Learning to Learn Kernels with Variational Random Features

Xiantong Zhen *1 2 Haoliang Sun *3 Yingjun Du *2 Jun Xu 4 Yilong Yin 3 Ling Shao 5 1 Cees Snoek 2

Abstract

We introduce kernels with random Fourier features in the meta-learning framework for few-shot learning. We propose meta variational random features (MetaVRF) to learn adaptive kernels for the base-learner, which is developed in a latent variable model by treating the random feature basis as the latent variable. We formulate the optimization of MetaVRF as a variational inference problem by deriving an evidence lower bound under the meta-learning framework. To incorporate shared knowledge from related tasks, we propose a context inference of the posterior, which is established by an LSTM architecture. The LSTM-based inference network effectively integrates the context information of previous tasks with task-specific information, generating informative and adaptive features. The learned MetaVRF is able to produce kernels of high representational power with a relatively low spectral sampling rate and also enables fast adaptation to new tasks. Experimental results on a variety of few-shot regression and classification tasks demonstrate that MetaVRF can deliver much better, or at least competitive, performance compared to existing meta-learning alternatives.

1. Introduction

Learning to learn, or meta-learning (Schmidhuber, 1992; Thrun & Pratt, 2012), offers a promising tool for few-shot learning (Andrychowicz et al., 2016; Ravi & Larochelle, 2017; Finn et al., 2017) and has recently generated increasing popularity in machine learning. The crux of meta-learning for few-shot learning is to extract prior knowledge from related tasks to enable fast adaptation to a new task with a limited amount of data. Generally speaking, existing meta-learning algorithms (Ravi & Larochelle, 2017; Bertinetto et al., 2019) design the meta-learner to extract meta-knowledge that improves the performance of the base-learner on individual tasks. Meta knowledge, like a good parameter initialization (Finn et al., 2017), or an efficient optimization update rule shared across tasks (Andrychowicz et al., 2016; Ravi & Larochelle, 2017) has been extensively explored in general learning framework, but how to define and use in few-shot learning remains an open question.

An effective base-learner should be powerful enough to solve individual tasks and able to absorb information provided by the meta-learner to improve its own performance. While potentially strong base-learners, kernels (Hofmann et al., 2008) have not yet been studied in the meta-learning scenario for few-shot learning. Learning adaptive kernels (Bach et al., 2004) in a data-driven way via random features (Rahimi & Recht, 2007) has demonstrated great success in regular learning tasks and remains of broad interest in machine learning (Sinha & Duchi, 2016; Hensman et al., 2017; Carratino et al., 2018; Bullins et al., 2018; Li et al., 2019). However, due to the limited availability of data, it is challenging for few-shot learning to establish informative and discriminant kernels. We thus explore the relatedness among distinctive but relevant tasks to generate rich random features to build strong kernels for base-learners, while still maintaining their ability to adapt quickly to individual tasks.

In this paper, we make three important contributions. First, we propose meta variational random features (MetaVRF), integrating, for the first time, kernel learning with random features and variational inference into the meta-learning framework for few-shot learning. We develop MetaVRF in a latent variable model by treating the random Fourier basis of translation-invariant kernels as the latent variable. Second, we formulate the optimization of MetaVRF as a variational inference problem by deriving a new evidence lower bound (ELBO) in the meta-learning setting, where the posterior over the random feature basis corresponds to the spectral distribution associated with the kernel. This formulation under probabilistic modeling provides a principled way of learning data-driven kernels with random Fourier features and more importantly, fits well in the meta-learning framework for few-shot learning allowing us to
Learning to Learn Kernels with Variational Random Features

Figure 1. The learning framework of our meta variational random features (MetaVRF). The meta-learner employs an LSTM-based inference network $\phi(\cdot)$ to infer the spectral distribution over $\omega^t$, the kernel from the support set $S^t$ of the current task $t$, and the outputs $h^{t-1}$ and $c^{t-1}$ of the previous task. During the learning process, the cell state in the LSTM is deployed to accumulate the shared knowledge through experiencing a set of prior tasks. The remember and forget gates in the LSTM episodically refine the cell state by absorbing information from each experienced task. For each individual task, the task-specific information extracted from its support set is combined with distilled information from the previous tasks to infer the adaptive spectral distribution of the kernels.

We demonstrate the effectiveness of the proposed MetaVRF by extensive experiments on a variety of few-shot regression and classification tasks. Results show that our MetaVRF achieves better, or at least competitive, performance compared to previous methods. Moreover, we conduct further analysis on MetaVRF to demonstrate its ability to be integrated with deeper architectures and its efficiency with relatively low sampling rates. We also apply MetaVRF to versatile and challenging settings with inconsistent training and test conditions, and it still delivers promising results, which further demonstrates its strong learning ability.

2. Method

We first describe the base-learner based on the kernel ridge regression in meta-learning for few-shot learning, and then introduce kernel learning with random features, based on which our meta variational random features are developed.

2.1. Meta-Learning with Kernels

We adopt the episodic training strategy commonly used for few-shot classification in meta-learning (Ravi & Larochelle, 2017), which involves meta-training and meta-testing stages. In the meta-training stage, a meta-learner is trained to enhance the performance of a base-learner on a meta-training set with a batch of few-shot learning tasks, where a task is usually referred as an episode (Ravi & Larochelle, 2017). In the meta-test stage, the base-learner is evaluated on a meta-testing set with different classes of data samples from the meta-training set.

For the few-shot classification problem, we sample $C$-way $k$-shot classification tasks from the meta-training set, where $k$
is the number of labelled examples for each of the \( C \) classes. Given the \( t \)-th task with a support set \( S^t = \{(x_i, y_i)\}_{i=1}^{C \times k} \) and query set \( Q^t = \{(\tilde{x}_i, \tilde{y}_i)\}_{i=1}^{m} \) (\( S^t, Q^t \subseteq \mathcal{X} \)), we learn the parameters \( \alpha^t \) of the predictor \( f_{\alpha^t} \) using a standard learning algorithm with kernel trick \( \alpha^t = \Lambda(\Phi(X), Y) \), where \( \Lambda \) is the base-learner and \( \Phi : \mathcal{X} \rightarrow \mathbb{R}^k \) is a mapping function from \( \mathcal{X} \) to a dot product space \( \mathcal{H} \). The similarity measure \( k(x, x') = \langle \Phi(x), \Phi(x') \rangle \) is usually called a kernel (Hofmann et al., 2008).

As in traditional supervised learning problems, the base-learner for the \( t \)-th single task is obtained by solving the following objective w.r.t. \( \alpha^t \): 

\[
\sum_{(x, y) \in Q^t} L \left( f_{\alpha^t}(\Phi(x)), y \right),
\]

where \( L(\cdot) \) can be any differentiable function, e.g., cross-entropy loss. In the meta-learning setting for few-shot learning, we usually consider a batch of tasks. Thus, the meta-learner is trained by optimizing the following objective function for a single task is obtained by solving the following objective w.r.t. \( \alpha^t \): 

\[
\sum_{t} \sum_{(x, y) \in Q^t} L \left( f_{\alpha^t}(\Phi(x)), y \right), \quad \text{s.t.} \quad \alpha^t = \Lambda(\Phi(X), Y),
\]

where \( \Phi^t \) is the feature mapping function which can be obtained by learning a task-specific kernel \( k_t \) for each task \( t \) with data-driven random Fourier features.

In this work, we employ kernel ridge regression, which has an efficient closed-form solution, as the base-learner \( \Lambda \) for few-shot learning. The kernel value in the Gram matrix \( K \in \mathbb{R}^{Ck \times Ck} \) is computed as \( k(x, x') = \Phi(x)\Phi(x')^\top \), where “\(^\top\)” is the transpose operation. The base-learner \( \Lambda \) for a single task is obtained by solving the following objective w.r.t. the support set \( \mathcal{X} \):

\[
\Lambda = \arg \min_{\alpha} \text{Tr}[(Y - \alpha K)(Y - \alpha K)^\top] + \lambda \text{Tr}[\alpha K\alpha^\top],
\]

which admits a closed-form solution

\[
\alpha = Y(\Lambda I + K)^{-1}
\]

The learned predictor is then applied to samples in the query set \( \tilde{X} \):

\[
\tilde{Y} = f_{\alpha}(\tilde{X}) = \alpha \tilde{K},
\]

Here, \( \tilde{K} = \Phi(X)\Phi(\tilde{X})^\top \in \mathbb{R}^{Ck \times m} \), with each element as \( k(x, \tilde{x}) \) between the samples from the support and query sets. Note that we also treat \( \Lambda \) in (3) as a trainable parameter by leveraging the meta-learning setting, and all these parameters are learned by the meta-learner.

Rather than using pre-defined kernels, we consider learning adaptive kernels with random Fourier features in a data-driven way. Moreover, we leverage the shared knowledge by exploring dependencies among related tasks to learn rich features for building up informative kernels.

### 2.2. Random Fourier Features

Random Fourier features (RFFs) were proposed to construct approximate translation-invariant kernels using explicit feature maps (Rahimi & Recht, 2007), based on Bochner’s theorem (Rudin, 1962).

**Theorem 1 (Bochner’s theorem) (Rudin, 1962)** A continuous, real-valued, symmetric and shift-invariant function \( k(x, x') \) on \( \mathbb{R}^d \) is a positive definite kernel if and only if it is the Fourier transform \( p(\omega) \) of a positive finite measure such that

\[
k(x, x') = \int_{\mathbb{R}^d} e^{i\omega^\top(x-x')} dp(\omega) = E_\omega[z_\omega(x)z_\omega(x')^\ast],
\]

where \( z_\omega(x) = e^{i\omega^\top x} \).

It is guaranteed that \( z_\omega(x)z_\omega(x')^\ast \) is an unbiased estimation of \( k(x, x') \) with sufficient RFF bases \( \{\omega\} \) drawn from \( p(\omega) \) (Rahimi & Recht, 2007).

For a predefined kernel, e.g., radius basis function (RBF), we use Monte Carlo sampling to draw bases from the spectral distribution, which gives rise to the explicit feature map:

\[
z(x) = \frac{1}{\sqrt{D}}[\cos(\omega_1^\top x + b_1), \cdots, \cos(\omega_D^\top x + b_D)],
\]

where \( \{\omega_1, \cdots, \omega_D\} \) are the random bases sampled from \( p(\omega) \), and \( \{b_1, \cdots, b_D\} \) are uniformly distributed in the range of \([0, 2\pi]\). Finally, the kernel values \( k(x, x') = z(x)z(x')^\top \) are computed as the dot product of their random feature maps with the same bases.

### 3. Meta Variational Random Features

We introduce our MetaVRF using a latent variable model in which we treat random Fourier bases as latent variables inferred from data. Learning kernels with random Fourier features is tantamount to finding the posterior distribution over random bases in a data-driven way. It is naturally cast into a variational inference problem, where the optimization objective is derived from an evidence lower bound (ELBO) under the meta-learning framework.

#### 3.1. Evidence Lower Bound

From a probabilistic perspective, under the meta-learning setting for few-shot learning, the random feature basis
is obtained by maximizing the conditional predictive log-likelihood of samples from the query set $Q$.

$$\max_p \sum_{(x, y) \in Q} \log p(y|x, S) = \max_p \sum_{(x, y) \in Q} \log \int p(y|x, S, \omega)p(\omega|x, S) d\omega. \quad (9)$$

We adopt a conditional prior distribution $p(\omega|x, S)$ over the base $\omega$ as in the conditional variational auto-encoder (CVAE) (Sohn et al., 2015) rather than an uninformative prior (Kingma & Welling, 2013; Rezende et al., 2014). By depending on the input $x$, we infer the bases that can specifically represent the data, while leveraging the context of the current task by conditioning on the support set $S$.

In order to infer the posterior $p(\omega|y, x, S)$ over $\omega$, which is generally intractable, we resort to using a variational distribution $q_\phi(\omega|S)$ to approximate it, where the base is conditioned on the support set $S$ by leveraging meta-learning. We obtain the variational distribution by minimizing the Kullback-Leibler (KL) divergence

$$D_{KL}[q_\phi(\omega|S)||p(\omega|y, x, S)]. \quad (10)$$

By applying the Bayes’ rule to the posterior $p(\omega|y, x, S)$, we derive the ELBO as

$$\log p(y|x, S) \geq E_{q_\phi(\omega|S)} \log p(y|x, S, \omega) - D_{KL}[q_\phi(\omega|S)||p(\omega|x, S)]. \quad (11)$$

The first term of the ELBO is the predictive log-likelihood conditioned on the observation $x$, $S$ and the inferred RFF bases $\omega$. Maximizing it enables us to make an accurate prediction for the query set by utilizing the inferred bases from the support set. The second term in the ELBO minimizes the discrepancy between the meta variational distribution $q_\phi(\omega|S)$ and the meta prior $p(\omega|x, S)$, which encourages samples from the support and query sets to share the same random Fourier bases. The full derivation of the ELBO is provided in the supplementary material.

We now obtain the objective by maximizing the ELBO with respect to a batch of $T$ tasks:

$$L = \frac{1}{T} \sum_{t=1}^T \left( \sum_{(x, y) \in Q^t} E_{q_\phi(\omega^t|S^t)} \log p(y|x, S^t, \omega^t) - D_{KL}[q_\phi(\omega^t|S^t)||p(\omega^t|x, S^t)] \right), \quad (12)$$

where $S^t$ is the support set of the $t$-th task associated with its specific bases $\{\omega^t_d\}_{d=1}^D$ and $(x, y) \in Q^t$ is the sample from the query set of the $t$-th task. Directly optimizing the above objective does not take into account the task dependency. Thus, we introduce context inference by conditioning the posterior on both the support set of the current task and the shared knowledge extracted from previous tasks.

Figure 2. Illustration of MetaVRF in a directed graphical model, where $(x, y)$ is a sample in the query set $Q^t$. The base $\omega^t$ of the $t$-th task is dependent on the support set $S^t$ of the current task and the context $C$ of related tasks.

3.2. Context Inference

We propose a context inference which puts the inference of random feature bases for the current task in the context of related tasks. We replace the variational distribution in (10) with a conditional distribution $q_\phi(\omega^t|S^t, C)$ that makes the bases $\{\omega^t_d\}_{d=1}^D$ of the current $t$-th task conditioned also on the context $C$ of related tasks.

The context inference gives rise to a new ELBO, as follows:

$$\log p(y|x, S^t) \geq E_{q_\phi(\omega|S^t, C)} \log p(y|x, S^t, \omega) - D_{KL}[q_\phi(\omega|S^t, C)||p(\omega|x, S^t)], \quad (13)$$

which can be represented in a directed graphical model as shown in Figure 2. In a practical sense, the KL term in (13) encourages the model to extract useful information from previous tasks for inferring the spectral distribution associated with each individual sample $x$ of the query set in the current task.

The context inference integrates the knowledge shared across tasks with the task-specific knowledge to build up adaptive kernels for individual tasks. The inferred random features are highly informative due to the absorbed information from prior knowledge of experienced tasks. The base-learner built on the inferred kernel with the informative random features can effectively solve the current task.

However, since there is usually a huge number of related tasks, it is non-trivial to model all these tasks simultaneously. We consider using recurrent neural networks to gradually accumulate information episodically along with the learning process by organizing tasks in a sequence. We propose an LSTM-based inference network by leveraging its innate capability of remembering long-term information (Gers & Schmidhuber, 2000). The LSTM offers a well-suited structure to implement the context inference. The cell state $c$ stores and accrues the meta knowledge shared among related tasks, which can also be updated when experiencing a new task in each episode during the course of learning; the output $h$ is used to adapt to each specific task.
To be more specific, we model the variational posterior $q_\phi(\omega^t | S^t, C)$ through $q_\phi(\omega | h^t)$ which is parameterized as a multi-layer perceptron (MLP) $\phi(h^t)$. Note that $h^t$ is the output from an LSTM that takes $S^t$ and $C$ as inputs. We implement the inference network with both vanilla and bidirectional LSTMs (Schuster & Paliwal, 1997; Graves & Schmidhuber, 2005). For a vanilla LSTM, we have
\[
[h^t, c^t] = g_{\text{LSTM}}(S^t, h^{t-1}, c^{t-1}),
\]
where $g_{\text{LSTM}}(\cdot)$ is a vanilla LSTM network that takes the current support set, the output $h^{t-1}$ and the cell state $c^{t-1}$ as the input. $S^t$ is the average over the feature representation vectors of samples in the support set (Zaheer et al., 2017). The feature representation is obtained by a shared convolutional network $\psi(\cdot)$. To incorporate more context information, we also implement the inference with a bidirectional LSTM, and we have $h^t = [h^t_\rightarrow, h^t_\leftarrow]$, where $h^t_\rightarrow$ and $h^t_\leftarrow$ are the outputs from forward and backward LSTMs, respectively, and $[\cdot, \cdot]$ indicates a concatenation operation.

Therefore, the optimization objective with the context inference is:
\[
L = \frac{1}{T} \sum_{t=1}^{T} \left( \sum_{(x,y) \in Q^t} \mathbb{E}_{q_\phi(\omega^t | h^t)} \log p(y | x, S^t, \omega^t) \right. \\
\left. - D_{\text{KL}}[q_\phi(\omega^t | h^t) \| p(\omega^t | x, S^t)] \right).
\]
where the variational approximate posterior $q_\phi(\omega^t | h^t)$ is taken as a multivariate Gaussian with a diagonal co-variance. Given the support set as input, the mean $\omega_\mu$ and standard deviation $\omega_\sigma$ are output from the inference network $\phi(\cdot)$. The conditional prior $p(\omega^t | x, S^t)$ is implemented with a prior network which takes an aggregated representation by using the cross attention (Kim et al., 2019) between $x$ and $S^t$. The details of the prior network are provided in the supplementary material. To enable back-propagation with the sampling operation during training, we adopt the reparameterization trick (Rezende et al., 2014; Kingma & Welling, 2013) as $\omega = \omega_\mu + \omega_\sigma \odot \epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$.

During the course of learning, the LSTMs accumulate knowledge in the cell state by updating their cells using information extracted from each task. For the current task $t$, the knowledge stored in the cell is combined with the task-specific information from the support set to infer the spectral distribution for this task. To accrue the information across all the tasks in the meta-training set, the output and the cell state of LSTMs are passed down across batches. As a result, the final cell state contains the distilled prior knowledge from all the tasks experienced in the meta-training set.

**Fast Adaptation.** Once meta-training ceases, the output and the cell state are directly used for a new incoming task in the meta-test set to achieve fast adaptation with a simple feed-forward computation operation. To be more specific, for a task with the support set $S^*$ in the meta-test set, we draw $D$ samples $\{\omega^{(l)}\}_{l=1}^D$ as the bases: $\omega^{(l)} \sim q(\omega | h^*)$, where $h^*$ is output from either a vanilla LSTM or a bidirectional LSTM, depending on which is used during the meta-training stage. The bases are adopted to compute the kernels on the support set and construct the classifier of the base-learner for the task, using (4). The classifier is then used to make predictions of samples in the query set for performance evaluation.

4. Related Work

**Meta-learning.** or learning to learn, endues machine learning models the ability to improve their performance by leveraging knowledge extracted from a number of prior tasks. It has received increasing research interest with breakthroughs in many directions (Finn et al., 2017; Rusu et al., 2019; Gordon et al., 2019; Aravind Rajeswaran, 2019). Gradient-based methods (e.g., MAML (Finn et al., 2017)) learn an appropriate initialization of model parameters and adapt it to new tasks with only a few gradient steps (Finn & Levine, 2018; Zintgraf et al., 2019; Rusu et al., 2019). Learning a shared optimization algorithm has also been explored in order to quickly learn of new tasks (Ravi & Larochelle, 2017; Andrychowicz et al., 2016; Chen et al., 2017).

**Metric learning** has been widely studied with great success for few-shot learning (Vinyals et al., 2016; Snell et al., 2017; Satorras & Estrach, 2018; Oreshkin et al., 2018; Allen et al., 2019). The basic assumption is that a common metric space is shared across related tasks. By extending the matching network (Vinyals et al., 2016) to few-shot scenarios, Snell et al. (Snell et al., 2017) constructed a prototype for each class by averaging feature representations of samples from the class in the metric space. The classification is conducted via matching the query samples to prototypes by computing their distances. To enhance the prototype representation, Allen et al. (Allen et al., 2019) proposed an infinite mixture of prototypes (IMP) to adaptively represent data distribution of each class by multiple clusters instead of using a single vector. In addition, Oreshkin et al. (Oreshkin et al., 2018) proposed task dependent adaptive metric for improved few-shot learning and established prototypes of classes conditioning on a task representation encoded by a task embedding network. Their results indicate the benefit of learning task-specific metric in few-shot learning.

While these meta-learning algorithms have made great progress in few-shot learning tasks, exploring prior knowledge from previous tasks remains an open challenge (Titsias et al., 2019). In this work, we introduce kernels based on random features as the base-learners, which enables us to acquire shared knowledge across tasks by modeling their dependency via the random feature basis of kernels.
Kernel learning with random Fourier features is a versatile and powerful tool in machine learning (Bishop, 2006; Hofmann et al., 2008; Shervashidze et al., 2011). Pioneering works (Bach et al., 2004; Gönen & Alpaydın, 2011; Duvenaud et al., 2013) learn to combine predefined kernels in a multi-kernel learning manner. Kernel approximation by random Fourier features (RFFs) (Rahimi & Recht, 2007) is an effective technique for efficient kernel learning (Gärtner et al., 2002), which has recently become increasingly popular (Sinha & Duchi, 2016; Carratino et al., 2018). Recent works (Wilson & Adams, 2013) learn kernels in the frequency domain by modeling the spectral distribution as a mixture of Gaussians and computing its optimal linear combination. Instead of modeling the spectral distribution with explicit density functions, other works focus on optimizing the random base sampling strategy (Yang et al., 2015; Sinha & Duchi, 2016). Nonetheless, it has been shown that accurate approximation of kernels does not necessarily result in high classification performance (Avron et al., 2016; Chang et al., 2017). This suggests that learning adaptive kernels with random features by data-driven sampling strategies (Sinha & Duchi, 2016) can improve the performance, even with a low sampling rate compared to using universal random features (Avron et al., 2016; Chang et al., 2017).

Our MetaVRF is the first work to introduce kernel learning with random features to the meta-learning framework for few-shot learning. The optimization of MetaVRF is naturally cast as a variational inference and the context inference offers a principled way to incorporate prior knowledge and achieve informative and adaptive kernels.

5. Experiments

We evaluate our MetaVRF on several few-shot learning problems for both regression and classification. We demonstrate the benefit of exploring task dependency by implementing a baseline MetaVRF (12) without using the LSTM, which infers the random base solely from the support set. We also conduct further analysis to validate the effectiveness of our MetaVRF by showing its performance with deep embedding architectures, different numbers of bases, and under versatile and challenging settings with inconsistent training and test conditions.

5.1. Few-Shot Regression

We conduct regression tasks with different numbers of shots \( k \), and compare our MetaVRF with MAML (Finn et al., 2017), a representative meta-learning algorithm. We follow the MAML work (Finn et al., 2017) to fit a target sine function \( y = A \sin(wx + b) \), with only a few annotated samples. \( A \in [0.1, 5] \), \( w \in [0.8, 1.2] \), and \( b \in [0, \pi] \) denote the amplitude, frequency, and phase, respectively, which follow a uniform distribution within the corresponding interval.

The goal is to estimate the target sine function given only \( n \) randomly sampled data points. In our experiments, we consider the input in the range of \( x \in [-5, 5] \), and conduct three tests under the conditions of \( k = 3, 5, 10 \). For a fair comparison, we compute the feature embedding using a small multi-layer perception (MLP) with two hidden layers of size 40, following the same settings used in MAML.

The results in Figure 3 show that our MetaVRF fits the function well with only three shots but performs better with an increasing number of shots, almost entirely fitting the target function with ten shots. Moreover, the results demonstrate the advantage of exploring task dependency by LSTM-based inference. MetaVRF with bi-LSTM performs better than regular LSTM since more context tasks are incorporated by bi-LSTM. In addition, we observe that MetaVRF performs better than MAML for all three settings with varying numbers of shots. We provide more results on few-shot regression tasks in the supplementary material.

5.2. Few-Shot Classification

The classification experiments are conducted on three commonly-used benchmark datasets, i.e., Omniglot (Lake et al., 2015), miniImageNet (Vinyals et al., 2016) and CIFAR-FS (Krizhevsky et al., 2009); for more details, please refer to the supplementary material. We extract image features using a shallow convolutional neural network with the same architecture as in (Gordon et al., 2019). We do not use any fully connected layers for these CNNs. The dimension of all feature vectors is 256. We also evaluate the random Fourier features (RFFs) and the radial basis function (RBF) kernel in which we take the bandwidth \( \sigma \) as the mean of pair-wise distances between samples in the support set of each task. The inference network \( \phi(\cdot) \) is a three-layer MLP with 256 units in the hidden layers and rectifier non-linearity where input sizes are 256 and 512 for the vanilla and bidirectional LSTMs, respectively.
Learning to Learn Kernels with Variational Random Features

The key hyperparameter for the number of bases $D$ in (7) is set to $D = 780$ for MetaVRF in all experiments, while we use RFFs with $D = 2048$ as this produces the best performance. The sampling rate in our MetaVRF is much lower than in previous works using RFFs, in which $D$ is usually set to be 5 to 10 times the dimension of the input features (Yu et al., 2016; Rahimi & Recht, 2007). We adopt a similar meta-testing protocol as (Gordon et al., 2019; Finn et al., 2017), but we test on 3,000 episodes rather than 600 and present the results with 95% confidence intervals. All reported results are produced by models trained from scratch. We compare with previous methods that use the same training procedures and similar shallow conventional CNN architectures as ours. The comparison results on three benchmark datasets are reported in Tables 1 and 2.

On all benchmark datasets, MetaVRF delivers the state-of-the-art performance. Even with a relatively low sampling rate, MetaVRF produces consistently better performance compared with RBF kernels, RFFs. MetaVRF with bi-LSTM outperforms the one with vanilla LSTM since it can leverage more information. It is worth mentioning that MetaVRF with bi-LSTM achieves good performance (54.2%) under the 5-way 1-shot setting on the miniImageNet dataset, surpassing the second best model by 1%. The MetaVRFs with bi-LSTM and vanilla LSTM consistently outperform the one without the LSTM, which demonstrates the effectiveness of using LSTM to explore task dependency. The much better performance over RBF kernels on all three datasets indicates the great benefit of adaptive kernels based on random Fourier features. The relatively inferior performance produced by RBF kernels is due to that the mean of pair-wise distances of support samples would not be able to provide a proper estimate of the kernel bandwidth since we only have a few samples in each task, for instance, five samples under the 5-way 1-shot setting in the support set.

Note that on Omniglot, the performance of existing methods saturates and MetaVRF with bi-LSTM achieves the best performance for most settings, including 5-way 1-shot, 5-way 5-shot, and 20-way 1-shot. It is also competitive under the 20-way 5-shot setting falling within the error bars of the state-of-the-arts. In Table 1, we also implement a MAML (64C) with 64 channels in each convolutional layer. However, while it obtains modest performance, we believe the increased model size leads to overfitting. Since in the original SNAIL, a very deep ResNet-12 network is used for embedding, we cite the result of SNAIL reported in Bertinetto et al. (2019) using similar shallow networks as ours. For fair comparison, we also cite the original results of R2-D2 (Bertinetto et al., 2019) using 64 channels.

### 5.3. Further Analysis

#### Deep embedding

Our MetaVRF is independent of the convolutional architectures for feature extraction and can work with deeper embeddings either pre-trained or trained from scratch. In general, the performance improves with more powerful feature extraction architectures. We evaluate our method using pre-trained embeddings in order to compare with existing methods using deep embedding architectures. To benchmark with those methods, we adopt the pre-trained embeddings from a 28-layer wide residual network (WRN-28-10) (Zagoruyko & Komodakis, 2016), in a similar fash-

---

**Table 1. Performance (%) on miniImageNet and CIFAR-FS.**

<table>
<thead>
<tr>
<th>Method</th>
<th><strong>miniImageNet, 5-way</strong></th>
<th></th>
<th><strong>CIFAR-FS, 5-way</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-shot</td>
<td>5-shot</td>
<td>1-shot</td>
<td>5-shot</td>
</tr>
<tr>
<td>Matching net (Vinyals et al., 2016)</td>
<td>44.2</td>
<td>57</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>MAML (Finn et al., 2017)</td>
<td>48.7±1.8</td>
<td>63.1±0.9</td>
<td>58.9±1.9</td>
<td>71.5±1.0</td>
</tr>
<tr>
<td>MAML (64C)</td>
<td>46.7±1.7</td>
<td>61.1±0.1</td>
<td>58.9±1.8</td>
<td>71.5±1.1</td>
</tr>
<tr>
<td>META-LSTM (Ravi &amp; Larochelle, 2017)</td>
<td>43.4±0.8</td>
<td>60.6±0.7</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Proto-LSTM (Snell et al., 2017)</td>
<td>47.4±0.6</td>
<td>65.4±0.5</td>
<td>55.5±0.7</td>
<td>72.0±0.6</td>
</tr>
<tr>
<td>Relation net (Sung et al., 2018)</td>
<td>50.4±0.8</td>
<td>65.3±0.7</td>
<td>55.0±1.0</td>
<td>69.3±0.8</td>
</tr>
<tr>
<td>SNAIL (32C) by (Bertinetto et al., 2019)</td>
<td>45.1</td>
<td>55.2</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>GNN (Garcia &amp; Bruna, 2018)</td>
<td>50.3</td>
<td>66.4</td>
<td>61.9</td>
<td>75.3</td>
</tr>
<tr>
<td>Platipus (Finn et al., 2018)</td>
<td>50.1±1.9</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Versa (Gordon et al., 2019)</td>
<td>53.3±1.8</td>
<td>67.3±0.9</td>
<td>62.5±1.7</td>
<td>75.1±0.9</td>
</tr>
<tr>
<td>R2-D2 (64C) (Bertinetto et al., 2019)</td>
<td>49.5±0.2</td>
<td>65.4±0.2</td>
<td>62.3±0.2</td>
<td>77.4±0.2</td>
</tr>
<tr>
<td>R2-D2 (Devos et al., 2019)</td>
<td>51.7±1.8</td>
<td>63.3±0.9</td>
<td>60.2±1.8</td>
<td>70.9±0.9</td>
</tr>
<tr>
<td>CAVIA (Zintgraf et al., 2019)</td>
<td>51.8±0.7</td>
<td>65.6±0.6</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>iMAML (Aravind Rajeswaran, 2019)</td>
<td>49.3±1.9</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>RBF Kernel</td>
<td>42.1±1.2</td>
<td>54.9±1.1</td>
<td>46.0±1.2</td>
<td>59.8±1.0</td>
</tr>
<tr>
<td>RFFs (2048d)</td>
<td>52.8±0.9</td>
<td>65.4±0.9</td>
<td>61.1±0.8</td>
<td>74.7±0.9</td>
</tr>
<tr>
<td>MetaVRF (w/o LSTM, 780d)</td>
<td>51.3±0.8</td>
<td>66.1±0.7</td>
<td>61.1±0.7</td>
<td>74.3±0.9</td>
</tr>
<tr>
<td>MetaVRF (vanilla LSTM, 780d)</td>
<td>53.1±0.9</td>
<td>66.8±0.7</td>
<td>62.1±0.8</td>
<td>76.0±0.8</td>
</tr>
<tr>
<td>MetaVRF (bi-LSTM, 780d)</td>
<td><strong>54.2±0.8</strong></td>
<td><strong>67.8±0.7</strong></td>
<td><strong>63.1±0.7</strong></td>
<td><strong>76.5±0.9</strong></td>
</tr>
</tbody>
</table>
Learning to Learn Kernels with Variational Random Features

Table 2. Performance (%) on Omniglot.

<table>
<thead>
<tr>
<th>Method</th>
<th>Omniglot, 5-way</th>
<th>Omniglot, 20-way</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-shot</td>
<td>5-shot</td>
</tr>
<tr>
<td>SIAMESE NET (Koch, 2015)</td>
<td>96.7 ± 0.1</td>
<td>98.4 ± 0.1</td>
</tr>
<tr>
<td>MATCHING NET (Vinyals et al., 2016)</td>
<td>98.1 ± 0.2</td>
<td>98.9 ± 0.2</td>
</tr>
<tr>
<td>MAML (Finn et al., 2017)</td>
<td>98.7 ± 0.1</td>
<td>99.9 ± 0.1</td>
</tr>
<tr>
<td>PROTO NET (Snell et al., 2017)</td>
<td>98.5 ± 0.2</td>
<td>99.5 ± 0.2</td>
</tr>
<tr>
<td>GNN (Garcia &amp; Bruna, 2018)</td>
<td>99.2 ± 0.2</td>
<td>99.7 ± 0.2</td>
</tr>
<tr>
<td>VERSA (Gordon et al., 2019)</td>
<td>99.7 ± 0.2</td>
<td>99.8 ± 0.2</td>
</tr>
<tr>
<td>R2-D2 (Bertinetto et al., 2019)</td>
<td>98.6</td>
<td>99.7</td>
</tr>
<tr>
<td>IMP (Allen et al., 2019)</td>
<td>98.4 ± 0.3</td>
<td>99.5 ± 0.1</td>
</tr>
</tbody>
</table>

Table 3. Performance (%) on miniImageNet (5-way)

<table>
<thead>
<tr>
<th>Method</th>
<th>1-shot</th>
<th>5-shot</th>
</tr>
</thead>
<tbody>
<tr>
<td>META-SGD (Li et al., 2017)</td>
<td>54.24 ± 0.03</td>
<td>70.86 ± 0.04</td>
</tr>
<tr>
<td>(Gidaris &amp; Komodakis, 2018)</td>
<td>56.20 ± 0.86</td>
<td>73.00 ± 0.64</td>
</tr>
<tr>
<td>(Bauer et al., 2017)</td>
<td>56.30 ± 0.40</td>
<td>73.90 ± 0.30</td>
</tr>
<tr>
<td>(Munkhdalai et al., 2017)</td>
<td>57.10 ± 0.70</td>
<td>70.04 ± 0.63</td>
</tr>
<tr>
<td>(Qiao et al., 2018)</td>
<td>59.60 ± 0.41</td>
<td>73.54 ± 0.19</td>
</tr>
<tr>
<td>LEO (Rusu et al., 2019)</td>
<td>61.76 ± 0.08</td>
<td>77.59 ± 0.12</td>
</tr>
<tr>
<td>SNAIL (Mishra et al., 2018)</td>
<td>55.71 ± 0.99</td>
<td>68.88 ± 0.92</td>
</tr>
<tr>
<td>TADAM (Oreshkin et al., 2018)</td>
<td>58.50 ± 0.30</td>
<td>76.70 ± 0.30</td>
</tr>
<tr>
<td>META-VRF (w/o LSTM, 780d)</td>
<td><strong>62.12 ± 0.07</strong></td>
<td>77.05 ± 0.28</td>
</tr>
<tr>
<td>META-VRF (vanilla LSTM, 780d)</td>
<td><strong>63.21 ± 0.06</strong></td>
<td><strong>77.83 ± 0.28</strong></td>
</tr>
<tr>
<td>META-VRF (bi-LSTM, 780d)</td>
<td><strong>63.80 ± 0.05</strong></td>
<td><strong>77.97 ± 0.28</strong></td>
</tr>
</tbody>
</table>

Figure 4. Performance with different numbers $D$ of bases. Our MetaVRF consistently achieves better performance than regular RFFs, especially with relatively low sampling rates.

Figure 5. Performance with varied ways and shots on Omniglot.

ion to (Rusu et al., 2019; Bauer et al., 2017; Qiao et al., 2018). We choose activations in the 21-st layer, with average pooling over spatial dimensions, as feature embeddings. The dimension of pre-trained embeddings is 640. We show the comparison results on the miniImageNet dataset for 5-way 1-shot and 5-shot settings in Table 3. Our MetaVRF with bi-LSTM achieves the best performance under both settings and largely surpasses LEO, a recently proposed meta-learning method, especially on the challenging 5-way 1-shot setting. Note that the MetaVRF with vanilla LSTM and without LSTM also produce competitive performance.

**Efficiency.** Regular random Fourier features (RFFs) usually require high sampling rates to achieve satisfactory performance. However, our MetaVRF can achieve high performance with a relatively low sampling rate compared, which guarantees its high efficiency. In Figure 4, we compare with regular RFFs using different sampling rates. We show the performance change of fully trained models using RFFs and our MetaVRF with bi-LSTM under a different number $D$ of bases. We show the comparison results for the 5-way 5-shot setting in Figure 4. MetaVRF with bi-LSTM consistently yields higher performance than regular RFFs with the same number of sample bases. The results verify the efficiency of our MetaVRF in learning adaptive kernels and the effectiveness in improving performance by exploring dependencies of related tasks.

**Versatility.** In contrast to most existing meta-learning methods, our MetaVRF can be used for versatile settings. We evaluate the performance of MetaVRF on more challenging scenarios where the number of ways $C$ and shots $k$ between training and testing are inconsistent. Specifically, we test the performance of MetaVRF on tasks with varied $C$ and $k$, when it is trained on one particular $C$-way-$k$-shot task. As shown in Figure 5, the results demonstrate that the
trained model can still produce good performance, even on the challenging condition with a far higher number of ways. In particular, the model trained on the 20-way-5-shot task can retain a high accuracy of 94% on the 100-way setting, as shown in Figure 5(a). The results also indicate that our MetaVRF exhibits considerable robustness and flexibility to a great variety of testing conditions.

6. Conclusion
In this paper, we introduce kernel approximation based on random Fourier features into the meta-learning framework for few-shot learning. We propose meta variational random features (MetaVRF), which leverage variational inference and meta-learning to infer the spectral distribution of random Fourier features in a data-driven way. MetaVRF generates random Fourier features of high representational power with a relatively low spectral sampling rate by using an LSTM based inference network to explore the shared knowledge. In practice, our LSTM-based inference network demonstrates a great ability to collect and absorb common knowledge from experience tasks, which enables quick adaptation to specific tasks for improved performance. Experimental results on both regression and classification tasks demonstrate the effectiveness for few-shot learning.

Acknowledgements
This research was supported in part by Natural Science Foundation of China (No. 61976060, 61871016, 61876098).

References


Learning to Learn Kernels with Variational Random Features


