$$

          // Optimization
          $\mathcal{L} \leftarrow H(y, Q(s, a; \theta_{i,j-1}^W) + regularizer$
          $\theta_i^W \leftarrow \theta_{i,j-1}^W - \alpha_p \frac{\partial \mathcal{L}}{\partial \theta_{i,j-1}^W}$
        **end for**
        $\theta_i^{T_n} \leftarrow \theta_{i,k}^W$
      **end for**
      // Across batch FO-MAML update:
      $\phi \leftarrow \phi - \eta \nabla_\phi \mathcal{L}(\theta_i^{T_n})$
      // Reset target action-value network:
      $\hat{Q} = Q$
    **end while**
  **end while**
  Update surrogate loss based on Task $T_n$ in Equation 2
**end while**
**return** $\theta, M$
**end procedure**

where identify indexes the task being trained on. This helps to overcome the issue of training a network on two different Q-functions, which has been reported to be very challenging even as a regression task (Rusu et al., 2015).

Our lifelong learning scenario follows (Kaplanis et al., 2018a), is composed of two tasks (Figure 4), Cart-Pole-v1 from the OpenAI Gym(Brockman et al., 2016) and Catcher, which are suitable for training on the same network as they have the same size of state and action space respectively. The agent is trained alternately on the two tasks (for 6 epochs with $2,000$ episodes in each epoch) and, as a measure of its ability to learn continually, the number of episodes for each agent to (re)learn the task after every switch is recorded. A task is deemed to have been (re)learnt if the agent reaches the terminal state within one episode ($500$ time steps). Reward per episode averaged over each epoch for each task is also recorded as another measure of evaluation.

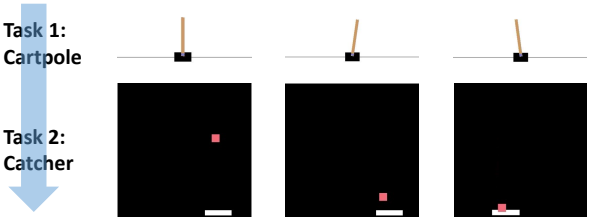



Figure 4: a sequence of frames from Cartpole and Catcher respectively. The goal in Cartploe is to prevent the pole, attached by an un-actuated joint to a cart, from falling over. In Catcher, the goal is to capture the falling pellet by moving the racket on the bottom of the screen.

The experience replay database has a size of 2000, from which 64 experiences are sampled for training with Adam (Kingma & Ba, 2014) at the end of every episode. Crucially, the database is cleared at the end of every epoch in order to ensure that the agent is only training on one task at a time. The agent is $\epsilon$-greedy with respect to the stochastic soft Q-learning policy and $\epsilon$ is decayed from 1 to almost 0 over the course of each epoch. Finally, 'soft' target network updates are used as in (Lillicrap et al., 2015), rather than hard periodic updates used in the original DQN.

For comparison, we choose naive DQN, DQN with BF (Kaplanis et al., 2018b), DQN with MER (Riemer et al., 2019) as our baselines. Details of experimental settings and hyperparameters are listed in Appendix B.

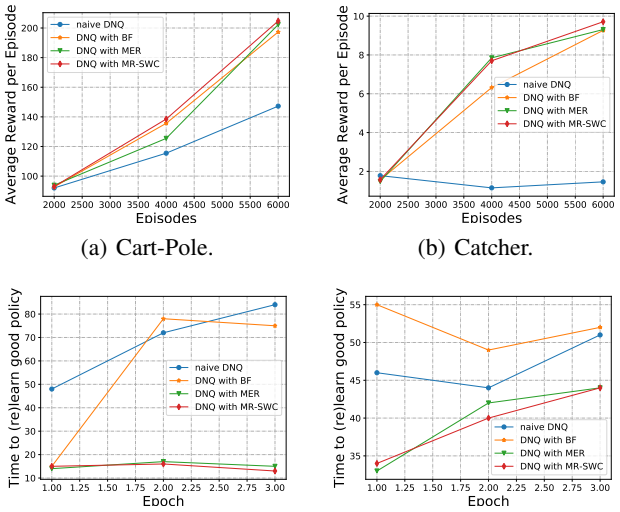
### 6.2. Evaluation Metrics

As mentioned in Section 3, our objectives include backward and future transfer. We follow (Kaplanis et al., 2018b) and consider average reward and #episode to relearn to evaluate the lifelong DQN performances based on the objectives.

**Average reward per episode:** With recording average reward for the agent, we can evaluate both backward and forward performances in lifelong reinforcement learning process.

**#Episode to (re)learn good policy:** This metric calculates how many episodes does the agent need to relearn an old task or learn a new policy. We record how many episodes the agents use to win the game. The speed of relearning old and learning new policies means how fast can the agent restores previous learnt memory (backward transfer) and fast adaptation on new tasks (forward transfer), respectively.

### 6.3. Results and Analysis




(a) Cart-Pole. (b) Catcher.

(c) Cart-Pole. (d) Catcher.

Figure 5: Experimental results on two evaluation metrics.

Experimental results are illustrated in Figure 5. We can find that naive DQN performs worst among with catastrophic forgetting. DQN with meta learning modules (MER and MR-SWC) are with high convergence speed compared with BF model (See 5(c) and 5(d)). We provide analysis on the results in Question 1&2 below.

**Question 1** *How does performance of DQN with MR-SWC compared with other baselines?* MR-SWC outperforms BF and MER slightly on average rewards. This indicates that weight consolidation reduces forgetting effectively. Meanwhile, MR-SWC has impressive results on convergence speed (See 5(c) and 5(d)) than BF model due to the meta representation module. In some check points, MR-SWC has faster learning speed than MER. This is mainly because MR-SWC only conducts across-batches meta updating and MER has two within and across batches meta updating. Within-batches meta updating may bring mis-information due to the existence of strong correlation in one batch.

**Question 2** *How to deal with the computational burden in MR-SWC?* Indeed, computation on Fisher information matrix brings much computational burden. However, we can consider utilize parallel cloud or edge computing techniques to achieve off-line deployment of this module in piratical applications (Du et al., 2019; Liu et al., 2017).

## 7. Discussion and future work

**More experimental protocols:** Due to the limitation on computational resources, we only conduct experiments on multiple task protocol. Prior works (Riemer et al., 2019) also simulate time-varying distribution in one single task (e.g., changing basket velocity in Catcher), where the domain or distribution gap is smaller than multiple tasks here.

**Potential extension to complex and piratical applications:** In this work, we show the piratical experiments under relatively easy games. Future enhancements to this work can focus on more complex applications. -e.g., develop lifelong learning algorithms for robots to make actions or decisions under complex non-stationary visual environments (similar to this 2-stage RL+Visual Control method for robots manipulation (Sermanet et al., 2017)).

**Other reinforcement learning paradigms:** In order to develop a simple but efficient method, we only utilize MR-SWC with DQN to achieve lifelong reinforcement learning in this work. We may deepen our work to apply MR-SWC to other reinforcement learning paradigms (off-policy, model-based, etc.).

## 8. Conclusion

In this work, we present a lifelong reinforcement learning method known as MR-SWC for learning optimal policy under time-varying environments. We show that combination of meta learning and regularization is capable of adapting unseen tasks and environments without forgetting previous learned tasks. We provide experiments in multiple games with DQN and compare with other three prior works. The numerical results show that our method can achieve 1) faster adaptation on new tasks; 2) better remembering previous knowledge with small memory buffer.

All the methods are implemented using Tensorflow (Abadi & et al, 2015) toolboxes with an Intel Core i9 CPU and 8 Nvidia RTX 2080 Ti GPUs.

## References

Abadi, M. and et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

Ammar, H. B., Tutunov, R., and Eaton, E. Safe policy search for lifelong reinforcement learning with sublinear regret. In *International Conference on Machine Learning*, pp. 2361–2369, 2015.

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, O. P., and Zaremba, W. Hindsight experience replay. In *Advances in neural information processing systems*, pp. 5048–5058, 2017.

Benna, M. K. and Fusi, S. Computational principles of synaptic memory consolidation. *Nature neuroscience*, 19 (12):1697, 2016.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Caccia, M., Rodriguez, P., Ostapenko, O., Normandin, F., Lin, M., Caccia, L., Laradji, I., Rish, I., Lacoste, A., Vazquez, D., et al. Online fast adaptation and knowledge accumulation: a new approach to continual learning. *arXiv preprint arXiv:2003.05856*, 2020.

Clavera, I., Nagabandi, A., Liu, S., Fearing, R. S., Abbeel, P., Levine, S., and Finn, C. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HyztsoC5Y7.

Du, B., Wu, C., and Huang, Z. Learning resource allocation and pricing for cloud profit maximization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 7570–7577, 2019.

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

Finn, C., Rajeswaran, A., Kakade, S., and Levine, S. Online meta-learning. *arXiv preprint arXiv:1902.08438*, 2019.

Forman, G. Tackling concept drift by temporal inductive transfer. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 252–259, 2006.

French, R. M. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

Haarnoja, T., Tang, H., Abbeel, P., and Levine, S. Reinforcement learning with deep energy-based policies. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1352–1361. JMLR. org, 2017.

Kamra, N., Gupta, U., and Liu, Y. Deep generative dual memory network for continual learning. *arXiv preprint arXiv:1710.10368*, 2017.

Kaplanis, C., Shanahan, M., and Clopath, C. Continual reinforcement learning with complex synapses. *arXiv preprint arXiv:1802.07239*, 2018a.

Kaplanis, C., Shanahan, M., and Clopath, C. Continual reinforcement learning with complex synapses. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2497–2506. PMLR, 2018b.

Kaplanis, C., Shanahan, M., and Clopath, C. Policy consolidation for continual reinforcement learning. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3242–3251, 2019.

Kemker, R., McClure, M., Abitino, A., Hayes, T. L., and Kanan, C. Measuring catastrophic forgetting in neural networks. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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 (PNAS)*, pp. 3521–3526, 2017.

Learning, L. M. Synthesis lectures on artificial intelligence and machine learning.

LeCun, Y., Bengio, Y., and Hinton, G. Deep learning. *nature*, 521(7553):436–444, 2015.

Li, Z. and Hoiem, D. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2017.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Lin, L.-J. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine learning*, 8(3-4):293–321, 1992.

Liu, N., Li, Z., Xu, J., Xu, Z., Lin, S., Qiu, Q., Tang, J., and Wang, Y. A hierarchical framework of cloud resource allocation and power management using deep reinforcement learning. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pp. 372–382. IEEE, 2017.

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 (PNAS)*, 115(44):467–475, 2018.

McCloskey, M. and Cohen, N. J. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pp. 109–165. Elsevier, 1989.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

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, 2015.

Pan, S. J. and Yang, Q. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10): 1345–1359, 2009.

Parisi, G. I., Tani, J., Weber, C., and Wermter, S. Lifelong learning of human actions with deep neural network self-organization. *Neural Networks*, 96:137–149, 2017.

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

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

Riemer, M., Cases, I., Ajemian, R., Liu, M., Rish, I., Tu, Y., , and Tesauro, G. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=B1gTShAct7.

Rusu, A. A., Colmenarejo, S. G., Gulcehre, C., Desjardins, G., Kirkpatrick, J., Pascanu, R., Mnih, V., Kavukcuoglu, K., and Hadsell, R. Policy distillation. *arXiv preprint arXiv:1511.06295*, 2015.

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

Schmidhuber, J. On learning how to learn learning strategies. 1995.

Sermanet, P., Lynch, C., Hsu, J., and Levine, S. Time-contrastive networks: Self-supervised learning from multi-view observation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 486–487. IEEE, 2017.

Shalev-Shwartz, S. et al. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, 4(2):107–194, 2012.

Shin, H., Lee, J. K., Kim, J., and Kim, J. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 2990–2999, 2017.

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

Sutton, R. S. and Barto, A. G. Reinforcement learning: An introduction. 2011.

Traoré, R., Caselles-Dupré, H., Lesort, T., Sun, T., Cai, G., Díaz-Rodríguez, N., and Filliat, D. Discorl: Continual reinforcement learning via policy distillation. *arXiv preprint arXiv:1907.05855*, 2019.

van de Ven, G. M. and Tolias, A. S. Generative replay with feedback connections as a general strategy for continual learning. *arXiv preprint arXiv:1809.10635*, 2018.

Wang, Z., She, Q., and Ward, T. E. Generative adversarial networks: A survey and taxonomy. *arXiv preprint arXiv:1906.01529*, 2019.

Watkins, C. J. C. H. Learning from delayed rewards. 1989.

Wixted, J. T. The psychology and neuroscience of forgetting. *Annu. Rev. Psychol.*, 55:235–269, 2004.

Yoon, J., Yang, E., Lee, J., and Hwang, S. J. Lifelong learning with dynamically expandable networks. *arXiv preprint arXiv:1708.01547*, 2017.

Zenke, F., Poole, B., and Ganguli, S. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70, pp. 3987–3995, 2017.

Zhai, M., Chen, L., Tung, F., He, J., Nawhal, M., and Mori, G. Lifelong gan: Continual learning for conditional image generation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2759–2768, 2019.

Zhang, S. and Sutton, R. S. A deeper look at experience replay. *arXiv preprint arXiv:1712.01275*, 2017.

Zhou, Z.-H., Zhang, M.-L., Huang, S.-J., and Li, Y.-F. Multi-instance multi-label learning. *Artificial Intelligence*, 176 (1):2291–2320, 2012.