Machine Learning (2023) 112:655–686
https://doi.org/10.1007/s10994-022-06283-9

Hierarchically structured task-agnostic continual learning

Heinke Hihn[1] · **Daniel A. Braun**[1]

Received: 17 February 2022 / Revised: 1 November 2022 / Accepted: 12 November 2022 /
Published online: 28 December 2022

Abstract
One notable weakness of current machine learning algorithms is the poor ability of models to solve new problems without forgetting previously acquired knowledge. The Continual Learning paradigm has emerged as a protocol to systematically investigate settings where the model sequentially observes samples generated by a series of tasks. In this work, we take a task-agnostic view of continual learning and develop a hierarchical information-theoretic optimality principle that facilitates a trade-off between learning and forgetting. We derive this principle from a Bayesian perspective and show its connections to previous approaches to continual learning. Based on this principle, we propose a neural network layer, called the Mixture-of-Variational-Experts layer, that alleviates forgetting by creating a set of information processing paths through the network which is governed by a gating policy. Equipped with a diverse and specialized set of parameters, each path can be regarded as a distinct sub-network that learns to solve tasks. To improve expert allocation, we introduce diversity objectives, which we evaluate in additional ablation studies. Importantly, our approach can operate in a task-agnostic way, i.e., it does not require task-specific knowledge, as is the case with many existing continual learning algorithms. Due to the general formulation based on generic utility functions, we can apply this optimality principle to a large variety of learning problems, including supervised learning, reinforcement learning, and generative modeling. We demonstrate the competitive performance of our method on continual reinforcement learning and variants of the MNIST, CIFAR-10, and CIFAR-100 datasets.

Keywords Continual learning · Mixture-of-experts · Variational Bayes · Information theory

Editor: Tong Zhang.

✉ Heinke Hihn
heinke.hihn@uni-ulm.de

Daniel A. Braun
daniel.braun@uni-ulm.de

[1] Institute of Neural Information Processing, Ulm University, Ulm, Germany

1 Introduction

Acquiring new skills and concepts without forgetting previously acquired knowledge is a hallmark of human and animal intelligence. Biological learning systems leverage task-relevant knowledge from preceding learning episodes to guide subsequent learning of new tasks to accomplish this. Artificial learning systems, such as neural networks, usually lack this crucial property and experience a problem coined "catastrophic forgetting" (McCloskey & Cohen, 1989). Catastrophic forgetting occurs when we naively apply machine learning algorithms to solve a sequence of tasks $T_{1:t}$, where the adaptation to task T_t prompts overwriting of the parameters learned for tasks $T_{1:t-1}$.

The Continual Learning (CL) paradigm (Thrun, 1998) has emerged as a way to investigate such problems systematically. We can divide CL approaches into four broad categories: generative approaches with memory consolidation, regularization, architecture and expansion methods, and algorithm-based methods. Generative methods train a generative model to learn the data-generating distribution to reproduce data of old tasks. Data sampled from the learned model is then part of the training process (Shin et al., 2017; Rebuffi et al., 2017). This strategy draws inspiration from neuroscience research regarding the reactivation of neuronal activity patterns representing previous experiences that are hypothesized to be vital for stabilizing new memories while retaining old memories (Wilson & McNaughton, 1994; Rasch & Born, 2007; van de Ven et al., 2016). In contrast, regularization methods (Kirkpatrick et al., 2017; Zenke et al., 2017; Ahn et al., 2019; Benavides-Prado et al., 2020; Han & Guo, 2021) introduce an additional constraint to the learning objective. The goal is to prevent changes in task-relevant parameters, where we may measure relevance as, e.g., the performance on previously seen tasks (Kirkpatrick et al., 2017; Zenke et al., 2017; Li et al., 2021; Cha et al., 2020). This approach can also be motivated through synaptic plasticity and elasticity changes of biological neurons when learning new tasks (Ostapenko et al., 2019). CL can also be achieved by modifying the design of a model during learning (Lin et al., 2019; Fernando et al., 2017; Rusu et al., 2016; Yoon et al., 2018; Golkar et al., 2019). Such methods include adding new layers to a neural network (Zacarias & Alexandre, 2018), re-routing data through a neural network based on task information (Collier et al., 2020), distilling parameters (Zhai et al., 2019; Liu et al., 2020), and adding new experts to a mixture-of-experts architecture (Lee et al., 2020). Lastly, algorithmic approaches aim to adapt the optimization algorithm itself to avoid catastrophic forgetting, e.g., by mapping the gradient updates into a different space (Zeng et al., 2019; Wang et al., 2021). Some methods provide a combination of approaches, as it seems plausible that a mixture of approaches will provide the best-performing systems, as these methods are often complementary (Biesialska et al., 2020; De Lange et al., 2021; Vijayan & Sridhar, 2021).

While there has been significant progress in the field of CL, there are still some major open questions (Parisi et al., 2019). For example, most existing algorithms share a significant drawback in that they require task-specific knowledge, such as the number of tasks and which task is currently at hand (Zenke et al., 2017; Shin et al., 2017; Nguyen et al., 2017; Kirkpatrick et al., 2017; Li & Hoiem, 2017; Rao et al., 2019; Sokar et al., 2021; Yoon et al., 2018; Han & Guo, 2021; Chaudhry et al., 2021, 2018). One prominent class of CL approaches sharing this drawback are multi-head approaches (El Khatib & Karray, 2019; Nguyen et al., 2017; Ahn et al., 2019), which build a set of shared layers but a separate output layer ("head") per task, deterministically activated by the current task index (see Fig. 1).

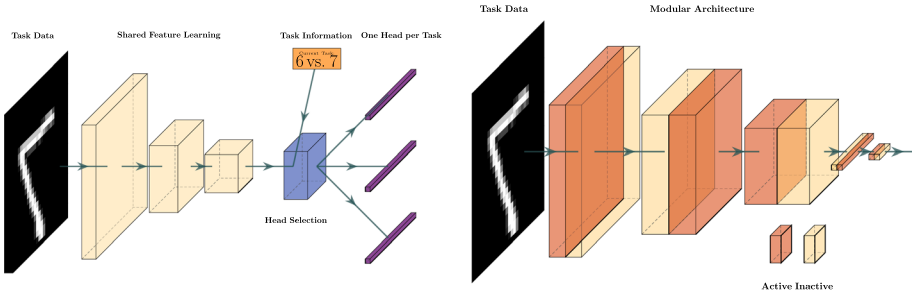

Fig. 1 Left: Multi-head architectures share a feature extractor block and select task-specific output heads based on the knowledge provided by a task oracle. Right: Our proposed architecture for single head CL. Each layer implements a modular system, such that task data can flow through distinct "paths" (highlighted) in the network

Extracting relevant task information is in general a difficult problem, in particular when distinguishing tasks without any contextual input (Hihn & Braun, 2020; Yao et al., 2019). Thus, providing the model with such task-relevant information yields overly optimistic results (Chaudhry et al., 2018). In order to deal with more realistic and challenging CL scenarios, therefore, models must learn to compensate for the lack of auxiliary information. The approach we propose in this work tackles this problem by formulating a hierarchical learning system, that allows us to learn a set of sub-modules specialized in solving particular tasks. To this end, we introduce hierarchical variational continual learning (HVCL) and devise the mixture-of-variational-experts layer (MoVE layers) as an instantiation of HVCL. MoVE layers consist of M experts governed by a gating policy, where each expert maintains a posterior distribution over its parameters alongside a corresponding prior. During each forward pass, the gating policy selects one expert per layer. This sparse selection reduces computation as only a small subset of the parameters must be updated during the back-propagation of the loss (Shazeer et al., 2017). To mitigate catastrophic forgetting we condition the prior distributions on previously observed tasks and add a penalty term on the Kullback-Leibler-Divergence between the expert posterior and its prior. This constraint facilitates a trade-off between learning and forgetting and allows us to design information-efficient decision-makers (Hihn & Braun, 2020).

When dealing with ensemble methods, two main questions arise naturally. The first one concerns the question of optimally selecting ensemble members using appropriate selection and fusion strategies (Kuncheva, 2004). The second one, is the question of how to ensure expert diversity (Kuncheva & Whitaker, 2003; Bian & Chen, 2021). We argue that ensemble diversity benefits continual learning and investigate two complementary diversity objectives: the entropy of the expert selection process and a similarity measure between different experts based on Wasserstein exponential kernels in the context of determinantal point processes (Kulesza et al., 2012). By maximizing the determinant of the expert similarity matrix, we can then "spread" the expert parameters optimally within the shared parameter space. To summarize, our contributions are the following: (1) we extend variational continual learning (Nguyen et al., 2017) to a hierarchical multi-prior setting, (2) we derive a computationally efficient method for task-agnostic continual learning from this general formulation, (3) to improve expert specialization and diversity, we introduce and evaluate novel diversity measures (4) we demonstrate our approach in supervised CL, generative CL, and continual reinforcement learning.

This paper is structured as follows: after introducing our method in Sect. 2, we design, perform, and evaluate the main experiments in Sect. 3. In Sect. 4, we discuss novel aspects of the current study in the context of previous literature and conclude with a final summary in Sect. 5.

2 Hierarchical variational continual learning

In this section we first introduce the concept of continual learning and related nomenclature formally and then extend the variational continual learning (VCL) setting introduced by Nguyen et al. (2017) to a hierarchical multi-prior setting and then introduce a neural network implementation as a generalized application of this paradigm in Sect. 2.1.

In CL the goal is to minimize the loss of all seen tasks given no (or limited) access to data from previous tasks:

$$\min \sum_{t=1}^{T} \mathbb{E}_{(x_t,y_t)\sim(X_t,Y_t)} \left[\ell(f_\theta(x_t, y_t)) \right], \tag{1}$$

where T is the total number of tasks, $\mathcal{D}_t = \{x_t^i, y_t^i\}_{i=1}^{N_t} = (X_t, Y_t)$ the dataset of task t, ℓ some loss function, and f_θ a *single* predictor parameterized by θ (e.g., a neural network). Here, task refers to an isolated training stage with a new dataset. This dataset may belong to a new set of classes, a new domain, or a new output space. We can divide these concepts based on the in- and output distributions (De Lange et al., 2021): in task-incremental learning the labels change $Y_t \neq Y_{t+1}$, but the label distributions remain $P(Y_t) = P(Y_{t+1})$, e.g., in a series of binary classification tasks, plus the task t is always known. In domain-incremental learning we have $Y_t = Y_{t+1}$ and $P(Y_t) = P(Y_{t+1})$, but t is unknown. Finally, in class-incremental learning we have $Y_t \subset Y_{t+1}$, but $P(Y_t) \neq P(Y_{t+1})$ and unknown t, e.g., MNIST with ten total classes but only two classes per task. For all settings it holds that $X_t \neq X_{t+1}$—see Fig. 2 for an illustration.

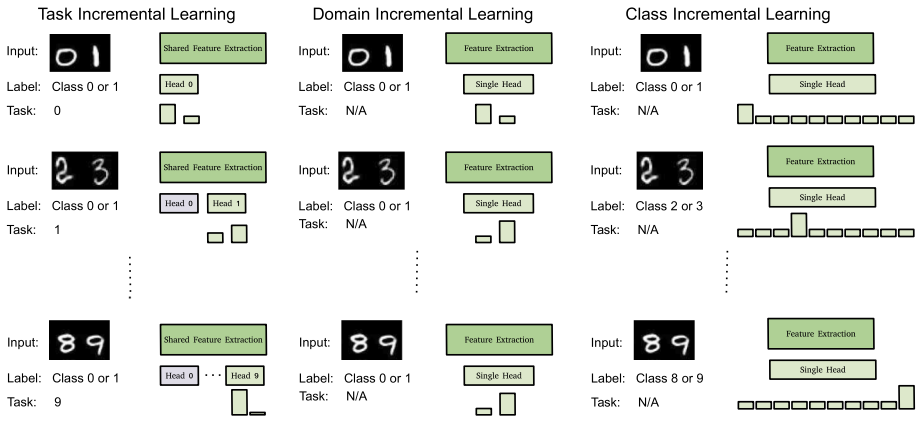

Fig. 2 Continual Learning setups. In Task Incremental Learning the number of classes is constant and task information is provided. In Domain Incremental Learning, the number of classes is also constant, but no task information is given. In Class Incremental Learning the total number of classes is constant, but the learning agent only observes a subset at a time, e.g., two out of ten total classes per task

The variational continual learning approach (Nguyen et al., 2017) describes a general learning paradigm wherein an agent stays close to an old strategy ("prior") it has learned on a previous task T_{t-1} while learning to solve a new task T_t ("posterior"). Given datasets of input-output pairs $\mathcal{D}_t = \{x_t^i, y_t^i\}_{i=0}^{N_t}$ of tasks $t \in \{1, ..., T\}$, the main learning objective of minimizing the log-likelihood $\log p_\theta(y_t^i|x_t^i)$ for task t is augmented with an additional loss term in the following way:

$$\mathcal{L}_{\text{VCL}}^t = \sum_{i=1}^{N_t} \mathbb{E}_\theta \left[\log p_\theta(y_t^i|x_t^i)\right] - \text{D}_{\text{KL}} \left[p_t(\theta)||p_{t-1}(\theta)\right], \tag{2}$$

where $p(\theta)$ is a distribution over the models parameter θ and N_t is the number of samples for task t. The prior constraint encourages the agent to find an optimal trade-off between solving a new task and retaining knowledge about old tasks. When the likelihood model is implemented by a neural network, a new output layer can be associated with each incoming task, resulting in a multi-head implementation. Over the course of T datasets, Bayes' rule then recovers the posterior

$$\begin{aligned} p(\theta|\mathcal{D}_{1:T}) &\propto p(\theta) \prod_{t=1}^{T} \prod_{i=1}^{N_t} p(y_t^i|\theta, x_t^i) \\ &= p(\theta) \prod_{t=1}^{T} p(\mathcal{D}_t|\theta) \\ &\propto p(\theta|\mathcal{D}_{1:T-1}) p(\mathcal{D}_T|\theta), \end{aligned} \tag{3}$$

which forms a recursion: the posterior after seeing T datasets is obtained by multiplying the posterior after $T-1$ with the likelihood and normalizing accordingly.

In their original work, the authors propose to use multi-headed networks, i.e., to train a new output layer for each incoming task. This strategy has two main drawbacks: (1) it introduces an organizational overhead due to the growing number of network heads, and (2) task boundaries must be known at all times, making it unsuitable for more complex continual learning settings. In the following we argue that we can alleviate these problems by combining multiple decision-makers with a learned selection policy to replace the deterministic head selection.

To extend VCL to the hierarchical case, we assume that samples are drawn from a set of M independent data generating processes, i.e. the likelihood is given by a mixture model $p(y|x) = \sum_{m=1}^{M} p(m|x)p(y|m, x)$. We define an indicator variable $z \in Z$, where $z_m^{i,t}$ is 1 if the output y_i^t of sample i from task t was generated by expert m and zero otherwise. The conditional probability of an output is then given by

$$p(y_t^i|x_t^i, \Theta) = \sum_{m=1}^{M} p(z_t^{i,m}|x_t^i, \vartheta) p(y_t^i|x_t^i, \omega_m), \tag{4}$$

where ϑ are the parameters of the selection policy, ω_m the parameters of the m-th expert, and $\Theta = \{\vartheta, \{\omega_m\}_{m=1}^M\}$ the combined model parameters. The posterior after observing T tasks is then given by

$$
\begin{aligned}
p(\Theta|\mathcal{D}_{1:T}) &\propto p(\vartheta)p(\omega)\prod_{t=1}^{T}\prod_{i=1}^{N_t}\sum_{m=1}^{M} p(z_t^{i,m}|x_t^i,\vartheta)p(y_t^i|x_t^i,\omega_m) \\
&= p(\Theta)\prod_{t=1}^{T} p(\mathcal{D}_t|\Theta) \\
&\propto p(\Theta|\mathcal{D}_{1:T-1})p(\mathcal{D}_T|\Theta).
\end{aligned} \tag{5}
$$

The Bayes posterior of an expert $p(\omega_m|\mathcal{D}_{1:T})$ is recovered by computing the marginal over the selection variables Z. Again, this forms a recursion, in which the posterior $p(\Theta|\mathcal{D}_{1:T})$ depends on the posterior after seeing $T-1$ tasks and the likelihood $p(\mathcal{D}_T|\Theta)$. We can now formulate the hierarchical variational continual learning objective for task t as minimizing the following loss:

$$
\begin{aligned}
\mathcal{L}_{\text{HVCL}}^t = \sum_{i=1}^{N_t} &\mathbb{E}_{p(\Theta)}\left[\log p(y_t^i|x_t^i,\Theta)\right] \\
&- \mathrm{D}_{\text{KL}}\left[p_t(\vartheta)||p_{1:t-1}(\vartheta)\right] \\
&- \mathrm{D}_{\text{KL}}\left[p_t(\omega)||p_{1:t-1}(\omega)\right],
\end{aligned} \tag{6}
$$

where N_t is the number of samples in task t, and the likelihood $p(y|x,\Theta)$ is defined as in Eq. (4). The Mixture-of-Variational-Experts layers we introduce in Sect. 2.1 are based on a generalization of this optimization problem.

2.1 Sparsely gated mixture-of-variational layers

As we plan to tackle not only supervised learning problems, but also reinforcement learning problems, we assume in the following a generic scalar utility function $\mathbf{U}(x, f_\theta(x))$ that depends both on the input x and the parameterized agent function $f_\theta(x)$ that generates the agent's output y. We assume that the agent function $f_\theta(x)$ is composed of multiple layers as depicted in Fig. 3. Our layer design builds on the sparsely gated Mixture-of-Expert (MoE) layers (Shazeer et al., 2017), which in turn draws on the Mixture-of-Experts paradigm introduced by Jacobs et al. (1991). MoEs consist of a set of M experts indexed by m and a gating network $p(m|x)$ whose output is a (sparse) M-dimensional vector. All experts have an identical architecture but separate parameters. Let $p(m|x)$ be the gating output and $p(y|m,x)$ the response of an expert m given input x. The layer's output is then given by a weighted sum of the experts responses, i.e., $p(y|x) = \sum_{m\in M} p(m|x)p(y|m,x)$. To save computation time we employ a top-k gating scheme, where only the k experts with highest gating activation are evaluated and use an additional penalty that encourages gating sparsity (see Sect. 2.1). In all our experiments we set $k=1$, to drive expert specialization (see Sect. 2.2.1) and reduce computation time. We implement the learning objective for task t as layer-wise regularization in the following way:

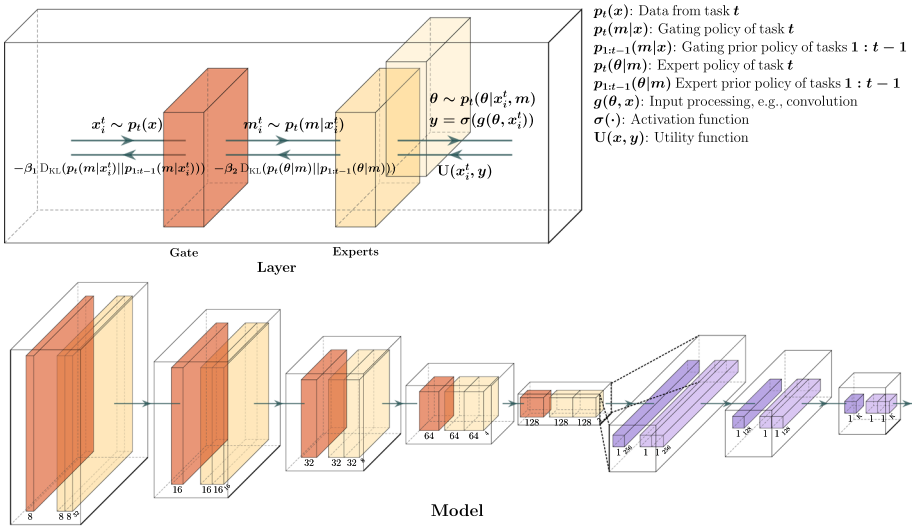

Fig. 3 This figure illustrates our proposed design. Each layer implements a top-k expert selection conditioned on the output of the previous layer. Each expert m maintains a distribution over its weights $p(\theta|m) = \mathcal{N}(\mu_m, \sigma_m)$ and a set of bias variables b_m. Left to right arrows represent sampling, while right to left arrows show the resulting utility and losses. Combining L layers with M experts gives L^M possible paths through the network. The architecture shown here is used in the CIFAR10 experiments

$$\begin{aligned}\mathcal{L}^t_{\text{MoVE}} = \sum_{i=1}^{N_t} \Bigg[& \mathbb{E}_{\Theta}\left[\mathbf{U}(x_i^t, f_\Theta(x_i^t))\right] \\ & - \sum_{l=1}^{L} \mathbb{E}_{p^l(m|x_i^t), q^l(\theta|m)} \Big[\beta_1 \, \mathrm{D_{KL}}\left[p_t^l(m|x_i^t)||p^l_{1:t-1}(m|x_i^t)\right] \\ & + \beta_2 \, \mathrm{D_{KL}}\left[p_t^l(\theta|m)||p^l_{1:t-1}(\theta|m)\right] \Big] \Bigg],\end{aligned} \tag{7}$$

where L is the total number of layers, $\Theta = \{\theta, \{\vartheta_m\}_{m=1}^M\}$ the combined parameters,and the temperature parameters β_1 and β_2 govern the layer-wise trade-off between utility (e.g., classification performance) and information-cost.

Thus, we allow for two major generalizations compared to Eq. 6: in lieu of the log-likelihood we allow for generic utility functions $\mathbf{U}(\cdot)$, and instead of applying the constraint on the gating parameters, we apply it directly on the gating output distribution $p(m|x)$. This implies, that the weights of the gating policy are not sampled. Otherwise the gating mechanism would involve two stochastic steps: one in sampling the weights and a second one in sampling the experts. This potentially high selection variance hinders expert specialization (see Sect. 2.2.1). Encouraging the gating policy to stay close to its prior also ensures that similar inputs are assigned to the same expert. Next we consider how we could extend objective 7 further by additional terms that encourage diversity between different experts.

2.2 Encouraging expert diversity

In the following, we argue that a diverse set of experts may mitigate catastrophic forgetting in continual learning, as experts specialize more easily in different tasks, which improves expert selection. Diversity measures enjoy an increasing interest in the reinforcement learning community but remain mainly understudied in continual learning (e.g., Bang et al., 2021). In the reinforcement learning literature diversity has been considered, for example, by encouraging skills or policies that are sufficiently different (Eysenbach et al., 2018; Parker-Holder et al., 2020), or by sampling trajectories that reflect goal diversity (Dai et al., 2021), which is an idea similar to well-known bagging techniques (Breiman, 1996). Moreover, diversity may arise from a sufficiently high variance weight initialization, but this can introduce computational instabilities during back-propagation, as we lose the variance reducing benefits of state-of-the-art initialization schemes (Glorot et al., 2010; He et al., 2015; Narkhede et al., 2021). Also, there is no guarantee that the expert parameters won't collapse again during training.

In the following, we present two expert diversity objectives. The first one arises directly from the main learning objective and is designed to act as a regularizer on the gating policy while the second one is a more sophisticated approach that aims for diversity in the expert parameter space. The latter formulation introduces a new class of diversity measures, as we discuss in more detail in Sect. 4.1. We designed additional experiments in Sect. 3.4 to investigate their influence on learning and the resulting policies and to further motivate the need for expert diversity.

2.2.1 Diversity through specialization

The relationship between objectives of the form described by Eq. (7) with the emergence of expert specialization has been previously investigated for simple learning problems (Genewein et al., 2015) and in the context of meta-learning (Hihn & Braun, 2020), but not in the context of continual learning. This class of models assumes a two-level hierarchical system of specialized decision-makers where first level decision-makers $p(m|x)$ select which second level decision-maker $p(y|m,x)$ serves as experts for a particular input x. By co-optimizing

$$\max_{p(y|x,m),p(m|x)} \mathbb{E}\left[\mathbf{U}(x,y)\right] - \beta_1 I(X;M) - \beta_2 I(X;Y|M), \tag{8}$$

the combined system finds an optimal partitioning of the input space X, where $I(\cdot|\cdot)$ denotes the (conditional) mutual information between random variables. In fact, the hierarchical VCL objective given by Eq. (6) can be regarded as a special case of the information-theoretic objective given by Eq. (8), if we interpret the prior as the learning strategy of task $t-1$ and the posterior as the strategy of task t, and set $\beta = 1$. All these hierarchical decision systems correspond to a multi prior setting, where different priors associated with different experts can specialize on different sub-regions of the *input space*. In contrast, specialization in the context of continual learning can be regarded as the ability of partitioning the *task space*, where each expert decision-maker m solves a subset of old tasks $T^m \subseteq T_{1:t}$. In both cases, expert diversity is a natural consequence of specialization if the gating policy $p(m|x)$ partitions between the experts. Using gradient descent on parameterized distributions, objective (8) can also be maximized in an online manner (Hihn & Braun, 2020).

In addition to the implicit pressures for specialization already implied by Eq. 7, here we investigate the effect of an additional entropy cost. Inspired by recent entropy regularization

techniques (Eysenbach et al., 2018; Galashov et al., 2019; Grau-Moya et al., 2019), we aim to improve the gating policy by introducing the entropy cost

$$\max_{p(m|x)} H(M|X) - H(M), \tag{9}$$

where M is the set of experts and X the inputs. By maximizing the conditional entropy $H(M|X) = -\sum_{m,x} p(x)p(m|x)\log p(m|x)$ we encourage high certainty in the expert gating and by minimizing the marginal entropy $H(M)$ we prefer solutions that minimize the number of active experts. In our implementation, we compute these values batch-wise, as the full entropies are not tractable. We evaluate these entropy penalties in Sect. 3.4.

2.2.2 Parameter diversity

Our second diversity formulation is based on differences in the parameter space, rather than in the output space as formalized by Eqs. (9) or (8). To find expert parameters that are pairwise different, we introduce a symmetric distance measure and we show how this measure can be efficiently computed and maximized. By maximizing distance in parameter space, we hope to find a set of expert parameters stretched over the space of possible parameters. This in turn helps to prevent collapsing to a state where all experts have similar parameters (and thus similar outputs), rendering the idea behind an ensemble useless.

Our idea builds on determinantal point processes (DPPs) (Kulesza et al., 2012), a mechanism that produces diverse subsets by sampling proportionally to the determinant of the kernel matrix of points within the subset (Macchi, 1975). A point process P on a ground set Y is a probability measure over finite subsets of Y. A sample from P may be the empty set, the entirety of Y, or anything in between. P is a determinantal point process if, when Y is a random subset drawn according to P, we have, for every$A \subset Y$, $P(A \subset Y) = \det(K_A)$ for some real, symmetric $N \times N$ matrix K indexed by the elements of Y. Here, $K_A = [K_{ij}]_{i,j\in A}$ denotes the restriction of K to the entries indexed by elements of A, and we adopt $\det(K_\emptyset) = 1$. Since P is a probability measure, all principal minors $\det(KA)$ of K must be non-negative, and thus K itself must be positive semi-definite. These requirements turn out to be sufficient: any K, $0 \leq K \leq I$, defines a DPP. We refer to K as the marginal kernel since it contains all the information needed to compute the probability of any subset A of Y. If $A = \{i\}$ is a singleton, then we have $P(i \in Y) = K_{i,i}$. In this case, the diagonal of K gives the marginal probabilities of inclusion for individual elements of Y. Diagonal entries close to 1 correspond to elements of Y selected with high probability. The matrix K is defined by a kernel function $k(x_0, x_1)$. A kernel is a two-argument real-valued function over $\mathcal{X} \times \mathcal{X}$ such that for any $x_0, x_1 \in \mathcal{X}$:

$$k(x_0, x_1) = \langle \phi(x_0), \phi(x_1) \rangle_{\mathcal{F}}, \tag{10}$$

where $\mathcal{X}$ is a vector space and $\mathcal{F}$ is a inner-product space such that $\forall x \in \mathcal{X} : \phi(x) \in \mathcal{F}$. Specifically, we use a exponential kernel based on the Wasserstein-2 distance $W(p, q)$ between two probability distributions p and q. The p^{th} Wasserstein distance between two probability measures p and q in $P_p(M)$ is defined as

$$W_p(p, q) := \left(\inf_{\gamma \in \Gamma(p,q)} \int_{M \times M} d(x, y)^p \, \mathrm{d}\gamma(x, y) \right)^{1/p}, \tag{11}$$

where $\Gamma(p, q)$ denotes the collection of all measures on $M \times M$ with marginals p and q on the first and second factors. Let p and q be two isotropic Gaussian distributions and

$W_2^2(q,p)$ the Wasserstein-2 distance between p and q. The exponential Wasserstein-2 kernel is then defined by

$$k(p,q) = \exp\left(-\frac{W_2^2(p,q)}{2h^2}\right), \tag{12}$$

where h is the kernel width. We show in Appendix B that Eq. (12) gives a valid kernel. This formulation has two properties that make it suitable for our purpose. Firstly, the Wasserstein distance is symmetric, i.e., $W_2^2(p,q) = W_2^2(q,p)$, which in turn will lead to a symmetric kernel matrix. This is not true for other similarity measures on probability distributions, such as D_{KL} (Cover & Thomas, 2012). Secondly, if p and q are Gaussian and mean-field approximations, i.e., covariance matrices Σ_p and Σ_q are given by diagonal matrices, such that $\Sigma_p = \text{diag}(d_p)$ and $\Sigma_q = \text{diag}(d_q)$, $W_2^2(p,q)$ can be computed in closed form as

$$W_2^2(p,q) = ||\mu_p - \mu_q||_2^2 + ||\sqrt{d_p} - \sqrt{d_q}||_2, \tag{13}$$

where $\mu_{p,q}$ are the means and d p, q the diagonal entries of distributions p and q. We provide a more detailed derivation of Eq. (13) in Appendix A. For each layer l with N experts the following regularization objective is added to the main objective:

$$\max \sum_{l=1}^{L} \det(K_l) = \sum_{l=1}^{L} \det \begin{pmatrix} 1 & k(p_1^l, p_2^l) & \cdots & k(p_1^l, p_N^l) \\ k(p_2^l, p_1^l) & 1 & \cdots & k(p_2^l, p_N^l) \\ \vdots & \vdots & \ddots & \vdots \\ k(p_N^l, p_0^l) & k(p_N^l, p_2^l) & \cdots & 1 \end{pmatrix} \tag{14}$$

where L is the number of Layers, $k(p_i^l, p_j^l)$ is the kernel of the i-th and j-th expert of layer l (see Eq. 10) and $\det(K)$ denotes the matrix determinant of the kernel matrix K. Note that the matrix is symmetric, which reduces computation time. Computing $\det(K)$ can have some pitfalls, which we discuss in Sect. 4.2.

From a geometric perspective, the determinant of the kernel matrix represents the volume of a parallelepiped spanned by feature maps corresponding to the kernel choice. We seek to maximize this volume, effectively filling the parameter space—see Fig. 4 for an illustration.

3 Experiments

While the correct and robust evaluation of continual learning algorithms is still a topic of discussion (Farquhar & Gal, 2018; Hsu et al., 2018), we follow the majority of studies to ensure a fair comparison. We evaluate our approach in current supervised learning benchmarks in Sect. 3.1, in a generative learning setting in Sect. 3.2, and in the continual reinforcement learning setup in Sect. 3.3. Additionally, we conduct ablation studies in Sect. 3.4 to investigate the influence of the diversity objective, the generator quality, the influence of the hyper-parameters β_1 and β_2, and regarding the number of experts. We give experimental details in Appendix C.

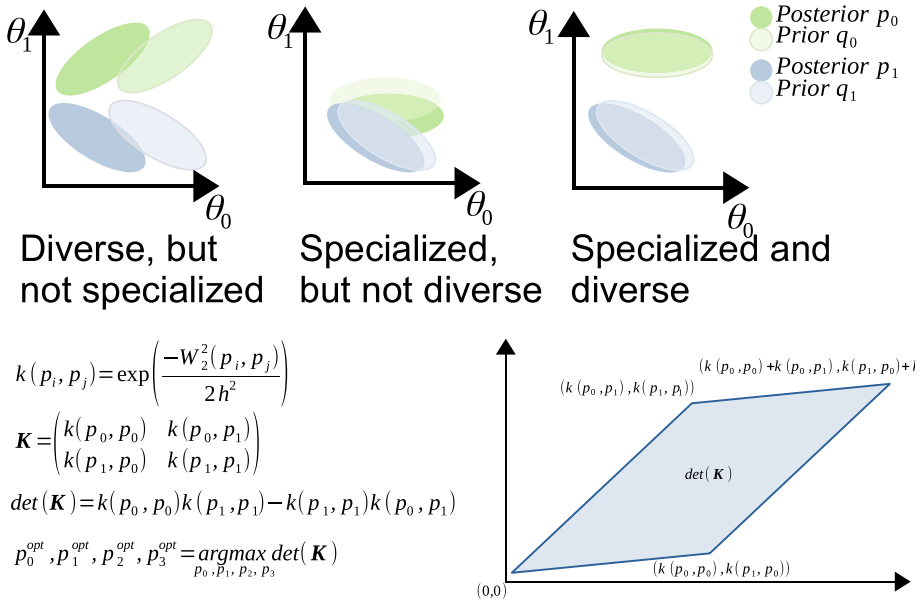

Fig. 4 Upper Row: We seek experts that are both specialized, i.e., their posterior p is close to their prior q, and diverse, i.e., posteriors $p_i, p_j \forall i \neq j$ are sufficiently distant from one another. Bottom Row: To this effect, we maximize the determinant of the kernel matrix K, effectively filling the feature space. In the case of two experts this would mean to maximize $det(\mathbf{K}) = 1 - K(p_0, p_1)$, which we can achieve by maximizing the Wasserstein-2 distance between the posteriors p_0 and p_1

3.1 Continual supervised learning scenarios

The basic setting of continual learning is defined as an agent which sequentially observes data from a series of tasks $\mathcal{T} = \{T_i\}_{i=1}^N$ and must learn T_i while maintaining performance on older tasks $T_{\leq i}$. We evaluate the performance of our method in this setting in split MNIST (Fig. 1), split CIFAR-10 and split CIFAR-100 (Fig. 2). We follow the domain incremental setup (van de Ven et al., 2020; Kessler et al., 2021; Raghavan & Balaprakash, 2021; Hsu et al., 2018; Mazur et al., 2021; He & Zhu, 2022), where the number of classes is constant and task information is not available, but we also compare against task-incremental methods (Zenke et al., 2017; Shin et al., 2017; Nguyen et al., 2017; Kirkpatrick et al., 2017; Li & Hoiem, 2017; Rao et al., 2019; Sokar et al., 2021; Yoon et al., 2018; Han & Guo, 2021; Chaudhry et al., 2021, 2018) in Appendix C Tables 3 and 4, where the task information is available, to give a complete overview of current methods. The different Continual Learning setups are given in Fig. 2 in more detail.

The first benchmark builds on the MNIST dataset. Five binary classification tasks from the MNIST dataset arrive in sequence: 0/1, 2/3, 4/5, 6/7, and 8/9. In time step t, the performance is measured as the average classification accuracy on all tasks up to task t. In permuted MNIST the task received at each time step t consists of labeled MNIST images whose pixels have undergone a fixed random permutation. The second benchmark is a variation of the CIFAR-10/100 datasets. In Split CIFAR-10, we divide the ten classes into five binary classification tasks. In total, CIFAR-10 consists of 60000 $32 \times 32 \times 3$ images in 10 classes, with 6000 images per class. There are 50,000 training images and 10,000 test

Table 1 Continual learning results in the split MNIST (S-MNIST) and permuted MNIST (P-MNIST) benchmark compared to current CL methods

Baselines	S-MNIST	P-MNIST
Dense neural network	86.15 (± 1.00)	17.26 (± 0.19)
Offline re-training	99.64 (± 0.03)	97.59 (± 0.02)
Single-head and task-agnostic methods		
Hierarchical VCL (ours)	97.50 (± 0.33)	97.07 (± 0.62)
Hierarchical VCL w/GR (ours)	98.60 (± 0.35)	97.47 (± 0.52)
Uncertainty guided CL w/BNN (Ebrahimi et al., 2020)	97.70 (± 0.03)	92.50 (± 0.01)
Brain-inspired replay through feedback† (van de Ven et al., 2020)	99.66 (± 0.13)	97.31 (± 0.04)
Hierarchical Indian buffet neural nets (Kessler et al., 2021)	91.00 (± 2.20)	93.70 (± 0.60)
Balanced continual learning (Raghavan & Balaprakash, 2021)	98.71 (± 0.06)	97.51 (± 0.05)
Target layer regularization (Mazur et al., 2021)	80.64 (± 1.25)	

Results were averaged over ten random seeds with the standard deviation given in parenthesis. Results on algorithms marked with † were taken from (van de Ven & Tolias, 2018), others from their original work. "Dense Neural Network" refers to simple NN, that has been trained naively with sequential data and represents a lower bound. "Offline re-training" refers to a NN that has been retrained on all tasks seen so far

Table 2 Continual learning results in the split CIFAR-10 and the split CIFAR-100 benchmark compared to current CL methods

Baselines	Split-CIFAR-10	CIFAR-100
Conv. neural network	66.62 (± 1.06)	19.80 (± 0.19)
Offline re-training	80.42 (± 0.95)	52.30 (± 0.02)
Single-head and task-agnostic methods		
Hierarchical VCL (ours)	78.41 (± 1.18)	33.10 (± 0.62)
Hierarchical VCL w/GR (ours)	81.00 (± 1.15)	37.20 (± 0.52)
Continual learning with dual regularizations (Han & Guo, 2021)	86.72 (± 0.30)	25.62 (± 0.22)
Natural continual learning (Kao et al., 2021)		38.79 (± 0.24)
Target layer regularization (Mazur et al., 2021)	74.89 (± 0.61	
Memory aware synapses (He & Zhu, 2022)	73.50 (± 1.54)	

Results were averaged over ten random seeds with the standard deviation given in the parenthesis. We report results of other methods as given in their original studies. "Conv. Neural Network" refers to simple CNN, that has been trained naively with sequential data and represents a lower bound. "Offline re-training" refers to a CNN that has been retrained on all tasks seen so far

images. CIFAR-100 is like the CIFAR-10, except it has 100 classes containing 600 images each. There are 500 training images and 100 testing images per class. Tasks are defined as a 10-way classification problem, thus forming ten tasks in total.

We achieve comparable results to current state-of-the-art approaches (see Tables 1 and 2) on all three supervised learning benchmarks.

3.2 Generative continual learning

Generative CL is a simple but powerful paradigm (Shin et al., 2017; Rebuffi et al., 2017; van de Ven et al., 2020). The main idea is to learn the data generating distribution and simulate data of previous tasks. We can extend our approach to the generative setting by modeling a variational autoencoder using the novel layers we propose in this work. We provide hyper-parameters and other experimental settings in Appendix C.

Modeling the latent variable z to capture the dynamics the data generating distribution $p(x)$ is difficult if $p(x)$ is multi-modal and authors have suggested the use of more complex distributions (Hadjeres et al., 2017; Ghosh et al., 2019; Vahdat & Kautz, 2020) as variational prior $p(z)$. We model the distribution of the latent variable z in the variational autoencoder by using a densely connected MoVE layer with 3 experts. Using multiple experts enables us to capture a richer class of distributions than a single Gaussian distribution could, as is usually the case in VAEs. We can interpret this as z following a Gaussian Mixture Model, whose components are mutually exclusive and modeled by experts. We integrate the generated data by optimizing a mixture of the loss on the new task data and the loss of the generated data:

$$L(\theta) = \frac{1}{2|B_t|} \sum_{b \in B_t} \ell(b) + \frac{1}{2|B_{1:t-1}|} \sum_{b \in |B_{1:t-1}|} \ell(b), \tag{15}$$

where B_t is batch of data from the current task, $B_{1:t-1}$ a batch of generated data (instead of stored data from previous tasks), and $\ell(b)$ a loss function on the batch b. We were able to improve our results in the supervised settings by incorporating a generative component as a replay mechanism, as we show in Table 1, and in Fig. 2.

3.3 Continual reinforcement learning

In the continual reinforcement learning (CRL) setting, the agent is tasked with finding an optimal policy in sequentially arriving reinforcement learning problems. To benchmark our method in this setting, we follow the experimental protocol of Ahn et al. (2019) and use a series of reinforcement learning problems from the PyBullet environments (Coumans & Bai, 2016–2021; Ellenberger, 2018–2019). In particular, we use the following: Walker2D, Half Cheetah, Ant, Inverted Double Pendulum, and Hopper. The environments we selected have different states and action dimensions. This implies we can't use a single neural network to model policies and value functions. To remedy this, we pad each state and action with zeros to have equal dimensions. The Ant environment has the highest dimensionality with a state dimensionality of 28 and an action dimensionality of 8. All others are zero-padded to have this dimensionality. We provide hyper-parameters and other settings in Appendix C.

Our approach to continual reinforcement learning can build upon any deep reinforcement learning algorithm (see Wang et al., 2020 for a review of current algorithms). Here, we chose soft actor-critic (SAC) (Haarnoja et al., 2018). We extend SAC by implementing all neural networks with MovE layers. When a new task arrives, the old posterior over the expert parameters and the gating posterior become the new priors. After each update step in task t, we evaluate the agent in all previous tasks $T_{1:t}$ for three episodes each. We divide the reward achieved during evaluation by the mean reward during training and report the cumulative normalized reward, which gives an upper bound of t in the t-th task.

We compare our approach against a simple continuously trained SAC implementation with dense neural networks, EWC (Kirkpatrick et al., 2017), and the recently published UCL (Ahn et al., 2019) method. UCL is similar to our approach in that it also employs Bayesian neural networks, but the weight regularization acts on a per-weight basis. Note that, in contrast to our approach, UCL and EWC both require task information to compute task-specific losses. Our results (see Fig. 5) show that our approach can sequentially learn new policies while maintaining an acceptable performance on previously seen tasks. We evaluate the methods by computing the following score $J(t)$ for each time step after training on task t is complete:

Fig. 5 Continual reinforcement learning. The upper row shows the cumulative normalized rewards over time across the five reinforcement learning tasks. Each vertical dotted line indicates a change in environment after 1 million frames. Performance is computed according to Eq. (16) after each frame and thus represents the total reward achieved by the agent summed over t tasks (with a maximum normalized reward of one per task). Thus, a value close to t indicates no forgetting, while 1.0 shows the total forgetting of old tasks. The lower row shows the absolute cumulative episodic reward in each of the five environments given pre-training on the preceding tasks. We compare against EWC (Kirkpatrick et al., 2017) and UCL (Ahn et al., 2019). The lower bound is given by a naively trained dense neural network using SAC (Haarnoja et al., 2018) without CL and without access to old environments

$$J(t) = \sum_{k=1}^{t} \frac{1}{NR_{\text{avg}}(k, \pi_k)} \sum_{n=1}^{N} R_n(k, \pi_t), \tag{16}$$

where N is the number of episodes to average over, and $R(k, \pi_k)$ is the cumulative episodic reward in environment k under policy π_k:

$$R(k, \pi_k) = \mathbb{E}_{a_i \sim \pi_k(s_i), s \sim p_k(s_{i+1}|a_i, s_i)} \left[r(s_i, a_i) \right], \tag{17}$$

where $p_k(s_{i+1}|a_i, s_i)$ are the dynamics of environment k and $r(s_i, a_i)$ is the immediate reward for executing action a_i in state s_i at time step i. The policy π_k refers to the policy trained on tasks up to k. The normalization is done by dividing by the average performance over ten episodes of the agent in task k, when the agent was trained on that task for the first time. Thus we get a score that measures how well the agent retains its performance on a past environment while learning to solve new problems. As we sum over past environments, an increasing score indicates a successful trade-off between learning and forgetting, while a decreasing or stagnating score indicates forgetting of old tasks.

Our method outperforms UCL (Ahn et al., 2019) and EWC (Kirkpatrick et al., 2017). In this setting naively training the agent sequentially (labeled "Dense") yields poor performance. This behavior indicates the complete forgetting of old policies. The bottom row of Fig. 5 shows the performance of our method in any particular environment, when pre-training on the preceding environments. It shows that the other methods (UCL and EWC) adapt more successfully to the individual tasks, which is however, coupled with catastrophic forgetting, when switching to the next task. In contrast, our method achieves a better trade-off between learning and forgetting. For this result, we did not optimize hyper-parameters for single task performance, but we simply set the hyper-parameters such that the training performance in each task was comparable in order of magnitude to the other methods. The evaluation across tasks with the same normalized metric shows then the superior ability of HVCL to maintain performance over a sequence of reinforcement learning tasks. Additionally, we note that the variance of the results achieved by our method are lower, suggesting a more stable and reliable training phase.

3.4 Ablation studies

To further investigate the methods we propose in this work, we designed a set of ablation experiments. In particular, we aim to demonstrate the importance of each component. To this effect, we run experiments investigating the generator quality in the generative CL setting, study the diversity bonuses in the supervised CL scenario, and take a closer look at the number of experts and the influence of the D_{KL} weights in the continual reinforcement learning setup.

3.4.1 Investigating diversity bonuses

In Sect. 2.2 we introduced a diversity objective to stabilize learning in a mixture-of-experts system. Additionally, we argued in favor of an entropy bonus to encourage a selection policy that favors high certainty and sparsity. To investigate the validity of these additions, we run a set of experiments on the Split CIFAR-10 dataset as described in Sect. 3, but with

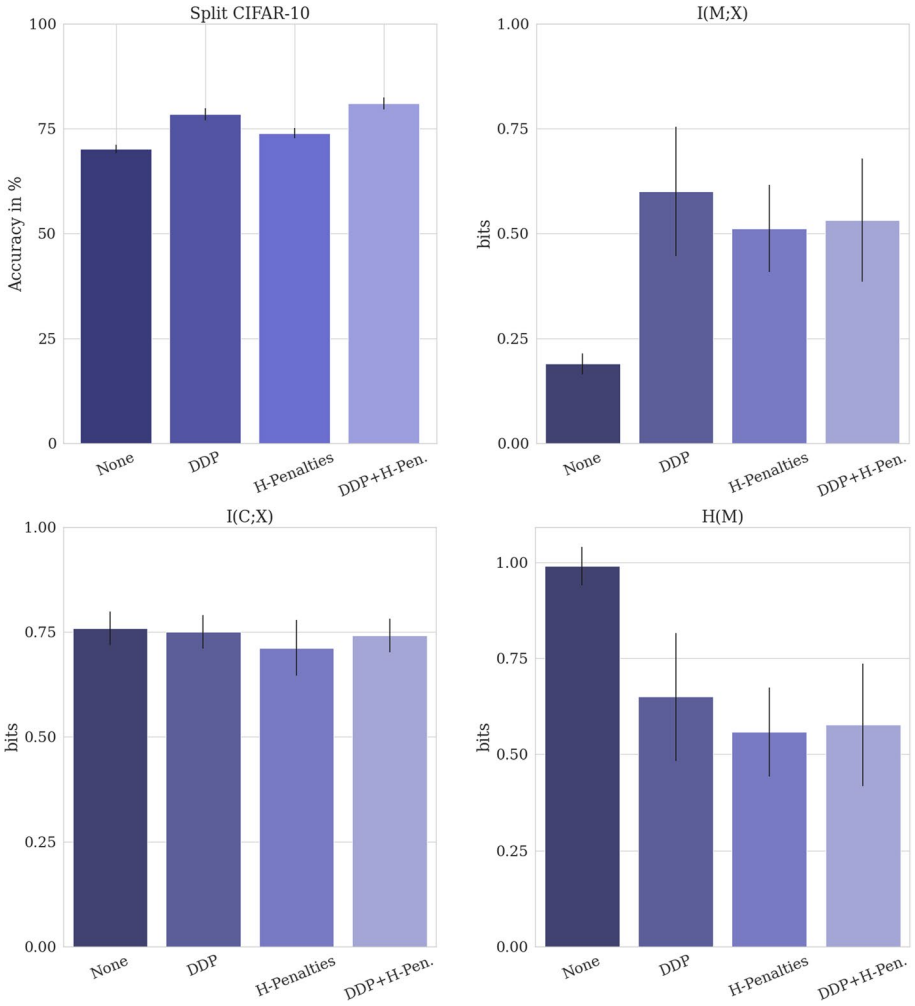

Fig. 6 Here we evaluate the proposed diversity measures in the split CIFAR-10 benchmark. We averaged every experiment over three random trials. Information-theoretic quantities $I(M; X)$, $I(C; X)$, and $H(X)$ were measured for each layer and averaged

different bonuses—see Fig. 6. In the baseline setup, we used no other objectives as those described by Eq. (6).

Apart from the classification accuracy, we are interested in three information-theoretic quantities that allow us to investigate the system closer. Firstly, the mutual information between the data generating distribution $p(x)$ and the expert selection $p(m|x)$ as measured by $I(M; X)$ indicates how much uncertainty over m the gating unit can reduce on average after observing an input x. A higher value means that inputs are differentiated better, which is what we would expect from a more diverse set of experts. $I(M; X)$ is the highest when we use a DPP-based diversity objective ("DDP"), while the entropy of selection policy $H(M)$ is lowest when we use an entropy-based diversity measure

("H-Penalties"), which both show that the objectives we introduced in this study yield the intended results. Combining both the DDP diversity bonus and the entropy penalty on the expert ("DDP+H-Pen.") enforces a trade-off between both objectives and yields the best empirical results. We average the results of ten random seeds in each setting.

3.4.2 Generator quality

We introduce a generative approach to continual learning in Sect. 3.2 by implementing a Variational Auto-encoder using our proposed layer design. This addition improved classification performance and mitigated catastrophic forgetting, as evidenced by the results shown in Fig. 2 and Table 1.

By borrowing methods from the generative learning community, we can investigate the performance further. The main focus lies on the quality of the generated images. We can not straightforwardly measure the accuracy, as artificial images lack labels. Thus we first use the trained classifier to obtain labels and compute metrics based on these self-generated labels. We opted for the Inception Score (IS) (Salimans et al., 2016), as it is widely used in the generative learning community. In this initially proposed formulation, the IS builds on the D_{KL} between the conditional and the marginal class probabilities as returned by a pre-trained Inception model (Szegedy et al., 2015). To investigate the quality of the generated images concerning the continually trained classifier, we use a different version of the Inception Score, which we defined as

$$IS_T(G_{1:T}) = \mathbb{E}_{x\sim G_{1:T}}\left[\mathrm{D}_{\mathrm{KL}}\left[p_{1:T}(y|x)||p_{1:T}(y)\right]\right], \tag{18}$$

where $G_{1:T}$ is the data generator trained on tasks up to T, $p_{1:T}(y|x)$ the conditional class distribution returned by the classifier trained up to task T, and $p_{1:T}(y)$ the marginal class distribution up to Task T. Note that, $IS(G_{1:T}) \leq \log_2 N_c$, where N_c is the number of classes. We show IS_T and the entropy of $p(y)$ in the split CIFAR-10 and CIFAR-100 setting in Fig. 7. In both cases, it can be seen that the generated pictures retain task-specific information, although there is a notable decrement across tasks.

3.4.3 Number of experts

Our method builds on a mixture of experts model and it is thus natural to assume that increasing the number of experts improves performance. Indeed, this is the case as we demonstrate in additional continual reinforcement learning experiments in Fig. 8. As Fig. 1 illustrates, adding experts to layers increases the number of possible information processing paths through the network. Equipped with a diverse and specialized set of parameters, each path can be regarded as a distinct sub-network that learns to solve tasks.

3.4.4 D_{KL} weights

As with any hyper-parameter, setting a specific value for β has a strong influence on the outcome of the experiments. Setting it too small will lead to the regularization term dominating the loss, and the experts can't learn a new task, as the new parameters remain close to the parameters of the previous task. A high value will drive the penalty term towards zero, which, in turn, will not preserve parameters from old tasks. In principle, there are three ways to choose β.

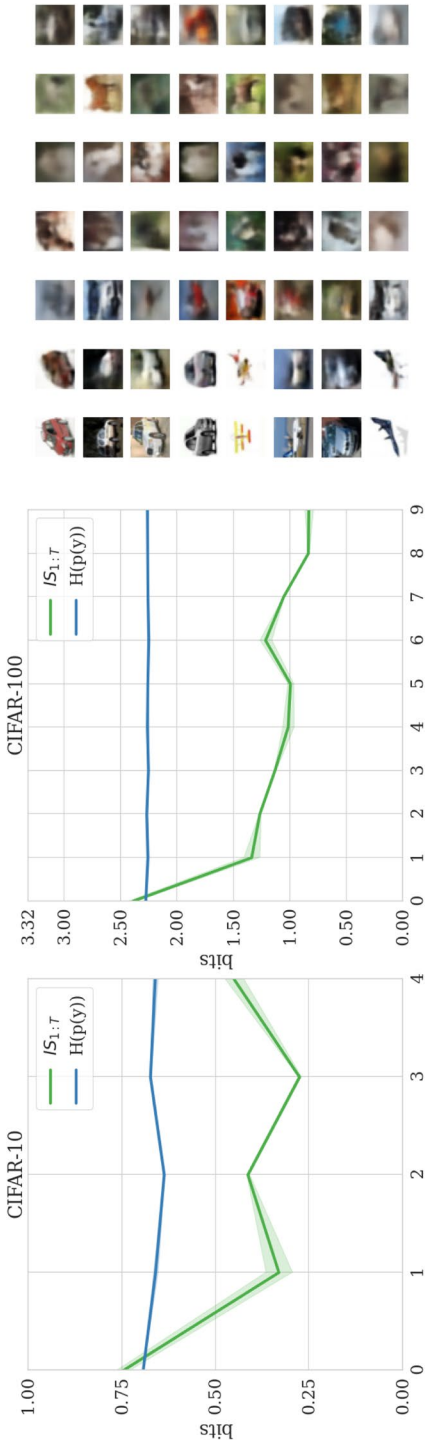

Fig. 7 Left and middle: Information-theoretic measures for the generator quality. Right: The first two rows show original images and the corresponding reconstructions. The following five rows show samples generated from the VAE after each task (automobiles vs. airplanes, birds vs. cats, deers vs. frogs, dogs vs. horses, and ships vs. trucks)

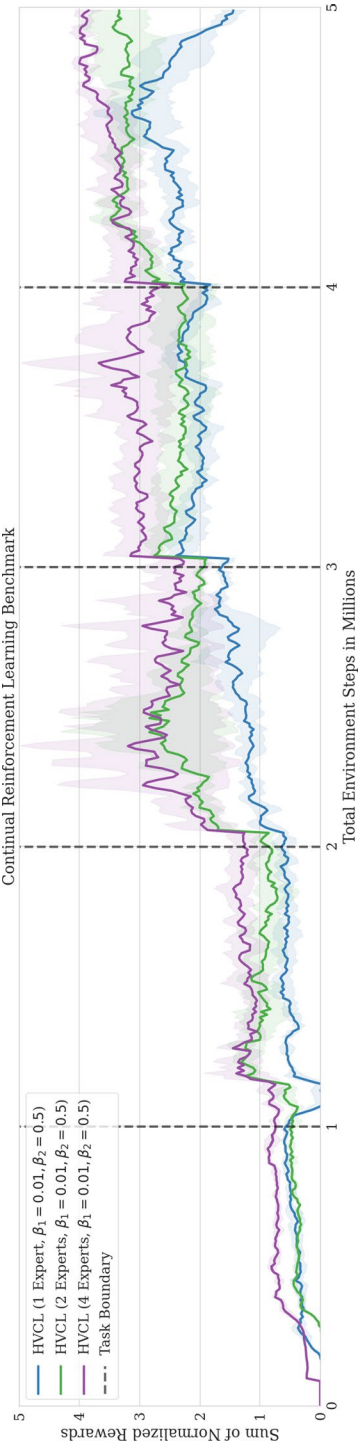

Fig. 8 Experimental results for systems with 1, 2, and 4 experts. As expected, adding experts mitigates forgetting. Each curve represents three trials in the continual reinforcement learning domain as described in Sect. 3.3

First, by setting β such that it satisfies an expert information-processing limit. This technique has the advantage that we can interpret this value, e.g., "each expert can process 1.57 bits of information on average, i.e., distinguishing between three options", but shifts the burden from picking β to setting a target entropy (see, e.g., Haarnoja et al., 2018; Grau et al., 2019 for an example of this approach). Second, employing a schedule for β, as, e.g., proposed by Fu et al. (2019). Last, another option is to run a grid search over a pre-defined range and choose the one that fits best. In our supervised learning experiments, we used a cyclic schedule for β_1 and β_2 (Fu et al., 2019) while we kept them fixed in the reinforcement learning experiments. To systematically investigate the influence of these parameters, we conducted additional experiments (see Fig. 9).

4 Discussion

4.1 Related work

The principle we propose in this work falls into a wider class of methods that deal more efficiently with learning and decision-making problems by integrating information-theoretic cost functions. Such information-constrained machine learning methods have enjoyed recent interest in a variety of research fields, such as reinforcement learning (Eysenbach et al., 2018; Ghosh et al., 2018; Leibfried & Grau-Moya, 2019; Hihn et al., 2019; Arumugam et al., 2020), MCMC optimization (Hihn et al., 2018; Pang et al., 2020), meta-learning (Rothfuss et al., 2018; Hihn & Braun, 2020), continual learning (Nguyen et al., 2017; Ahn et al., 2019), and self-supervised learning (Thiam et al., 2021; Tsai et al., 2021).

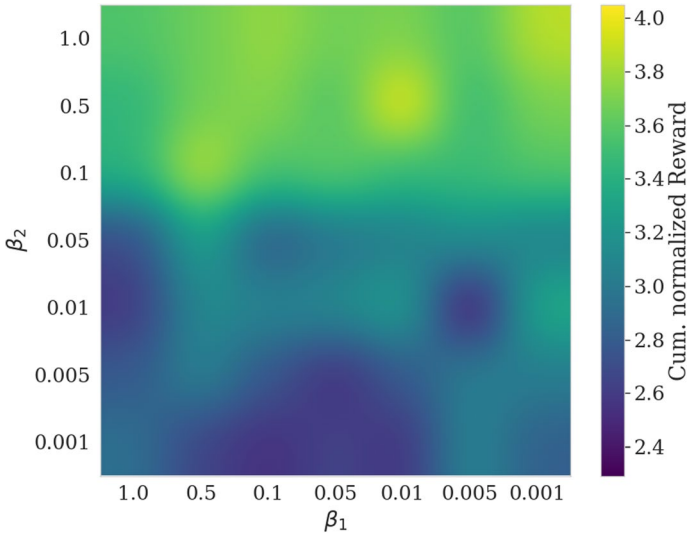

Fig. 9 In this figure, we show the influence of the D_{KL} weights β_1 and β_2 in the continual reinforcement learning setting. Setting the expert D_{KL} weight $\beta_2 < 0.1$ results in poor performance, as posteriors deviate too much from their priors. On the other hand, a lower gating D_{KL} weight β_1 allows for a flexible expert allocation and improves performance

The hierarchical structure we employ is a variant of the Mixture of Experts (MoE) model. Jacobs et al. (1991) introduced MoE as tree-structured models for complex classification and regression problems, where the underlying approach is the divide and conquer paradigm. As in our approach, three main building blocks define MoEs: gates, experts, and a probabilistic weighting to combine expert predictions. Learning proceeds by finding a soft partitioning of the input space and assigning partitions to experts performing well on the partition. MoEs for machine learning have seen a growing interest, with a recent surge stemming from the introduction of the sparsely-gated mixture-of-experts layer (Shazeer et al., 2017). In its initial form, this layer is optimized to divide the inputs equally among experts and then sparsely activate the top-k experts per input. This allowed training systems with billions of total parameters, as only a small subset was active at any given time. In our work, we removed the incentive to equally distribute inputs, as we aim to find specialized experts, which contradicts a balanced load. The computational advantage remains, as we still activate only the top-1 expert.

We extended the sparse MoE layer to continual learning by re-formulating its main principle as a hierarchical extension of variational continual learning (Nguyen et al., 2017). Our main contribution is removing the need for multi-headed networks by moving the head selection to the gating network. Our method is similar to the approach described in Hihn and Braun (2020) but differs in two key aspects. Firstly, we provide a more stable learning procedure as our layers can readily offer end-to-end training, which alleviates problems such as expert class imbalance and brittle converging properties reported in the previous study. Secondly, we implement the information-processing constraints on the parameters instead of the output of the experts, thus shifting the information cost from decision-making to learning. A method similar to ours is conditional computing for continual learning (Lin et al., 2019). The authors propose to condition the parameters of a neural network on the input samples by learning a (deterministic) function that groups inputs and maps a set of parameters to each group. Our approach differs in two main ways. First, our method can capture uncertainty allowing us to learn stochastic tasks. Second, our design can incorporate up to 2^n paths (or groupings) through a neural net with n layers, making it more flexible than learning a mapping function. Another routing approach is routing networks by Collier et al. (2020). The authors propose to use mixture-of-experts layers that they train with a novel algorithm called co-training. In co-training, an additional data structure keeps track of all experts assigned to a specific task, as these are trained differently than those unassigned so far. In contrast, our method does not require additional training procedures and does not produce any organizational overhead. PathNet (Fernando et al., 2017) is a modular neural network architecture where an evolutionary algorithm combines modules (e.g., convolution, max pooling, activation) to solve each task. This approach requires two training procedures: one for the evolutionary algorithm and one to adapt the path modules. Our method allows efficient end-to-end training. Lee et al. (2020) propose a Mixture-of-Experts model for continual learning, in which the number of experts increases dynamically. Their method utilizes Dirichlet-Process-Mixtures (Antoniak, 1974) to infer the number of experts. The authors argue that since the gating mechanism is itself a classifier, training it in an online fashion would result in catastrophic forgetting. To remedy this, they implement a generative model per expert m to model $p(m|x)$ and approximate the output as $p(y|x) \approx \sum_m p(y|x)p(m|x)$. In our work, we have demonstrated that it is possible to implement a gating mechanism based only on the input by coupling it with an information-theoretic objective to prevent catastrophic forgetting.

To stabilize expert training we introduced a diversity objective. Diversity measures have witnessed increasing interest in the reinforcement learning community. The ”diversity is

all you need" (DIAYN) paradigm (Eysenbach et al., 2018) proposes to formulate an information-processing hierarchy similar to Eq. (8), where the information bottleneck on the latent variable discards irrelevant information while an entropy bonus increases diversity. DIAYN acts as intrinsic motivation in environments with no rewards and enables efficient policy learning. Lupu et al. (2021) investigate diversity as a means to create a set of policies in a multi-agent environment. They define diversity between two policies based on a generalization of the Jensen-Shannon divergence and optimize this objective and the main goal defined by the environment simultaneously. Parker et al. (2020) introduce a DDP-based method, that aims to promote diversity in the policy space by defining it as a measure of the different states a policy may reach from a given starting state. Dai et al. (2021) propose another DDP-based method, where the idea is to augment the sampling process in hindsight experience replay (Andrychowicz et al., 2017) (a method for off-policy reinforcement learning) with a diversity bonus. The method we propose to encourage expert diversity differs from previous methods as we define diversity in parameter space instead of the policy outcomes or inputs. In combination with a D_{KL} penalty, this allows us to optimize for efficient information-processing, which facilities efficient continual learning. Additionally, as we define it on parameters instead of actions, we can apply it straightforwardly to any problem formulation, as our extensive experiments show.

Currently, there are only few methods that perform well in supervised continual learning and continual reinforcement learning (e.g., Ahn et al., 2019; Jung et al., 2020; Cha et al., 2020). These methods require task information, as the either keep a set of separate task-specific heads (Ahn et al., 2019; Jung et al., 2020) or compute task-specific losses (Cha et al., 2020). We were able to achieve comparable results to Uncertainty-based continual learning (UCL) (Ahn et al., 2019). This makes our method one of the first task-agnostic CL approaches to do so.

4.2 Critical issues and future work

One drawback of our method is the number of hyper-parameters it introduces. Specifically, the weighting factors for the gating and the expert D_{KL} constraint, the diversity bonus and its kernel parameters, and the number of experts and the top-k settings. To optimize them in our experiments, we ran a hyperparameter optimization algorithm on a reduced variant of the problem and fine-tuned them on the complete dataset. To further mitigate this problem, we used a scheduling scheme for the D_{KL} weights, as discussed in more detail in Sect. 3.4.4.

Moreover, variational inference in neural networks can be computationally expensive (Gal and Ghahramani 2016; Zhang et al., 2018; Freitas et al., 2000), as one has to draw samples for each forward pass to approximate gradients. We tackle this issue in three ways. Firstly, we use $D-$dimensional Gaussian mean-field approximate posteriors $q_t(\theta) = \prod_{d=1}^{D} \mathcal{N}(\theta_l | p_{t,d}, \sigma_{t,d}^2)$ to model distributions. This allows us to compute the D_{KL} (as well as the diversity measures) in closed form. Secondly, we use the flip-out estimator (Wen et al., 2018) to approximate the gradients, which is known to have a low variance. In practice, we draw a single sample to approximate the expectation. Lastly, our top$-k$ sampling allows us to only activate k of the posteriors, which means that the remaining parameters have zero gradient, and accordingly do not have to be updated (Tensorflow 2022). As a consequence, our HVCL approach requires a similar amount of training episodes and computation time as non-probabilistic network representations.

We have observed in simple toy problems that expert allocation and optimization greatly depend on the initialization of the experts. A sufficiently diverse expert initialization yielded better results while requiring fewer iterations. We could not directly transfer this to more complex learning problems because this would introduce computational instabilities during back-propagation, as we lose the variance reducing benefits of state-of-the-art initialization schemes (Glorot et al., 2010; He et al., 2015; Narkhede et al., 2021). This is also part of the reason why we chose to implement a diversity objective instead of a novel initializer. We leave this as a topic for future work.

To optimize the diversity measure, we have to compute the determinant of the kernel matrix (see Sect. 2.2) by evaluating the kernel function for each expert pair and finding the derivative of its determinant. While we showed that the Wasserstein-2 distance between Gaussian distributions with diagonal covariances is available in closed form (see Eq. 13), and that the resulting kernel matrix is symmetric, there is still the issue of the derivative of the determinant. Following Jacobi's formula to do so, requires finding the inverse of the kernel matrix. This operation fails when the kernel matrix is not invertible, which is equivalent to the determinant being zero. This is the case if diversity is also zero, i.e., the expert parameters are pairwise nearly identical. We countered this by setting the kernel width h (see Eq. 12) to a sufficiently large value, which we found by a simple grid search. This problem requires further investigation, as it can impede the complete optimization process.

Our model requires a fixed number of experts. In a more realistic continual learning setting, the number of tasks may grow such that the system may benefit from additional experts. This could be realized by taking a Bayesian non-parametric approach which treats the number of experts as a variable. Dirichlet-Process-Mixtures (Antoniak, 1974) offer such flexibility, with recent applications to meta-learning (Jerfel et al., 2019), and to continual learning (Lee et al., 2020).

5 Conclusion

We introduced a novel hierarchical approach to task-agnostic continual learning, derived an immediate application, and extensively evaluated this method in supervised continual learning and continual reinforcement learning. The method we introduced builds on a hierarchical Bayesian inference view of continual learning and is a direct extension of Variational Continual Learning (VCL). This adaptive mechanism allowed us to remove the need for extrinsic task-relevant information and to operate in a task-agnostic way. While we removed this limitation, we achieved results competitive to task-aware and to task-agnostic algorithms. These insights allowed us to design a diversity objective that stabilizes learning and further reduces the risk of catastrophic forgetting. In particular, we could show that enforcing expert diversity through additional objectives stabilizes learning and further reduces the risk of catastrophic forgetting. Essentially, catastrophic forgetting is avoided by having multiple experts that are responsible for different tasks and are activated in different contexts and trained only with their assigned data, which avoids weight overwriting and negative interference. Finally, having more diverse experts leads to crisper task partitioning with less interference.

As our method builds on generic utility functions, we can apply it independently of the underlying optimization problem, which makes our method one of the first to do so and achieve competitive results in continual supervised learning benchmarks based on variants of the MNIST, CIFAR-10, and CIFAR-100 datasets. In continual reinforcement learning,

we evaluated our method on a series of challenging simulated robotic control tasks. We also demonstrated how our method gives rise to a generative continual learning method. Additionally, we conducted ablation experiments to analyze our approach in a detailed and systematic way. These experiments confirmed that the additional objectives we introduced enhance expert partitioning and enforce a sparse expert selection policy. This leads to specialized and diverse experts, which alleviates catastrophic forgetting. We also investigated the impact of the hyper-parameters our method introduces to highlight how they help to mitigate catastrophic forgetting. In the generative setting, we took a closer look at the performance of the generative model by introducing a continual learning version of the widely used Inception Score. Finally, we showed that increasing the number of experts reduces forgetting in the continual reinforcement learning scenarios.

Appendix A: Wasserstein-distance between two Gaussians

The W_2^2 distance between two Gaussians is given by

$$W_2^2(p,q) = \|\mu_p - \mu_q\|_2^2 + \|\sqrt{d_p} - \sqrt{d_q}\|_2, \tag{19}$$

Proof: Let $p = \mathcal{N}(\mu_p, \Sigma_p)$ and $q = \mathcal{N}(\mu_q, \sigma_q)$ be two Guassian distributions. The Wasserstein-2 distance between p and q is then given by

$$W_2^2(p,q) = \|\mu_p - \mu_q\|_2^2 + \mathcal{B}(\Sigma_p, \Sigma_q), \tag{20}$$

where $\mathcal{B}$ is the Bures metric between two positive semi-definite matrices:

$$\mathcal{B}(\Sigma_p, \Sigma_q) = tr(\Sigma_p + \Sigma_q - 2(\Sigma_p^{1/2}\Sigma_q\Sigma_p^{1/2})^{1/2}, \tag{21}$$

where $tr(A)$ is the trace of a matrix A and $A^{1/2}$ is the matrix square root. Matrix square roots are computationally expensive to compute and there can potentially be an infinite number of solutions. In the case where p and q are Gaussian mean-field approximations, i.e., all dimensions are independent, Σ_p and Σ_q are given by diagonal matrices, such that $\Sigma_p = \text{diag}(d_p)_i$ and $\Sigma_q = \text{diag}(d_q)_i$. The Bures metric then reduces to the Hellinger distance between the diagonals d_p and d_q, and we have:

$$W_2^2(p,q) = \|\mu_p - \mu_q\|_2^2 + \|\sqrt{d_p} - \sqrt{d_q}\|_2. \tag{22}$$

The full Hellinger distance is given by $\frac{1}{\sqrt{2}}\|\sqrt{d_p} - \sqrt{d_q}\|_2$, but we chose to ommit the constant factor during optimization.

Appendix B: Wasserstein-2 exponential kernel

The exponential Wasserstein-2 kernel between isotropic Gaussian distributions p and q with kernel width h defined by

$$k(p,q) = \exp\left(-\frac{W_2^2(p,q)}{2h^2}\right)$$

is a valid kernel function.

Proof: The simplest way to show a kernel function k is valid is by deriving k from other valid kernels. We can express the Wasserstein distance as the sum of two norms as shown in Eq. (22). The euclidean norm and the Hellinger distance both form inner product spaces and are thus valid kernel functions. Their sum is also a valid kernel function, which makes the Wasserstein distance on isotropic Gaussians a valid kernel. If $k(p, q)$ is a valid kernel, then $\exp(k(p,q))$ is also a valid kernel.

Appendix C: Experiment details

To implement variational layers we use Gaussian distributions. For simplicity we use a $D-$dimensional Gaussian mean-field approximate posterior $q_t(\theta) = \prod_{d=1}^{D} \mathcal{N}(\theta_t | p_{t,d}, \sigma_{t,d}^2)$. We use the flip-out estimator (Wen et al., 2018) to approximate the gradients. In practice, we draw a single sample to approximate the expectation.

C.1 MNIST experiments

For split MNIST experiments we used dense layers for both the VAE and the classifier. The VAE encoder contains two layers with 256 units each, followed by 64 units (64 units for the mean and 64 units for log-variance) for the latent variable, and two layers with 256 units for the decoder, followed by an output layer with $28 * 28 = 784$ units. This assumes isotropic Gaussians as priors and posteriors over the latent variable and allows to compute the $\mathrm{D_{KL}}$ if closed form. We used only one expert for the VAE with $\beta_1 = 0.002$, $\beta_2 = 0.75$, a diversity bonus weight of 0.01 and leaky ReLU activations (Maas et al., 2013) in the hidden layers. We trained with a batch size 256 for 150 epochs. The VAE output activation function is a sigmoid and we trained it using a binary cross-entropy loss between the normalized pixel values of the original and the reconstructed images. We used no other regularization methods on the VAE. We used 10.000 generated samples after each task.

The classifier consists of two dense layers, each with 256 units with leaky ReLU activations (Maas et al., 2013) and dropout (Srivastava et al., 2014) layers, followed by an output layer with two units. All layers of the classifier have two experts. We trained with batch size 256 for 150 epochs using Adam (Kingma and Ba 2015) with a learning rate of 6×10^{-4}. In the permuted MNIST setting we used the same architecture, but increased the number of units to 512.

C.2 CIFAR-10 experiments

The VAE encoder consisted of five convolutional layers with stride 4 with two experts, each with 8, 16, 32, 64, and 128 units, followed by two dense units with two experts, each with 256 units. The latent variable has 128 dimensions, which we model by two dense layers: one with 128 units for the mean and one with 128 units for the log-variance. The dense layer modeling the mean has three experts, the layer for the log-variance one expert. We assume isotropic Gaussian distributions as priors and posteriors over the latent variable,

Table 3 Continual learning results with additional task-aware methods in the split MNIST (S-MNIST) and permuted MNIST (P-MNIST) benchmark compared to current CL methods

Baselines	S-MNIST	P-MNIST
Dense neural network	86.15 (± 1.00)	17.26 (± 0.19)
Offline re-training + task oracle	99.64 (± 0.03)	97.59 (± 0.02)
Single-head and task-agnostic methods		
Hierarchical VCL (ours)	97.50 (± 0.33)	97.07 (± 0.62)
Hierarchical VCL w/GR (ours)	98.60 (± 0.35)	97.47 (± 0.52)
Uncertainty guided CL w/BNN (Ebrahimi et al., 2020)	97.70 (± 0.03)	92.50 (± 0.01)
Brain-inspired replay through feedback† (van de Ven et al., 2020)	99.66 (± 0.13)	97.31 (± 0.04)
Hierarchical Indian buffet neural nets (Kessler et al., 2021)	91.00 (± 2.20)	93.70 (± 0.60)
Balanced continual learning (Raghavan and Balaprakash 2021)	98.71 (± 0.06)	97.51 (± 0.05)
Target layer regularization (Mazur et al., 2021)	80.64 (± 1.25)	
Multi-head and task-aware methods		
Synaptic intelligence† (Zenke et al., 2017)	99.14 (± 0.49)	94.75 (± 0.49)
Deep generative replay + distill.† (Shin et al., 2017)	99.59 (± 0.40)	97.51 (± 0.04)
Variational continual learning† (Nguyen et al., 2017)	98.50 (± 1.78)	96.60 (± 1.34)
Elastic weight consolidation† (Kirkpatrick et al., 2017)	85.48 (± 5.36)	94.74 (± 0.22)
Learning without forgetting† (Li & Hoiem, 2017)	99.60 (± 0.13)	69.84 (± 2.00)
Continual unsupervised representation learning (Rao et al., 2019)	99.10 (± 0.06)	
Self-attention meta-learning for CL (Sokar et al., 2021)	97.95 (± 0.07)	
Dynamically expandable networks (Yoon et al., 2018)	99.26 (± 0.01)	

Results were averaged over ten random seeds with the standard deviation given in parenthesis. Results on algorithms marked with † were taken from van de Ven and Tolias (2018), others from their original work

which allows us to compute the D_{KL} if closed form. The decoder mirrors the encoder and has two dense layers followed by 5 de-convolutional layers with stride 4 (the last layer has stride 3). All hidden layers use a leaky ReLU activation function (Maas et al., 2013). The VAE output activation function is a sigmoid and we trained it using a binary cross-entropy loss between the normalized pixel values of the original and the reconstructed images. We used no other regularization methods on the VAE. We used 10.000 generated samples after each task.

The classifier architecture is similar to the encoder architecture. We used five convolutional layers, followed by two dense layers. All layers used two experts. The convolutional layers have 8, 16, 32, 64, and 128 units per experts, while the dense layers both have 256 units per layer. We used leaky ReLU as an activation function for the hidden layers and softmax for the output layer. We trained the classifier using a binary cross-entropy loss between the true and the predicted label. We trained with batch size 256 for 1000 epochs using the Adam optimizer with a learning rate of 3×10^{-4}.

C.3 Reinforcement learning experiment details

Each task was trained for one million time steps. We use two layer networks (actor and critics) with 64 units per layer. Each layer has four experts followed by leaky ReLU (Maas et al., 2013) activation functions. Each We set each SAC related hyper-parameter as proposed in the original publication (Haarnoja et al., 2018). We update every 5000

Table 4 Continual learning results with additional task-aware methods in the split CIFAR-10 and the split CIFAR-100 benchmark compared to current CL methods

Baselines	Split-CIFAR-10	CIFAR-100
Conv. neural network	66.62 (± 1.06)	19.80 (± 0.19)
Offline re-training + task oracle	80.42 (± 0.95)	52.30 (± 0.02)
Single-head and task-agnostic methods		
Hierarchical VCL (ours)	78.41 (± 1.18)	33.10 (± 0.62)
Hierarchical VCL w/GR (ours)	81.00 (± 1.15)	37.20 (± 0.52)
Continual learning with dual regularizations (Han and Guo 2021)	86.72 (± 0.30)	25.62 (± 0.22)
Natural continual learning (Kao et al., 2021)		38.79 (± 0.24)
Target layer regularization (Mazur et al., 2021)	74.89 (± 0.61	
Memory aware synapses (He & Zhu, 2022)	73.50 (± 1.54)	
Multi-head and task-aware methods		
Gradient episodic memory (Lopez-Paz & Ranzato, 2017)	79.10 (± 1.60)	40.60 (± 1.90)
Meta-consolidation for continual learning (Kj & Balasubramanian, 2020)	82.90 (± 1.20)	43.50 (± 0.60)
Contrastive CL with feature propagation (Han & Guo, 2021)	86.33 (± 1.47)	65.19 (± 0.65)
Hindsight anchor learning (Chaudhry et al., 2021)	75.19 (± 2.57)	47.88 (± 2.76)
Synaptic intelligence (Zenke et al., 2017)	63.31 (± 3.79)	36.33 (± 4.23)
Averaged gradient episodic memory (Chaudhry et al., 2018)	74.07 (± 0.76)	46.88 (± 1.81)

Results were averaged over ten random seeds with the standard deviation given in the parenthesis. We report results of other methods as given in their original studies

environment steps with a batch size of 256. We use a Prioritized Replay Buffer (Schaul et al., 2015) with 1 Million sample capacity. For UCL (Ahn et al., 2019), we used the implementation provided by the authors for our experiments and use the hyper-parameters suggested in the publication. Note that the UCL implementation rests on a PPO (Schulman et al., 2017) backbone. Our CRL experiments do not use any form of additional replay (except for the replay buffer used by SAC).

Author contributions HH Conceptualization, Methodology, Formal analysis, Software, Visualization, Validation, Investigation, Writing—Original draft preparation—Reviewing and Editing, DAB Conceptualization, Supervision, Funding acquisition, Writing—Reviewing and Editing.

Funding Open Access funding enabled and organized by Projekt DEAL. This work was supported by the European Research Council, grant number ERC-StG-2015-ERC, Project ID: 678082, "BRISC: Bounded Rationality in Sensorimotor Coordination".

Data availability Datasets and Environments used in this study: MNIST http://yann.lecun.com/exdb/mnist/, CIFAR10/1000: https://www.cs.toronto.edu/~kriz/cifar.html, PyBullet Gymperium: https://www.github.com/benelot/pybullet-gym, Tensorflow: https://www.tensorflow.org/, NumPy https://numpy.org/doc/, SciPy https://scipy.org/.

Declarations

Conflict of interest All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.

Ethics approval Not applicable.

Code availability Open Source code implementing MoVE Layers is available under https://github.com/hhihn/HVCL/.

Consent to participate Not applicable.

Consent for publication Not applicable.

References

Ahn, H., Cha, S., Lee, D., & Moon, T. (2019). Uncertainty-based continual learning with adaptive regularization. In *Proceedings of the 33rd international conference on neural information processing systems* (pp. 4392–4402).

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, P., & Zaremba, W. (2017). Hindsight experience replay. In *Proceedings of the 31st international conference on neural information processing systems* (pp. 5055–5065).

Antoniak, C. E. (1974). Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *The Annals of Statistics*, 1152–1174.

Arumugam, D., Henderson, P., & Bacon, P.-L. (2020). An information-theoretic perspective on credit assignment in reinforcement learning. In *Workshop on biological and artificial reinforcement learning (NeurIPS 2020)*.

Bang, J., Kim, H., Yoo, Y., Ha, J.-W., & Choi, J. (2021). Rainbow memory: Continual learning with a memory of diverse samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 8218–8227).

Benavides-Prado, D., Koh, Y. S., & Riddle, P. (2020). Towards knowledgeable supervised lifelong learning systems. *Journal of Artificial Intelligence Research, 68,* 159–224.

Bian, Y., & Chen, H. (2021). When does diversity help generalization in classification ensembles. *IEEE Transactions on Cybernetics*.

Biesialska, M., Biesialska, K., & Costa-jussà, M. R. (2020). Continual lifelong learning in natural language processing: A survey. In *Proceedings of the 28th international conference on computational linguistics* (pp. 6523–6541).

Breiman, L. (1996). Bagging predictors. *Machine Learning, 24,* 123–140. https://doi.org/10.1023/A:1018054314350.

Cha, S., Hsu, H., Hwang, T., Calmon, F., & Moon, T. (2020). Cpr: Classifier-projection regularization for continual learning. In *International conference on learning representations*.

Chaudhry, A., Dokania, P. K., Ajanthan, T., & Torr, P. H. (2018). Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European conference on computer vision (ECCV)* (pp. 532–547).

Chaudhry, A., Gordo, A., Dokania, P., Torr, P., & Lopez-Paz, D. (2021). Using hindsight to anchor past knowledge in continual learning. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 35, pp. 6993–7001).

Chaudhry, A., Ranzato, M., Rohrbach, M., & Elhoseiny, M. (2018). Efficient lifelong learning with a-gem. In *International conference on learning representations*.

Collier, M., Kokiopoulou, E., Gesmundo, A., & Berent, J. (2020). Routing networks with co-training for continual learning. In *ICML 2020 workshop on continual learning*.

Coumans, E., & Bai, Y. (2016–2021). PyBullet, a Python module for physics simulation for games, robotics and machine learning. http://pybullet.org

Cover, T. M., & Thomas, J. A. (2012). *Elements of information theory*. Wiley.

Dai, T., Liu, H., Arulkumaran, K., Ren, G., & Bharath, A. A. (2021). Diversity-based trajectory and goal selection with hindsight experience replay. In *Pacific rim international conference on artificial intelligence* (pp. 32–45). Springer.

De Lange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., et al. (2021). A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 44*(7), 3366–3385.

Ebrahimi, S., Elhoseiny, M., Darrell, T., & Rohrbach, M. (2020). Uncertainty-guided continual learning with bayesian neural networks. In *International conference on learning representations*.

El Khatib, A., & Karray, F. (2019). Strategies for improving single-head continual learning performance. In F. Karray, A. Campilho, & A. Yu (Eds.), *Image analysis and recognition* (pp. 452–460). Cham: Springer.

Ellenberger, B. (2018–2019). PyBullet Gymperium. https://www.github.com/benelot/pybullet-gym

Eysenbach, B., Gupta, A., Ibarz, J., & Levine, S. (2018). Diversity is all you need: Learning skills without a reward function. In *International conference on learning representations*.

Farquhar, S., & Gal, Y. (2018). Towards robust evaluations of continual learning. In *Lifelong learning: A reinforcement learning approach (ICML 2018)*.

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

Freitas, J. D., Niranjan, M., Gee, A. H., & Doucet, A. (2000). Sequential monte Carlo methods to train neural network models. *Neural Computation, 12*(4), 955–993.

Fu, H., Li, C., Liu, X., Gao, J., Celikyilmaz, A., & Carin, L. (2019). Cyclical annealing schedule: A simple approach to mitigating kl vanishing. In *Proceedings of the 2019 conference of the north american chapter of the association for computational linguistics: Human language technologies, Volume 1 (long and short papers)* (pp. 240–250).

Gal, Y., & Ghahramani, Z. (2016). Bayesian convolutional neural networks with Bernoulli approximate variational inference. In *ICLR 2016 workshop track*.

Galashov, A., Jayakumar, S. M., Hasenclever, L., Tirumala, D., Schwarz, J., Desjardins, G., Czarnecki, W. M., Teh, Y. W., Pascanu, R., & Heess, N. (2019). Information asymmetry in kl-regularized rl. In *Proceedings of the international conference on representation learning*

Genewein, T., Leibfried, F., Grau-Moya, J., & Braun, D. A. (2015). Bounded rationality, abstraction, and hierarchical decision-making: An information-theoretic optimality principle. *Frontiers in Robotics and AI, 2,* 27.

Ghosh, D., Singh, A., Rajeswaran, A., Kumar, V., & Levine, S. (2018). Divide-and-conquer reinforcement learning. In *International conference on learning representations*.

Ghosh, P., Sajjadi, M. S., Vergari, A., Black, M., & Scholkopf, B. (2019). From variational to deterministic autoencoders. In *International conference on learning representations.*

Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In Teh, Y. W., & Titterington, M. (Eds.), *Proceedings of the thirteenth international conference on artificial intelligence and statistics. proceedings of machine learning research* (Vol. 9, pp. 249–256). PMLR, Chia Laguna Resort, Sardinia, Italy. https://proceedings.mlr.press/v9/glorot10a.html

Golkar, S., Kagan, M., & Cho, K. (2019). Continual learning via neural pruning. In *NeurIPS 2019 workshop neuro AI*.

Grau-Moya, J., Leibfried, F., & Vrancx, P. (2019). Soft q-learning with mutual-information regularization. In *Proceedings of the international conference on learning representations*.

Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning* (pp. 1861–1870).

Hadjeres, G., Nielsen, F., & Pachet, F. (2017). Glsr-vae: Geodesic latent space regularization for variational autoencoder architectures. In *2017 IEEE symposium series on computational intelligence (SSCI)* (pp. 1–7). IEEE.

Han, X., & Guo, Y. (2021a). Continual learning with dual regularizations. In *Joint European conference on machine learning and knowledge discovery in databases* (pp. 619–634). Springer.

Han, X., & Guo, Y. (2021b). Contrastive continual learning with feature propagation. arXiv:2112.01713

He, J., & Zhu, F. (2022). Online continual learning via candidates voting. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision* (pp. 3154–3163).

He, K., Zhang, X., Ren, S., & Sun, J.(2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision* (pp. 1026–1034).

Hihn, H., Gottwald, S., & Braun, D. A. (2019). An information-theoretic on-line learning principle for specialization in hierarchical decision-making systems. In *2019 IEEE 58th conference on decision and control (CDC)* (pp. 3677–3684). IEEE.

Hihn, H., Gottwald, S., Braun, & D. A. (2018). Bounded rational decision-making with adaptive neural network priors. In *IAPR workshop on artificial neural networks in pattern recognition* (pp. 213–225). Springer.

Hihn, H.,Braun, & D. A. (2020a). Hierarchical expert networks for meta-learning. In *4th ICML workshop on life long machine learning*.

Hihn, H., & Braun, D. A. (2020b). Specialization in hierarchical learning systems. *Neural Processing Letters, 52*(3), 2319–2352.

Hsu, Y.-C., Liu, Y.-C., Ramasamy, A., & Kira, Z. (2018). Re-evaluating continual learning scenarios: A categorization and case for strong baselines. In *Continual learning workshop, 32nd conference on neural information processing systems*.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural Computation, 3*(1), 79–87.

Jerfel, G., Grant, E., Griffiths, T., & Heller, K. A. (2019). Reconciling meta-learning and continual learning with online mixtures of tasks. In *Advances in neural information processing systems* (pp. 9122–9133).

Jung, S., Ahn, H., Cha, S., & Moon, T. (2020). Continual learning with node-importance based adaptive group sparse regularization. *Advances in Neural Information Processing Systems, 33,* 3647–3658.

Kao, T.-C., Jensen, K., van de Ven, G., Bernacchia, A., & Hennequin, G. (2021). Natural continual learning: Success is a journey, not (just) a destination. *Advances in Neural Information Processing Systems*, *34*.

Kessler, S., Nguyen, V., Zohren, S., & Roberts, S. J. (2021). Hierarchical Indian buffet neural networks for Bayesian continual learning. In *Uncertainty in artificial intelligence* (pp. 749–759). PMLR.

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In *Proceedings of the 3rd international conference on learning representations*.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., et al. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences, 114*(13), 3521–3526.

Kj, J., & Balasubramanian, N. V. (2020). Meta-consolidation for continual learning. *Advances in Neural Information Processing Systems, 33,* 14374–14386.

Kulesza, A., Taskar, B., et al. (2012). Determinantal point processes for machine learning. *Foundations and Trends in Machine Learning, 5*(2–3), 123–286.

Kuncheva, L. I. (2004). *Combining pattern classifiers: Methods and algorithms*. Wiley.

Kuncheva, L. I., & Whitaker, C. J. (2003). Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning, 51*(2), 181–207.

Lee, S., Ha, J., Zhang, D., & Kim, G. (2020). A neural dirichlet process mixture model for task-free continual learning. In *International conference on learning representations*. https://openreview.net/forum?id=SJxSOJStPr

Leibfried, F., & Grau-Moya, J. (2019). Mutual-information regularization in Markov decision processes and actor-critic learning. In *Proceedings of the conference on robot learning*.

Li, H., Krishnan, A., Wu, J., Kolouri, S., Pilly, P. K., & Braverman, V. (2021). Lifelong learning with sketched structural regularization. In *Asian conference on machine learning* (pp. 985–1000). PMLR.

Li, Z., & Hoiem, D. (2017). Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 40*(12), 2935–2947. https://doi.org/10.1109/TPAMI.2017.2773081.

Lin, M., Fu, J., & Bengio, Y. (2019). Conditional computation for continual learning. In *NeurIPS 2018 continual learning workshop*.

Liu, Y., Su, Y., Liu, A. -A., Schiele, B., & Sun, Q. (2020). Mnemonics training: Multi-class incremental learning without forgetting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 12245–12254).

Lopez-Paz, D., & Ranzato, M. (2017). Gradient episodic memory for continual learning. In *Advances in neural information processing systems* (pp. 6467–6476).

Lupu, A.,Cui, B., Hu, H., & Foerster, J. (2021). Trajectory diversity for zero-shot coordination. In *International conference on machine learning* (pp. 7204–7213). PMLR.

Maas, A. L., Hannun, A. Y., & Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. In *Proceedings of the Icml* (Vol. 30, p. 3).

Macchi, O. (1975). The coincidence approach to stochastic point processes. *Advances in Applied Probability, 7*(1), 83–122.

Mazur, M., Pustelnik, Ł., Knop, S., Pagacz, P., & Spurek, P. (2021). Target layer regularization for continual learning using cramer-wold generator. arXiv:2111.07928

McCloskey, M., & Cohen, N. J. (1989). Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation, 24,* 109–165. https://doi.org/10.1016/S0079-7421(08)60536-8.
Narkhede, M. V., Bartakke, P. P., & Sutaone, M. S. (2021). A review on weight initialization strategies for neural networks. *Artificial Intelligence Review*, 1–32.
Nguyen, C. V., Li, Y., Bui, T. D., & Turner, R. E. (2017). Variational continual learning. In *Proceedings of the international conference on representation learning*.
Ostapenko, O., Puscas, M., Klein, T., Jahnichen, P., & Nabi, M. (2019). Learning to remember: A synaptic plasticity driven framework for continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 11321–11329).
Pang, B., Han, T., Nijkamp, E., Zhu, S.-C., & Wu, Y. N. (2020). Learning latent space energy-based prior model. *Advances in Neural Information Processing Systems33*.
Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., & Wermter, S. (2019). Continual lifelong learning with neural networks: A review. *Neural Networks, 113,* 54–71.
Parker-Holder, J., Pacchiano, A., Choromanski, K. M., & Roberts, S. J. (2020). Effective diversity in population based reinforcement learning. *Advances in Neural Information Processing Systems, 33*.
Raghavan, K., & Balaprakash, P. (2021). Formalizing the generalization-forgetting trade-off in continual learning. *Advances in Neural Information Processing Systems*, *34*.
Rao, D., Visin, F., Rusu, A., Pascanu, R., Teh, Y. W., & Hadsell, R. (2019). Continual unsupervised representation learning. *Advances in Neural Information Processing Systems*, *32*.
Rasch, B., & Born, J. (2007). Maintaining memories by reactivation. *Current Opinion in Neurobiology, 17*(6), 698–703.
Rebuffi, S. -A., Kolesnikov, A., Sperl, G., & Lampert, C. H. (2017) icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2001–2010).
Rothfuss, J., Lee, D., Clavera, I., Asfour, T., & Abbeel, P. (2018). Promp: Proximal meta-policy search. In *International conference on learning representations*.
Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., & Hadsell, R. (2016). Progressive neural networks. In *NIPS deep learning symposium*.
Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., & Chen, X. (2016). Improved techniques for training gans. *Advances in Neural Information Processing Systems, 29,* 2234–2242.
Schaul, T., Quan, J., Antonoglou, I., & Silver, D. (2015). Prioritized experience replay. arXiv:1511.05952
Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. arXiv:1707.06347
Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., & Dean, J. (2017). Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *Proceedings of the international conference on learning representations (ICLR)*.
Shin, H., Lee, J. K., Kim, J., & Kim, J. (2017). Continual learning with deep generative replay. In *Advances in neural information processing systems* (pp. 2990–2999).
Sokar, G., Mocanu, D. C., & Pechenizkiy, M. (2021). Self-attention meta-learner for continual learning. In *Proceedings of the 20th international conference on autonomous agents and multiagent systems* (pp. 1658–1660).
Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research, 15*(1), 1929–1958.
Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1–9).
Tensorflow 2.0 Documentation (2022). https://www.tensorflow.org/addons/api_docs/python/tfa/optimizers/LazyAdam
Thiam, P.,Hihn, H., Braun, D. A., Kestler, H. A., & Schwenker, F. (2021). Multi-modal pain intensity assessment based on physiological signals: A deep learning perspective. *Frontiers in Physiology, 12*.
Thrun, S. (1998). Lifelong learning algorithms. In *Learning to learn* (pp. 181–209). Springer.
Tsai, Y.-H. H., Wu, Y., Salakhutdinov, R., & Morency, L.-P. (2021). Self-supervised learning from a multi-view perspective. In *International conference on learning representations*.
Vahdat, A., & Kautz, J.: Nvae: A deep hierarchical variational autoencoder. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., & Lin, H. (Eds.), *Advances in neural information processing systems* (Vol. 33, pp. 19667–19679). Curran Associates, Inc., Online Conference (2020). https://proceedings.neurips.cc/paper/2020/file/e3b21256183cf7c2c7a66be163579d37-Paper.pdf

van de Ven, G. M., Siegelmann, H. T., & Tolias, A. S. (2020). Brain-inspired replay for continual learning with artificial neural networks. *Nature Communications, 11*(1), 1–14.

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

van de Ven, G. M., Trouche, S., McNamara, C. G., Allen, K., & Dupret, D. (2016). Hippocampal offline reactivation consolidates recently formed cell assembly patterns during sharp wave-ripples. *Neuron, 92*(5), 968–974.

Vijayan, M., & Sridhar, S. S. (2021). Continual learning for classification problems: A survey. In V. Krishnamurthy, S. Jaganathan, K. Rajaram, & S. Shunmuganathan (Eds.), *Computational intelligence in data science* (pp. 156–166). Springer.

Wang, H.-n., Liu, N., Zhang, Y.-y., Feng, D.-w., Huang, F., Li, D.-s., & Zhang, Y.-m. (2020). Deep reinforcement learning: A survey. *Frontiers of Information Technology and Electronic Engineering*, 1–19.

Wang, S., Li, X., Sun, J., & Xu, Z. (2021). Training networks in null space of feature covariance for continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 184–193).

Wen, Y., Vicol, P., Ba, J., Tran, D., & Grosse, R. (2018). Flipout: Efficient pseudo-independent weight perturbations on mini-batches. In *International conference on learning representations*.

Wilson, M. A., & McNaughton, B. L. (1994). Reactivation of hippocampal ensemble memories during sleep. *Science, 265*(5172), 676–679.

Yao, H., Wei, Y., Huang, J., & Li, Z. (2019). Hierarchically structured meta-learning. In *Proceedings of the international conference on machine learning* (pp. 7045–7054).

Yoon, J., Yang, E., Lee, J., & Hwang, S. J. (2018). Lifelong learning with dynamically expandable networks. In *6th International conference on learning representations, ICLR 2018.*

Zacarias, A., & Alexandre, L. A. (2018). Sena-cnn: Overcoming catastrophic forgetting in convolutional neural networks by selective network augmentation. In *IAPR workshop on artificial neural networks in pattern recognition* (pp. 102–112). Springer.

Zeng, G., Chen, Y., Cui, B., & Yu, S. (2019). Continual learning of context-dependent processing in neural networks. *Nature Machine Intelligence, 1*(8), 364–372.

Zenke, F., Poole, B., & Ganguli, S. (2017). Continual learning through synaptic intelligence. *Proceedings of Machine Learning Research, 70,* 3987.

Zhai, M., Chen, L., Tung, F., He, J., Nawhal, M., & Mori, G. (2019). Lifelong gan: Continual learning for conditional image generation. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 2759–2768).

Zhang, G., Sun, S., Duvenaud, D., & Grosse, R. (2018). Noisy natural gradient as variational inference. In *International conference on machine learning* (pp. 5852–5861). PMLR.