Improving Generative Variational Continual Learning by Changing Modelling Distributions and Controlling Regularisation

Candidate number: 1053670
Uncertainty in Deep Learning Mini-project

Abstract

This work aims to explore various changes and potential improvements in the Variational Continual Learning framework in its generative application. The aspects considered involve a change of overall underlying distributions from Gaussian to Laplace and an addition of a hyperparameter to control regularisation. Experiments support the fact that changing Gaussian distributions to Laplace is beneficial to avoid forgetting while maintaining high performance on newly learnt tasks. Controlling the "memory" term in the loss function with a hyperparameter turns out not to have a significant effect on the overall performance but is crucial to prove that Laplacian distributions can be superior to Gaussians in some settings of VCL, and on some particular datasets.

1 Introduction

Online learning is a fundamental task in machine learning in which data arrives continually (possibly in a non-i.i.d. manner) and is to be processed gradually rather than all at once. Its significance comes from the fact that it allows learning from data collected potentially over long periods of time, incorporating new discoveries while maintaining memory of what has already been seen. The main issue it is faced with is the so called "catastrophic forgetting" problem − adaptation to more recent, possibly evolved or even completely new tasks, makes algorithms forget what has already been learnt from previous datasets. Constantly retraining on all previous datasets is very costly and breaks the idea of continual learning. A combination of online learning and the arising field of generative learning finds increasing interest both in research and real-world applications. Image and sound generation are wide fields of exploration that can certainly benefit from developed robust online learning algorithms, allowing less expensive and more efficient learning over time.

Different approaches have been taken in continual learning to deal with the "catastrophic forgetting" problem all of which deploy one (or more) of the following strategies: using regularisation, using a change in the structure of the network for each new task presented, using periodic memory enhancement. Variational Continual Learning [5] is a very general framework which overcomes the aforementioned continual learning obstacle to some extent. It counts towards the regularisation approaches as its objective function splits into two parts − one trying to increase the likelihood of observing the data from the current task, and another trying to guarantee preserved memory of previous tasks and datasets. It allows much freedom in the choice of architecture and fine tuning, and so exploring it further is worth considering. This work exposes the results of some experiments aimed at enhancing the performance of generative VCL for MNIST digit generation − namely changing all underlying distributions from Gaussians to Laplace and controlling the regularisation term with a hyperparameter.

2 Related Work

The most prominent problem of continual learning is the potential significant performance loss on previous tasks after training on a new, completely different task, and is called "catastrophic forgetting". Approaches towards continual learning generally split into the following categories (or a combination of them): regularisation-based techniques − ones that manage to retain previous information via a specific loss function, dynamic architecture methods − ones that deploy a change in architecture for each new task, and finally memory enhancement ones − reprocessing data from previous tasks regularly to avoid forgetting.

When it comes to regularisation-based techniques, [5] makes a good comparison between adaptations of traditional discriminative continual learning algorithms such as EWC [2] and SI [10] to the generative setting, together with VCL. Pure regularisation-based methods are now more or less far from state-of-the-art but provide good opportunities for further improving the performance of more sophisticated strategies. Strong examples of dynamic architecture methods are DEN [9] and RCL [8]. Memory enhancement methods explore periodically training a model with a "coreset" dataset − a dataset consisting of observations from previous tasks. There are different options for a choice of a "coreset" − keep random data from each task in the "coreset", keep special data from each task in the "coreset" (for example cluster centers), or even use synthetically generated data or partial encodings of real data as a "coreset"; "coreset" techniques are usually used to supplement one of the other two strategies. Generally, most of the advanced generative architectures such as GANs and Diffusion models have now been applied to online learning tasks as in [3], [7] and [4], achieving big success and constantly setting new benchmarks in the field. All in all, generative continual learning is an active topic of research which is expected to develop vastly in the upcoming years.

3 Proposed approach

The goal of this study is to investigate performance changes for generative VCL after a change in underlying distributions from Gaussian to Laplace, as well as after adding an explicit regularisation hyperparameter to deal with the tradeoff between learning our current dataset and preserving memory of what has already been learnt. For the second aspect of this investigation, despite losing the full mathematical coherence and the good probabilistic interpretation of traditional VCL, we might observe that controlling the memory preserving regularisation term in the objective function with a hyperparameter is beneficial. Experiments are performed to ascertain what values of the hyperparameter provide biggest performance enhancements.

4 Theory

Consider a generative model in which we aim to learn a distribution $p(\mathbf{x} \mid y)$ over the input $\mathbf{x}$ given its correct label y. Suppose further that data arrives in an online manner − that is, we first receive a dataset $D_1 = (\mathbf{x}_i^1, L_1)_{i=1}^{n_1}$ in which all observations have labels L_1, after that we receive a dataset $D_2 = (\mathbf{x}_i^2, L_2)_{i=1}^{n_2}$ with all observations labelled L_2, and so on. Our task is to manage to learn all tasks well enough while avoiding the "catastrophic forgetting" problem as much and for as long as possible.

Considering each task separately, in order to learn a good distribution $p_{L_i}(\mathbf{x})$, we need to maximise the likelihood of observing $(\mathbf{x}_m^i)_{m=1}^{n_i}$. Provided $\mathbf{x}_m^i$'s are all independent, it suffices to maximise $\log p(\mathbf{x}_1^i,\, \mathbf{x}_2^i,\, \ldots, \mathbf{x}_{n_i}^i) = \sum_{m=1}^{n_i} \log p(\mathbf{x}_m^i)$. A classical approach in generative learning is to assume our observation $\mathbf{x}$ is a random function of a random noise latent variable $\mathbf{z} \sim N(0, \boldsymbol{I}_{dim_z})$, and a random parameters tensor $\boldsymbol{\theta}$, coming from a prior distribution $p(\boldsymbol{\theta})$. Thus, if all $\mathbf{z}$'s are independent of $\boldsymbol{\theta}$, by Bayes' law we have that:

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}) \;=\; \log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})}{p_{\boldsymbol{\theta}}(\mathbf{z} \mid \mathbf{x})} \;=\; \log \frac{p_{\boldsymbol{\theta}}(\mathbf{x} \mid \mathbf{z})\, p(\mathbf{z})}{p_{\boldsymbol{\theta}}(\mathbf{z} \mid \mathbf{x})}$$

The distribution $p_{\boldsymbol{\theta}}(\mathbf{z} \mid \mathbf{x})$ is not straightforward to evaluate. For that reason, we take the approach of approximating it. In particular, we consider $\mathbf{z}$ as a random function of $\mathbf{x}$ whose additional parameters are collected in a tensor $\boldsymbol{\phi}$ – possibly deterministic, possibly random (this way we implement a

Variational AutoEncoder [1]). From there, instead of the quantity $\log p_{\boldsymbol{\theta}}(\mathbf{x})$, we consider:

$$\begin{aligned}\mathcal{L}\left(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}\right) &= \mathbb{E}_{\mathbf{z}\sim p_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}\,|\,\mathbf{z})\, p(\mathbf{z})}{p(\mathbf{z}\,|\,\mathbf{x}, \boldsymbol{\phi})}\right] \\ &= \log p_{\boldsymbol{\theta}}(\mathbf{x}) - \mathsf{KL}\left(p_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x}) \,||\, p_{\boldsymbol{\theta}}(\mathbf{z}\,|\,\mathbf{x})\right) \quad (1) \\ &= -\mathsf{KL}(p_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x}) \,||\, p_{\boldsymbol{\theta}}(\mathbf{z}))) + \mathbb{E}_{\mathbf{z}\sim p_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}\,|\,\mathbf{z})\right] \quad (2)\end{aligned}$$

By the representation in (1), since the KL divergence of two distributions is always non-negative, we have that $\mathcal{L}\left(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}\right)$ is always no more than $\log p_{\boldsymbol{\theta}}(\mathbf{x})$. We choose to optimise (maximise) over this quantity as an approximation to $\log p_{\boldsymbol{\theta}}(\mathbf{x})$ (its variational lower bound [1]). In terms of the representation in (2), our objective function when training on dataset D_i, becomes (to be maximised) [1]:

$$\frac{1}{n_i}\sum_{m=1}^{n_i}\left(-\mathsf{KL}(p_{\boldsymbol{\phi}}(\mathbf{z}_m\,|\,\mathbf{x}_m^i) \,||\, p_{\boldsymbol{\theta}}(\mathbf{z}_m)) + \mathbb{E}_{\mathbf{z}\sim p_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}_m^i)}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}_m^i\,|\,\mathbf{z}_m)\right]\right)$$

In the case of [5], it is assumed that $\mathbf{x} \sim \mathsf{Bernoulli}(f_{\boldsymbol{\theta}}(\mathbf{x}))$, where $f_{\boldsymbol{\theta}}$ is a function of a deep neural network, with all weights and biases collected in the parameters tensor $\boldsymbol{\theta}$. This deep neural network comprises two heads − one task-specific and one shared between all tasks. The purpose of that architecture is to allow simultaneous adaptation to the current task while managing to maintain good performance ("memory") of previous tasks. $\boldsymbol{\theta}$ is considered to come from a Gaussian prior with independent components. For the encoder component, it is assumed that $\mathbf{z}\,|\,\mathbf{x} \sim N(\mu_{\text{post}}, \sigma^2_{\text{post}})$, where $\{\mu_{\text{post}}, \sigma_{\text{post}}\}$ is the output of a function $g_{\boldsymbol{\phi}}(\mathbf{x})$ with $g_{\boldsymbol{\phi}}$ being a MLP parametrised by $\boldsymbol{\phi}$.

The KL divergence part of the above loss can be evaluated directly using the fact that the KL divergence between two univariate Gaussian random variables $N_1(\mu_1, \sigma_1)$ and $N_2(\mu_2, \sigma_2)$ is given by $\mathsf{KL}(N_1 \,||\, N_2) = \log\left(\frac{\sigma_2}{\sigma_1}\right) + \frac{\sigma_1^2 + (\mu_1 - \mu_2)^2}{2\sigma_2^2} - \frac{1}{2}$. The expectation can be approximated using Monte Carlo integration i.e. by observing a sequence $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_K$ of random variables coming from the distribution $p_{\boldsymbol{\phi}}(\mathbf{v}|\mathbf{x})$ and using $\frac{1}{K}\sum_{j=1}^{K}\log p_{\boldsymbol{\theta}}(\mathbf{x}\,|\,\mathbf{v}_j)$ as an approximation for it.

It remains to incorporate the VCL inter-task regularisation term − namely, in order to guarantee remembrance of previous tasks, we add the KL divergence between the current shared parameters and their prior (which is the posterior after the previous task), possibly controlled by some hyperparameter λ. Again assuming Gaussian distributions, this divergence can be directly evaluated. In the case of $\lambda = 1$, a good probabilistic interpretation exists and the whole loss can be seen as a Bayesian Inference approximation for the posterior of the parameters tensor $\boldsymbol{\theta}$ − having

$$\mathsf{KL}\left(p_{\text{post}}(\boldsymbol{\theta}) \,||\, \mathsf{normalised}\left(p_{\text{prior}}(\boldsymbol{\theta}) \times p(\text{data}\,|\,\boldsymbol{\theta})\right)\right) \approx 0$$

is equivalent to minimising

$$\mathsf{KL}(p_{\text{post}}(\boldsymbol{\theta}) \,||\, p_{\text{prior}}(\boldsymbol{\theta})) - \mathbb{E}_{\boldsymbol{\theta}\sim p_{\text{post}}(\boldsymbol{\theta})}\left[\log p(\text{data}\,|\,\boldsymbol{\theta})\right]$$

Thus, finally, in order to perform Variational Continual Learning to learn task i, we need to optimise over (minimise) the loss:

$$\mathbb{E}_{\boldsymbol{\theta}\sim p_{\text{post}}(\boldsymbol{\theta})}\left[\frac{1}{n_i}\sum_{m=1}^{n_i}\left(\mathsf{KL}(p_{\boldsymbol{\phi}}(\mathbf{z}_m\,|\,\mathbf{x}_m^i) \,||\, p_{\boldsymbol{\theta}}(\mathbf{z}_m)) - \mathbb{E}_{\mathbf{z}\sim p_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}_m^i)}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}_m^i\,|\,\mathbf{z}_m)\right]\right)\right] + \mathsf{KL}(p_{\text{post}}(\boldsymbol{\theta}) \,||\, p_{\text{prior}}(\boldsymbol{\theta}))$$

The first aspect of investigation of this work is to test whether a change of underlying distributions can be beneficial for learning with smaller forgetting rates while maintaining high log likelihoods of the observed data (or its variational lower bound $\mathcal{L}$ to be more precise). If we assume that all latent variables $\mathbf{z}$ come from a $\mathsf{Laplace}(0, \boldsymbol{I}_{dim_z})$ distribution and $\mathbf{z}|\mathbf{x} \sim \mathsf{Laplace}(\mu_{\text{post}}, b_{\text{post}})$ with $\{\mu_{\text{post}}, b_{\text{post}}\}$ being the output of the encoder learnt function $g(\mathbf{x})$, we can approach the problem as described above but now using the fact that the KL divergence between two univariate Laplacian random variables $L_1(\mu_1, b_1)$ and $L_2(\mu_2, b_2)$ is given by $\mathsf{KL}\left(L_1 \,||\, L_2\right) = \frac{b_1 \exp\left(-\frac{|\mu_1 - \mu_2|}{b_1}\right) + |\mu_1 - \mu_2|}{b_2} + \log\frac{b_2}{b_1} - 1$. The second aspect of investigation of this work is to determine whether controlling the regularisation term with a hyperparameter λ is beneficial for better learning with smaller forgettance rates. Thus, to test that we consider the loss above multiplying the second term by λ.

5 Experiments

We conduct experiments for MNIST digit generation. Data is separated into datasets $D_0, D_1, \ldots, D_9$ with dataset D_i containing all training images labelled i, and those datasets arrive in an online manner. For test evaluation, we use an importance sampling estimate for the log likelihood (or $\mathcal{L}$ more precisely) as an accuracy metric.

We deploy the architecture considered in [5] in order to present results comparable to the ones presented there. The image generating architecture comprises a task-specific head and a head shared among all tasks. Each of those contains 2 hidden layers of dimension 500. The dimension of the latent variables is taken to be 50. The encoder architecture is constructed to mimic the generator and so contains 4 hidden layers of dimension 500. Yet, it is deterministic, in contrast to the generator, which has random parameters used to measure uncertainty. From an optimisation perspective, an ADAM optimiser is used with a learning rate of 10^{-4}. The number of epochs used is 200.

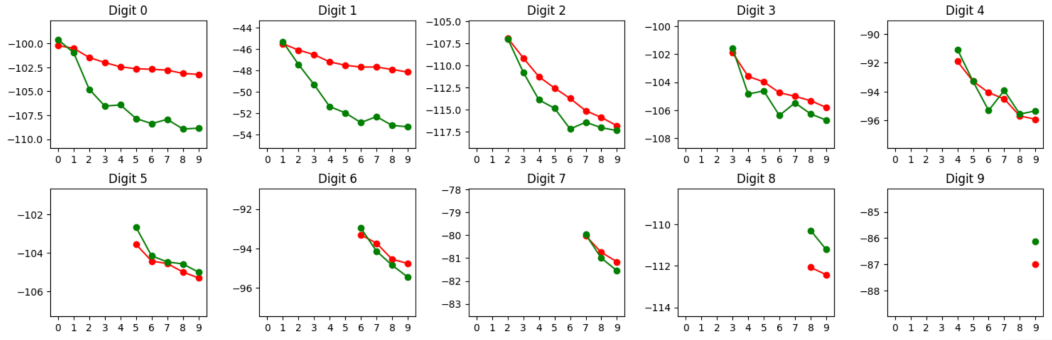

Fig 1. Comparison of test forgetting rates for all digits in the Laplace case (red curves) and in the Gaussian case (green curves) with $\lambda = 1$.

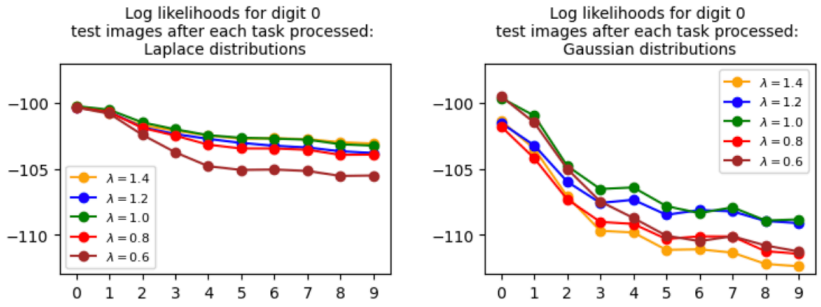

Fig 2. Average log likelihoods of observing test images labelled 0 after learning each new task. Left graph presents results in the Laplace case, while right graph – Gaussian case, with $\lambda \in \{0.6, 0.8, 1.0, 1.2, 1.4\}$.

Performing the described experiments (Fig 1.), we observe that by changing all distributions – weights' and biases' priors and latent variables' distributions, from Gaussian to Laplace, we obtain a better long-term performance. Forgettance rates are visibly diminished at the cost of rare very insignificant losses in likelihood. This supports the claim that a change of distributions is benefitial. An explanation for that might be that the total KL divergence $\mathsf{KL}(p_{\text{post}}(\boldsymbol{\theta}) \,||\, p_{\text{prior}}(\boldsymbol{\theta}))$ in the case of Laplace distributions is smaller than the same quantity in the case of Gaussian distributions. This serves as a good motivation to perform the second experiment – namely verify whether just controlling the aforementioned term in the Gaussian setting with a hyperparameter has the same effect as changing the underlying distributions to Laplace. The results in Fig 2. show that this is not true – performance does not enhance (or generally significantly change) if we make the inter-task regularisation term in the Gaussian case have the same magnitude as the inter-task regularisation term in the Laplace case. Therefore, Laplace distributions are generally more suitable than Gaussians when training with VCL for MNIST digit generation, judging by our experiments.

6 Discussion and Conclusion

Variational Continual Learning is a very powerful approach to online learning that to a good extent deals with the catastrophic forgetting problem. The aim of this work is to research different potential changes that might enhance the performance of VCL in learning to generate MNIST digits. It turns out that altering all distributions from Gaussians to Laplace is beneficial for further overcoming potential forgetting. It is certainly very important to test those discoveries on other datasets with more tasks − for example generating small Latin characters, to assess the real forgettance rates. This is not done now due to limitations in the computational resources and the aim to still provide results comparable to the ones in the original VCL paper. [5]

7 VCL Implementation

Code for running the experiments can be found in [6]. This is a Pytorch implementation of VCL influenced by [5].

References

[1] Diederik P Kingma. "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

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

[3] Kevin J Liang et al. "Generative adversarial network training is a continual learning problem". In: *arXiv preprint arXiv:1811.11083* (2018).

[4] Sergi Masip et al. "Continual learning of diffusion models with generative distillation". In: *arXiv preprint arXiv:2311.14028* (2023).

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

[6] *Personal VCL implementation*. `https://anonymous.4open.science/r/UDL_exam-B272/UDL_exam_code.ipynb`.

[7] Hanul Shin et al. "Continual learning with deep generative replay". In: *Advances in neural information processing systems* 30 (2017).

[8] Ju Xu and Zhanxing Zhu. "Reinforced continual learning". In: *Advances in neural information processing systems* 31 (2018).

[9] Jaehong Yoon et al. "Lifelong learning with dynamically expandable networks". In: *arXiv preprint arXiv:1708.01547* (2017).

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