VIABLE: Fast Adaptation via Backpropagating Learned Loss

Leo Feng[*] University of Oxford

Luisa Zintgraf University of Oxford

Bei Peng University of Oxford

Shimon Whiteson University of Oxford Latent Logic

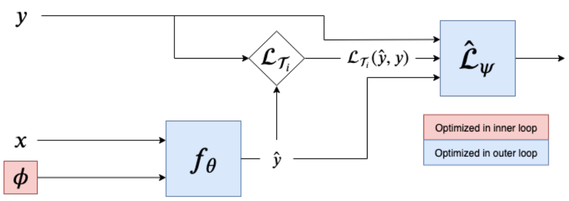
1 Introduction

Meta-learning is a popular and general way to tackle few-shot learning problems, i.e., learning how to solve unseen tasks given only little data. Many meta-learning methods can be characterised as meta-gradient-based [7, 13, 17, 30]. Briefly speaking, meta-gradient-based methods work as follows. During training, at each iteration, these methods perform a gradient-based task-specific update (often referred to as the "inner loop"). Then, for the meta-update, so-called *meta-gradients* are computed by backpropagating through these inner loop updates (which therefore involves taking higher order gradients). At test time, on a new task, only the inner-loop update is performed using a few gradient updates. In few-shot learning, typically, the loss function applied at test time is the one we are ultimately interested in minimising, such as the mean-squared-error loss for a regression problem. However, given we have few samples at test time, we argue that the loss function we want to minimise is not necessarily the loss function most suitable for computing gradients in a few-shot setting. Such a loss function is naive in the sense that it treats each datapoint independently, disregarding any relationships between them. This can be particularly problematic when only few datapoints are given and include, e.g., outliers or correlated points. Furthermore, it can be prone to cause over- or underfitting [14], depending on the stepsize and number of gradient steps. Therefore, we propose to instead *learn* the test-time loss function for meta-gradient-based methods for few-shot adaptation. In this work, we introduce *fast adaptation* ***via b**ackprogating **le**arned loss* (VIABLE), a generic meta-learning extension which builds on existing meta-gradient-based methods by learning a differentiable loss function using meta-gradients. This loss function replaces the pre-defined inner-loop loss function and is meta-learned such that it maximises performance (i.e., minimises the pre-defined loss) within a few gradient steps and with little data. We show that learning a loss function capable of leveraging relational information between samples reduces underfitting, and significantly improves performance and sample efficiency on a simple regression task. In addition, we show VIABLE is scalable by evaluating on the Mini-Imagenet dataset [16]. Since we typically use neural networks as function approximators, we will refer to the network making predictions as the *prediction network* and the learned loss function as the *loss network*.

Learning a loss function has been explored in a variety of ways in machine learning fields [1, 5, 6, 10, 19, 22, 25, 27, 28] including reinforcement learning and semi-supervised learning. In this paper, we are concerned with the few-shot supervised learning setting. Closest related to our method is recent work by Chebotar et al. [5], who propose ML^3, in which they learn a loss function in a similar fashion as VIABLE. In contrast to our work, ML^3 is not designed for few-shot learning and instead uses the learned loss function to learn a prediction network *from scratch* per task. VIABLE on the other hand can be applied on top of any meta-gradient-based meta-learning techniques designed for few-shot learning. Also closely related is work by Sung et al. [22], who propose meta-critics. In addition to also learning *from scratch* per task, during meta-training, the meta-critic (loss network) is updated after each batch of task-specific actor (prediction network) updates; while in VIABLE, the loss network is frozen during task-specific updates and thus requires far fewer updates in total. Most importantly, compared to the above methods, we propose to learn a loss function that is designed to

[*] Correspondence to: leo.feng@keble.ox.ac.uk

3rd Workshop on Meta-Learning at NeurIPS 2019, Vancouver, Canada.



y
$\mathcal{L}_{\mathcal{T}_i}$
$\mathcal{L}_{\mathcal{T}_i}(\hat{y}, y)$
$\hat{\mathcal{L}}_{\psi}$
Optimized in inner loop
Optimized in outer loop
x
ϕ
f_θ
$\hat{y}$

Figure 1: Overview of VIABLE with a *simple loss network* applied to CAVIA, where f_θ is the prediction network, $\hat{\mathcal{L}}_\psi$ is the loss network, and $\mathcal{L}_{\mathcal{T}_i}$ is the original task-specific loss function.

operate on *the entire dataset* at once, thus leveraging relational information between datapoints. We achieve this by using a relation network [18] that looks at pairwise combinations of datapoints. As we show in this paper, this leads to a significant improvement in terms of performance.

2 Background

We consider the problem setting of meta-learning for supervised learning problems. In supervised learning, we learn a model $f : x \mapsto \hat{y}$ that maps data points x that have a true label y to predictions $\hat{y}$. In few-shot learning problems, during each meta-training iteration, a batch of N tasks $\mathbf{T} = \{\mathcal{T}_i\}_{i=1}^N$ is sampled from a task distribution $p(\mathcal{T})$. A task $\mathcal{T}_i$ is a tuple ($\mathcal{X}$, $\mathcal{Y}$, $\mathcal{L}$, q), where $\mathcal{X}$ is the input space, $\mathcal{Y}$ is the output space, $\mathcal{L}$ is the task-specific loss function, and $q(x, y)$ is a distribution over data points. During each meta-training iteration, for each $\mathcal{T}_i \in \mathbf{T}$, we sample from $q_{\mathcal{T}_i}$: $\mathcal{D}_i^{\text{train}} = \{(x, y)^{i,m}\}_{m=1}^{M_i^{\text{train}}}$ and $\mathcal{D}_i^{\text{test}} = \{(x, y)^{i,m}\}_{m=1}^{M_i^{\text{test}}}$, where M_i^{train} and M_i^{test} are the fixed number of training and test datapoints respectively. The training data is used to perform updates on the model f. Afterwards, the updates are evaluated on the test data and f or the update rule are adjusted.

2.1 Context Adaptation via Meta-Learning: CAVIA

In theory, VIABLE can be generically applied to meta-gradient-based methods. In this paper, we evaluate on CAVIA [30] because it applies the inner-loop update only on a small set of so-called context parameters instead of the entire network, making it easier to optimise. CAVIA aims to learn two distinct sets of parameters: task-specific context parameters ϕ and task-agnostic parameters θ. At every meta-training iteration (inner loop), CAVIA starts from a fixed value ϕ_0, typically $\phi_0 = 0$, and updates its context-parameters ϕ for each task $\mathcal{T}_i$ in the current batch $\mathbf{T}$ of tasks as follows[†]:

$$\phi_i = \phi_0 - \alpha \nabla_\phi \frac{1}{M_i^{\text{train}}} \sum_{(x,y) \in \mathcal{D}_i^{\text{train}}} \mathcal{L}_{\mathcal{T}_i}(f_{\phi_0, \theta}(x), y) \tag{1}$$

In the meta-update step (outer loop), the model parameters θ are updated with respect to the performance after the inner-loop update:

$$\theta \leftarrow \theta - \beta \nabla_\theta \frac{1}{N} \sum_{\mathcal{T}_i \in \mathbf{T}} \frac{1}{M_i^{\text{test}}} \sum_{(x,y) \in \mathcal{D}_i^{\text{test}}} \mathcal{L}_{\mathcal{T}_i}(f_{\phi_i, \theta}(x), y) \tag{2}$$

At test time, model parameters θ are frozen and only the task-specific parameters ϕ are updated.

3 Fast Adaptation via Backpropagating Learned Loss: VIABLE

We introduce VIABLE, a generic meta-learning extension that aims to adapt a loss function applicable to meta-gradient-based methods. During training, at each iteration, VIABLE trains an existing meta-gradient-based method (referred to as *prediction network*) by performing gradient updates using the output of a differentiable learned loss function (referred to as *loss network*). During the meta-update step, the meta-gradients are calculated and used to update the *loss network*. In this section, we

[†] We outline CAVIA for one gradient update step, but it can be extended to several gradient steps.

consider two variants of loss networks: a simple loss network and an extension inspired by relation networks [18] which leverages relationships between datapoints.

Simple Loss Network. First, we consider a simple loss network $\hat{\mathcal{L}}_\psi$ which takes as input the target y, the prediction $\hat{y}$, and pre-defined task-specific loss $\mathcal{L}_{\mathcal{T}_i}(\hat{y}, y)$, and outputs a loss value. In the inner loop of the meta-gradient-based method, we replace the pre-defined task-specific loss with the output of our loss network. In this case, we replace CAVIA's inner loop update (see (1)) with:

$$\phi_i = \phi_0 - \alpha \nabla_\phi \frac{1}{M_i^{\text{train}}} \sum_{(x,y)\in\mathcal{D}_i^{\text{train}}} \hat{\mathcal{L}}_\psi(\mathcal{L}_{\mathcal{T}_i}(f_{\phi_0,\theta}(x), y), f_{\phi_0,\theta}(x), y) \tag{3}$$

The task-specific parameters ϕ are updated by backpropagating the learned loss *through the original loss and the outputs of the prediction network.* In the outer loop, we update the parameters of the loss network ψ along with the task-agnostic parameters of the prediction network θ (see (2)):

$$\psi \leftarrow \psi - \gamma \nabla_\psi \frac{1}{N} \sum_{\mathcal{T}_i\in\mathbf{T}} \frac{1}{M_i^{\text{test}}} \sum_{(x,y)\in\mathcal{D}_i^{\text{test}}} \mathcal{L}_{\mathcal{T}_i}(f_{\phi_i,\theta}(x), y) \tag{4}$$

Relation Loss Network. Note that the pre-specified loss function $\mathcal{L}_{\mathcal{T}_i}$ and the aforementioned simple loss network naively calculate an independent loss per sample and average, ignoring any possible relationships between datapoints. For example, in the case of an outlier with a large disagreeing gradient compared to the other samples, simply averaging the gradients may negatively impact the model's performance post-update. In addition, there is substantial evidence in few-shot learning showing that incorporating relational information between samples improves predictions [11, 17, 23, 26]. Thus, we believe that loss functions can improve upon gradient-based methods by providing the prediction network with relational information between samples, especially in gradient-based methods like MAML which treat their datapoints as independent during prediction. To show this, we introduce a relation loss network which takes as input the pairwise combinations of x, y, $\hat{y}$, $\mathcal{L}_{\mathcal{T}_i}(\hat{y}, y)$. Thus, we replace CAVIA's inner loop update (see (1)) with:

$$\phi_i = \phi_0 - \alpha \nabla_\phi \frac{1}{(M_i^{\text{train}})^2} \sum_{(x_j,y_j)\in\mathcal{D}_i^{\text{train}}} \sum_{(x_k,y_k)\in\mathcal{D}_i^{\text{train}}} \hat{\mathcal{L}}_\psi(\mathcal{L}_{\mathcal{T}_i}(\hat{y}_j, y_j), x_j, \hat{y}_j, y_j, \mathcal{L}_{\mathcal{T}_i}(\hat{y}_k, y_k), x_k, \hat{y}_k, y_k) \tag{5}$$

where $\hat{y}_j = f_{\phi_0,\theta}(x_j)$. Similar to the simple loss network, in the outer loop, we update the loss network and the task-agnostic parameters of the prediction network (see (4) and (2)).

4 Experiments

In this section, we evaluate the benefits of replacing the existing loss function in meta-gradient-based meta-learning methods with an adapted loss trained with VIABLE. We show that: 1) a loss function that leverages relational information between samples yields a substantial increase in performance over loss functions without relational information, 2) VIABLE improves the sample efficiency and reduces underfitting in a simple regression task, and 3) VIABLE is scalable by evaluating on the Mini-Imagenet dataset. For these experiments, we denote simVIABLE as applying VIABLE with a simple loss network to CAVIA, and relVIABLE as applying VIABLE with a relation loss network to CAVIA. Note that we do not evaluate against ML^3 since it is not designed for few-shot learning and thus would require more samples. We describe the specifics of our implementation in the Appendix.

4.1 Regression

We begin with a regression problem of fitting sine curves from Finn et al. [7]. A task is defined by the amplitude and phase of the sine curve which are uniformly sampled from $[0.1, 0.5]$ and $[0, \pi]$ respectively. During training, for each task, k (default $k = 10$) datapoints are uniformly sampled from $x \in [-5, 5]$ and given to the model to perform inner loop updates. The task specific loss is mean-squared-error (MSE) loss. In these experiments, we perform a single inner-loop update.

Improved performance. Both versions of VIABLE significantly outperform CAVIA. With 2 context parameters, CAVIA achieves a loss of 0.21, simVIABLE achieves 0.14, and relVIABLE achieves 0.02, which suggests that leveraging relational information between samples can substantially improve the effectiveness of the loss function. See Appendix C.2 for the full results.

Improved data efficiency. For this experiment, we uniformly sample $k \in \{0, \ldots, 20\}$ (the number of training sample points) during training. We observe in Table 1 that relVIABLE achieves better performance with 4 sample points than CAVIA does with 20. In Figure 2, we see that with only a single gradient update, CAVIA underfits on the 4 test points while relVIABLE fits the curve closely.

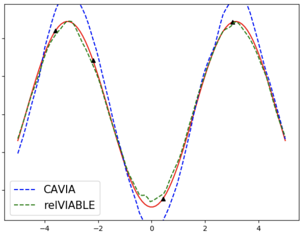


Figure 2: Test with 4 data points

	Number of Sample Points					
Method	0	1	2	3	4	20
CAVIA	**3.13**	1.69	0.93	0.58	0.47	0.13
simVIABLE	**3.13**	1.57	0.85	0.45	0.37	0.09
relVIABLE	3.14	**1.44**	**0.52**	**0.17**	**0.11**	**0.02**

Table 1: Results for the sine curve regression task. Shown is the MSE for varying number of sample points.

4.2 Classification

We show that this method can scale to problems which require larger networks by testing it on the few-shot image classification benchmark Mini-Imagenet [16].

Setup. In Rusu et al. [17], a Wide Residual Network (WRN) [29] is trained with supervised classification on the meta-train set; the network is then frozen and feature representations of the Mini-Imagenet dataset is extracted. Following their training protocol, we use the same embeddings and meta-learn on both the meta-train and meta-validation sets, with early-stopping on meta-validation.

	5-way accuracy	
Method	1-shot	5-shot
Matching Networks [26]	46.6%	60.0%
MAML [7]	$48.70 \pm 1.84\%$	$63.11 \pm 0.92\%$
Meta-SGD* [13]	$54.24 \pm 0.03\%$	$70.86 \pm 0.04\%$
LEO* [17]	$61.76 \pm 0.08\%$	$77.59 \pm 0.12\%$
MetaOptNet-SVM-trainval † [12]	$\mathbf{64.09 \pm 0.62\%}$	$\mathbf{80.00 \pm 0.45\%}$
CAVIA*	$58.10 \pm 0.51\%$	$67.07 \pm 0.45\%$
simVIABLE*	$57.88 \pm 0.49\%$	$69.32 \pm 0.41\%$
relVIABLE*	$58.26 \pm 0.50\%$	$70.23 \pm 0.41\%$

Table 2: Few-shot classification results on Mini-Imagenet (average accuracy with 95% confidence intervals). † Is the current state-of-the-art. * Used the feature embeddings from Rusu et al. [17]

Results. Table 2 shows that simVIABLE offers a notable 2.25% improvement over CAVIA while relVIABLE offers a substantial 3.16% increase in accuracy in 5-way 5-shot experiments. In both variants of VIABLE, 5-way 1-shot experiments are within confidence intervals. We suspect that learning a loss for 1-shot experiments does not offer a significant advantage due to a single sample being all the information the model is provided regarding a class of images. For example, there is no concept of an outlier with a single sample. In the regression experiments, Table 1 shows similar results where the learned loss provides minor improvements over CAVIA for a single sample point.

5 Conclusion and Future Work

We proposed VIABLE, a general-purpose meta-learning extension applicable to existing meta-gradient-based meta-learning methods. We show that learning a loss capable of leveraging relations between samples through VIABLE improves upon CAVIA by mitigating underfitting and yielding substantial improvements to sample efficiency and performance. Furthermore, we show VIABLE is scalable by evaluating on the Mini-Imagenet dataset. For future work, we are interested in applying this extension to other existing meta-learning methods such as MAML and LEO, and evaluating variants of loss networks which utilise more than just pairwise relations such as an attention network.

Acknowledgements

We thank Andrei Rusu for useful feedback on working with the LEO image embeddings [17]. This work was supported by a generous equipment grant from NVIDIA. Luisa Zintgraf is supported by the Microsoft Research PhD Scholarship Program. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement number 637713).

References

[1] M. Andrychowicz, M. Denil, S. Gomez, M. W. Hoffman, D. Pfau, T. Schaul, B. Shillingford, and N. De Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems*, pages 3981–3989, 2016.

[2] A. Antoniou, H. Edwards, and A. Storkey. How to train your maml. *arXiv preprint arXiv:1810.09502*, 2018.

[3] D. Bahdanau, P. Brakel, K. Xu, A. Goyal, R. Lowe, J. Pineau, A. Courville, and Y. Bengio. An actor-critic algorithm for sequence prediction. *Fifth International Conference on Learning Representations (ICLR 2017)*, 2017.

[4] H. S. Behl, A. G. Baydin, and P. H. Torr. Alpha maml: Adaptive model-agnostic meta-learning. *arXiv preprint arXiv:1905.07435*, 2019.

[5] Y. Chebotar, A. Molchanov, S. Bechtle, L. Righetti, F. Meier, and G. Sukhatme. Meta-learning via learned loss. In *ICML Multi-Task and Lifelong Reinforcement Learning Workshop*, 2019.

[6] Y. Duan, J. Schulman, X. Chen, P. L. Bartlett, I. Sutskever, and P. Abbeel. Rl2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

[7] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135. JMLR. org, 2017.

[8] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine. One-shot visual imitation learning via meta-learning. *arXiv preprint arXiv:1709.04905*, 2017.

[9] C. Finn, K. Xu, and S. Levine. Probabilistic model-agnostic meta-learning. In *Advances in Neural Information Processing Systems*, pages 9516–9527, 2018.

[10] R. Houthooft, Y. Chen, P. Isola, B. Stadie, F. Wolski, O. J. Ho, and P. Abbeel. Evolved policy gradients. In *Advances in Neural Information Processing Systems*, pages 5400–5409, 2018.

[11] G. Koch. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, 2015.

[12] K. Lee, S. Maji, A. Ravichandran, and S. Soatto. Meta-learning with differentiable convex optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10657–10665, 2019.

[13] Z. Li, F. Zhou, F. Chen, and H. Li. Meta-sgd: Learning to learn quickly for few-shot learning. *arXiv preprint arXiv:1707.09835*, 2017.

[14] N. Mishra, M. Rohaninejad, X. Chen, and P. Abbeel. A simple neural attentive meta-learner. *Sixth International Conference on Learning Representations (ICLR 2018)*, 2018.

[15] T. Nguyen and S. Sanner. Algorithms for direct 0–1 loss optimization in binary classification. In *International Conference on Machine Learning*, pages 1085–1093, 2013.

[16] S. Ravi and H. Larochelle. Optimization as a model for few-shot learning. In *Fifth International Conference on Learning Representations (ICLR 2017)*, 2017.

[17] A. A. Rusu, D. Rao, J. Sygnowski, O. Vinyals, R. Pascanu, S. Osindero, and R. Hadsell. Meta-learning with latent embedding optimization. In *Seventh International Conference on Learning Representations (ICLR 2019)*, 2019.

[18] A. Santoro, D. Raposo, D. G. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, and T. Lillicrap. A simple neural network module for relational reasoning. In *Advances in Neural Information Processing Systems*, pages 4967–4976, 2017.

[19] C. N. d. Santos, K. Wadhawan, and B. Zhou. Learning loss functions for semi-supervised learning via discriminative adversarial networks. In *NeurIPS Learning with Limited Data Workshop*, 2017.

[20] S. Shen, Y. Cheng, Z. He, W. He, H. Wu, M. Sun, and Y. Liu. Minimum risk training for neural machine translation. *arXiv preprint arXiv:1512.02433*, 2015.

[21] Y. Song, A. Schwing, R. Urtasun, et al. Training deep neural networks via direct loss minimization. In *International Conference on Machine Learning*, pages 2169–2177, 2016.

[22] F. Sung, L. Zhang, T. Xiang, T. Hospedales, and Y. Yang. Learning to learn: Meta-critic networks for sample efficient learning. *arXiv preprint arXiv:1706.09529*, 2017.

[23] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1199–1208, 2018.

[24] M. Taylor, J. Guiver, S. Robertson, and T. Minka. Softrank: optimizing non-smooth rank metrics. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, pages 77–86. ACM, 2008.

[25] V. Veeriah, M. Hessel, Z. Xu, J. Rajendran, R. L. Lewis, J. Oh, H. P. van Hasselt, D. Silver, and S. Singh. Discovery of useful questions as auxiliary tasks. In *Advances in Neural Information Processing Systems*, pages 9306–9317, 2019.

[26] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, pages 3630–3638, 2016.

[27] J. X. Wang, Z. Kurth-Nelson, D. Tirumala, H. Soyer, J. Z. Leibo, R. Munos, C. Blundell, D. Kumaran, and M. Botvinick. Learning to reinforcement learn. *CogSci*, 2017.

[28] L. Wu, F. Tian, Y. Xia, Y. Fan, T. Qin, L. Jian-Huang, and T.-Y. Liu. Learning to teach with dynamic loss functions. In *Advances in Neural Information Processing Systems*, pages 6466–6477, 2018.

[29] S. Zagoruyko and N. Komodakis. Wide residual networks. In *British Machine Vision Conference*, 2016.

[30] L. Zintgraf, K. Shiarli, V. Kurin, K. Hofmann, and S. Whiteson. Fast context adaptation via meta-learning. In *International Conference on Machine Learning*, pages 7693–7702, 2019.

VIABLE: Fast Adaptation via Backpropagating Learned Loss

Supplementary Material

A Pseudocode

Algorithm 1 simVIABLE: VIABLE applied to CAVIA with a simple loss network

```
Require: Distribution over tasks $p(\mathcal{T})$
Require: Step sizes $\alpha$, $\beta$, $\gamma$
Require: Initial model $\mathcal{L}_\psi$ with $\psi$ intitialised randomly and model $f_{\phi_0,\theta}$ with $\theta$ initialised randomly
    and $\phi_0 = 0$
while not done do
    Sample batch of tasks $\mathbf{T} = \{\mathcal{T}_i\}_{i=1}^N$ where $\mathcal{T}_i \sim p$
    for all $\mathcal{T}_i \in \mathbf{T}$ do
        $\mathcal{D}_i^{\text{train}}, \mathcal{D}_i^{\text{test}} \sim q_{\mathcal{T}_i}$
        $\phi_0 = 0$
        $\phi_i = \phi_0 - \alpha \nabla_\phi \frac{1}{M_i^{\text{train}}} \sum\limits_{(x,y)\in\mathcal{D}_i^{\text{train}}} \hat{\mathcal{L}}_\psi(\mathcal{L}_{\mathcal{T}_i}(f_{\phi_0,\theta}(x), y), f_{\phi_0,\theta}(x), y)$
    end for
    $\psi \leftarrow \psi - \gamma \nabla_\psi \frac{1}{N} \sum\limits_{\mathcal{T}_i \in \mathbf{T}} \frac{1}{M_i^{\text{test}}} \sum\limits_{(x,y)\in\mathcal{D}_i^{\text{test}}} \mathcal{L}_{\mathcal{T}_i}(f_{\phi_i,\theta}(x, y))$
    $\theta \leftarrow \theta - \beta \nabla_\theta \frac{1}{N} \sum\limits_{\mathcal{T}_i \in \mathbf{T}} \frac{1}{M_i^{\text{test}}} \sum\limits_{(x,y)\in\mathcal{D}_i^{\text{test}}} \mathcal{L}_{\mathcal{T}_i}(f_{\phi_i,\theta}(x, y))$
end while
```

Algorithm 2 relVIABLE: VIABLE applied to CAVIA with a relation loss network

```
Require: Distribution over tasks $p(\mathcal{T})$
Require: Step sizes $\alpha$, $\beta$, $\gamma$
Require: Initial model $\mathcal{L}_\psi$ with $\psi$ intitialised randomly and model $f_{\phi_0,\theta}$ with $\theta$ initialised randomly
    and $\phi_0 = 0$
while not done do
    Sample batch of tasks $\mathbf{T} = \{\mathcal{T}_i\}_{i=1}^N$ where $\mathcal{T}_i \sim p$
    for all $\mathcal{T}_i \in \mathbf{T}$ do
        $\mathcal{D}_i^{\text{train}}, \mathcal{D}_i^{\text{test}} \sim q_{\mathcal{T}_i}$
        $\phi_0 = 0$
        $\phi_i = \phi_0 - \alpha \nabla_\phi \frac{1}{(M_i^{\text{train}})^2} \sum_{\substack{(x_j,y_j)\in\mathcal{D}_i^{\text{train}} \\ (x_k,y_k)\in\mathcal{D}_i^{\text{train}}}} \hat{\mathcal{L}}_\psi(\mathcal{L}_{\mathcal{T}_i}(\hat{y}_j, y_j), x_j, \hat{y}_j, y_j, \mathcal{L}_{\mathcal{T}_i}(\hat{y}_k, y_k), x_k, \hat{y}_k, y_k)$
    end for
    $\psi \leftarrow \psi - \gamma \nabla_\psi \frac{1}{N} \sum\limits_{\mathcal{T}_i \in \mathbf{T}} \frac{1}{M_i^{\text{test}}} \sum\limits_{(x,y)\in\mathcal{D}_i^{\text{test}}} \mathcal{L}_{\mathcal{T}_i}(f_{\phi_i,\theta}(x, y))$
    $\theta \leftarrow \theta - \beta \nabla_\theta \frac{1}{N} \sum\limits_{\mathcal{T}_i \in \mathbf{T}} \frac{1}{M_i^{\text{test}}} \sum\limits_{(x,y)\in\mathcal{D}_i^{\text{test}}} \mathcal{L}_{\mathcal{T}_i}(f_{\phi_i,\theta}(x, y))$
end while
```

B Additional Related Work

Meta-gradient based Methods. A common form of meta-learning is to adapt parameters in two interleaving phases that can be characterised as the task-specific updates (often referred to as the "inner loop") and the meta-updates (often referred to as the "outer loop"). At test time, on a new task, only the task-specific updates are applied. Finn et al. [7] introduces a meta-gradient-based method (MAML) that aims to learn a model initialisation that allows for fast adaptation to a new task given a few task-specific updates. Many methods that are inspired by or built on top of MAML can also be classified as meta-gradient-based [2, 4, 8, 9, 13, 30]. Another meta-gradient-based method, CAVIA [30] extends MAML by splitting the model parameters are into task-specific (context) parameters and task-agnostic parameters, resulting in fewer parameters to optimize in test time. Rusu et al. [17] introduces a meta-gradient-based method LEO that learns to produce network weights from task-specific embeddings. In this paper, we focus on CAVIA due to its structure being simple and easy to optimise.

Learning a Loss Function. Specially designed loss functions have been important in improving performance of many tasks such as classification [15], machine translation [3, 20], ranking [24], and object detection [21]. In recent years, there has been interest in exploring methods for learning a good loss function automatically in a variety of machine learning fields [1, 5, 6, 10, 19, 22, 25, 27, 28], including reinforcement learning and semi-supervised learning. In this work, we focus on meta-learning, specifically the few-shot supervised learning setting. Closely related is meta-critics [22] and ML^3 [5], who both learn a form of loss network. In contrast to their works, we are not required to learn our prediction network *from scratch* per task. Furthermore, VIABLE is applicable to any meta-gradient-based meta-learning techniques designed for few-shot learning, and, in contrast to meta-critics, we do not require adaptation for our *loss network* at test time. Most importantly, compared to the above methods, we propose to learn a loss function that is designed to operate on *the entire dataset* at once, thus leveraging relational information between datapoints. We achieve this by using a relation network [18] that looks at pairwise combinations of datapoints. As we show in this paper, this leads to a significant improvement in terms of performance.

C Regression

C.1 Details

In the sine curve regression task, we follow the architecture used in the original paper for CAVIA [30] (a neural network with two hidden layers and 40 nodes each). Unless otherwise stated, by default we use 5 context parameters. In addition, a batch of 25 tasks is used per meta-update. We train for 50,000 iterations, with early stopping on a meta-validation set of 100 newly sampled tasks. During testing, we presented the model with p (default $p = 10$) datapoints from 1000 newly sampled tasks and measured MSE over 100 linearly spaced test points. In the meta-update step, the task-agnostic parameters of the prediction network is updated using the Adam optimiser with the standard learning rate of 0.001 which is annealed every 5,000 steps by multiplying it by 0.9.

To allow a fair comparison, in VIABLE we use the same architecture as CAVIA for the prediction network. For both the relation loss network and the simple loss network, we use a neural network with three hidden layers of 32 nodes each. In the meta-update step, the parameters of the loss network is learned along with the task-agnostic parameters of the prediction network using the Adam optimiser with the standard learning rate of 0.001 which is annealed every 5,000 steps by multiplying it by a factor of 0.9.

Both VIABLE and CAVIA are trained with a single inner-loop gradient step with an inner loop learning rate of 1.0.

C.2 Additional Results

	Number of Context Parameters				
Method	1	2	3	4	5
MAML	0.29 (0.02)	0.24 (0.02)	0.24 (0.02)	0.23 (0.02)	0.23 (0.02)
CAVIA	0.84 (0.06)	0.21 (0.02)	0.20 (0.02)	0.19 (0.02)	0.19 (0.02)
simVIABLE	0.75 (0.05)	0.14 (0.01)	0.15 (0.01)	0.14 (0.01)	0.16 (0.01)
relVIABLE	**0.57 (0.05)**	**0.02 (0.00)**	**0.04 (0.00)**	**0.03 (0.00)**	**0.01 (0.00)**

Table 3: Results for the sine curve regression task. Shown is the mean-squared-error (MSE) for varying number of context parameters, with 95% confidence intervals in brackets.

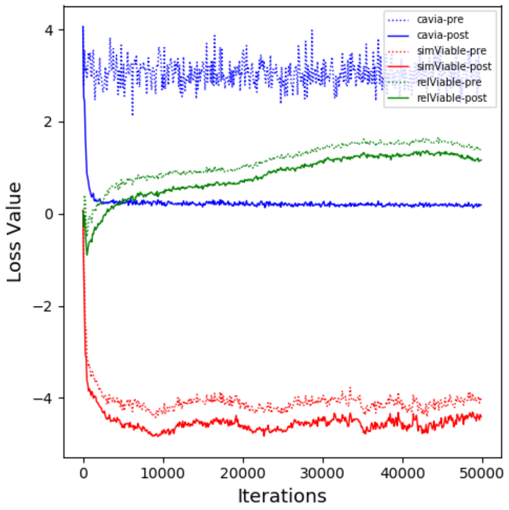


Figure 3: Pre-update and post-update test-time loss of each method on the sine curve task. The task specific loss of CAVIA is mean-squared-error (MSE) loss. The task specific loss of VIABLE is the output of the learned loss network.

D Classification

D.1 Problem Setting

In N-way K-shot classification, a task is a random selection of N classes. The model gets to see K examples per class from which the model is expected to learn to classify unseen images from the N classes. The Mini-Imagenet dataset is divided into training, validation, and test metasets with 64 classes, 16 classes, and 20 classes respectively in which there are 600 images per class. We use an open-source dataset of Mini-Imagenet embeddings made available by [17]. The embeddings are each of size 640.

D.2 Model Details

In CAVIA, our model uses a single hidden layer of size 800 and 100 context parameters. To ensure fairness, we use the same architecture for the prediction network in VIABLE. In simVIABLE, our loss network consisted of two hidden layers of 64 nodes each, and in relVIABLE, it consisted of two hidden layers of 1500 nodes each. Both VIABLE and CAVIA are trained with two inner-loop gradient steps along with an inner-learning rate of 1.0. In the meta-update step, VIABLE (prediction network and loss network) and CAVIA are both trained using the Adam optimiser with the standard learning rate of 0.001 which is also annealed every 5,000 steps by multiplying it by a factor of 0.9.

D.3 Further Experiments

We perform an additional experiment that evaluates CAVIA and VIABLE's ability to generalise to different amount of shots than seen during training. In this experiment, we train on 5-way 5-shot tasks and evaluate on 5-way k-shot where k varies from 1 to 9. Table 4 shows both variants of VIABLE significantly outperform CAVIA in generalising at test time to tasks which have a different amount of data than during meta-training. In the case of $k = 1$, the relation loss network calculates a loss using the same input in a pair with itself.

	Number of Shots: 5-way k-shot								
Method	1	2	3	4	5	6	7	8	9
CAVIA	50.36 ± 0.49%	58.94 ± 0.46%	62.64 ± 0.46%	65.61 ± 0.44%	67.07 ± 0.45%	68.32 ± 0.43%	69.13 ± 0.43%	70.16 ± 0.43%	69.95 ± 0.44%
simVIABLE	53.94 ± 0.49%	62.19 ± 0.44%	65.52 ± 0.43%	68.16 ± 0.41%	69.32 ± 0.41%	70.10 ± 0.40%	71.03 ± 0.40%	72.01 ± 0.39%	71.79 ± 0.39%
relVIABLE	**55.03 ± 0.48%**	**63.02 ± 0.44%**	**66.59 ± 0.42%**	**68.79 ± 0.42%**	**70.23 ± 0.41%**	**71.06 ± 0.39%**	**71.90 ± 0.40%**	**72.57 ± 0.39%**	**72.80 ± 0.39%**

Table 4: Results for Mini-Imagenet. Shown is the accuracy for 5-way k-shot while being pre-trained on 5-way 5-shot, with 95% confidence intervals in brackets.