Supplementary Materials for **"Learning to Learn Kernels with Variational Random Features"**

Xiantong Zhen, Haoliang Sun, Yingjun Du, Jun Xu, Yilong Yin, Ling Shao, Cees Snoek

A. Derivations of the ELBO

For a singe task, we begin with maximizing log-likelihood of the conditional distribution $p(\mathbf{y}|\mathbf{x}, \mathcal{S})$ to derive the ELBO of MetaVRF. By leveraging Jensen's inequality, we have the following steps as

$$\begin{aligned}
\log p(\mathbf{y}|\mathbf{x}, \mathcal{S}) &= \log \int p(\mathbf{y}|\mathbf{x}, \mathcal{S}, \boldsymbol{\omega}) p(\boldsymbol{\omega}|\mathbf{x}, \mathcal{S}) d\boldsymbol{\omega} && (1) \\
&= \log \int p(\mathbf{y}|\mathbf{x}, \mathcal{S}, \boldsymbol{\omega}) p(\boldsymbol{\omega}|\mathbf{x}, \mathcal{S}) \frac{q_\phi(\boldsymbol{\omega}|\mathcal{S})}{q_\phi(\boldsymbol{\omega}|\mathcal{S})} d\boldsymbol{\omega} && (2) \\
&\geq \int \log \left[\frac{p(\mathbf{y}|\mathbf{x}, \mathcal{S}, \boldsymbol{\omega}) p(\boldsymbol{\omega}|\mathbf{x}, \mathcal{S})}{q_\phi(\boldsymbol{\omega}|\mathcal{S})} \right] q_\phi(\boldsymbol{\omega}|\mathcal{S}) d\boldsymbol{\omega} && (3) \\
&= \underbrace{\mathbb{E}_{q_\phi(\boldsymbol{\omega}|\mathcal{S})} \log \left[p(\mathbf{y}|\mathbf{x}, \mathcal{S}, \boldsymbol{\omega}) \right] - D_{\mathrm{KL}}[q_\phi(\boldsymbol{\omega}|\mathcal{S}) || p(\boldsymbol{\omega}|\mathbf{x}, \mathcal{S})]}_{\text{ELBO}}. && (4)
\end{aligned}$$

The ELBO can also be derived from the perspective of the KL divergence between the variational posterior $q_\phi(\boldsymbol{\omega}|\mathcal{S})$ and the posterior $p(\boldsymbol{\omega}|\mathbf{y}, \mathbf{x}, \mathcal{S})$:

$$\begin{aligned}
D_{\mathrm{KL}}[q_\phi(\boldsymbol{\omega}|\mathcal{S}) || p(\boldsymbol{\omega}|\mathbf{y}, \mathbf{x}, \mathcal{S})] &= \mathbb{E}_{q_\phi(\boldsymbol{\omega}|\mathcal{S})} \left[\log q_\phi(\boldsymbol{\omega}|\mathcal{S}) - \log p(\boldsymbol{\omega}|\mathbf{y}, \mathbf{x}, \mathcal{S}) \right] \\
&= \mathbb{E}_{q_\phi(\boldsymbol{\omega}|\mathcal{S})} \left[\log q_\phi(\boldsymbol{\omega}|\mathcal{S}) - \log \frac{p(\mathbf{y}|\boldsymbol{\omega}, \mathbf{x}, \mathcal{S}) p(\boldsymbol{\omega}|\mathbf{x}, \mathcal{S})}{p(\mathbf{y}|\mathbf{x}, \mathcal{S})} \right] \\
&= \log p(\mathbf{y}|\mathbf{x}, \mathcal{S}) + \mathbb{E}_{q_\phi(\boldsymbol{\omega}|\mathcal{S})} \left[\log q_\phi(\boldsymbol{\omega}|\mathcal{S}) - \log p(\mathbf{y}|\boldsymbol{\omega}, \mathbf{x}, \mathcal{S}) - \log p(\boldsymbol{\omega}|\mathbf{x}, \mathcal{S}) \right] \\
&= \log p(\mathbf{y}|\mathbf{x}, \mathcal{S}) - \mathbb{E}_{q_\phi(\boldsymbol{\omega}|\mathcal{S})} \left[\log p(\mathbf{y}|\boldsymbol{\omega}, \mathbf{x}, \mathcal{S}) \right] + D_{\mathrm{KL}}[q_\phi(\boldsymbol{\omega}|\mathcal{S}) || p(\boldsymbol{\omega}|\mathbf{x}, \mathcal{S})] \geq 0.
\end{aligned} \quad (5)$$

Therefore, the lower bound of the $\log p(\mathbf{y}|\mathbf{x}, \mathcal{S})$ is

$$\log p(\mathbf{y}|\mathbf{x}, \mathcal{S}) \geq \mathbb{E}_{q_\phi(\boldsymbol{\omega}|\mathcal{S})} \log \left[p(\mathbf{y}|\mathbf{x}, \mathcal{S}, \boldsymbol{\omega}) \right] - D_{\mathrm{KL}}[q_\phi(\boldsymbol{\omega}|\mathcal{S}) || p(\boldsymbol{\omega}|\mathbf{x}, \mathcal{S})], \quad (6)$$

which is consistent with (4).

B. Cross attention in the prior network

In $p(\boldsymbol{\omega}|\mathbf{x}, \mathcal{S})$, both $\mathbf{x}$ and $\mathcal{S}$ are inputs of the prior network. In order to effectively integrate the two conditions, we adopt the cross attention (Kim et al., 2019) between $\mathbf{x}$ and each element in $\mathcal{S}$. In our case, we have the key-value matrices $K = V \in \mathbb{R}^{C \times d}$, where d is the dimension of the feature representation, and C is the number of categories in the support set. We adopt the instance pooling by taking the average of samples in each category when the shot number $k > 1$.

For the query $Q_i = \mathbf{x} \in \mathbb{R}^d$, the Laplace kernel returns attentive representation for $\mathbf{x}$:

$$\textbf{Laplace}(Q_i, K, V) := W_i V \in \mathbb{R}^d, \quad W_i := \mathrm{softmax}(-\left\| Q_i - K_{j\cdot} \right\|_1)_{j=1}^{C} \quad (7)$$

The prior network takes the attentive representation as the input.

C. More experimental details

We train all models using the Adam optimizer (Kingma & Ba, 2014) with a learning rate of 0.0001. The other training setting and network architecture for regression and classification on three datasets are different as follows.

C.1. Inference networks

The architecture of the inference network with vanilla LSTM for the regression task is in Table C.1. The architecture of the inference network with bidirectional LSTM for the regression task is in Table C.2. For few-shot classification tasks, all models share the same architecture with vanilla LSTM, as in Table C.3, For few-shot classification tasks, all models share the same architecture with bidirectional LSTM, as in Table C.4.

C.2. Prior networks

The architecture of the prior network for the regression task is in Table C.5. For few-shot classification tasks, all models share the same architecture, as in Table C.6.

C.3. Feature embedding networks

Regression. The fully connected architecture for regression tasks is shown in Table D.7. We train all three models (3-shot, 5-shot, 10-shot) over a total of $20,000$ iterations, with 6 episodes per iteration.

Classification. The CNN architectures for Omniglot, CIFAR-FS, and *mini*ImageNet are shown in Table D.8, D.9, and D.10. The difference of feature embedding architectures for different datasets is due the different image sizes.

Table C.1. The inference network $\phi(\cdot)$ based on the vanilla LSTM used for regression.

Output size	**Layers**
40	Input samples feature
40	fully connected, ELU
40	fully connected, ELU
40	LSTM cell, Tanh to μ_w, $\log \sigma_w^2$

Table C.2. The inference network $\phi(\cdot)$ based on the bidirectional LSTM for regression.

Output size	**Layers**
80	Input samples feature
40	fully connected, ELU
40	fully connected, ELU
40	LSTM cell, Tanh to μ_w, $\log \sigma_w^2$

Table C.3. The inference network $\phi(\cdot)$ based on the vanilla LSTM for Omniglot, *mini*ImageNet, CIFAR-FS.

Output size	**Layers**
$k \times 256$	Input feature
256	instance pooling
256	fully connected, ELU
256	fully connected, ELU
256	fully connected, ELU
256	LSTM cell, tanh to μ_w, $\log \sigma_w^2$

D. Few-shot classification datasets

Omniglot (Lake et al., 2015) is a benchmark of few-shot learning that contain 1623 handwritten characters (each with 20 examples). All characters are grouped in 50 alphabets. For fair comparison against the state of the arts, we follow the same

Table C.4. The inference network $\phi(\cdot)$ based on the bidirectional LSTM for Omniglot, *mini*ImageNet, CIFAR-FS.

Output size	Layers
$k \times 512$	Input feature
256	instance pooling
256	fully connected, ELU
256	fully connected, ELU
256	fully connected, ELU
256	LSTM cell, tanh to μ_w, $\log \sigma_w^2$

Table C.5. The prior network for regression.

Output size	Layers
40	fully connected, ELU
40	fully connected, ELU
40	fully connected to μ_w, $\log \sigma_w^2$

data split and pre-processing used in Vinyals et al. (2016). The training, validation, and testing are composed of a random split of $[1100, 200, 423]$. The dataset is augmented with rotations of 90 degrees, which results in 4000 classes for training, 400 for validation, and 1292 for testing. The number of examples is fixed as 20. All images are resized to 28×28. For a C-way, k-shot task at training time, we randomly sample C classes from the 4000 classes. Once we have C classes, $(k+15)$ examples of each are sampled. Thus, there are $C\times k$ examples in the support set and $C\times15$ examples in the query set. The same sampling strategy is also used in validation and testing.

***mini*ImageNet** (Vinyals et al., 2016) is a challenging dataset constructed from ImageNet (Russakovsky et al., 2015), which comprises a total of 100 different classes (each with 600 instances). All these images have been downsampled to 84×84. We use the same splits of Ravi & Larochelle (2017), where there are $[64, 16, 20]$ classes for training, validation and testing. We use the same episodic manner as Omniglot for sampling.

CIFAR-FS (CIFAR100 few-shots) (Bertinetto et al., 2019) is adapted from the CIFAR-100 dataset (Krizhevsky et al., 2009) for few-shot learning. Recall that in the image classification benchmark CIFAR-100, there are 100 classes grouped into 20 superclasses (each with 600 instances). CIFAR-FS use the same split criteria $(64, 16, 20)$ with which *mini*ImageNet has been generated. The resolution of all images is 32×32.

D.1. Other settings

The settings including the iteration numbers and the batch sizes are different on different datasets. The detailed information is given in Table D.11.

E. More results on few-shot regression

We provide more experimental results for the tasks of few-shot regression in Figure E.1. The proposed MetaVRF again performs much better than regular random Fourier features (RFFs) and the MAML method.

Table C.6. The prior network for Omniglot, *mini*ImageNet, CIFAR-FS

Output size	**Layers**
256	Input query feature
256	fully connected, ELU
256	fully connected, ELU
256	fully connected to μ_w, $\log \sigma_w^2$

Table D.7. The fully connected network $\psi(\cdot)$ used for regression.

Output size	**Layers**
1	Input training samples
40	fully connected, RELU
40	fully connected, RELU

Table D.8. The CNN architecture $\psi(\cdot)$ for Omniglot.

Output size	Layers
$28\times28\times1$	Input images
$14\times14\times64$	*conv2d* (3×3, stride=1, SAME, RELU), dropout 0.9, *pool* (2×2, stride=2, SAME)
$7\times7\times64$	*conv2d* (3×3, stride=1, SAME, RELU), dropout 0.9, *pool* (2×2, stride=2, SAME)
$4\times4\times64$	*conv2d* (3×3, stride=1, SAME, RELU), dropout 0.9, *pool* (2×2, stride=2, SAME)
$2\times2\times64$	*conv2d* (3×3, stride=1, SAME, RELU), dropout 0.9, *pool* (2×2, stride=2, SAME)
256	flatten

Table D.9. The CNN architecture $\psi(\cdot)$ for CIFAR-FS

Output size	Layers
$32\times32\times3$	Input images
$16\times16\times64$	*conv2d* (3×3, stride=1, SAME, RELU), dropout 0.5, *pool* (2×2, stride=2, SAME)
$8\times8\times64$	*conv2d* (3×3, stride=1, SAME, RELU), dropout 0.5, *pool* (2×2, stride=2, SAME)
$4\times4\times64$	*conv2d* (3×3, stride=1, SAME, RELU), dropout 0.5, *pool* (2×2, stride=2, SAME)
$2\times2\times64$	*conv2d* (3×3, stride=1, SAME, RELU), dropout 0.5, *pool* (2×2, stride=2, SAME)
256	flatten

Table D.10. The CNN architecture $\psi(\cdot)$ for *mini*ImageNet

Output size	Layers
$84\times84\times3$	Input images
$42\times42\times64$	*conv2d* (3×3, stride=1, SAME, RELU), dropout 0.5, *pool* (2×2, stride=2, SAME)
$21\times21\times64$	*conv2d* (3×3, stride=1, SAME, RELU), dropout 0.5, *pool* (2×2, stride=2, SAME)
$10\times10\times64$	*conv2d* (3×3, stride=1, SAME, RELU), dropout 0.5, *pool* (2×2, stride=2, SAME)
$5\times5\times64$	*conv2d* (3×3, stride=1, SAME, RELU), dropout 0.5, *pool* (2×2, stride=2, SAME)
$2\times2\times64$	*conv2d* (3×3, stride=1, SAME, RELU), dropout 0.5, *pool* (2×2, stride=2, SAME)
256	flatten

Table D.11. The iteration numbers and batch sizes on different datasets.

Dataset	**Iteration**	**Batch size**
Regression	$20,000$	25
Omniglot	$100,000$	6
CIFAR-FS	$200,000$	8
*mini*ImageNet	$150,000$	8

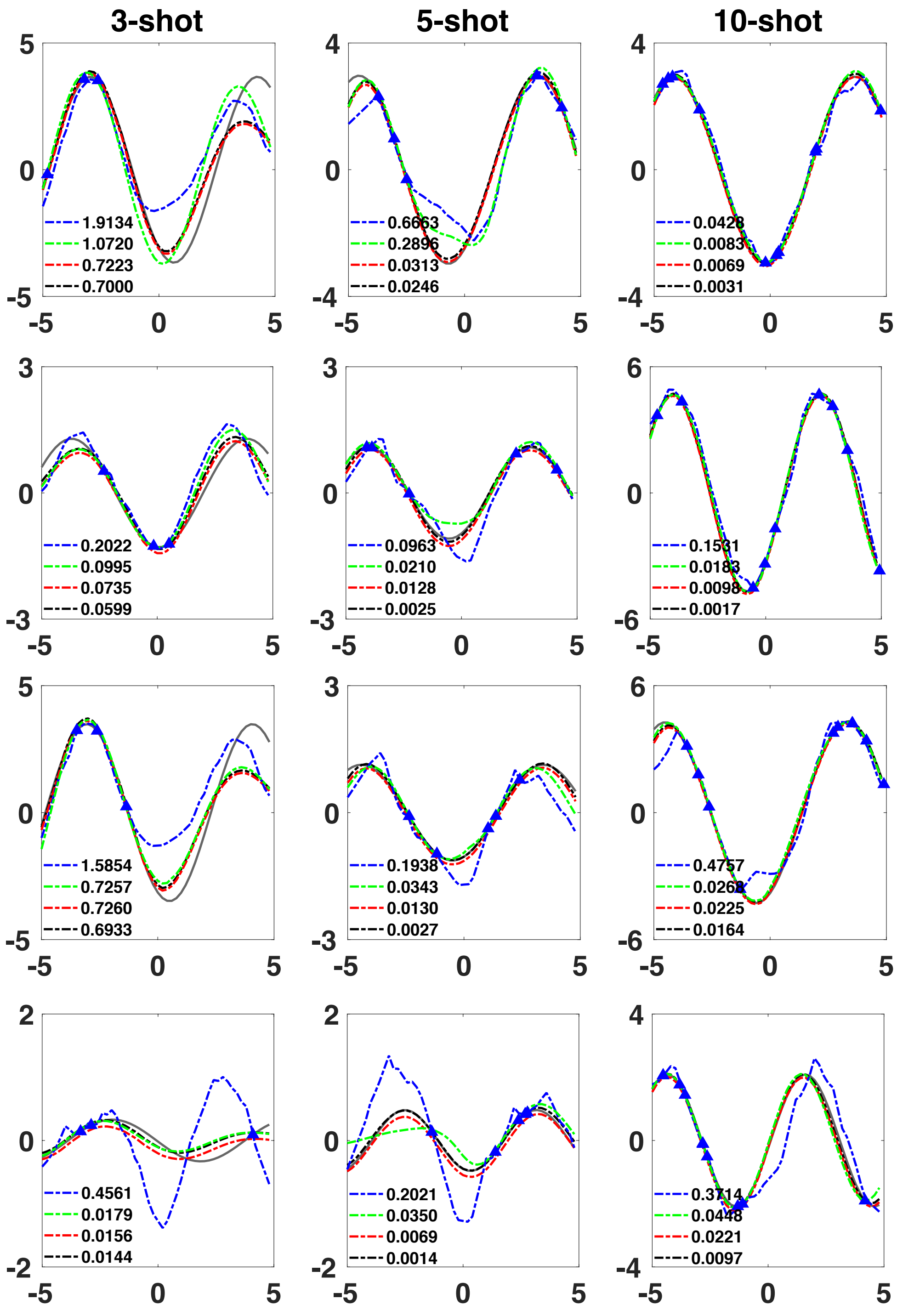

Figure E.1. More results of few-shot regression. (- - - MetaVRF with bi-LSTM; - - - MetaVRF with LSTM; - - - MetaVRF w/o LSTM; - - - MAML; —— Ground Truth; ▲ Support Samples.)

References

Bertinetto, L., Henriques, J. F., Torr, P. H., and Vedaldi, A. Meta-learning with differentiable closed-form solvers. In *International Conference on Learning Representations*, 2019.

Kim, H., Mnih, A., Schwarz, J., Garnelo, M., Eslami, A., Rosenbaum, D., Vinyals, O., and Teh, Y. W. Attentive neural processes. In *International Conference on Learning Representations*, 2019.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Krizhevsky, A. et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Lake, B. M., Salakhutdinov, R., and Tenenbaum, J. B. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

Ravi, S. and Larochelle, H. Optimization as a model for few-shot learning. In *International Conference on Learning Representations*, 2017.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015.

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