MIXTURE-OF-VARIATIONAL-EXPERTS FOR CONTINUAL LEARNING

Heinke Hihn* **& Daniel A. Braun**
Institute of Neural Information Processing
Ulm University, Ulm, Germany
{heinke.hihn,daniel.braun}@uni-ulm.de

ABSTRACT

One weakness of machine learning algorithms is the poor ability of models to solve new problems without forgetting previously acquired knowledge. The Continual Learning (CL) 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 discuss this principle from a Bayesian perspective and show its connections to previous approaches to CL. 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. 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 in continual supervised learning and in continual reinforcement learning.

1 INTRODUCTION

Acquiring new skills 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 adaptation to task T_t prompts overwriting of 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 three broad categories: rehearsal and memory consolidation, regularization and weight consolidation, and architecture and expansion methods. Rehearsal methods train a generative model to learn the data-generating distribution to reproduce data of old tasks (Shin et al., 2017; Rebuffi et al., 2017). In contrast, regularization methods (e.g., Kirkpatrick et al., 2017; Ahn et al., 2019; Han & Guo, 2021a) introduce an additional constraint to the learning objective to prevent changes in task-relevant parameters. Finally, CL can be achieved by modifying the design of a model during learning (e.g., Lin et al., 2019; Rusu et al., 2016; Golkar et al., 2019).

Despite recent progress in CL, there are still 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. Approaches sharing this drawback are multi-head methods (e.g., El Khatib & Karray, 2019; Nguyen et al., 2017; Ahn et al., 2019) and methods that compute a per-task loss, which requires storing old weights and task information (e.g.,

*Corresponding Author
Code is available at https://sites.google.com/view/hvcl

Kirkpatrick et al., 2017; Zenke et al., 2017; Sokar et al., 2021; Yoon et al., 2018; Chaudhry et al., 2021; Han & Guo, 2021b). Extracting relevant task information is a difficult problem, in particular when distinguishing tasks without contextual input (Hihn & Braun, 2020b). Thus, providing the model with such task-relevant information yields overly optimistic results (Chaudhry et al., 2018a).

To deal with more realistic CL scenarios, therefore, models must learn to compensate for the lack of auxiliary information. The approach we propose here 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 (HCVL) 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. A 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.

In ensemble methods two main questions arise. 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).

To summarize, our contributions are the following: (i) we extend VCL to a hierarchical multi-prior setting, (ii) we derive a computationally efficient method for task-agnostic continual learning from this general formulation, (iii) to improve expert specialization and diversity, we introduce and evaluate novel diversity measures.

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

2 Hierarchical Variational Continual Learning

In this section we first 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 Section 2.1.

VCL describes a general learning paradigm wherein an agent stays close to an old strategy ("prior") it has learned on a previous task $t-1$ while learning to solve a new task 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:

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

where $p(\theta)$ is a distribution over the models parameter θ and N_t is the number of samples for task t. The constraint encourages the agent to find an optimal trade-off between solving a new task and retaining knowledge about old tasks. Over the course of T datasets, Bayes' rule then recovers the posterior

$$p(\theta|\mathcal{D}_{1:T}) \propto p(\theta|\mathcal{D}_{1:T-1}) p(\mathcal{D}_T|\theta) \tag{2}$$

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.

This multi-head strategy has two main drawbacks: (i) it introduces an organizational overhead due to the growing number of network heads, and (ii) task boundaries must be known at all times, making it unsuitable for more complex continual learning settings. In the following we argue that

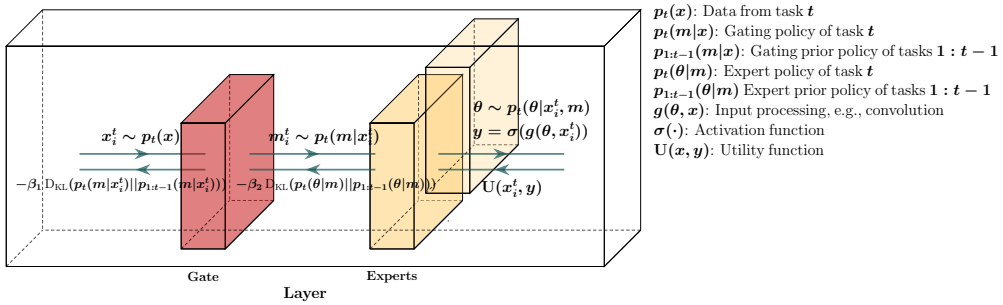

Figure 1: This figure illustrates our proposed design. Each layer implements a top-k expert selection conditioned on the output of the previous layer. Each expert m maintains a distribution over its weights $p(\theta|m) = \mathcal{N}(\mu_m, \sigma_m)$ and a set of bias variables b_m.

we can alleviate these problems by combining multiple decision-makers with a learned selection policy.

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:

$$p(y_t^i|x_t^i, \Theta) = \sum_{m=1}^{M} p(z_t^{i,m}|x_t^i, \vartheta)p(y_t^i|x_t^i, \omega_m), \tag{3}$$

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

$$\begin{aligned} p(\Theta|\mathcal{D}_{1:T}) &\propto p(\vartheta)p(\omega)\prod_{t=1}^{T}\prod_{i=1}^{N_t}\sum_{m=1}^{M} p(z_t^{i,m}|x_t^i, \vartheta)p(y_t^i|x_t^i, \omega_m) \\ &= p(\Theta)\prod_{t=1}^{T} p(\mathcal{D}_t|\Theta) \propto p(\Theta|\mathcal{D}_{1:T-1})p(\mathcal{D}_T|\Theta). \end{aligned} \tag{4}$$

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 $p(\mathcal{D}_T|\Theta)$. Finally, we formulate the HVCL objective for task t as:

$$\mathcal{L}_{\text{HVCL}}^t = \sum_{i=1}^{N_t} \mathbb{E}_{p(\Theta)}\left[\log p(y_t^i|x_t^i, \Theta)\right] - \mathrm{D_{KL}}\left[p_t(\vartheta)||p_{1:t-1}(\vartheta)\right] - \mathrm{D_{KL}}\left[p_t(\omega)||p_{1:t-1}(\omega)\right], \tag{5}$$

where N_t is the number of samples in task t, and the likelihood $p(y, x|\Theta)$ is defined as in equation 3.

2.1 Sparsely Gated Mixture-of-Variational Layers

As we aim 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 the agents output function $f_\theta(x)$ is composed of multiple layers. Our layer design builds on the sparsely gated Mixture-of-Expert (MoE) layers (Shazeer et al., 2017), which in turn draws on the paradigm introduced by Jacobs et al. (1991). MoEs consist of a set of m experts 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. To save computation time we employ a top-k gating scheme, where only the k experts with

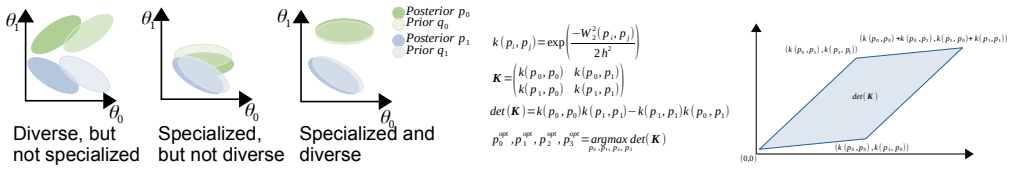

Figure 2: Left: We seek experts that are both specialized, i.e., their posterior p is close to their prior q, and diverse, i.e., posteriors are sufficiently distant from one another. Right: To this effect, we maximize the determinant of the kernel matrix K, effectively filling the feature space. In the 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.

highest gating activation are evaluated and use an additional penalty that encourages gating sparsity (see Section 2.1). In all our experiments we set $k = 1$, to drive expert specialization (see Section 2.2.1) and reduce computation time. We implement the learning objective for task t as layer-wise regularization in the following way:

$$\mathcal{L}^t_{\text{MoVE}} = \sum_{i=1}^{N_t} \Bigg[\mathbb{E}_{p(\Theta)} \left[\mathbf{U}(x_i^t, f_\Theta(x_i^t)) \right] - \sum_{l=1}^{L} \mathbb{E}_{p_t^l(m|x_i^t), p_t^l(\theta|m)} \Bigg[\beta_1 \, \mathbf{D}_{\text{KL}} \left[p_t^l(m|x_i^t) || p_{1:t-1}^l(m|x_i^t) \right] + \beta_2 \, \mathbf{D}_{\text{KL}} \left[p_t^l(\theta|m) || p_{1:t-1}^l(\theta|m) \right] \Bigg] \Bigg], \tag{6}$$

where L is the total number of layers, $\Theta = \{\theta, \{\vartheta_m\}_{m=1}^M\}$ the combined parameters,and the temperature parameters $\beta_{1,2}$ govern the layer-wise trade-off between utility and information-cost.

Thus, we allow for two major generalizations compared to equation 5: in lieu of the log-likelihood we allow for generic utility functions, 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 expert index. This potentially high selection variance hinders expert specialization (see Section 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 6 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. 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 Section 4.

2.2.1 Diversity through Specialization

The relationship between objectives of the form described by equation 6 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, 2020b), but not in the context of continual learning. We assume a two-level hierarchical system of specialized decision-makers where low-level decision-makers $p(m|x)$ select which high-level decision-maker $p(y|m, x)$ serve as experts for a particular input x. By co-optimizing

$$\max_{p(y|x,m), p(m|x)} \mathbb{E}[\mathbf{U}(x, y)] - \beta_1 I(X; M) - \beta_2 I(X; Y|M), \tag{7}$$

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 equation 5 can be regarded as a special case of the information-theoretic objective given by equation 7, if we interpret the prior as the learning strategy of task $t-1$ and the posterior as the strategy of task t, and set $\beta_1 = \beta_2 = 1$. 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.

In addition to the implicit pressures for specialization already implied by equation 6, 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 $-\frac{\beta_3}{N_b}\sum_{x\in\mathcal{B}} H(M)$, where M is the set of experts and X the inputs in a mini-batch $\mathcal{B}$ of size N_b, and β_3 a weight. Thus, by minimizing the marginal entropy $H(M)$ we prefer solutions that minimize the number of active experts. We compute these values batch-wise, as the full entropies over $p(x)$ are not tractable.

2.2.2 Parameter Diversity

Our second diversity formulation 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. P is a DPP 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 $det(K_\emptyset) = 1$. 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. Since P is a probability measure, all principal minors of K must be nonnegative, and thus K itself must be positive semidefinite, which we can achieve by constructing K by a kernel function $k(x_0, x_1)$, such that for any $x_0, x_1 \in \mathcal{X}$:

$$k(x_0, x_1) = \langle \phi(x_0), \phi(x_1) \rangle_{\mathcal{F}}, \tag{8}$$

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

$$W_p(p, q) := \left(\inf_{\gamma\in\Gamma(p,q)} \int_{M\times M} d(x, y)^p \, \mathrm{d}\gamma(x, y) \right)^{1/p}, \tag{9}$$

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

$$k(p, q) = \exp\left(-\frac{W_2^2(p, q)}{2h^2} \right), \tag{10}$$

where h is the kernel width. We show in Appendix B that equation 10 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 $\mathrm{D_{KL}}$ (Cover & Thomas, 2012). Secondly, if p and q are Gaussian and mean-field approximations, i.e., covariance matrices are given by diagonal matrices, i.e., $\Sigma_p = \mathrm{diag}(d_p)$ and $\Sigma_q = \mathrm{diag}(d_q)$, $W_2^2(p, q)$ can be computed in closed form as

$$W_2^2(p, q) = ||\mu_p - \mu_q||_2^2 + ||\sqrt{d_p} - \sqrt{d_q}||_2, \tag{11}$$

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 equation 11 in Appendix A. 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 Figure 2 for an illustration.

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)
Task-Agnostic		
HVCL (ours)	97.50 (±0.33)	97.07 (±0.62)
HVCL w/ GR (ours)	98.60 (±0.35)	97.47 (±0.52)
UGCL w/ BNN (Ebrahimi et al., 2020)	97.70 (±0.03)	92.50 (±0.01)
Brain-inspired RtF (van de Ven et al., 2020)	99.66 (±0.13)	97.31 (±0.04)
HIBNN (Kessler et al., 2021)	91.00 (±2.20)	93.70 (±0.60)
BCL (Raghavan & Balaprakash, 2021)	98.71 (±0.06)	97.51 (±0.05)
TLR (Mazur et al., 2021)	80.64 (±1.25)	
Task-Aware		
DGR+distill. (Shin et al., 2017)	99.59 (±0.40)	97.51 (±0.04)
VCL (Nguyen et al., 2017)	98.50 (±1.78)	96.60 (±1.34)
CURL (Rao et al., 2019)	99.10 (±0.06)	
SAML (Sokar et al., 2021)	97.95 (±0.07)	
DEN (Yoon et al., 2018)	99.26 (±0.01)	

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)
Task-Agnostic		
HVCL (ours)	78.41 (±1.18)	33.10 (±0.62)
HVCL w/ GR (ours)	81.00 (±1.15)	37.20 (±0.52)
CL-DR (Han & Guo, 2021a)	86.72 (±0.30)	25.62 (±0.22)
NCL (Kao et al., 2021)		38.79 (±0.24)
TLR (Mazur et al., 2021)	74.89 (±0.61	
MAS (He & Zhu, 2022)	73.50 (±1.54)	
Task-Aware		
GEM (Lopez-Paz & Ranzato, 2017)	79.10 (±1.60)	40.60 (±1.90)
MCCL (KJ & N Balasubramanian, 2020)	82.90 (±1.20)	43.50 (±0.60)
CCLwFP (Han & Guo, 2021b)	86.33 (±1.47)	65.19 (±0.65)
HAL (Chaudhry et al., 2021)	75.19 (±2.57)	47.88 (±2.76)
SI (Zenke et al., 2017)	63.31 (±3.79)	36.33 (±4.23)
AGEM (Chaudhry et al., 2018b)	74.07 (±0.76)	46.88 (±1.81)

Table 1: Results in the supervised CL benchmarks. 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.

3 EXPERIMENTS

We evaluate our approach in current supervised CL benchmarks in Section 3.1, in a generative learning setting in Section 3.2, and in the CRL setup in Section 3.3. We give experimental details in Appendix C and additional ablation studies in Appendix D.

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 while maintaining performance on older tasks. We evaluate the performance of our method in this setting in split MNIST, permuted MNIST, split CIFAR-10/100 (see Table 1). We follow the domain incremental setup (van de Ven et al., 2020), where task information is not available, but we also compare against task-incremental methods, where the task information is available, to give a complete overview of current methods.

The first benchmark builds on the MNIST dataset. Five binary classification tasks from the MNIST dataset arrive in sequence and at 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. CIFAR-100 is like the CIFAR-10, except it has 100 classes and 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 Table 1 on all supervised learning benchmarks.

3.2 GENERATIVE CONTINUAL LEARNING

Generative CL is a simple but powerful paradigm (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 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 simple 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 We were able to improve our results in the supervised settings by incorporating a generative component, as we show in Table 1. We show additional empirical results in Appendix D.4.

Figure 3: In this figure we show results in the CRL benchmark. To measure the continual learning performance in the RL settings, we normalize rewards and plot the sum of normalized rewards. The maximum of 5.0 indicates no forgetting, while 1.0 shows the total forgetting of old tasks. We comapre against EWC (Kirkpatrick et al., 2017), and UCL (Ahn et al., 2019).

3.3 Continual Reinforcement Learning

In the continual reinforcement learning (CRL) setting, the agent is tasked with finding an optimal policy in sequentially arriving RL problems. To benchmark our method, we follow the experimental protocol of Ahn et al. (2019) and use a series of tasks from the PyBullet environments (Ellenberger, 2018–2019). The environments we selected have different states and action dimensions. This implies we can't use a single model to learn 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.

Here, we extend SAC (Haarnoja et al., 2018) 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 UCL and EWC both require task information to compute task-specific losses. Our results (see Figure 3) show that our approach can sequentially learn new policies while maintaining an acceptable performance on previously seen 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.

4 Discussion

The principle we propose in this work falls into a wider class of methods that deal 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, e.g., 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, 2020a), continual learning (Nguyen et al., 2017; Ahn et al., 2019), and self-supervised learning (Thiam et al., 2021; Tsai et al., 2021).

Currently, there are only few methods that perform well in supervised CL and in CRL (e.g., Ahn et al., 2019; Jung et al., 2020; Cha et al., 2020). These methods require task information, as they 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). This makes our method one of the first task-agnostic CL approaches to drop this requirement while still performing competitively.

The hierarchical structure we employ is a variant of the Mixture-of-Experts (MoE) model (Jacobs et al., 1991), specifically an extension of the sparsely-gated MoE layers (Shazeer et al., 2017). Sparsely-gated MoE layers enforce a balanced load between experts. 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.

Our method is similar to the approach described by Hihn & Braun (2020b) but differs in two key aspects. Firstly, we provide a more stable learning procedure as our layers can readily offer end-to-end training. 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.

Several methods in the current CL literature rest on modular architectures (e.g., Fernando et al., 2017; Collier et al., 2020; Lin et al., 2019; Lee et al., 2020). Lin et al. (2019) propose to condition model parameters on the inputs by learning a (deterministic) grouping function. 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. Lee et al. (2020) propose a MoE model for continual learning, in which the number of experts increases dynamically, utilizing 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.

Recently, several diversity measures have been proposed. Parker-Holder et al. (2020) introduce a DDP-based method based on the different states a given policy may reach. Dai et al. (2021) propose to augment the sampling process in hindsight experience replay (Andrychowicz et al., 2017) with a DPP-diversity bonus. The method we propose differs from previous methods as we define diversity in parameter space instead of the policy outcomes or inputs. Additionally, as we define it on parameters instead of actions, we can apply it straightforwardly to any problem formulation, as our experiments show.

5 Conclusion

We introduced a hierarchical approach to task-agnostic continual learning, derived an application, and extensively evaluated this method in supervised CL and CRL. While we removed the task-information limitation, we achieved results competitive to task-aware and to task-agnostic algorithms. We argued that both VCL and hierarchical VCL have strong connections to an information-theoretic formulation of bounded rationality. We designed a diversity objective that stabilizes learning and further reduces the risk of catastrophic forgetting. Our method builds on generic utility functions, we can apply it independently of the underlying problem, which makes our method one of the first to do so.

Acknowledgment

This work was supported by the European Research Council, grant number ERC-StG-2015-ERC, Project ID: 678082, "BRISC: Bounded Rationality in Sensorimotor Coordination".

REFERENCES

Hongjoon Ahn, Sungmin Cha, Donggyu Lee, and Taesup Moon. Uncertainty-based continual learning with adaptive regularization. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pp. 4392–4402, 2019.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 5055–5065, 2017.

Charles E Antoniak. Mixtures of dirichlet processes with applications to bayesian nonparametric problems. *The annals of statistics*, pp. 1152–1174, 1974.

Dilip Arumugam, Peter Henderson, and Pierre-Luc Bacon. An information-theoretic perspective on credit assignment in reinforcement learning. In *Workshop on Biological and Artificial Reinforcement Learning (NeurIPS 2020)*, 2020.

Yijun Bian and Huanhuan Chen. When does diversity help generalization in classification ensembles. *IEEE Transactions on Cybernetics*, 2021.

Sungmin Cha, Hsiang Hsu, Taebaek Hwang, Flavio Calmon, and Taesup Moon. Cpr: Classifier-projection regularization for continual learning. In *International Conference on Learning Representations*, 2020.

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

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. In *International Conference on Learning Representations*, 2018b.

Arslan Chaudhry, Albert Gordo, Puneet Dokania, Philip Torr, and David Lopez-Paz. Using hindsight to anchor past knowledge in continual learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 6993–7001, 2021.

Mark Collier, Efi Kokiopoulou, Andrea Gesmundo, and Jesse Berent. Routing networks with co-training for continual learning. In *ICML 2020 Workshop on Continual Learning*, 2020.

Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

Tianhong Dai, Hengyan Liu, Kai Arulkumaran, Guangyu Ren, and Anil Anthony Bharath. Diversity-based trajectory and goal selection with hindsight experience replay. In *Pacific Rim International Conference on Artificial Intelligence*, pp. 32–45. Springer, 2021.

Sayna Ebrahimi, Mohamed Elhoseiny, Trevor Darrell, and Marcus Rohrbach. Uncertainty-guided continual learning with bayesian neural networks. In *International Conference on Learning Representations*, 2020.

Alaa El Khatib and Fakhri Karray. Strategies for improving single-head continual learning performance. In Fakhri Karray, Aurélio Campilho, and Alfred Yu (eds.), *Image Analysis and Recognition*, pp. 452–460, Cham, 2019. Springer International Publishing. ISBN 978-3-030-27202-9.

Benjamin Ellenberger. Pybullet gymperium. `https://github.com/benelot/pybullet-gym`, 2018–2019.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*, 2018.

Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. In *arXiv preprint arXiv:1701.08734*, 2017.

Hao Fu, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, Asli Celikyilmaz, and Lawrence Carin. 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, 2019.

Alexandre Galashov, Siddhant M Jayakumar, Leonard Hasenclever, Dhruva Tirumala, Jonathan Schwarz, Guillaume Desjardins, Wojciech M Czarnecki, Yee Whye Teh, Razvan Pascanu, and Nicolas Heess. Information asymmetry in kl-regularized rl. In *Proceedings of the International Conference on Representation Learning*, 2019.

Tim Genewein, Felix Leibfried, Jordi Grau-Moya, and Daniel Alexander Braun. Bounded rationality, abstraction, and hierarchical decision-making: An information-theoretic optimality principle. *Frontiers in Robotics and AI*, 2:27, 2015.

Dibya Ghosh, Avi Singh, Aravind Rajeswaran, Vikash Kumar, and Sergey Levine. Divide-and-conquer reinforcement learning. In *International Conference on Learning Representations*, 2018.

Siavash Golkar, Michael Kagan, and Kyunghyun Cho. Continual learning via neural pruning. In *NeurIPS 2019 Workshop Neuro AI*, 2019.

Jordi Grau-Moya, Felix Leibfried, and Peter Vrancx. Soft q-learning with mutual-information regularization. In *Proceedings of the International Conference on Learning Representations*, 2019.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pp. 1861–1870, 2018.

Xuejun Han and Yuhong Guo. Continual learning with dual regularizations. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 619–634. Springer, 2021a.

Xuejun Han and Yuhong Guo. Contrastive continual learning with feature propagation. *arXiv preprint arXiv:2112.01713*, 2021b.

Jiangpeng He and Fengqing Zhu. Online continual learning via candidates voting. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 3154–3163, 2022.

Heinke Hihn and Daniel A Braun. Hierarchical expert networks for meta-learning. In *4th ICML Workshop on Life Long Machine Learning*, 2020a.

Heinke Hihn and Daniel A. Braun. Specialization in hierarchical learning systems. *Neural Processing Letters*, 52(3):2319–2352, 2020b. ISSN 1573-773X. URL https://doi.org/10.1007/s11063-020-10351-3.

Heinke Hihn, Sebastian Gottwald, and Daniel A Braun. Bounded rational decision-making with adaptive neural network priors. In *IAPR Workshop on Artificial Neural Networks in Pattern Recognition*, pp. 213–225. Springer, 2018.

Heinke Hihn, Sebastian Gottwald, and Daniel A Braun. 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, 2019.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

Sangwon Jung, Hongjoon Ahn, Sungmin Cha, and Taesup Moon. Continual learning with node-importance based adaptive group sparse regularization. *Advances in Neural Information Processing Systems*, 33:3647–3658, 2020.

Ta-Chu Kao, Kristopher Jensen, Gido van de Ven, Alberto Bernacchia, and Guillaume Hennequin. Natural continual learning: success is a journey, not (just) a destination. *Advances in Neural Information Processing Systems*, 34, 2021.

Samuel Kessler, Vu Nguyen, Stefan Zohren, and Stephen J Roberts. Hierarchical indian buffet neural networks for bayesian continual learning. In *Uncertainty in Artificial Intelligence*, pp. 749–759. PMLR, 2021.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, 2015.

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

Joseph KJ and Vineeth N Balasubramanian. Meta-consolidation for continual learning. *Advances in Neural Information Processing Systems*, 33:14374–14386, 2020.

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

Ludmila I Kuncheva. *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons, 2004.

Ludmila I Kuncheva and Christopher J Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning*, 51(2):181–207, 2003.

Soochan Lee, Junsoo Ha, Dongsu Zhang, and Gunhee Kim. A neural dirichlet process mixture model for task-free continual learning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SJxSOJStPr.

Felix Leibfried and Jordi Grau-Moya. Mutual-information regularization in markov decision processes and actor-critic learning. In *Proceedings of the Conference on Robot Learning*, 2019.

Min Lin, Jie Fu, and Yoshua Bengio. Conditional computation for continual learning. In *NeurIPS 2018 Continual Learning Workshop*, 2019.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in neural information processing systems*, pp. 6467–6476, 2017.

Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, pp. 3, 2013.

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

Marcin Mazur, Łukasz Pustelnik, Szymon Knop, Patryk Pagacz, and Przemysław Spurek. Target layer regularization for continual learning using cramer-wold generator. *arXiv preprint arXiv:2111.07928*, 2021.

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pp. 109–165. Elsevier, 1989. doi: 10.1016/S0079-7421(08)60536-8.

Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. In *Proceedings of the International Conference on Representation Learning*, 2017.

Bo Pang, Tian Han, Erik Nijkamp, Song-Chun Zhu, and Ying Nian Wu. Learning latent space energy-based prior model. *Advances in Neural Information Processing Systems*, 33, 2020.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

Jack Parker-Holder, Aldo Pacchiano, Krzysztof M Choromanski, and Stephen J Roberts. Effective diversity in population based reinforcement learning. *Advances in Neural Information Processing Systems*, 33, 2020.

Krishnan Raghavan and Prasanna Balaprakash. Formalizing the generalization-forgetting trade-off in continual learning. *Advances in Neural Information Processing Systems*, 34, 2021.

Dushyant Rao, Francesco Visin, Andrei Rusu, Razvan Pascanu, Yee Whye Teh, and Raia Hadsell. Continual unsupervised representation learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010, 2017.

Jonas Rothfuss, Dennis Lee, Ignasi Clavera, Tamim Asfour, and Pieter Abbeel. Promp: Proximal meta-policy search. In *International Conference on Learning Representations*, 2018.

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. In *NIPS Deep Learning Symposium*, 2016.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29: 2234–2242, 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. In *arXiv preprint arXiv:1707.06347*, 2017.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

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

Ghada Sokar, Decebal Constantin Mocanu, and Mykola Pechenizkiy. Self-attention meta-learner for continual learning. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 1658–1660, 2021.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

Patrick Thiam, Heinke Hihn, Daniel A Braun, Hans A Kestler, and Friedhelm Schwenker. Multi-modal pain intensity assessment based on physiological signals: A deep learning perspective. *Frontiers in Physiology*, 12, 2021.

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

Yao-Hung Hubert Tsai, Yue Wu, Ruslan Salakhutdinov, and Louis-Philippe Morency. Self-supervised learning from a multi-view perspective. In *International Conference on Learning Representations*, 2021.

Gido M van de Ven, Hava T Siegelmann, and Andreas S Tolias. Brain-inspired replay for continual learning with artificial neural networks. *Nature communications*, 11(1):1–14, 2020.

Yeming Wen, Paul Vicol, Jimmy Ba, Dustin Tran, and Roger Grosse. Flipout: Efficient pseudo-independent weight perturbations on mini-batches. In *International Conference on Learning Representations*, 2018.

Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. In *6th International Conference on Learning Representations, ICLR 2018*. International Conference on Learning Representations, ICLR, 2018.

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

A WASSERSTEIN-DISTANCE BETWEEN TWO GAUSSIANS

The W_2^2 distance between two Gaussians is given by

$$W_2^2(p,q) = ||\mu_p - \mu_q||_2^2 + ||\sqrt{d_p} - \sqrt{d_q}||_2, \tag{12}$$

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

$$W_2^2(p,q) = ||\mu_p - \mu_q||_2^2 + \mathcal{B}(\Sigma_p, \Sigma_q), \tag{13}$$

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

$$\mathcal{B}(\Sigma_p, \Sigma_q) = tr(\Sigma_p + \Sigma_q - 2(\Sigma_p^{1/2}\Sigma_q\Sigma_p^{1/2})^{1/2}, \tag{14}$$

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 and 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:

$$W_2^2(p,q) = ||\mu_p - \mu_q||_2^2 + ||\sqrt{d_p} - \sqrt{d_q}||_2. \tag{15}$$

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 equation 15. 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.

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 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 & 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, 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 the same network architecture as suggested by the authors UCL: two layer networks (actor and critics) with 16 units each. 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). 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 replay (except for the replay buffer used by SAC).

D Additional Ablation Experiments

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.

D.1 Investigating Diversity Bonuses

In Section 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 Section 3, but with different bonuses – see Figure 4. In the baseline setup, we used no other objectives as those described by Equation equation 5.

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

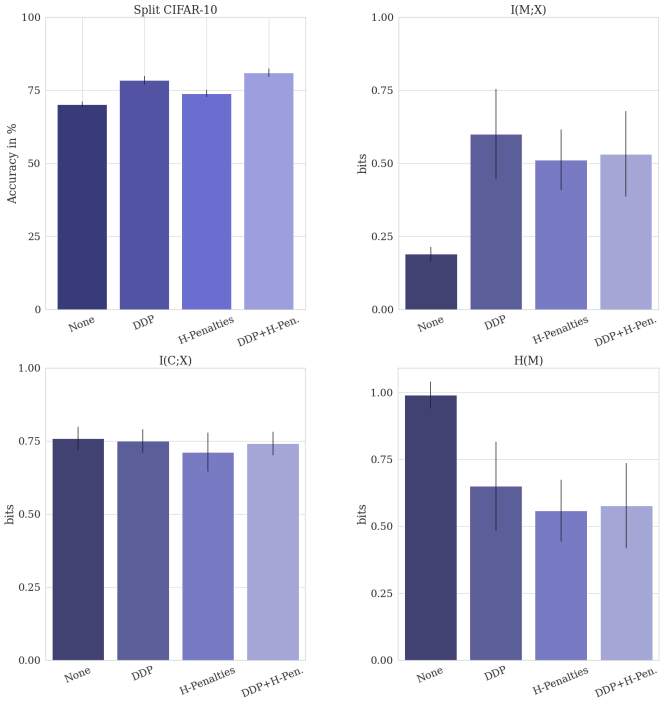

Figure 4: Here we evaluate the proposed diversity measures in the split CIFAR-10 benchmark. We averaged every experiment over three random trials. Information-theoretic quantities $I(M;X), I(C;X)$, and $H(X)$ were measured for each layer and averaged.

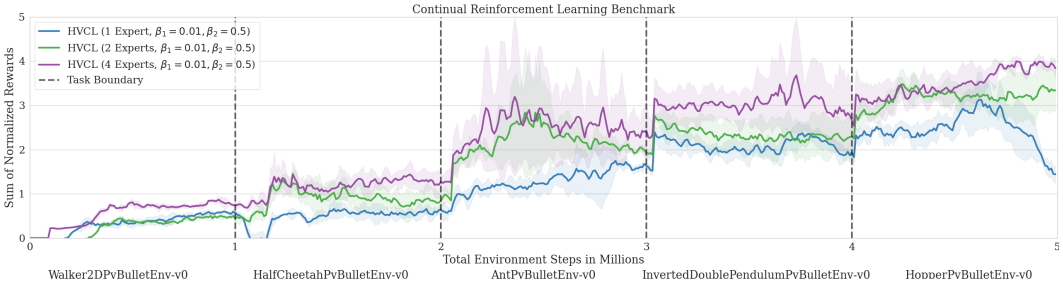

Figure 5: Experimental results for systems with 1, 2, and 4 experts. As expected, adding experts mitigates forgetting. Each curve represents three trials in the continual reinforcement learning domain as described in Section 3.3.

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 ("DDP+H-Pen.") enforces a trade-off between both objectives and yields the best empirical results. We achieve the best results with a DDP diversity bonus combined with an entropy penalty on the expert selection. We average the results of ten random seeds in each setting.

D.2 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. As Figure D.2 illustrates, adding experts to layers increases the number of possible information processing paths through the network. Equipped with

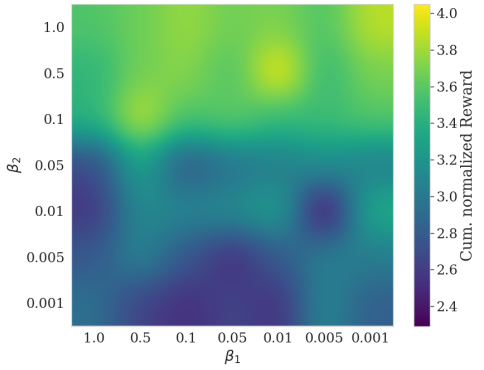

Figure 6: In this figure, we show the influence of the D_{KL} weights β_1 and β_2 in the continual reinforcement learning setting. Setting the expert D_{KL} weight $\beta_2 < 0.1$ results in poor performance, as posteriors deviate too much from their priors. On the other hand, a lower gating D_{KL} weight β_1 allows for a flexible expert allocation and improves performance.

a diverse and specialized set of parameters, each path can be regarded as a distinct sub-network that learns to solve tasks.

D.3 D_{KL} WEIGHTS

As with any hyper-parameter, setting a specific value for $\beta_{1,2}$ 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 $\beta_{1,2}$.

First, by setting $\beta_{1,2}$ 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 $\beta_{1,2}$ to setting a target entropy (see, e.g., Haarnoja et al. (2018) and Grau-Moya et al. (2019) for an example of this approach). Second, employing a schedule for $\beta_{1,2}$, 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 Figure D.3).

D.4 GENERATIVE CL

We introduce a generative approach to continual learning in Section 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 Table 1. We train by integrating artificially generated data into the process by optimizing a mixture loss:

$$L(\theta) = \frac{1}{2|B_t|} \sum_{b \in B_t} \ell(b) + \frac{1}{2|B_{1:t}|} \sum_{b \in |B_{1:t}|} \ell(b), \tag{16}$$

where B_t is batch of data from the current task, $B_{1:t}$ a batch of generated data, and $\ell(b)$ a loss function on the batch b.

Borrowing methods from generative learning, we investigate the performance of our proposed VAE design further, with the main focus on the quality of the generated images. We can not measure the accuracy directly, 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

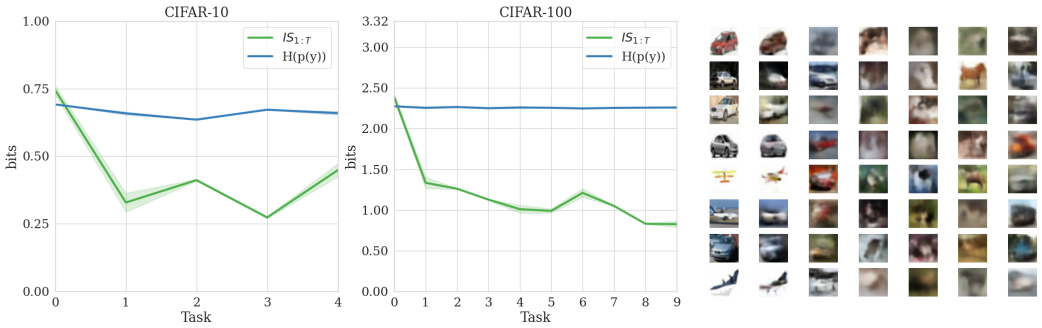

Figure 7: Left and middle: measures for the generator quality. Right: The first two rows depict original images and the corresponding reconstructions. The following five rows are images sampled from the VAE prior after each task (automobiles vs. airplanes, birds vs. cats, deers vs. frogs, dogs vs. horses, and ships vs. trucks).

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

$$IS_T(G_{1:T}) = \mathbb{E}_{x \sim G_{1:T}} \left[\mathrm{D}_{\mathrm{KL}} \left[p_{1:T}(y|x) || p_{1:T}(y) \right] \right], \tag{17}$$

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 split CIFAR-100 setting in Figure 7.