Lifelong Learning of Task-Parameter Relationships for Knowledge Transfer

Shikhar Srivastava Mohammad Yaqub
Karthik Nandakumar
Mohamed Bin Zayed University of Artificial Intelligence
Abu Dhabi, UAE
{shikhar.srivastava,mohammad.yaqub,karthik.nandakumar}@mbzuai.ac.ae

Abstract

The ability to acquire new skills and knowledge continually is one of the defining qualities of the human brain, which is critically missing in most modern machine vision systems. In this work, we focus on knowledge transfer in the lifelong learning setting. We propose a lifelong learner that models the similarities between the optimal weight spaces of tasks and exploits this in order to enable knowledge transfer across tasks in a continual learning setting. To characterize the "task-parameter relationships", we propose a metric called adaptation rate integral (ARI), which measures the expected rate of adaptation over a finite number of steps for a (task, parameter) pair. These task-parameter relationships are learned using an auxiliary network trained on guided explorations of parameter space. The learned auxiliary network is then used to heuristically select the best parameter sets on seen tasks, which are consolidated using a hypernetwork. Given a new (unseen) task, knowledge transfer occurs through the selection of the most suitable parameter set from the hypernetwork that can be rapidly finetuned. We show that the proposed approach can improve knowledge transfer between tasks across standard benchmarks without any increase in overall model capacity, while naturally mitigating catastrophic forgetting.

1. Introduction

An embodied agent that operates in the unbounded partially-observable real-world with its diversity of tasks, must possess the ability to acquire knowledge and skills continually. An embodied and therefore finite agent, cannot feasibly grow its skill set as a set of disjoint abilities learnt afresh for each task. For complex tasks, this form of learning would be prohibitively expensive. Conversely, learning a *single* skill that can be modulated and maintained to be effective on the large diversity of tasks is infeasible (a broad proof is provided in the No Free Lunch Theorems; [42]). A *skill set* is therefore necessary - one that allows the agent

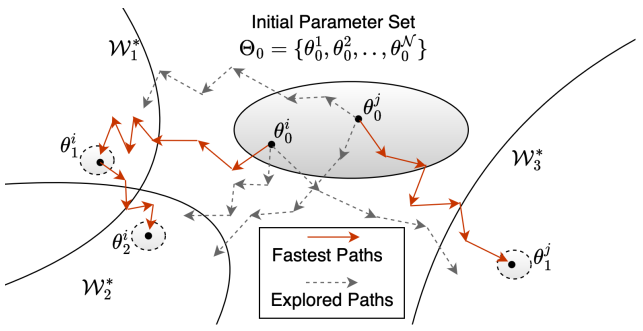


Figure 1. Exploiting shared structure in the optimal weight manifolds of tasks: Parameters trained on tasks with a greater vicinity of their optimal weight manifolds ($\mathcal{W}*$), may be transferred with a higher adaptation rate (illustrated as a faster path). Parameters adapted from the initialization set $\Theta_0 \sim p(\Theta)$ (centered, in grey) for tasks τ_1, τ_2, τ_3 in sequence. In this work, we model relationships between parameters in weight space and adaptation path lengths (and vicinity) to the optimal weight manifold of tasks.

to benefit from the *redundancies* between similar tasks that require similar skills. Skills optimized for previous tasks may reasonably be leveraged to improve learning quality and efficiency on unseen yet related tasks.

Such mechanisms are widely reflected in fundamental neuro-cognitive processes in human learning and memory [3,38]. The re-use of neural circuitry across diverse cognitive tasks appears to be a central organizational principle of the brain [2]. Shared structures and properties across tasks and environments are exploited. Similar tasks are solved rapidly and more effectively by re-using acquired skills, while novel experiences are prioritized within the learning bandwidths [3, 6]. A continual consolidation of knowledge occurs [1, 32, 41] aimed at retaining salient, widely and frequently re-usable knowledge/skills. This is particularly evident in memory organization, knowledge consolidation, and the role of novelty and forgetting in the memory [32, 38]. The aim of our work is to incorporate these mechanisms within the continual learning (CL) setting.

Modern machine learning (ML) algorithms still struggle to replicate this human ability to learn continually and utilize existing knowledge to learn faster without forgetting. The phenomenon of *catastrophic forgetting* [11, 28] - the difficulty in the retention of old information when new information is acquired, is commonly observed in training continually across multiple domains. This is a fundamental consequence of the transfer-interference trade-off [31] - for a *singular* finite network continually adapted to a shifting distribution, catastrophic forgetting is inevitable. The field of *continual learning* (CL) has therefore focused largely on mitigating catastrophic forgetting - with limited success in enabling knowledge transfer [21, 26, 30, 43] due in part to the single network constraints generally applied in CL [29].

In contrast, an important characteristic of the human learning process is the elevated 'quality of convergence' onto new tasks that share observed structures - a knowledge transfer that enables a higher rate and quality of learning (varying with the degree and type of shared task structures). In a lifelong learning setting, an efficient use of continually acquired knowledge necessitates a benefit to such quality of adaptation on new, albeit related tasks. In the limit, a continually learning agent that most rapidly converges on a new task, effectively already understands the task. This manner of optimization benefits an agent's supervision/data requirements in learning a new task.

We motivate this re-use of acquired skills for improved adaptation quality on new tasks, as an important trait of a generally learning agent. A faster 'rate of adaptation' (or convergence) in ML can be formulated as a shorter adaptation trajectory to the basin of convergence. This can be achieved by an initialization with a higher affinity to the basin or by enforcing a more direct trajectory [10] through auxiliary feedback/constraints. In contrast, work on mitigating catastrophic forgetting is focused on retaining parameters within the (optimal) weight manifolds learned for a task [31]. In this work, we begin with the hypothesis that the shared structure between tasks may be captured in the optimal weight spaces of the (networks trained on the) tasks. We also argue that in order to allow skill re-use for a new task, a heuristic to search the explored (or previously learned) weights set for the most promising weights is required.

To this end, we make the following contributions:

- To measure 'adaptation quality' for a (parameter θ, task τ), we formulate a metric - *adaptation rate integral* (ARI) that captures the convergence rate and performance of a parameter θ trained on a task τ.

- We develop a heuristic that can efficiently search the *observed* space of parameters (base model weights trained on previous tasks) by estimating the adaptation quality of any (θ, τ) pair using an auxiliary network.

- Finally, in order to efficiently store the parameters learned from previous tasks, we employ a meta model that stores all observed parameters in its representation. This approach does not require any additional model capacity compared to a single (base) model.

We leverage these contributions to incrementally learn and explore the observed space of parameters, improving the degree of knowledge transfer as well as the retained accuracy of the continual learner. We show that the benefits to knowledge transfer come with no increase in overall model capacity, while mitigating catastrophic forgetting naturally.

2. Preliminaries

In a supervised learning setting, a hypothesis $h : \mathcal{X} \rightarrow \mathcal{Y}, h \in \mathcal{H}$ is learned on the input and label spaces $\mathcal{X}$, and $\mathcal{Y}$ ($\mathcal{H}$ is the hypothesis space). The learner $h \equiv f(\cdot;\theta) : \mathcal{X} \rightarrow \mathcal{Y}$ can be defined as a ML model f parameterized by θ. The input and label spaces are related by a joint probability distribution $P(\boldsymbol{x}, y) \mid \boldsymbol{x} \in \mathcal{X}$ and $y \in \mathcal{Y}$. The empirical risk minimization (ERM) principle formulates this problem of learning h as a minimization of the population risk $e_P(h) = \mathcal{P}(h(\boldsymbol{x}) \neq y)$. For a finite sample set $\boldsymbol{S} \equiv (\boldsymbol{X}, \boldsymbol{Y}) = \{\boldsymbol{x}_i, y_i\}_{i\in[1,N]}$ of size N seen by the learner h, the empirical risk is $\hat{e}_{\boldsymbol{S}}(h) = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}_{h(x_i)\neq y_i}$ where $\mathbf{1}_z$ is an indicator function. The constrained ERM optimization problem is therefore defined as:

$$\text{ERM Optimization: } \min_{\forall\, h\in\mathcal{H}} \ \hat{e}_{\boldsymbol{S}}(h). \tag{1}$$

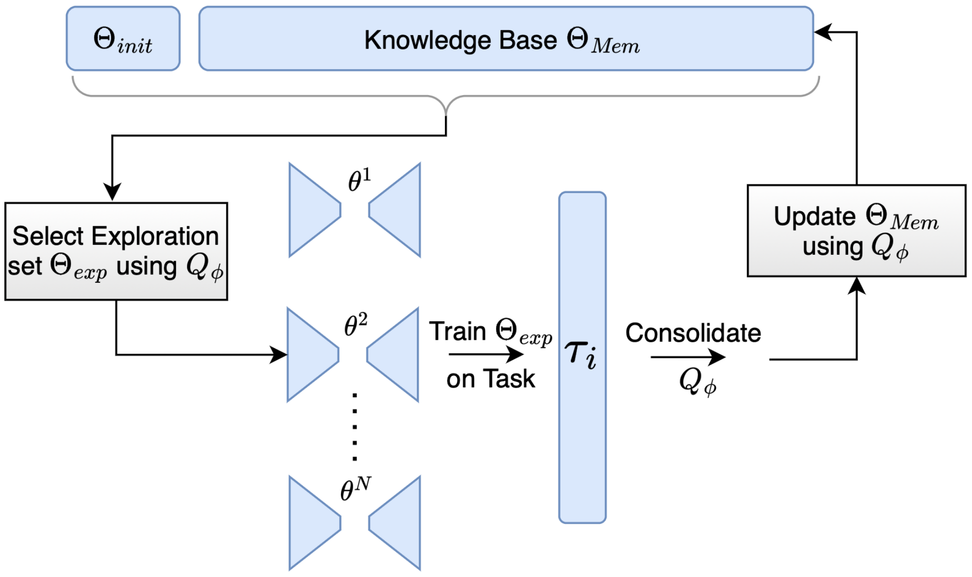
While in a typical continual learning (CL) setting [12], the learner observes a sequence of M tasks $[\tau_1, \tau_2, ..., \tau_M]; \tau_i \sim P(\tau)$ sampled from a distribution of tasks $P(\tau)$. Here, a task τ_i is defined as a set of N_i input, label pairs: $\boldsymbol{S}_i \equiv (\boldsymbol{X}_i, \boldsymbol{Y}_i) = \{\boldsymbol{x}_n, y_n\}_{n\in[1,N_i]}$. Typically, the objective is to find parameters θ that minimize the *cumulative loss* on all m seen tasks $\tau_{[1:m]}$, while having limited access to data $\boldsymbol{S}_i$ from previous tasks τ_i $(i < m)$. The CL objective is:

$$\begin{aligned}\min_{\theta} \sum_{i=1}^{m} \mathbb{E}_{\boldsymbol{S}_i} \left[l_i(\, f(\boldsymbol{X}_i;\theta), \boldsymbol{Y}_i) \right] = \min_{\theta} \\ \mathbb{E}_{\boldsymbol{S}_{[1:m]}} \left[L_m(f(\boldsymbol{X}_{[1:m]};\theta), \boldsymbol{Y}_{[1:m]}) \right]\end{aligned} \tag{2}$$

where l_i is the loss on task τ_i and L_m is the cumulative sum of all m task-specific losses $L_m = \sum_{i=1}^{m} \ l_i$.

3. Methodology

Our proposed continual learning algorithm operates in the meta space of parameters. It uses a heuristic to incrementally explore the meta space and consolidate parameters that are estimated to do well on the distribution of observed tasks within a knowledge base. Note that a parameter (θ) refers to the set of weights of a 'base' model trained on a task.



Θ_{init}
Knowledge Base Θ_{Mem}
Select Exploration set Θ_{exp} using Q_ϕ
θ^1
θ^2
θ^N
Train Θ_{exp} on Task
$\mathcal{T}_i$
Consolidate Q_ϕ
Update Θ_{Mem} using Q_ϕ

Figure 2. Overall Method: During training, a set of parameters are trained on the given task and their corresponding ARIs are calculated. After filtering using the estimated ARI metric, the selected parameters are stored in the hypernetwork for future retrieval. At Inference, from the parameter set Θ_{Mem} stored in the hypernetwork, the parameter θ_i with the maximum predicted ARI for the given inference task τ_k is fetched, finetuned, and evaluated.

3.1. Proposed CL Algorithm

The proposed approach relies on (i) a small auxiliary network Q_ϕ that estimates the 'quality of convergence' of a parameter θ trained on a task τ and (ii) a meta model F_Φ that efficiently stores parameters as a knowledge base (Θ_{Mem}). The algorithm consists of two phases: *training* and *inference*. **Training** consists of an Exploration and a Consolidation phase. Let Θ_{init} be a set of randomly initialized parameters.

1. *Exploration*: A combined training set Θ_{exp} of parameters from the knowledge base and initialization set ($\Theta_{exp} \subseteq \Theta_{Mem} \cup \Theta_{init}$) is selected and adapted on task τ. A measure of the convergence quality of all Θ_{exp} parameters on task τ is calculated and retained.

2. *Consolidation*: The calculated convergence qualities from all observed (parameter, task) pairs is used to train the auxiliary network Q_ϕ. Finally, the subset of the combined knowledge base and adapted training set ($\Theta_{Mem} \cup \hat{\Theta}_{exp}$), that has the highest estimated quality of convergence on the observed distribution of tasks, is retained in the updated knowledge base Θ_{Mem}.

Inference consists of selecting the parameter from the knowledge base, with the highest estimated quality of convergence for the new task, fine-tuning and evaluating it.

A training budget Q can be enforced by simply using the auxiliary network Q_ϕ to select the best candidate networks for training on a task. Further, as the architectures are the same, data streams (identical &) independent and only the initializations differ, training of the parameter sets is fully parallelizable with negligible overhead to training time complexity.

3.2. Convergence Quality Measure: Adaptation Rate Integral (ARI)

To formalize the 'quality of convergence' of a given parameter to the task space, we introduce the Adaptation Rate Integral (ARI) metric. In a typical supervised learning setting, solving the ERM optimization (eq. 1) involves iteratively modifying an initial hypothesis $h(0)$ in discrete steps $h(t+1) = h(t) + \alpha(t).\delta(t)$. Here, $t \in [0, T]$ denotes the step index, $h(t)$ represents the hypothesis at step t, $\alpha(t)$ denotes the learning rate, and $\delta(t)$ is typically an estimate of the gradient of the empirical risk. We define the *adaptation rate integral* or ARI, as simply the step-averaged area under the curve of $(1 - \hat{e}_{\boldsymbol{S}}(h(t)))\ \forall\ t \in [0, T]$. For a continuous step space with infinitesimally small step sizes dt,

$$\psi\ (\text{ARI}) \quad = 1 - \frac{1}{T}\int_0^T \hat{e}_{\boldsymbol{S}}(h(t))\ dt \tag{3}$$

If the optimizer is allowed a maximum of T steps to solve eq. 1, the adaptation rate integral (ARI) is maximized when the step averaged empirical risk on sample set $\boldsymbol{S}$ over T steps is minimum. Ideally, a learning algorithm should converge to a good quality solution (one that achieves global minimum of the empirical risk) in the fewest possible steps. The ARI value attempts to measure both the quality of the converged solution and how fast this solution can be reached. The challenge lies in how to estimate the value of ψ given an initial hypothesis $h(0)$ and the optimizer, without explicitly constructing the complete adaptation trajectory $\boldsymbol{h}$.

3.2.1 Proposed CL Objective: *ARI Maximization*

Let the initial parameters of function $f(\cdot;\theta)$ adapting on task τ with a loss function l be θ_0. Then, an SGD operator $U(\theta_0)$ acting on parameter θ_0 is defined as follows: $U(\theta_0) = \theta_1 = \theta_0 - \alpha\nabla_{\theta_0} l(\theta_0)$. Thus, the parameters θ_T obtained after T adaptation steps can be composed as $U_T(\theta_0) = U \circ U \cdots \circ U(\theta_0) = \theta_T$. In the CL setting, for a model f with initial parameters θ_0 adapted on task τ_i over T steps, the *adaptation rate integral* is defined as:

$$\psi(\theta_0, \tau_i, T) = 1 - \frac{1}{T}\sum_{t=0}^{T} \hat{e}_{\boldsymbol{S}_i}(f(\boldsymbol{X}_i;\theta_t), \boldsymbol{Y}_i)\ ;\ U_t(\theta_0) = \theta_t \tag{4}$$

Now, in order to learn the optimal initial parameters $\theta_0^* \in \Theta$ that maximizes the rate of adaptation on a distribution of tasks $\tau \sim \mathcal{P}(\tau)$, the learning objective using *ARI maximization* becomes:

$$\max_{\theta_0 \in \Theta} \mathbb{E}_{\tau_i \sim \mathcal{P}(\tau)} \left[\psi(\theta_0, \tau_i, T)\right] = \max_{\theta_0 \in \Theta} \mathbb{E}_{\tau_i \sim \mathcal{P}(\tau)} [1 - \frac{1}{T}\sum_{t=0}^{T} \hat{e}_{\boldsymbol{S}_i}(f(\boldsymbol{X}_i;\theta_t), \boldsymbol{Y}_i)] \tag{5}$$

For the continual learning setting, the ARI maximization objective can thus be expressed as:

$$\max_{\theta_0 \in \Theta} \sum_{i=1}^{M} \psi(\theta_0, \tau_i, T) = \max_{\theta_0 \in \Theta} \Psi(\theta_0, \tau_{[1:m]}, T) \tag{6}$$

where the learner has a budget of T iterations/steps to converge to each task τ, and $\Psi(\theta_0, \tau_{[1:m]}, T) = \sum_{i=1}^{m} \psi(\theta_0, \tau_i, T)$. In simple terms, given a new task τ where a learner's initial parameter θ_0 is evolved across T adaptation steps $\theta_T = U_T(\theta_0)$, we would like to find the optimal $\theta_0 \in \Theta$ such that the expected step-averaged empirical risk on τ across the T steps is minimized (Eq. 5). The proposed objective explicitly attempts to maximize the quality of adaptation of the learner to the observed task set as well as new tasks, which are sampled from the same task distribution.

3.3. Overall Lifelong Learner & Components

As a lifelong learner, our proposed approach operates as a compressed knowledge base $\Theta_{Mem} = \{\theta^1, \theta^2, \cdots, \theta^{\mathcal{N}}\}$ of explored base model parameters that are optimized for maximal 'adaptation quality' to the distribution of tasks $\tau \sim P(\tau)$ (while requiring the parameter budget of a single base model). Typically, models are often over-parameterized in continual learning with wider layers to mitigate the degree of interference. We circumvent the problem of interference, by searching the space of base model parameters and maintaining the subset of parameters with maximal expected ARI on the observed task distribution.

Algorithm 1 Lifelong Learning Algorithm

Input: Observed Tasks: $\tau_{[1:i-1]} = \{\tau_1, \tau_2, \cdots, \tau_{i-1}\}$, memory budget $\mathcal{B}$, training budget $\mathcal{Q}$
Require: ARI Estimator: Q_ϕ, meta model F_Φ and embedding vectors $E_{\mathcal{M}}$, model and task encoders: $\mathcal{E}_{\text{param}}, \mathcal{E}_{\text{task}}$ that generate model, task encodings $\boldsymbol{\eta}_\theta, \boldsymbol{\eta}_\tau$ resp. , Generated initialization set $\Theta_{init} = \{\theta_0^1, \theta_0^2, .., \theta_0^{\mathcal{N}}\}$
Ensure: $|\Theta_M| \leq \mathcal{B}$ and $|\Theta_{exp}| \leq \mathcal{Q}$

```
 1: Knowledge Base $\Theta_{Mem}$, Exploration Set $\Theta_{exp}$, Buffer $S = \{\}$
 2: for Training on task $\tau_i$ do
 3:     $\Theta_{Mem} \leftarrow$ Retrieve from meta-model $F_\Phi$
                                                   ▷ Exploration
 4:     Select exploration set $\Theta_{exp}$:
          $\Theta_{exp} \leftarrow \arg\min_{\Theta_{exp}} \sum_{\theta_0 \in \Theta_{exp}} Q_\phi(\boldsymbol{\eta}_{\theta_0}, \boldsymbol{\eta}_{\tau_i})$
            where $\Theta_{exp} \subseteq (\Theta_{Mem} \cup \Theta_{init})$, $|\Theta_{exp}| \leq \mathcal{Q}$
 5:     Train parameters in exploration set $\Theta_{exp}$:
          $\Theta_{exp} \leftarrow \{U_T(\theta_{exp})\}, \forall\, \theta_{exp} \in \Theta_{exp}$
 6:     Calculate and store true ARIs into buffer:
          $S = S \cup \{\psi(\theta_{exp}, \tau_i, T), \boldsymbol{\eta}_{\theta_{exp}}, \boldsymbol{\eta}_{\tau_i}\} \forall\, \theta_{exp} \in \Theta_{exp}$
                                                   ▷ Consolidation
 7:     Train ARI Estimator $Q_\phi$ on S,
          $\phi \leftarrow \phi - \nu\nabla_\phi \sum_{\{\psi, \boldsymbol{\eta}_\theta, \boldsymbol{\eta}_\tau\} \in S} ||\psi - Q_\phi(\boldsymbol{\eta}_\theta, \boldsymbol{\eta}_\tau)||_2^2$ (Eq. 7)
 8:     Consolidate $\Theta_{Mem}$:
          $\Theta_{Mem} \leftarrow \arg\min_{\Theta_{Mem}} \sum_{\theta \in \Theta_{Mem}} Q_\phi(\boldsymbol{\eta}_\theta, \boldsymbol{\eta}_{\tau_{[1:i]}})$
            where $\Theta_{Mem} \subseteq (\Theta_{exp} \cup \Theta_{Mem})$, $|\Theta_{Mem}| \leq \mathcal{B}$
 9: end for
                                                   ▷ Inference
10: for Inference on task $\tau_m$ do
11:     $\Theta_{Mem} \leftarrow \{F_\Phi(e_i) \forall e_i \in E_{\mathcal{M}}\}$ Generate from meta-model
12:     $\theta_* \leftarrow \arg\max_{\theta_* \in \Theta_{Mem}} Q_\phi(\boldsymbol{\eta}_{\theta_*}, \boldsymbol{\eta}_{\tau_m})$
13:     $\theta_* \leftarrow$ Finetune $\theta_*$ using exemplar memory $D$, then infer on $\tau_m$
14: end for
```

The proposed overall lifelong learning algorithm is detailed in Algorithm 1. Below, we detail the various components of the CL learner:

3.3.1 ARI Estimator

As searching exhaustively through a large space of parameters is prohibitively expensive, we propose a heuristic method to search the parameter space efficiently for parameters that most effectively adapt to observed and new tasks. Such a heuristic would effectively characterize the model-task relationships.

We learn a small-capacity auxiliary network Q_ϕ that estimates the true ARI $\psi(\theta, \tau, T)$ for a (parameter, task) pair $(\theta, \tau), \theta \in \Theta, \tau \sim P(\tau)$. Thus, when a new task τ is observed, an estimation of adaptation rate of the parameter θ to the task τ can be generated off hand by the auxiliary network, without requiring a full training pass over the task. However, this first requires a projection of the parameters and tasks into a shared representational space γ, which is informative of a parameter θ's adaptation rates for a task as well as a task τ's characteristics. We learn two auxiliary encoders - a parameter encoder $\mathcal{E}_{\text{param}}$ and a task encoder $\mathcal{E}_{\text{task}}$. Following the approach in [17], we extract functional signatures of the parameters $g_\alpha(\theta)$ and the task $g_\beta(\tau)$ and pass them to the encoders to generate the respective embeddings $\boldsymbol{\eta}_\theta = \mathcal{E}_{\text{param}}(g_\alpha(\theta))$ and $\boldsymbol{\eta}_\tau = \mathcal{E}_{\text{task}}(g_\beta(\tau))$. These embeddings are then used by the ARI estimator network Q_ϕ, which learns to regress the true ARI measured when parameter θ is adapted on task τ for T steps via solving the following optimization objective.

$$\min_{\phi} ||\psi(\theta_j, \tau_i, T) - Q_\phi(\boldsymbol{\eta}_{\theta_j}, \boldsymbol{\eta}_{\tau_i})||_2^2 \;\; \forall\, \theta_j \in \Theta_{\mathit{Mem}}, \tau_i \in \tau_{[1:t]}, \tag{7}$$

where Θ_{Mem} is the set of explored parameters (knowledge base) that has been adapted on the m observed tasks $\tau[1:m]$. Note that ϕ includes the parameters of the two encoders $\mathcal{E}_{\text{param}}$ and $\mathcal{E}_{\text{task}}$ as well as the regression model Q_ϕ. Given an embedding space that captures degrees of structural/domain similarity across the task distribution, the auxiliary network Q_ϕ generalizes to predict the rate of adaptability to new tasks similar to the observed set. The training objective (Eq. 7) is continually 'consolidated' and we hypothesize that the ARI estimator may naturally become more stable as number of observed tasks increases.

3.4. Knowledge Base Representation using Hypernetworks

In order to efficiently maintain the set of explored parameters (knowledge base) Θ_{Mem} without explicitly storing them, we leverage a hypernetwork meta model F_Φ. Hypernetworks [14] are meta models that learn to map embedding vectors to parameters [40], and can be thought of as weight generators. They have been shown to efficiently retain a large parameter set with no decrease in the evaluated performance of the parameters [40].

We train the hypernetwork F_Φ to map the knowledge base Θ_{Mem} to a set of learned embedding vectors $\{e_i\}_{i=1}^{n(\Theta)}$ that can be thought of as indices for parameters in Θ_{Mem}. Thus, given a parameter index i, the parameter θ_i can be be readily generated using the hypernetwork as $\theta_i = F_\Phi(e_i)$. This effectively reduces the storage complexity of the knowledge base from $|\Theta_{\mathit{Mem}}|$ to $|F_\Phi|$, where $|F_\Phi| \approxeq |\theta|, \theta \in \Theta_{\mathit{Mem}}$. We discuss the details of the hyper-network architecture and hyperparameters in Section 4.4.

4. Experiments

Following earlier continual learning literature [5, 26], and owing to compute restrictions involved in training base models parallely, we conduct experiments and ablations using 4 smaller scale continual learning benchmarks - SplitMNIST [4], PermutedMNIST [43], Split CIFAR10 [43] and Split CIFAR100 [30]. We evaluate against several standard baselines in continual learning - ER [31], GEM [26], A-GEM [5], iCaRL [30], EWC [21] and MER [31].

4.1. Datasets

The benchmarks considered for the continual learning experiments are described briefly below:

Split-MNIST: The MNIST dataset [25] is split into 5 incremental tasks of 2 consecutive classes each [4].

Permuted-MNIST consists of 10 tasks each with a random spatial permutation of the MNIST dataset. Each task contains 1000 images for training and the original MNIST test set for evaluation [26].

Split CIFAR-10 and **Split CIFAR-100**: Similarly, CIFAR-10 [22] and CIFAR-100 datasets are split into 10 and 20 independent incremental tasks by grouping 5 and 10 classes together into a task, respectively. Class grouping follows the standard protocol in [22]. Each task contains 2500 images for training, with the original test set for evaluation.

4.2. Baselines

We briefly describe the baseline formulations below:

Fine-tuning model - A naive baseline where a single model is trained continually on all tasks.

Experience Replay (ER) [31] - A strong replay-based baseline where samples from previous tasks are interleaved with samples of the current task.

GEM [26] and Averaged GEM (A-GEM) [5] - GEM and its efficient approximation A-GEM methods explicitly bound the model's loss on samples of previous tasks (stored in a memory buffer).

iCaRL [30] - uses a nearest-mean-of-exemplars classification strategy, with a knowledge distillation loss over feature representations of past tasks to limit catastrophic forgetting.

EWC [21] - A regularization based approach to mitigate catastrophic forgetting, where the parameters are crucial to the performance on previous tasks, as measured by the Fisher Information Matrix (FIM), are not modified. We consider the approach in online EWC [34] to calculate FIM as a moving average.

MER [31] - formulates replay as a meta-learning optimization where gradient alignment is enforced to minimize interference between task gradients and encourage transfer.

Table 1. Average Accuracy (ACC), Backward Transfer (BWT) and Forward Transfer (FWT) across 4 benchmarks. Values averaged across 3 runs (Standard deviations are reported in the supplementary material.)

Datasets →	Split MNIST			Permuted MNIST			Split CIFAR-10			Split CIFAR-100		
Methods ↓	ACC	FWT	BWT	ACC	FWT	BWT	ACC	FWT	BWT	ACC	FWT	BWT
Fine-tuning	30.2	1.2	−51.2	63.4	0.0	−60.1	41.3	0.9	−30.1	22.1	1.0	−24.0
ER	84.6	45.5	−5.4	76.6	10.2	−9.0	79.8	50.1	−3.2	44.6	49.0	−5.4
EWC [21]	43.9	4.2	−25.5	72.36	2.1	−14.9	59.1	13.6	−18.2	34.1	29.2	−11.7
GEM [26]	81.1	31.0	−8.3	82.8	8.0	−6.7	75.3	34.0	−14.7	44.3	45.8	−5.5
A-GEM [5]	77.8	25.4	−11.0	78.2	7.3	−10.5	70.7	32.1	−16.8	39.9	32.1	−5.9
iCaRL [30]	85.4	56.2	−3.1	–	–	–	66.2	27.4	−5.1	31.9	15.7	−1.2
MER [31]	86.1	62.5	−4.5	81.4	9.1	−5.8	80.5	54.8	−2.9	46.2	50.1	−2.2
Proposed (Ours*)	88.7	67.1	−1.1	84.2	10.4	+0.9	81.9	58.0	−1.7	48.8	54.2	−1.0

4.3. Evaluation Metrics

Following previous works [4,26], we consider the metrics of Average Accuracy (ACC), Backward Transfer (BWT), and Forward Transfer (FWT) for all experiments. Once the model is trained on task t_i, it is evaluated on the test-sets of all tasks in task-set T, resulting in a matrix $R \in \mathcal{R}^{T\times T}$. Each element R_{ij} is the test-classification accuracy of the model on task t_j after learning on examples from task t_i. Average accuracy (ACC $\in [0, 100]$) after learning task T can be defined as: ACC $= \frac{1}{T}\sum_{i=1}^{T} R_{T,i}$. The degree of Backward Transfer (BWT) is defined as: BWT $= \frac{1}{T-1}\sum_{i=1}^{T-1} R_{T,i} - R_{i,i}$. A positive BWT implies that learning a new task t improves the performance of the model on a previously seen task k. A significant negative BWT indicates that *catastrophic forgetting* has occurred. Forward Transfer (FWT) $= \frac{1}{T-1}\sum_{i=2}^{T} R_{i-1,i} - \hat{b}_i$ quantifies the degree to which learning a task t affects the performance of the model on future tasks $k > t$. A positive FWT signifies that the model will be successful in performing 'zero-shot' predictions on unseen tasks $k > t$ as a result of learning given task t.

4.4. Experimental Settings

We follow the protocols in [26] for our choice of experimental settings and build on the implementation provided by [12] and [40] to implement our baselines and hypernetwork meta-models respectively.

For MNIST experiments, we follow [26] and use a two layer, 100-neuron each, fully-connnected neural network with ReLU activation for the MNIST datasets. As a meta model, we use a fully-connected two-hidden layer ($[100, 100]$) chunked hypernetwork [40] with a chunk size of 200 and embedding vectors of size 8. The meta model contains 59,668 weights in comparison to a single base model with 89,400 weights. All experiments occur in the task-aware setting, and all baselines including ours are implemented using a single-headed base network. Our base models are trained in parallel for every task with parallel streams of training data from each task, and inference is done on the base model selected from the Knowledge Base Θ_{Mem}.

For CIFAR, we use a modified version of the ResNet18 [15] with one-third the feature maps across all layers, as in [26]. As a meta model, we use a larger hypernetwork with structured chunking that internally maintains 6 smaller composite two hidden layer ($[25, 25]$) hypernetworks, for a total of 166,610 weights (compared to 181,495 for a single base model). The baselines for CIFAR are all multi-headed and task-aware, while base models of our method are trained as single-headed networks (for each task).

The hypernetworks are trained with embedding vectors of size 8. Similar to the baseline [26], we maintain a small exemplar memory D of size 200 for MNIST experiments and 400 for CIFAR. For our ARI estimator, we follow [18] in generating task or parameter signatures by the activations of a pre-trained ResNet18 on samples per class (of each task) or the activations of the parameters on random gaussian noise, respectively. The activations are normalized and padded to a size of 2048, before being projected to 128 dimensions. The ARI estimator is a simple two hidden layer ($[100, 100]$) network with ReLU activations. We enforce a maximum size of 20 parameter sets on the knowledge base, and a training budget of 10 base models per task (see ablation in Sec. 5.1). Our initialization set contains 3 randomly generated parameters.

4.5. Training Details

To train the base model across all baselines we use the Adam [20] optimizer set with an initial learning rate of 0.001, weight decay of 0.001, and a batch size of 10, similar to baseline methods [21,26,30]. We also utilize a small buffer S to store the collected (ARI, task & parameter encoding) tuples. For our method, the ARI values for each parameter, task pair are calculated based on empirical steps required till

Table 2. Ablation: Influence of # of base models ($\mathcal{B}$) in the knowledge base across all benchmarks. Selected default configuration is highlighted in light blue, with cases having at least 1 model per task highlighted in grey. Values have been averaged across 2 runs.

Datasets $\rightarrow$	Split MNIST			Permuted MNIST			Split CIFAR-10			Split CIFAR-100		
Methods $\downarrow$	ACC	FWT	BWT	ACC	FWT	BWT	ACC	FWT	BWT	ACC	FWT	BWT
Fine-tuning	30.2	1.2	-51.2	63.4	0.0	-60.1	41.3	0.9	-30.1	22.1	1.0	-24.0
$\mathcal{B} = 1$	22.5	0.4	-55.0	40.5	2.0	-72.3	31.1	0.1	-65.0	14.0	0.0	-45.5
$\mathcal{B} = 2$	38.9	2.3	-35.1	62.1	1.7	-35.2	56.2	10.4	-25.1	29.3	22.9	-10.3
$\mathcal{B} = 5$	80.1	30.2	-9.4	74.5	5.5	-11.1	68.9	28.5	-20.8	37.7	30.5	-15.5
$\mathcal{B} = 10$	85.2	54.1	-2.5	81.0	6.7	-2.0	77.2	38.9	-3.0	43.2	34.1	-3.2
$\mathcal{B} = 20$	88.7	67.1	-1.1	84.2	10.4	$+0.9$	81.9	58.0	-1.7	48.8	54.2	-1.0
$\mathcal{B} = 30$	87.2	63.9	-1.9	82.9	8.3	$+0.1$	80.3	52.3	-1.1	48.9	57.2	-1.8

Table 3. Ablation Study: Evaluating the ARI Estimator

Datasets $\rightarrow$	Split MNIST			Permuted MNIST			Split CIFAR-10			Split CIFAR-100		
Methods $\downarrow$	ACC	FWT	BWT	ACC	FWT	BWT	ACC	FWT	BWT	ACC	FWT	BWT
Random Selection	60.1	15.7	-37.2	45.4	1.5	-55.1	49.2	0.0	-32.3	31.4	4.3	-18.1
ARI Estimator (Ours*)	88.7	67.1	-1.1	84.2	10.4	$+0.9$	81.9	58.0	-1.7	48.8	54.2	-1.0

convergence. The ARI estimator is trained for 1000 epochs with a batch size of 250 (parameter & task embeddings, ARI) tuples. Finally, the meta-models are trained using the SGD optimizer till convergence (an MSE error of 1e-3), and perform a single pass of the entire parameter set in one batch.

5. Results and Discussion

We discuss the performance comparison between our method and the selected baselines, provide an ablation study of the memory size of the knowledge base, the heuristic used to selected the candidate base model for inference, as well as the memory complexity utilized by the method.

The ACC, BWT, and FWT metrics for various methods are summarized in Table 1. Our approach achieves near consistent improvements in both Accuracy (ACC) as well as knowledge transfer (BWT and FWT) with large gains in BWT in particular with signficantly lower catastrophic forgetting and a positive BWT in the case of the Permuted-MNIST benchmark. It is arguably the case that a lower catastrophic forgetting naturally results from the presence of multiple parameters retrievable from previously observed tasks. However, the improvements in FWT across all benchmarks show that the candidates retrieved for training on new tasks observe improvements over a single learner. Additionally we note that the baseline MER [31] performs comparably in terms of ACC in Split-MNIST and Split-CIFAR10 however a clear benefit to FWT and BWT is observed in all benchmarks. As expected, iCaRL [30] achieves lower forgetting on most datasets compared to other baselines, except MER and the proposed method. This can be attributed to its auxiliary knowledge distillation loss [16, 19]. Next, we discuss the ablation on the heuristic used to pick the best candidate models from the knowledge base, as well as the memory budget of the knowledge base in our approach.

5.1. Ablation: Trivial heuristic to pick candidate models

In our ablation study, we evaluate the benefit from our ARI estimator. Table 3 shows the performance of our approach with the proposed ARI estimator along with the performance of the same method with a random selection heuristic. In this random selection heuristic, the parameters from the hypernetwork are selected during training and inference using a random heuristic. We observe a clear and large reduction in the average accuracy as well as an increase in the degree of forgetting observed. Without the proposed ARI estimator based heuristic, the approach collapses as it fails to select the appropriate parameters for the test tasks. Random selection also prevents retaining the optimal weights in the hypernetwork (memory). We posit that the proposed ARI estimator models the relationships between the learned task weights (of the base models trained on the task), the task features and the expected ARI for this (model, task) pair.

5.2. Ablation: Number of base models in knowledge base

In Table 2, we show an ablation on the number of base model parameters $\mathcal{B}$ stored by the hypernetwork (knowledge

# of params in →	GEM [26]	Ours*
Split MNIST	89610	59668
Permuted MNIST	89610	59668
Split CIFAR 10	181495	166610
Split CIFAR 100	181495	166610

Table 4. Parameter requirements of the proposed method vs. GEM [26] baseline.

base memory budget $\mathcal{B} = \Theta_{Mem}$). For each budget $\mathcal{B}$, the number of models in the knowledge base are at most $\mathcal{B}$ during the continual training, and the number of candidate models remain same as other experiments (training budget of 10 base models). We observe that the performance of the proposed method largely increases across all metrics (FWT, BWT, ACC) steeply till a peak at budget $\mathcal{B} = 20$ after which it falls gradually. Additionally, the performance of the approach deteriorates completely with a singular base model budget. This suggests that multiple base models are necessary for good model performance. On the other hand, the ARI estimator based search heuristic is less precise for a large number of base models.

5.3. Memory Complexity

Across all benchmarks, we utilize the same or smaller base models as compared to established baselines [26] (as shown in Table 4). Only the parameters of the hypernetwork themselves are retained in memory, and the weights of the trained base models are discarded after training. Thus, the total parameter size stored in memory remains constant and equal to the parameters of the hypernetwork, which is $1.4\times$ (in case of MNIST) and equal or less than the parameters of a single base model in other benchmarks.

6. Related Literature

6.1. Neuroscience

Vital to the process of learning and 'memorization' in humans is the continual familiarity-based modification of input instances within the Hippocampus [1, 3, 23, 41]. Input patterns that are similar to a familiar/stored pattern are modified to either be more similar (pattern completion) or differentiated (pattern separation) to those stored patterns [23]. New memory is allocated if the input instance is sufficiently distinct from stored instances. In case there is high overlap between input instance and one of the stored instances, the input instance is modified to be closer to the matched stored instance.

6.2. Continual Learning

Over the last few years, several directions of work have attempted to address these issues of catastrophic forgetting and beneficial knowledge transfer in a continual learning setting. Methods retain past knowledge either by replaying stored [26, 30] or generated samples [36, 39], regularizing task-specific weights [4, 24, 37], or scaling parameters to account for new tasks [8, 27, 35]. While attempting to explicitly prevent the learning dynamics that cause the loss of task-specific knowledge, approaches have inevitably focused on modelling the transfer-interference trade-off between gradients of different tasks [7, 26, 33]. In [29], authors introduce the idea of building a growing zoo of small capacity multi-tasking models, where synergistic tasks share models, enabling transfer of knowledge between them.

6.3. Meta Learning Approaches for CL

Recently, there has been work applying meta-learning approaches successfully to the continual setting [9, 12, 19, 31]. Online-aware Meta Learning (OML) [9] introduced the application of Meta Learning approaches to the lifelong learning setting. A meta-objective was used to learn the task distribution in an offline manner, which could then be leveraged for efficient online continual learning. Meta-learning approaches to continual learning such as MER [31] and La-MAML [12] leverage gradient alignments to enforce compatibility of tasks within a finite capacity. MER [31] enforces gradient alignment between observed and future tasks using replay while La-MAML [13] incrementally modulates parameter-specific learning rates based on gradient alignment across tasks to reduce forgetting. MERLIN [19] introduced the idea of 'meta-consolidation' and replay in the meta space of parameters learned by the meta model across all tasks. Parameters are generated by the meta model trained on all observed base model parameters, and ensembled to make predictions.

7. Conclusion

In this work, we motivate the quality of adaptation to improve knowledge transfer in the lifelong learning setting. We propose to represent task-model relationships as the expected adaptation rate of a (model, task) pair. In order to leverage (model, task) relationships, we replace a single network used for lifelong learning with an equivalent set of small-capacity networks such that the overall model capacity is conserved. We show that the proposed approach can transfer knowledge to new tasks without any increase in overall model capacity, while naturally mitigating catastrophic forgetting.

References

[1] Pablo Alvarez and Larry R Squire. Memory consolidation and the medial temporal lobe: a simple network model. *Proceedings of the national academy of sciences*, 91(15):7041–7045, 1994. 1, 8

[2] Michael L Anderson. Neural reuse: A fundamental organizational principle of the brain. *Behavioral and brain sciences*, 33(4):245–266, 2010. 1

[3] Alfonso Caramazza and Jennifer R Shelton. Domain-specific knowledge systems in the brain: The animate-inanimate distinction. *Journal of cognitive neuroscience*, 10(1):1–34, 1998. 1, 8

[4] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 532–547, 2018. 5, 6, 8

[5] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*, 2018. 5, 6

[6] Marc N Coutanche and Sharon L Thompson-Schill. Rapid consolidation of new knowledge in adulthood via fast mapping. *Trends in cognitive sciences*, 19(9):486–488, 2015. 1

[7] Danruo Deng, Guangyong Chen, Jianye Hao, Qiong Wang, and Pheng-Ann Heng. Flattening sharpness for dynamic gradient projection memory benefits continual learning. *Advances in Neural Information Processing Systems*, 34:18710–18721, 2021. 8

[8] Tom Diethe, Tom Borchert, Eno Thereska, Borja Balle, and Neil Lawrence. Continual learning in practice. *arXiv preprint arXiv:1903.05202*, 2019. 8

[9] Chelsea Finn, Aravind Rajeswaran, Sham Kakade, and Sergey Levine. Online meta-learning. In *International Conference on Machine Learning*, pages 1920–1930. PMLR, 2019. 8

[10] Sebastian Flennerhag, Pablo G Moreno, Neil D Lawrence, and Andreas Damianou. Transferring knowledge across learning processes. *arXiv preprint arXiv:1812.01054*, 2018. 2

[11] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999. 2

[12] Gunshi Gupta, Karmesh Yadav, and Liam Paull. La-maml: Look-ahead meta learning for continual learning. *arXiv preprint arXiv:2007.13904*, 2020. 2, 6, 8

[13] Gunshi Gupta, Karmesh Yadav, and Liam Paull. Look-ahead meta learning for continual learning. *Advances in Neural Information Processing Systems*, 33:11588–11598, 2020. 8

[14] David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. *arXiv preprint arXiv:1609.09106*, 2016. 5

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 6

[16] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 7

[17] Wonyong Jeong, Hayeon Lee, Gun Park, Eunyoung Hyung, Jinheon Baek, and Sung Ju Hwang. Task-adaptive neural network search with meta-contrastive learning, 2021. 5

[18] Wonyong Jeong, Hayeon Lee, Geon Park, Eunyoung Hyung, Jinheon Baek, and Sung Ju Hwang. Task-adaptive neural network search with meta-contrastive learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 6

[19] KJ Joseph and Vineeth N Balasubramanian. Meta-consolidation for continual learning. *arXiv preprint arXiv:2010.00352*, 2020. 7, 8

[20] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[21] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017. 2, 5, 6

[22] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Technical Report*, 2009. 5

[23] Dharshan Kumaran, Demis Hassabis, and James L McClelland. What learning systems do intelligent agents need? complementary learning systems theory updated. *Trends in cognitive sciences*, 20(7):512–534, 2016. 8

[24] Richard Kurle, Botond Cseke, Alexej Klushyn, Patrick van der Smagt, and Stephan Günnemann. Continual learning with bayesian neural networks for non-stationary data. In *International Conference on Learning Representations*, 2019. 8

[25] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998. 5

[26] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in neural information processing systems*, pages 6467–6476, 2017. 2, 5, 6, 8

[27] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative prun-

ing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7765–7773, 2018. 8

[28] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989. 2

[29] Rahul Ramesh and Pratik Chaudhari. Model zoo: A growing brain that learns continually. In *International Conference on Learning Representations*, 2021. 2, 8

[30] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017. 2, 5, 6, 7, 8

[31] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. *arXiv preprint arXiv:1810.11910*, 2018. 2, 5, 6, 7, 8

[32] Victoria JH Ritvo, Nicholas B Turk-Browne, and Kenneth A Norman. Nonmonotonic plasticity: how memory retrieval drives learning. *Trends in cognitive sciences*, 23(9):726–742, 2019. 1

[33] Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. *arXiv preprint arXiv:2103.09762*, 2021. 8

[34] Jonathan Schwarz, Jelena Luketina, Wojciech M Czarnecki, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. *arXiv preprint arXiv:1805.06370*, 2018. 5

[35] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. *arXiv preprint arXiv:1801.01423*, 2018. 8

[36] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems*, pages 2990–2999, 2017. 8

[37] Michalis K Titsias, Jonathan Schwarz, Alexander G de G Matthews, Razvan Pascanu, and Yee Whye Teh. Functional regularisation for continual learning with gaussian processes. *arXiv preprint arXiv:1901.11356*, 2019. 8

[38] Tyler M Tomita, Morgan D Barense, and Christopher J Honey. The similarity structure of real-world memories. *bioRxiv*, 2021. 1

[39] Gido M van de Ven and Andreas S Tolias. Generative replay with feedback connections as a general strategy for continual learning. *arXiv preprint arXiv:1809.10635*, 2018. 8

[40] Johannes Von Oswald, Christian Henning, João Sacramento, and Benjamin F Grewe. Continual learning with hypernetworks. *arXiv preprint arXiv:1906.00695*, 2019. 5, 6

[41] Matthew A Wilson and Bruce L McNaughton. Reactivation of hippocampal ensemble memories during sleep. *Science*, 265(5172):676–679, 1994. 1, 8

[42] David H Wolpert and William G Macready. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82, 1997. 1

[43] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. *Proceedings of machine learning research*, 70:3987, 2017. 2, 5