Uncertainty in Deep Learning Miniproject Report

1 Introduction

We study continual learning, a setting where a model encounters a sequence of tasks over time, rather than learning to solve a single fixed task. Formally, given a sequence of tasks $D = \{D_1, \ldots, D_T\}$, where each task $D_t = \{(x_i^t, y_i^t)\}_{i=1}^{N_t}$ consists of input-output pairs with $x_i^t \in \mathcal{X}$ and $y_i^t \in \mathcal{Y}$, the problem is to learn a hypothesis $f_\theta \colon \mathcal{X} \to \mathcal{Y}$, parameterized by θ, that **performs well on all tasks** in D, assuming **access to only one task** D_t at any given time during training. In continual learning, the model must adapt to new tasks and data (*plasticity*), while also retain knowledge from previously learned tasks to avoid *catastrophic forgetting* [4]. This learning setting is important because in many real-world scenarios, such as in autonomous driving or robotics, tasks and data continuously arrive in a stream. Also, in medical applications, we cannot always store or revisit all old data due to privacy, storage, or computational limitations, while continual learning allows us to adapt on-the-fly with limited access to old data. Prior works [4, 6] have shown that if the continual learning setup is approached naively, the methods trained with gradient descent suffer from catastrophic forgetting. Various methods [6, 8, 9, 7, 3] were developed to address this, mainly targeting discriminative learning. We extend the VCL [8] method to regression. While studying the derivation of VCL [8], we observed that the proposed objective can be interpreted as a lower bound on a suitable quantity. Our novel contribution is a tighter lower bound on the same quantity, inspired by IWAE [2] method.

2 Background

Regularization-based methods: Elastic Weight Consolidation (EWC) [6] approximates the posterior of parameters in the previous task with a Gaussian distribution whose mean is given by the learned parameters on the previous task and a diagonal precision given by the diagonal of the Fisher information matrix, evaluated at these parameters. This approximation contains information about which parameters were important for the previous task. When adapting to a new task, a regularization term is added to the new task objective. The regularization term is a difference between the current model parameters and the optimal parameters from the previous task, re-weighted by the Fisher information matrix, evaluated at the previous task. Synaptic Intelligence (SI) [11] tracks (1) the contribution of each individual parameter to reduction in the task loss, and (2) the extent to which each parameter deviates from its learned optimal value on the previous task. These quantities, combined with the difference between current model parameters and the optimal ones for the previous task, form a regularization term, encouraging the model to be accurate on earlier tasks, while learning a new task.

Replay-based methods: Generative Replay (GR) [9] consists of a generator, aiming to learn cumulative distribution of the observed data, and the solver that solves the downstream task/s. When the generator receives the new task data, it firstly generates the samples for previous tasks. Then, it is re-trained to match the distribution of the novel data (from the new task) and the previous task data (previous generator samples). The solver is trained using a mix of real and replayed (generated) data.

Variational methods: When encountering a new task, Variational Continual Learning (VCL) [8] VCL uses the posterior from the previous task as the prior for the new one. In particular, cumulative tasks posterior distribution $p(\theta|D_{1:t})$ is approximated by a variational distribution $q_t(\theta)$ that is trained to match $\frac{1}{Z_t} q_{t-1}(\theta) p(D_t|\theta)$, using Monte Carlo variational inference [5]. This objective arises from a principled derivation: $p(\theta \mid D_{1:t}) \propto p(\theta \mid D_{1:t-1})\, p(D_t \mid \theta) \approx q_{t-1}(\theta)\, p(D_t \mid \theta)$, that is justified by Bayes' theorem. Improved VCL [10] applies seemingly trivial engineering tricks to the original VCL, such as better initialization and longer training. However, these result in considerably better accuracy. Variational Generative Replay (VGR) [3] extends VCL with a GAN that generates synthetic samples from previous tasks. EVCL [1] adds EWC [6] regularization term to the original VCL objective. Temporal-Difference VCL (TD-VCL) [7] presents the alternative objective designed to address compounding approximation errors over successive recursions on $q_t(\theta)$ in the original VCL.

Candidate number: 1068707

3 Methodology

The paper presents two methods, namely VCL (Alg 1) and Coreset VCL (Alg 2). Both methods rely on the fact that $p(\theta|D_{1:t}) \propto p(\theta|D_{1:t-1})p(D_t|\theta)$, by Bayes Theorem. The paper proposes to approximate $p(\theta|D_{1:t})$ with a variational distribution $q_t(\theta)$ that is trained to match $\frac{1}{Z_t}q_{t-1}(\theta)p(D_t|\theta)$, where $q_{t-1}(\theta)$ is a variational approximation of $p(\theta|D_{1:t-1})$ and Z_t is the intractable constant. These variational approximations are built recursively using Monte Carlo variational inference [5], following Alg 1 or Alg 2. Parameters θ are initialized at the maximum likelihood estimate of θ on the initial task. Coreset VCL extends the VCL method by incorporating a small episodic memory - called a coreset. Concretely, the coreset is the subset of the training data held out during training but reintroduced only during evaluation. More precisely, prior to evaluation on the test data, the model is fine-tuned on the coreset data. However, the version of the model parameters *before* fine-tuning is used for subsequent approximations. The paper presents two strategies for selecting the coreset: "random" and "k-center". The "random" strategy selects the coreset uniformly at random from the training set, while "k-center" builds the coreset greedily and iteratively by adding, at each step, the training example that is farthest from the current coreset. In this work, we reproduce both methods and coreset selection strategies.

4 Novel extension

Required extension: Firstly, we assume that the goal is to solve PermutedMNIST and SplitMNIST using identical neural architectures, but with a regression objective instead of classification. For full generality, we present our method and its derivation assuming multi-head regression, since single-head regression is merely a special case. Our network is based on the discriminative network from VCL [8]. It consists of the lower-level feature extractor ϕ_{θ_S} and the task-specific "regression head network", predicting the parameters of our probabilistic model of $y|\boldsymbol{\theta}, \mathbf{x}$, for task t, parameterized by weights $\theta_t = \{\mathbf{W}_{\mu,t}, \mathbf{W}_{\Sigma,t}, \boldsymbol{b}_{\mu,t}, \boldsymbol{b}_{\Sigma,t}\}$. The full set of parameters of our network is therefore $\theta = \{\theta_S, \theta_1 \dots \theta_T\}$. Following the generative story for regression discussed in the course, we assume the following task-specific generative model of regression target $\mathbf{y}$, given parameters $\boldsymbol{\theta}$ and input $\mathbf{x}$,

$$p_t(\mathbf{y} \mid \boldsymbol{\theta}, \mathbf{x}) = \mathcal{N}\Big(\mathbf{y}\,;\, \mu_{\theta_t}(\phi_{\theta_S}(\boldsymbol{x})),\, \boldsymbol{\Sigma}_{\theta_t}(\phi_{\theta_S}(\boldsymbol{x}))\Big),$$

where $\mu_{\boldsymbol{\theta}_t}(\phi_{\theta_S}(\mathbf{x}))$ is the task-specific D-dimensional mean, and $\boldsymbol{\Sigma}_{\boldsymbol{\theta}_t}(\phi_{\theta_S}(\mathbf{x}))$ is the task-specific $D \times D$ covariance, both possibly depending on $\mathbf{x}$. To model uncertainty in model parameters, we use the Gaussian mean-field approximation from the original paper. Thus, we can model both epistemic uncertainty (through VCL's mean-field approximation) and task-specific aleatoric uncertainty. Consequently, the full derivation from the paper, until and including the following line, holds for us.

$$\mathcal{L}^t_{VCL}(q_t(\theta)) = \sum_{n=1}^{N_t} \mathbb{E}_{\boldsymbol{\theta}\sim q_t(\theta)} \left[\log p(y_t^{(n)}|\boldsymbol{\theta}, \mathbf{x}_t^{(n)})\right] - \mathrm{KL}(q_t(\theta)\|q_{t-1}(\theta))$$

For notational clarity, define $\mu_{\boldsymbol{\theta},t}(\mathbf{x}) = \mu_{\boldsymbol{\theta}_t}(\phi_{\theta_S}(\mathbf{x}))$ and similarly $\boldsymbol{\Sigma}_{\boldsymbol{\theta},t}(\mathbf{x}) = \boldsymbol{\Sigma}_{\boldsymbol{\theta}_t}(\phi_{\theta_S}(\mathbf{x}))$. Under our model, $p(y_t^{(n)}|\boldsymbol{\theta}, \mathbf{x}_t^{(n)}) = p_t(\mathbf{y}_t^{(n)} \mid \boldsymbol{\theta}, \mathbf{x}_t^{(n)})$, so the above stated $\mathcal{L}_t^{\mathrm{VCL}}(q_t(\boldsymbol{\theta}))$ objective becomes

$$\begin{aligned}\mathcal{L}_t^{\mathrm{VCL}}(q_t(\boldsymbol{\theta})) = &-\frac{1}{2}\sum_{n=1}^{N_t} \mathbb{E}_{q_t(\boldsymbol{\theta})}\left[\log\det\Big(\boldsymbol{\Sigma}_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big) + \Big(\mathbf{y}_t^{(n)} - \mu_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big)^\top \Big(\boldsymbol{\Sigma}_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big)^{-1}\Big(\mathbf{y}_t^{(n)} - \mu_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big)\right] \\ &- \mathrm{KL}\Big(q_t(\boldsymbol{\theta}) \,\|\, q_{t-1}(\boldsymbol{\theta})\Big) + \text{const.}\end{aligned}$$

We provide the full derivation in the supplementary. In our experiments, we choose not to model aleatoric uncertainty. In other words, for simplicity of implementation, we assume $\Sigma_\theta(\boldsymbol{x}) = \boldsymbol{I}_{D\times D}$. However, we explain how to implement the general case of our solution in the supplementary.

Novel Theoretical Extension: We observe that the VCL objective can be interpreted as ELBO. Instead of using VAE-like [5] derivation used in the original paper, we apply importance sampling ideas from IWAE [2]. Our contribution is the novel alternative loss, derived in the supplementary. We prove that our objective, $\mathcal{L}^t_{\text{IW-VCL}}$, is a tighter lower bound on the log of the expected task likelihood under the prior, with desirable convergence properties as the number of samples K increases.

$\mathcal{L}^{t,K}_{\text{IW-VCL}}(q_t(\theta)) = \mathbb{E}_{\theta_{1:K}\sim q_t(\theta)}\left[\log\left(\sum_{k=1}^K \exp\left(\sum_{n=1}^{N_t}\log p\big(y_t^{(n)} \mid \theta_k, x_t^{(n)}\big) + \log\frac{q_{t-1}(\theta_k)}{q_t(\theta_k)}\right)\right) - \log K\right].$

5 Experiments

We successfully reproduced both the discriminative and generative experiments on MNIST-derived datasets, namely Figure 2, Figure 3 and Figure 4 in the paper, as well as the subset of Figures 6 and 7. For computational reasons, we limited our evaluation to the proposed method and the two strongest baselines: EWC and SI. Due to page limit, we provide detailed reproduction methodology in the supplementary material and focus here on the reproduced results and their analysis. As the accuracy differences between methods were small, especially on SplitMNIST, we provide all the plots in the supplementary material, allowing us to present larger plots better matching those in the paper. CODE

Discriminative

Figure 2 shows the average test set accuracy on all observed tasks, on Permuted MNIST. As reported in the paper, VCL outperforms EWC and SI. For us, after 10 tasks, VCL achieves 90.46% average accuracy, VCL + Random Coreset 92.61%, VCL + K-Center Coreset 93.31%, while EWC achieves 84.10% and SI 87.91%. These values almost perfectly match the reported values, except for SI where we got almost 88% instead of the reported 86%. Figure 3 shows that increasing coreset size improves the accuracy, although it asympotes for the larger coresets. With the coreset of size 5000, VCL achieves 95.59% average accuracy, as reported in the paper. Figure 4 shows test set accuracy on all tasks for the Split MNIST experiment. For computational reasons, we report averages over three runs with different seeds, instead of ten runs as in the paper. VCL outperforms EWC but performs worse than SI. Interestingly, the gap between VCL and EWC is much smaller than in the paper. This could be attributed to the author's implementation of EWC, because our VCL achieves better 97.92% accuracy than the reported 97.0%, while our EWC achieves much better 96.11% accuracy than the reported 63.1%. For both VCL and SI, the task-wise accuracy trends we observed closely aligned with the reported results, enforcing our confidence in the implementation and the analysis presented.

Generative

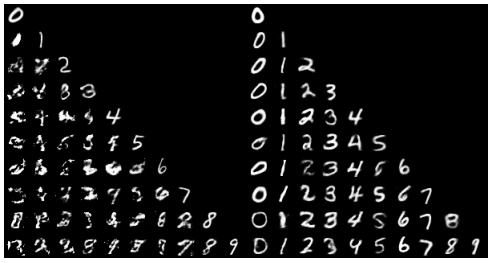

Figure 1: Naive VAE (left) vs VCL-VAE (right).

Due to limited resources, we were only able to implement and evaluate the naive VAE and VCL. Figure 1 shows that, unlike the naive VAE, VCL-VAE can successfully generate previously seen digits after observing new ones. This is also reflected in the per-digit test log-likelihood plot (Figure 6.2.1), which shows likelihoods very similar to the reported values. The task-wise test log-likelihood trend we observed aligns with the reported results, further reinforcing our confidence in the correctness of the implementation and the efficacy of the proposed method (VCL).

Required Extension: We evaluate all the methods using identical architectures and hyperparameters as in the case of classification, but we train with regression loss. Figure 6 and 7 show average test RMSE [1]. All methods degrade as the number of observed tasks increases. However, VCL consistently outperforms EWC and remains competitive with SI. On PermutedMNIST, Coreset VCL and SI are the best. But, on SplitMNIST, SI outperforms all other methods by a relatively large margin of 0.1.

Conclusion: We conclude that the proposed method performs exactly as it is reported in the paper. However, additional information on how the baselines were implemented and evaluated is needed, as subtle implementation details could impact their accuracy (see supplementary material for further details). Thus, the gap between the proposed method and the baselines may be smaller than reported, particularly on SplitMNIST (see Figure 4). Reproducing the results in the paper is challenging due to VCL's sensitivity on the variational approximation initialization. Without the initialization details from the follow-up paper [10], we were unable to perfectly reproduce the proposed method. Nevertheless, we are confident in our reproduction since our results closely match those reported. A key limitation of our work is that we did not experimentally evaluate our theoretical extension, IW-VCL. Given more time, we would have experimentally evaluated our proposed theoretical extension, IW-VCL, and likely explored more recent and challenging benchmarks, such as ImageNet or CIFAR.

[1] Given $\{\hat{\boldsymbol{y}}_i\}_{i=1}^N$ with ground truths $\{\boldsymbol{y}_i\}_{i=1}^N$, $\text{RMSE}(\{\hat{\boldsymbol{y}}_i\}_{i=1}^N, \{\boldsymbol{y}_i\}_{i=1}^N) = \sqrt{\frac{1}{N}\sum_{i=1}^N ||\hat{\boldsymbol{y}}_i - \boldsymbol{y}_i||_2^2}$

6 Supplementary Material

6.1 Discriminative Results

6.1.1 PermutedMNIST

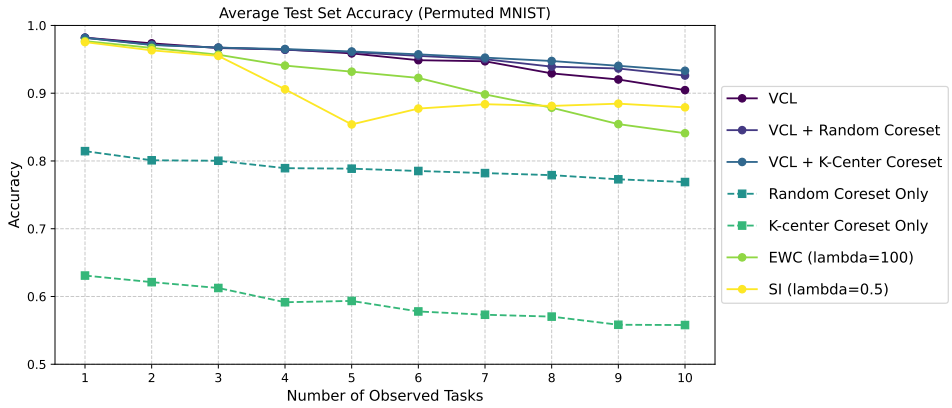

Figure 2: Average test set accuracy on all observed tasks in the Permuted MNIST experiment. This corresponds to Figure 2 in VCL [8].

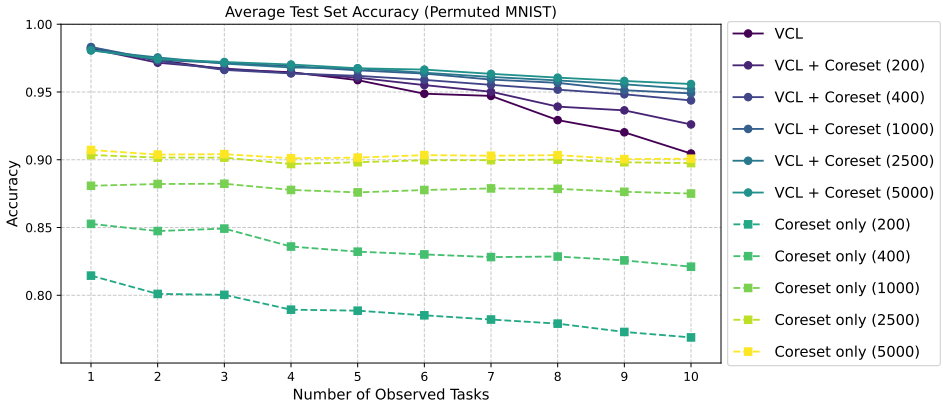

Figure 3: Comparison of the effect of coreset sizes in the Permuted MNIST experiment. This corresponds to Figure 3 in VCL [8].

6.1.2 SplitMNIST

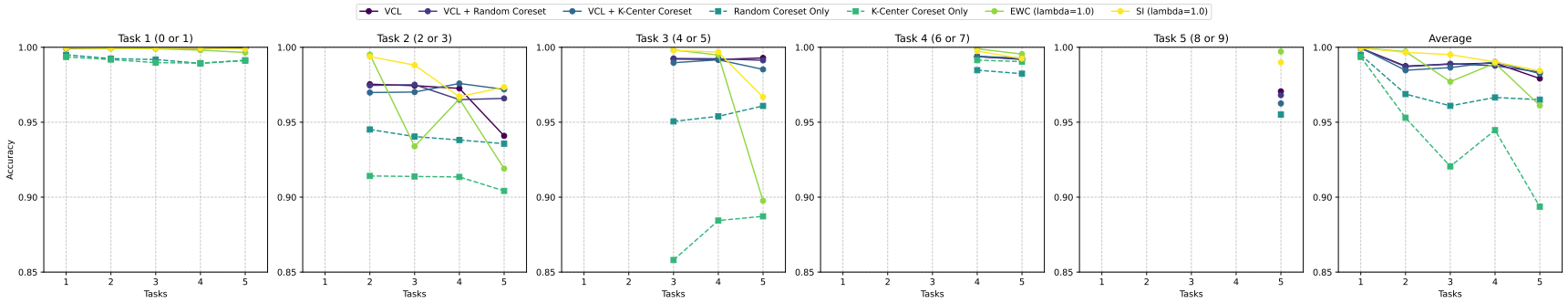

Figure 4: Test set accuracy on all tasks for the Split MNIST experiment. The last column shows the average accuracy over all tasks. Corresponds to Figure 4 in VCL.

6.2 Generative results

6.2.1 MNIST

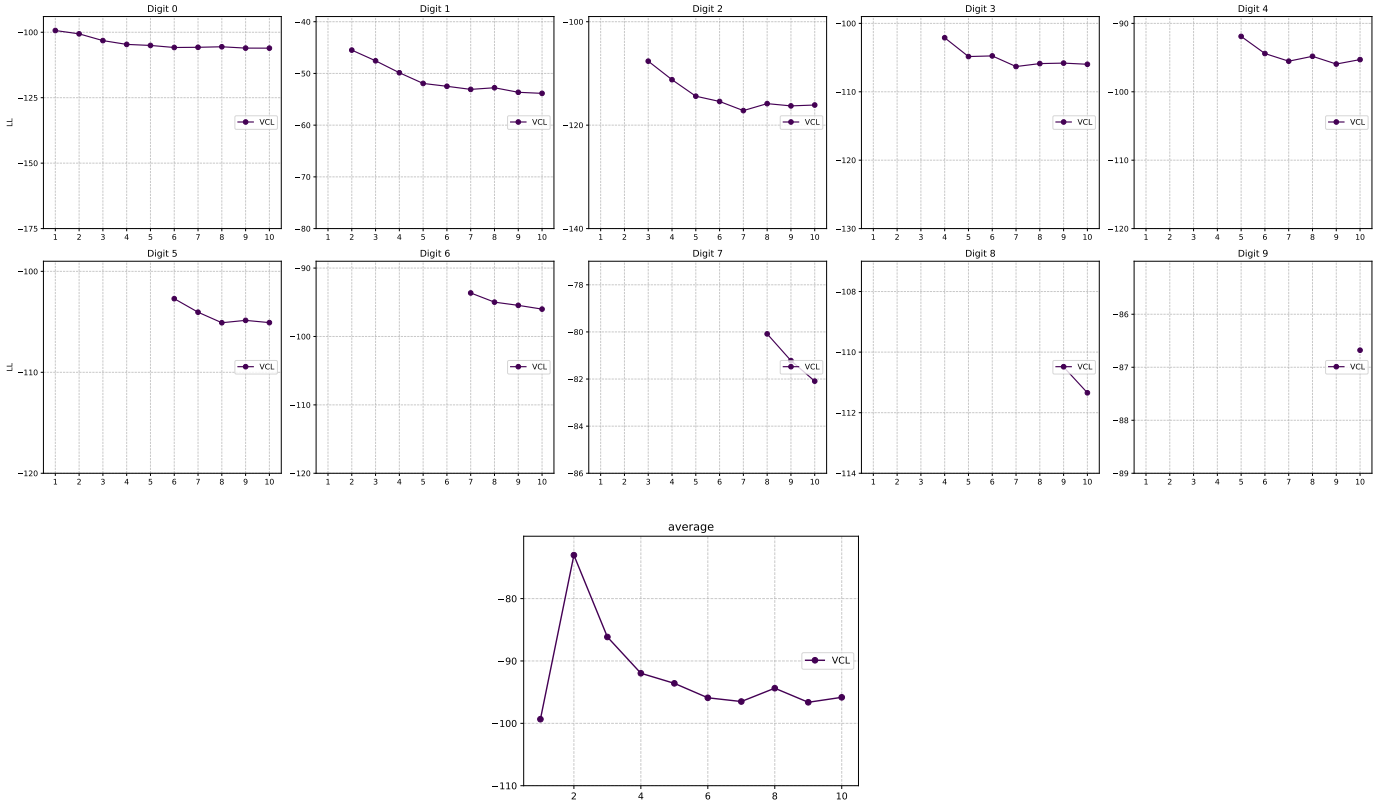

Figure 5: Test Log-Likelihood of VCL-VAE. Corresponds to Figure 7 (a) in the original paper [8].

6.3 Required Extension

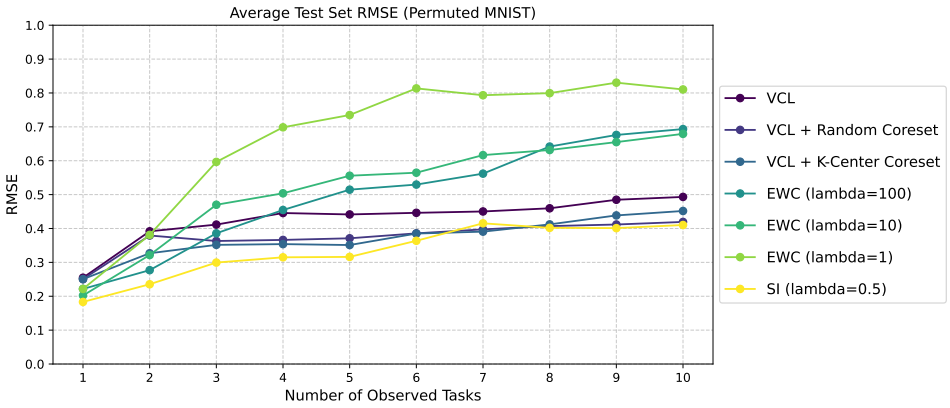

Figure 6: Average test set RMSE on all observed tasks in the Permuted MNIST experiment.

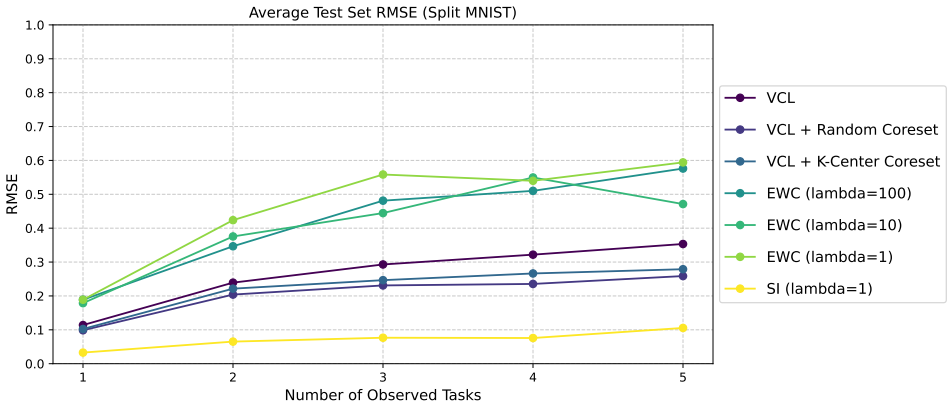

Figure 7: Average test set RMSE on all observed tasks in the Split MNIST experiment.

6.4 General reproduction details and link to implementation

We implemented VCL and EWC from scratch in PyTorch, following the architectures and initialization details outlined in the original paper [8]. However, we initialized the variational approximations using the method suggested in the follow-up paper [10], as we found this initialization to be crucial for the reported accuracy. Without it, our implementation with the default PyTorch weight initialization performed consistently worse. For generative models, the original paper does not specify a batch size, so we experimented with batch sizes of 32, 64 and 128. We found that a batch size of 64 yielded results closest to those reported and therefore report these results in our work. For PermutedMNIST, all methods were run with the same seed (17). For SplitMNIST, due to computational constraints, we report average metrics obtained from three independent runs using seeds 0, 137, and 256.

Interestingly, our initial implementation of SI in PyTorch consistently performed worse than reported in the original paper, and this is something we still cannot explain. To debug our implementation, we found an existing SI implementation at `https://github.com/spiglerg/TF_ContinualLearningViaSynapticIntelligence`, which is specific to PermutedMNIST. Initially, this implementation performed significantly better than ours, though we still don't understand why. We modified this implementation by incorporating architectures from the original paper, adjusting the initialization to match the one discussed above, and aligning our evaluation protocol with that used in VCL and EWC implementations. Additionally, we extended this implementation to SplitMNIST by implementing a multi-head network.

The link to our code repository is `https://github.com/deeplearningtester/udl`.

6.5 VCL

6.5.1 Proposed methods for reproduction

Algorithm 1 Variational Continual Learning (VCL)

Input: Prior $p(\theta)$.
Output: Variational and predictive distributions at each step $\{q_t(\theta), p(y^*|\boldsymbol{x}^*, D_{1:t})\}_{t=1}^T$.

$q_0(\theta) \leftarrow p(\theta)$ ▷ Initialize variational approximation
for $t = 1$ **to** T **do**
  $D_t \leftarrow$ next dataset ▷ Observe new dataset
  $q_t(\theta) \leftarrow \arg\min_{q \in \mathcal{Q}} \mathrm{KL}\left(q(\theta) \parallel \frac{1}{Z} q_{t-1}(\theta) p(D_t|\theta)\right)$ ▷ Update variational dist
  $p(y^*|\boldsymbol{x}^*, D_{1:t}) \leftarrow \mathbb{E}_{q_t(\theta)}[p(y^*|\theta, \boldsymbol{x}^*)]$ ▷ Predict at test input $\boldsymbol{x}^*$
end for

Algorithm 2 Coreset Variational Continual Learning (VCL)

Input: Prior $p(\theta)$.
Output: Variational and predictive distributions at each step $\{q_t(\theta), p(y^*|\boldsymbol{x}^*, D_{1:t})\}_{t=1}^T$.

$C_0 \leftarrow \emptyset, \tilde{q}_0 \leftarrow p$ ▷ Initialize the coreset and variational approximation
for $t = 1$ **to** T **do**
  $D_t \leftarrow$ next dataset
  $C_t \leftarrow$ update_coreset(C_{t-1}, D_t) ▷ Random or Greedy K-Center update
  $\tilde{q}_t(\theta) \leftarrow \arg\min_{q \in \mathcal{Q}} \mathrm{KL}\left(q(\theta) \parallel \frac{1}{Z} \tilde{q}_{t-1}(\theta) p(D_t \cup C_{t-1} \setminus C_t|\theta)\right)$ ▷ Update variational dist
  $q_t(\theta) \leftarrow \arg\min_{q \in \mathcal{Q}} \mathrm{KL}\left(q(\theta) \parallel \frac{1}{Z} \tilde{q}_t(\theta) p(C_t|\theta)\right)$ ▷ Compute predictive variational dist
  $p(y^*|\boldsymbol{x}^*, D_{1:t}) \leftarrow \mathbb{E}_{q_t(\theta)}[p(y^*|\theta, \boldsymbol{x}^*)]$ ▷ Predict at test input $\boldsymbol{x}^*$
end for

6.6 EWC

We use the following identity to estimate Fisher information using PyTorch autograd.

$$F = \mathbb{E}\left[-\frac{\partial^2(\log(p(\theta|D)))}{\partial\theta^2}\bigg|_{\theta_D^*}\right] = \mathbb{E}\left[\left(\left(\frac{\partial(\log(p(\theta|D)))}{\partial\theta}\right)\left(\frac{\partial(\log(p(\theta|D)))}{\partial\theta}\right)^T\right)\bigg|_{\theta_D^*}\right] \quad (1)$$

6.6.1 Implementation subtleties

There are a few subtleties associated with the implementation of Equation (1). For example, when computing $\log p(\theta|D)$, should we use the ground truth labels y, the labels sampled from model's distribution using categorical sampling, or should we simply choose the pseudo label given by the argmax under the model? We did not have enough compute resources to ablate this. We went for the simplest option - choosing the ground truth y. Also, the paper mentions that a certain number of examples is used to estimate Fisher information. However, is this estimation data held out during the training or not? We used data within the training set, but it is worth considering the alternative option.

6.7 Required Extension

6.7.1 Our probabilistic model

Following the generative story for regression discussed in the course, we assume the following task-specific generative model of $\mathbf{y}$, given $\boldsymbol{\theta}$ and $\mathbf{x}$,

$$p_t(\mathbf{y} \mid \boldsymbol{\theta}, \mathbf{x}) = \mathcal{N}\Big(\mathbf{y}\,;\, \mu_{\theta_t}(\phi_{\theta_S}(\boldsymbol{x})),\, \boldsymbol{\Sigma}_{\theta_t}(\phi_{\theta_S}(\boldsymbol{x}))\Big),$$

where $\phi_{\theta_S}(\mathbf{x})$ is the shared feature extractor discussed in the main body, $\mu_{\boldsymbol{\theta}_t}(\phi_{\theta_S}(\mathbf{x}))$ is the task-specific D-dimensional mean, and $\boldsymbol{\Sigma}_{\boldsymbol{\theta}_t}(\phi_{\theta_S}(\mathbf{x}))$ is a task-specific $D \times D$ covariance, both possibly depending on $\mathbf{x}$. For notational clarity, define $\mu_{\boldsymbol{\theta},t}(\mathbf{x}) = \mu_{\boldsymbol{\theta}_t}(\phi_{\theta_S}(\mathbf{x}))$ and similarly $\boldsymbol{\Sigma}_{\boldsymbol{\theta},t}(\mathbf{x}) = \boldsymbol{\Sigma}_{\boldsymbol{\theta}_t}(\phi_{\theta_S}(\mathbf{x}))$. Then, for a single observation in task t:

$$\log p_t(\mathbf{y}_t^{(n)} \mid \boldsymbol{\theta}, \mathbf{x}_t^{(n)}) = -\frac{D}{2}\log(2\pi) - \frac{1}{2}\log\det\Big(\boldsymbol{\Sigma}_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big) - \frac{1}{2}\Big(\mathbf{y}_t^{(n)} - \mu_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big)^\top \Big(\boldsymbol{\Sigma}_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big)^{-1}\Big(\mathbf{y}_t^{(n)} - \mu_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big)$$

Since the first term (the constant $-\frac{D}{2}\log(2\pi)$) does not depend on $\boldsymbol{\theta}$, it can be removed from the optimization. Hence, up to an additive constant, we have

$$\log p_t(\mathbf{y}_t^{(n)} \mid \boldsymbol{\theta}, \mathbf{x}_t^{(n)}) \propto -\frac{1}{2}\log\det\Big(\boldsymbol{\Sigma}_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big) - \frac{1}{2}\Big(\mathbf{y}_t^{(n)} - \mu_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big)^\top \Big(\boldsymbol{\Sigma}_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big)^{-1}\Big(\mathbf{y}_t^{(n)} - \mu_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big).$$

6.7.2 Application to VCL Objective

For each data point $(\mathbf{x}_t^{(n)}, \mathbf{y}_t^{(n)})$ in task t, the expected log-likelihood under the variational posterior $q_t(\boldsymbol{\theta})$ is

$$\mathbb{E}_{q_t(\boldsymbol{\theta})}\Big[\log p_t\big(\mathbf{y}_t^{(n)} \mid \boldsymbol{\theta}, \mathbf{x}_t^{(n)}\big)\Big] \propto -\frac{1}{2}\mathbb{E}_{q_t(\boldsymbol{\theta})}\Big[\log\det\Big(\boldsymbol{\Sigma}_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big) + \Delta_{t,n}(\boldsymbol{\theta})\Big]$$

where we define the squared Mahalanobis distance

$$\Delta_{t,n}(\boldsymbol{\theta}) := \Big(\mathbf{y}_t^{(n)} - \mu_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big)^\top \Big(\boldsymbol{\Sigma}_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big)^{-1}\Big(\mathbf{y}_t^{(n)} - \mu_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big).$$

Summing over the N_t examples and applying the linearity of expectation yields

$$\sum_{n=1}^{N_t}\mathbb{E}_{q_t(\boldsymbol{\theta})}\Big[\log p_t\big(\mathbf{y}_t^{(n)} \mid \boldsymbol{\theta}, \mathbf{x}_t^{(n)}\big)\Big] \propto -\frac{1}{2}\sum_{n=1}^{N_t}\mathbb{E}_{q_t(\boldsymbol{\theta})}\Big[\log\det\Big(\boldsymbol{\Sigma}_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big) + \Delta_{t,n}(\boldsymbol{\theta})\Big].$$

The final VCL objective combines the task-specific terms with the KL regularization:

$$\begin{aligned}\mathcal{L}_t^{\mathrm{VCL}}(q_t(\boldsymbol{\theta})) = &-\frac{1}{2}\sum_{n=1}^{N_t}\mathbb{E}_{q_t(\boldsymbol{\theta})}\Big[\log\det\Big(\boldsymbol{\Sigma}_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big) + \Big(\mathbf{y}_t^{(n)} - \mu_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big)^\top \Big(\boldsymbol{\Sigma}_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big)^{-1}\Big(\mathbf{y}_t^{(n)} - \mu_{\boldsymbol{\theta},t}(\mathbf{x}_t^{(n)})\Big)\Big] \\ &- \mathrm{KL}\Big(q_t(\boldsymbol{\theta}) \,\|\, q_{t-1}(\boldsymbol{\theta})\Big) + \text{const.}\end{aligned}$$

6.7.3 Implementation

The main difficulty in implementation is predicting $\boldsymbol{\Sigma}$, because we do not want to evaluate determinants or perform matrix inversion, due to their prohibitive computational cost. To predict $\boldsymbol{\Sigma}$, we take advantage of the fact it is a covariance matrix, and hence positive definite. Consequently, $\boldsymbol{\Sigma}^{-1}$ is also positive definite. By elementary linear algebra, it admits Cholesky decomposition, $\boldsymbol{\Sigma}^{-1} = \boldsymbol{L}\boldsymbol{L}^T$, where $\boldsymbol{L}$ is a **lower triangular** matrix. By elementary linear algebra, $\det(\boldsymbol{\Sigma}) = \frac{1}{\det(\boldsymbol{\Sigma}^{-1})}$ and $\det(\boldsymbol{\Sigma}^{-1})$ is simply the product of diagonal entries of $\boldsymbol{L}$. Thus, we predict $\boldsymbol{L}$ with a neural network.

6.8 IW-VCL

6.8.1 VCL Objective as ELBO

The VCL objective for task t is given by

$$\mathcal{L}_{\text{VCL}}^t(q_t(\theta)) = \sum_{n=1}^{N_t} \mathbb{E}_{q_t(\theta)} \left[\log p\big(y_t^{(n)} \mid \theta, x_t^{(n)}\big) \right] - \text{KL}\big(q_t(\theta) \,\|\, q_{t-1}(\theta)\big). \tag{2}$$

Recall that the Kullback-Leibler (KL) divergence is defined by

$$\text{KL}\big(q_t(\theta) \,\|\, q_{t-1}(\theta)\big) = \mathbb{E}_{q_t(\theta)} \left[\log \frac{q_t(\theta)}{q_{t-1}(\theta)} \right].$$

Multiplying by -1 yields,

$$-\text{KL}\big(q_t(\theta) \,\|\, q_{t-1}(\theta)\big) = \mathbb{E}_{q_t(\theta)} \left[\log \frac{q_{t-1}(\theta)}{q_t(\theta)} \right].$$

Substituting this back into Equation (2), we obtain:

$$\begin{aligned}
\mathcal{L}_{\text{VCL}}^t(q_t(\theta)) &= \sum_{n=1}^{N_t} \mathbb{E}_{q_t(\theta)} \left[\log p\big(y_t^{(n)} \mid \theta, x_t^{(n)}\big) \right] + \mathbb{E}_{q_t(\theta)} \left[\log \frac{q_{t-1}(\theta)}{q_t(\theta)} \right] \\
&= \mathbb{E}_{q_t(\theta)} \left[\sum_{n=1}^{N_t} \log p\big(y_t^{(n)} \mid \theta, x_t^{(n)}\big) + \log \frac{q_{t-1}(\theta)}{q_t(\theta)} \right] \qquad (3) \\
&= \mathbb{E}_{q_t(\theta)} \left[\log \left(\prod_{n=1}^{N_t} p\big(y_t^{(n)} \mid \theta, x_t^{(n)}\big) \frac{q_{t-1}(\theta)}{q_t(\theta)} \right) \right]. \qquad (4)
\end{aligned}$$

By Jensen's inequality,

$$\mathcal{L}_{\text{VCL}}^t(q_t(\theta)) \leq \log \left(\mathbb{E}_{q_{t-1}(\theta)} \left[\prod_{n=1}^{N_t} p\big(y_t^{(n)} \mid \theta, x_t^{(n)}\big) \right] \right) = \log \mathbb{E}_{q_{t-1}(\theta)} \left[p\big(D_t \mid \theta\big) \right],$$

which is the $\log$ of the expected likelihood of the task D_t under the prior $q_{t-1}(\theta)$. Since this quantity is not interpretable, we provide an interpretation of this quantity in the Subsection 6.8.5.

Can we derive a lower bound on $\log \mathbb{E}_{q_{t-1}(\theta)} \left[p\big(D_t \mid \theta\big) \right]$ **tighter** than $\mathcal{L}_{\text{VCL}}^t(q_t(\theta))$?

6.8.2 Deriving the alternative objective

Define

$$w(\theta) = \frac{q_{t-1}(\theta)}{q_t(\theta)} \prod_{n=1}^{N_t} p\big(y_t^{(n)} \mid \theta, x_t^{(n)}\big). \tag{5}$$

Recall that if $\theta_1, \ldots, \theta_K$ are K independent and identically distributed samples from $q_t(\theta)$, then by linearity of expectation,

$$\mathbb{E}_{\theta_{1:K} \sim q_t(\theta)} \left[\frac{1}{K} \sum_{k=1}^{K} w(\theta_k) \right] = \frac{1}{K} \sum_{k=1}^{K} \mathbb{E}_{\theta_k \sim q_t(\theta)} \left[w(\theta_k) \right] = \mathbb{E}_{\theta \sim q_t(\theta)} \left[w(\theta) \right].$$

Applying this result and Jensen's inequality yields

$$\begin{aligned}
\log \mathbb{E}_{\theta \sim q_{t-1}(\theta)} \left[p\big(D_t \mid \theta\big) \right] &= \log \mathbb{E}_{\theta \sim q_t(\theta)} [w(\theta)] \\
&= \log \mathbb{E}_{\theta_{1:K} \sim q_t(\theta)} \left[\frac{1}{K} \sum_{k=1}^{K} w(\theta_k) \right] \\
&\geq \mathbb{E}_{\theta_{1:K} \sim q_t(\theta)} \left[\log \left(\frac{1}{K} \sum_{k=1}^{K} w(\theta_k) \right) \right].
\end{aligned} \tag{6}$$

Plugging back the definition of $w(\theta_k)$ gives us the alternative objective,

$$\mathcal{L}^{t,K}_{\text{IW-VCL}}(q_t(\theta)) = \mathbb{E}_{\theta_{1:K}\sim q_t(\theta)}\left[\log\left(\frac{1}{K}\sum_{k=1}^{K}\frac{q_{t-1}(\theta_k)}{q_t(\theta_k)}\prod_{n=1}^{N_t}p\big(y_t^{(n)}\mid\theta_k,x_t^{(n)}\big)\right)\right]. \tag{7}$$

In practice, due to numerical stability, we work with log densities. To this end, we apply the following logsumexp trick.

$$\begin{aligned}
\mathcal{L}^{t,K}_{\text{IW}-\text{VCL}}(q_t(\theta)) &= \mathbb{E}_{\theta_{1:K}\sim q_t(\theta)}\left[\log\left(\frac{1}{K}\sum_{k=1}^{K}\frac{q_{t-1}(\theta_k)}{q_t(\theta_k)}\prod_{n=1}^{N_t}p\big(y_t^{(n)}\mid\theta_k,x_t^{(n)}\big)\right)\right]\\
&= \mathbb{E}_{\theta_{1:K}\sim q_t(\theta)}\left[\log\left(\frac{1}{K}\sum_{k=1}^{K}\exp\left(\sum_{n=1}^{N_t}\log p\big(y_t^{(n)}\mid\theta_k,x_t^{(n)}\big)+\log q_{t-1}(\theta_k)-\log q_t(\theta_k)\right)\right)\right]\\
&= \mathbb{E}_{\theta_{1:K}\sim q_t(\theta)}\left[\log\left(\sum_{k=1}^{K}\exp\left(\sum_{n=1}^{N_t}\log p\big(y_t^{(n)}\mid\theta_k,x_t^{(n)}\big)+\log\frac{q_{t-1}(\theta_k)}{q_t(\theta_k)}\right)\right)-\log K\right]
\end{aligned}$$

6.8.3 Remark

When $K=1$, $\mathcal{L}^{t,1}_{\text{IW}-\text{VCL}}(q_t(\theta)) = \mathcal{L}^{t}_{\text{VCL}}(q_t(\theta))$, so the original $\mathcal{L}^{t}_{\text{VCL}}(q_t(\theta))$ objective is a special case of the proposed $\mathcal{L}^{t,K}_{\text{IW}-\text{VCL}}(q_t(\theta))$.

6.8.4 $\mathcal{L}^{t}_{\textbf{IW-VCL}}(q_t(\theta))$ is at least as tight as $\mathcal{L}^{t}_{\textbf{VCL}}(q_t(\theta))$

Theorem 1. *Let $\mathcal{L}^{t,K}_{\textit{IW-VCL}}(q_t(\theta))$ denote our proposed objective with K i.i.d samples from $q_t(\theta)$. For $\boldsymbol{K \geq M}$, $\mathcal{L}^{t,K}_{\textit{IW-VCL}}(q_t(\theta)) \geq \mathcal{L}^{t,M}_{\textit{IW-VCL}}(q_t(\theta))$.*

Proof. We adapt the proof of Theorem 1 in [2].

We have, by Jensen's Inequality,

$$\begin{aligned}
\mathcal{L}^{t,K}_{\text{IW-VCL}}(q_t(\theta)) &= \mathbb{E}_{\theta_{1:K}\sim q_t(\theta)}\left[\log\left(\frac{1}{K}\sum_{k=1}^{K}w(\theta_k)\right)\right]\\
&= \mathbb{E}_{\theta_{1:K}\sim q_t(\theta)}\left[\log\left(\mathbb{E}_{I\subseteq\{1,\dots,K\},|I|=M}\frac{1}{M}\sum_{k=1}^{K}w(\theta_k)\mathbf{1}\{k\in I\}\right)\right]\\
&\geq \mathbb{E}_{\theta_{1:K}\sim q_t(\theta)}\left[\mathbb{E}_{I\subseteq\{1,\dots,K\},|I|=M}\left[\log\left(\frac{1}{M}\sum_{m\in I}w(\theta_m)\right)\right]\right]\\
&= \mathbb{E}_{\theta_{1:M}\sim q_t(\theta)}\left[\log\left(\frac{1}{M}\sum_{m=1}^{M}w(\theta_m)\right)\right]\\
&= \mathcal{L}^{t,M}_{\text{IW-VCL}}(q_t(\theta)).
\end{aligned}$$

In the derivation above, we assumed that I is an independently and uniformly sampled subset of $\{1,\dots,K\}$ of size M.

Setting $M=1$ yields implies that for every $K\geq 1$, $\mathcal{L}^{t,K}_{\text{IW-VCL}}(q_t(\theta)) \geq \mathcal{L}^{t}_{\text{VCL}}(q_t(\theta))$. ■

6.8.5 Interpretation of $\log \mathbb{E}_{q_{t-1}(\theta)}\left[p\big(D_t \mid \theta\big)\right]$

Assume that $q_{t-1}(\theta) = p(\theta \mid D_{1:t-1})$. By marginalization,

$$\begin{aligned}
\log \mathbb{E}_{q_{t-1}(\theta)}\left[p\big(D_t \mid \theta\big)\right] &= \log \int_\theta p\big(D_t \mid \theta\big) q_{t-1}(\theta)\, d\theta \\
&= \log \int_\theta p\big(D_t \mid \theta\big) p(\theta \mid D_{1:t-1})\, d\theta \\
&= \log \int_\theta p\big((D_t, \theta) \mid D_{1:t-1}\big)\, d\theta \\
&= \log p(D_t \mid D_{1:t-1})
\end{aligned}$$

Thus, if our variational approximation were perfect (i.e. $q_{t-1}(\theta) = p(\theta \mid D_{1:t-1})$), then the quantity $\log \mathbb{E}_{q_{t-1}(\theta)}\left[p\big(D_t \mid \theta\big)\right]$ is the log likelihood of the task D_t given the observed tasks $D_1 \ldots D_{t-1}$.

References

[1] Hunar Batra and Ronald Clark. *EVCL: Elastic Variational Continual Learning with Weight Consolidation*. 2024. arXiv: 2406.15972 [cs.LG]. URL: https://arxiv.org/abs/2406.15972.

[2] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. *Importance Weighted Autoencoders*. 2016. arXiv: 1509.00519 [cs.LG]. URL: https://arxiv.org/abs/1509.00519.

[3] Sebastian Farquhar and Yarin Gal. *A Unifying Bayesian View of Continual Learning*. 2019. arXiv: 1902.06494 [stat.ML]. URL: https://arxiv.org/abs/1902.06494.

[4] Ian J. Goodfellow et al. *An Empirical Investigation of Catastrophic Forgetting in Gradient-Based Neural Networks*. 2015. arXiv: 1312.6211 [stat.ML]. URL: https://arxiv.org/abs/1312.6211.

[5] Diederik P Kingma and Max Welling. *Auto-Encoding Variational Bayes*. 2022. arXiv: 1312.6114 [stat.ML]. URL: https://arxiv.org/abs/1312.6114.

[6] James Kirkpatrick et al. "Overcoming catastrophic forgetting in neural networks". In: *Proceedings of the national academy of sciences* 114.13 (2017), pp. 3521–3526.

[7] Luckeciano C Melo, Alessandro Abate, and Yarin Gal. "Temporal-Difference Variational Continual Learning". In: *arXiv preprint arXiv:2410.07812* (2024).

[8] Cuong V Nguyen et al. "Variational continual learning". In: *arXiv preprint arXiv:1710.10628* (2017).

[9] Hanul Shin et al. *Continual Learning with Deep Generative Replay*. 2017. arXiv: 1705.08690 [cs.AI]. URL: https://arxiv.org/abs/1705.08690.

[10] Siddharth Swaroop et al. "Improving and understanding variational continual learning". In: *arXiv preprint arXiv:1905.02099* (2019).

[11] Friedemann Zenke, Ben Poole, and Surya Ganguli. "Continual learning through synaptic intelligence". In: *International conference on machine learning*. PMLR. 2017, pp. 3987–3995.