A Model-Agnostic Randomized Learning Framework based on Random Hypothesis Subspace Sampling

Yiting Cao [1] **Chao Lan** [1]

Abstract

We propose a model-agnostic randomized learning framework based on Random Hypothesis Subspace Sampling (RHSS). Given any hypothesis class, it randomly samples k hypotheses and learns a near-optimal model from their span by simply solving a linear least square problem in $O(nk^2)$ time, where n is the number of training instances. On the theory side, we derive the performance guarantee of RHSS from a generic subspace approximation perspective, leveraging properties of metric entropy and random matrices. On the practical side, we apply the RHSS framework to learn kernel, network and tree based models. Experimental results show they converge efficiently as k increases and outperform their model-specific counterparts including random fourier feature, random vector functional link and extra tree on real-world data sets.

1. Introduction

Randomized machine learning is a research topic that studies how to randomize the learning process, often with an aim of improving learning efficiency. Representative techniques range from random projection (Vempala, 2005) for efficient dimensionality reduction to extremely randomized decision tree (Geurts et al., 2006), and from random Fourier feature (Rahimi & Recht, 2008) for efficient kernel methods to random vector functional link (Needell et al., 2020) for efficient network training. These techniques have received adequate research interests over the past decades.

When inspecting the literature, we notice that most randomized learning techniques are model-specific. For example, in (Geurts et al., 2006), tree generation is randomized by using random features to split tree nodes; in (Rahimi & Recht, 2008), a kernel machine is randomized by using random Fourier features to approximate kernel functions; in (Needell et al., 2020), neural network training is randomized by fixing all but the output weights to random values. While these techniques have achieved promising results on their designated models respecitvely, it remains unclear how they could be applied on other models or guide the design of randomized learners for them.

Random projection (Vempala, 2005) is a randomized dimensionality reduction technique, which projects data into a lower dimensional feature space through random linear projections. It can be applied to speed up downstream learning and considered as model-agnostic to the learner. However, the speedup is often limited because the learner is not randomized and could remain inefficient especially if its time complexity does not depend heavily on the original feature dimension such as kernel methods.

The above observations reveal the lack of a model-agnostic randomized learning framework that not only ties the existing techniques for certain models but also provides guidance on designing randomized learners for other models. We believe such framework will help to significantly advance the research and application of randomized machine learning. This motivates the present study.

A major contribution of this paper is the design of a model-agnostic randomized learning framework based on Random Hypothesis Subspace Sampling (RHSS). Given any hypothesis class, it randomly samples k hypotheses and learns a model in their span that best approximates the target model on a set of n training instances. Importantly, this learning process can always be cast as a simple linear least square problem and solvable in $O(nk^2)$ time. In practice, small k often suffices for good performance, which makes RHSS-based learning extremely efficient no matter how complex the given hypothesis class is.

On the theory side, we derive the performance guarantee of RHSS from a generic subspace approximation perspective, leveraging properties of metric entropy and random matrices. Under proper conditions, we show the best model learned from the span of k randomly sampled hypotheses can approximate any target model on a fixed data set by up

[1]School of Computer Science, University of Oklahoma, Norman, OK, USA. Correspondence to: Chao Lan <clan@ou.edu>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

to an $O(k^{-c})$ error with high probability, where c is a constant in $(0, 1)$. Although this bound is not as tight as those developed for model-specific randomized learners such as in (Rudi & Rosasco, 2017; Avron et al., 2017), in experiments we observe RHSS has similar or even better performance than their counterparts. Nonetheless, it remains an open question on how to bridge the gap theoretically.

On the practical side, we demonstrate the applications of RHSS on kernel, neural network and tree based models, and discuss their connections to the existing randomized learners that are specifically designed for these models, including random Fourier features, random vector functional link and extra tree. In experiments, we compare the proposed RHSS-based learners with standard learners and model-specific randomized learners. We see they approach standard learners efficiently as k increases, and often outperform their model-specific counterparts on real-world data sets.

The rest of this paper is organized as follows: Section 2 reviews related work; Section 3 presents the proposed RHSS framework; Section 4 presents its theoretical analysis and Section 5 demonstrates its applications; experimental results are shown in Section 6 and conclusion in Section 7.

2. Related Work

2.1. Random Fourier Feature

Random Fourier Features (RFF) is designed to speed up kernel methods (Rahimi & Recht, 2008). It approximates kernel functions using inner products of explicit feature vectors generated through random Fourier functions, and thus bypasses the need of working with the Gram matrix. With n training instances and k random features, RFF reduces the typical time complexity of learning a kernel machine from $O(n^3)$ to $O(nk^2)$, where k is often smaller than n. Because of its outstanding efficiency, RFF has been intensively studied in the past e.g. (Yang et al., 2012; Szabó, 2015; Rudi & Rosasco, 2017; Avron et al., 2017; Li, 2017; Li et al., 2019).

It remains unclear, however, that how RFF can be applied on non-kernel machines such as network or tree that do not necessarily work with Gram matrices. One may use it as a feature preprocessing technique such as generating a tree based on Fourier features, but there is little guarantee on the accuracy or efficiency of downstream learning. (See Section 2.4 for more discussion on the limitations of randomized feature preprocessing.) Comparatively, the proposed RHSS framework applies to both kernel and non-kernel machines.

2.2. Random Vector Functional Link

Random Vector Functional Link (RVFL) is designed to speed up multi-layer perceptron learning (Pao et al., 1994; Igelnik & Pao, 1995). It only optimizes the weights between the last hidden layer and the output layer, and randomly sets the other weights for the final network. With n training instances and m neurons in the last hidden layer, RVFL can efficiently learn the network in $O(nm^2)$ time. Although proposed in the last century, RVFL is re-gaining research interests in recent years (Zhang & Suganthan, 2016; Needell et al., 2020; Gallicchio & Scardapane, 2020).

Apparently, it is unclear how RVFL can be applied on non-network models such as kernel machine or tree that are not constructed by ordered layers of weights. Comparatively, the proposed RHSS framework applies to both network and non-network models. Interestingly, when applied on multi-layer perceptron, RHSS can be viewed as applying the RVFL principle on a specially constructed network. See Figure 1 and related discussions in Section 5.2.

2.3. Extra Tree

Extra tree is designed to speed up tree learning (Geurts et al., 2006). It randomly selects features to split nodes when generating each tree, and outputs the average of multiple trees as the final model. By avoiding the search of optimal features for node splitting, extra tree is more efficient than standard tree learning and has received successful applications (Desir et al., 2012; Maier et al., 2015).

Similar to RFF and RVFL, however, it is unclear how extra tree can be applied on non-tree models that do not have nodes to split. Comparatively, the proposed RHSS framework applies to both tree and non-tree models. When applied on tree, RHSS uses the same method as extra tree to generate multiple trees, but then outputs an optimally weighted average of them as the final model.

From the tree ensemble perspective, RHSS and extra tree are both connected to random forest. The latter is a powerful non-randomized tree learning method, which also averages multiple trees but each tree finds optimal features (from a sub-pool) to split nodes. We do not expect randomized tree learners to beat random forest in accuracy, yet our experimental results suggest they provide well approximations while being significantly more efficient to learn.

In light of the above discussion, we also see some connection between RHSS and boosting, since the latter also finds an optimal weighted average of models. Yet, they have a fundamental difference that boosting optimizes each model (thus not a randomized learner) whereas RHSS randomly picks each model. Besides, models in boosting are often dependent whereas models in RHSS are i.i.d. sampled.

2.4. Random Projection

Random Projection (RP) is design to speed up dimensionality reduction. It maps a set of data into a lower dimensional feature space through randomly generated linear projections

(Vempala, 2005). Compared to other reduction methods, RP is more efficient in that it avoids the search of optimal projections which often has a high time complexity such as $O(p^3)$ for p input features in PCA. Besides, it is proved that data distortion in the randomly projected feature space is likely bounded (Arriaga & Vempala, 2006) and thus RP will not significantly deteriorate the performance of downstream learning (Fradkin & Madigan, 2003; Maillard & Munos, 2012; Durrant & Kabán, 2013).

RP is often applied to speed up downstream learning and considered as model-agnostic to the learner. However, the speedup is often limited because the learner could remain inefficient. For example, after using RP to reduce feature dimension, learning a kernel machine still takes $O(n^3)$ time with n training instances and RVFL still takes $O(nm^2)$ time with m hidden neurons. Comparatively, the proposed RHSS directly speeds up the learner (through approximation). When applied on linear hypothesis class, RHSS is equivalent to RP followed by the learning of a linear model.

From a broader randomized feature projection perspective, another related work is randomized kernel locality sensitive hashing (R-KLSH) (Garg et al., 2019b;a). It designs randomized hash functions (with parameters) to generate binary features, then optimizes them for label prediction, and finally use them to train multiple tree models that are at the end assembled into a random forest.

Both R-KLSH and RP generate features with randomness for downstream learning, but they have two fundamental difference. First, features of RP are random while features of R-KLSH are semi-random since they are optimized from data for the prediction task. Second, RP speeds up learning by generating few features, but R-KLSH often needs to generate redundant features and achieves speedup based on the binary property of these features that can be efficiently exploited by tree models – from this perspective, R-KLSH is model-specific. These, plus the difference between RP and RHSS, are the difference between R-KLSH and RHSS.

3. The RHSS-based Learning Framework

In this section, we present the RHSS-based randomized learning framework. Its basic idea is to randomly sample some hypotheses and then learn an optimal model in their span. Specifically, given a hypothesis class and a set of n training instances, RHSS operates in four steps:

(1) Randomly sample k hypotheses from the class.

(2) Apply each sampled hypothesis on all training instances and obtain an n-dimensional vector of its predicted labels.

(3) Learn an optimal linear combination of the prediction vectors that best approximates the vector of true labels.

Framework 1 The RHSS-based Learning Framework

Input: hypothesis class H, sampling distribution D, a labeled set $(x_1, y_1), \ldots, (x_n, y_n)$, hyper-parameter k

1: Independently sample $h_1, \ldots, h_k \in H$ based on D.
2: Calculate $\tilde{h}_i = [h_i(x_1), \ldots, h_i(x_n)]^T$ for each i.
3: Optimize coefficients $\alpha_1, \ldots, \alpha_k \in \mathbb{R}$ by solving

$$\min_{\alpha_1,\ldots,\alpha_k} \left\| \sum_{i=1}^{k} \alpha_i \tilde{h}_i - \tilde{y} \right\|^2, \tag{1}$$

where $\tilde{y} = [y_1, \ldots, y_n]^T$.

Output: Model $f = \sum_{i=1}^{k} \alpha_i h_i$.

(4) Output the combined hypothesis.

Detailed learning process is elaborated in Framework 1. In the framework, x_i is the i_{th} instance in the training set and y_i is its label. The number of sampled hypotheses k is a hyper-parameter. The design of specific hypothesis sampling approach is dependent on the hypothesis class, and we will show three examples in Section 5.

Once an output model f is obtained, we can apply it on a testing point z by first applying the sampled hypotheses to obtain $h_1(z), \ldots, h_k(z)$ and then calculating the prediction as $f(z) = \alpha_1 h_1(z) + \ldots + \alpha_k h_k(z)$.

As one can see, the RHSS framework is fairly simple and easy to apply as learning is always formulated as a linear least square problem solvable in $O(nk^2)$ time, no matter how complex the hypothesis class is. In experiments, we observe that small k often suffices for good performance, which makes RHSS-based learning very efficient.

Next, we inspect the theoretical guarantees of RHSS and demonstrate its applications on three hypothesis classes.

4. Theoretical Analysis of RHSS

In this section, we inspect the theoretical guarantee of RHSS from a generic subspace approximation perspective, i.e., it is equivalent to approximating a target set using a random subspace spanned by the columns of a random matrix.

In the following, we first introduce a set of analytic tools, some borrowed from the literature of metric entropy and random matrices and some developed by ourselves (with proofs given in the appendix). Then, we present the main theoretical results and discuss their implications.

4.1. Preliminaries

Our analysis of random subspace approximation will be performed on *Grassmannian*. Let $G_{n,\ell}$ be a Grassmannian

consisting of all the ℓ-subspaces of $\mathbb{R}^n$, and the distance between any two $U, V \in G_{n,\ell}$ is measured by

$$d_{G,r}(U,V) = \Delta(U \cap S_r, V \cap S_r), \tag{2}$$

where Δ is the Hausdorff distance defined as $\Delta(X,Y) = \max\{\sup_{x\in X}\inf_{y\in Y}\|x-y\|, \sup_{y\in Y}\inf_{x\in X}\|x-y\|\}$ with $\|\cdot\|$ being the ℓ_2 norm, and S_r is an $n-1$ dimensional sphere with radius r. The Remark 5 in (Szarek, 1982) suggests $d_{G,r}$ is also a metric.

To analyze the approximation error, we will use the covering number of $G_{n,\ell}$. Let N_ε be the ε-covering number of $G_{n,\ell}$ w.r.t. $d_{G,r}$, which is defined as the smallest number of ε-balls whose union contains $G_{n,\ell}$, that is,

$$N_\varepsilon = \arg\min_m \cup_{i=1}^m B_\varepsilon(U_i) \supseteq G_{n,\ell}, \tag{3}$$

where $B_\varepsilon(U_i) = \{V \in G_{n,\ell} \mid d_{G,r}(U_i,V) \le \varepsilon\}$ is an ε-ball centered at $U_i \in G_{n,\ell}$ and with radius ε; moreover, $B_\varepsilon(U_1),\dots,B_\varepsilon(U_{N_\varepsilon})$ is called an ε-covering of $G_{n,\ell}$. The following lemma is a scaled version of the Proposition 8 in (Szarek, 1982), originally proposed in (Szarek), which bounds the covering number.

Lemma 4.1. *There exist universal constants c, C such that*

$$\left(\frac{cr}{\varepsilon}\right)^{\ell(n-\ell)} \le N_\varepsilon \le \left(\frac{Cr}{\varepsilon}\right)^{\ell(n-\ell)}, \tag{4}$$

for any $\varepsilon \in (0, \sqrt{2}]$.

Our analysis also involves the use of a subspace to approximate a finite set. Inspired by the Kolmogorov n-width theory (Pinkus, 2012), we define the distance from a subspace $U \in G_{n,\ell}$ to a finite set $A \subseteq \mathbb{R}^n$ as

$$d_S(A,U) = \sup_{a\in A}\inf_{u\in U}||a-u||. \tag{5}$$

We remark that $\inf_{U\in G_{n,\ell}} d_S(A,U)$ is the Kolmogorov ℓ-width of A in $\mathbb{R}^n$, and its value is zero whenever the cardinality of A is no greater than ℓ (because one can always use elements of A as part of a basis to construct U). In addition, we develop the following pseudo-triangular inequality based on this distance. Its proof is in Appendix A.

Lemma 4.2. *For any $U, V \in G_{\ell,n}$ and finite $A \subseteq S_r$,*

$$d_S(A,U) \le d_S(A,V) + d_{G,r}(V,U). \tag{6}$$

The subspace we analyze will be random, and is actually the row space of a random matrix $\tilde{H}$ constructed as follows: let $H_1,\dots,H_k$ be the random hypotheses from which the k hypotheses in Framework 1 are sampled respectively, and $x_1,\dots,x_n$ be a given data set. Construct

$$\tilde{H} = \begin{bmatrix} H_1(x_1) & \dots & H_1(x_n) \\ \vdots & \ddots & \vdots \\ H_k(x_1) & \dots & H_k(x_n) \end{bmatrix} = \begin{bmatrix} \tilde{H}_{1:} \\ \vdots \\ \tilde{H}_{k:} \end{bmatrix}, \tag{7}$$

where $\tilde{H}_{i:} = [H_i(x_1),\dots,H_i(x_n)]$ is the i_{th} row and k is typically way smaller than n.

Our analysis will rely on a fixed dimension of the random subspace, and this only occurs with certain probability that can be characterized by the following property of random matrix, which is from (Vershynin, 2010) Theorem 5.39.

Theorem 4.3. *Let M be a k-by-n matrix whose rows are independent sub-gaussian isotropic random vectors in $\mathbb{R}^n$. Let $\sigma_{\min}$ be the smallest singular value of M. Then*

$$\Pr\{\sigma_{\min} < \sqrt{k} - c\sqrt{n} - t\} \le 2\exp(-Ct^2), \tag{8}$$

for any $t \ge 0$, where $c, C > 0$ are constants depending only on the maximum subgaussian norm of the rows.

Using the above theorem, we develop the following property of $\tilde{H}$. Its proof is in Appendix B.

Lemma 4.4. *For random matrix $\tilde{H}$ in (7), if $H_1,\dots,H_k$ are i.i.d. and each $\tilde{H}_{i:}$ follows a sub-Gaussian distribution and has an invertible expected outer product, then*

(i) $\tilde{H}_{1:},\dots,\tilde{H}_{k:}$ are i.i.d..

(ii) There exist constants a, b depending on the largest sub-Gaussian norm and expected outer product of $\tilde{H}_{i:}$, such that a sample of $\tilde{H}$ has linearly independent rows with probability at least $1 - 2\exp(-b(\sqrt{k} - a\sqrt{n})^2)$.

Our analysis also relies on the following assumption.

Assumption 4.5. There exists an ε-covering of $G_{n,\ell}$, denoted by $B_\varepsilon(U_1),\dots,B_\varepsilon(U_{N_\varepsilon})$ for some $U_1,\dots,U_{N_\varepsilon} \in G_{n,\ell}$, such that any ℓ-dimensional row span of $\tilde{H}$ is uniformly distributed in $B_\varepsilon(U_1),\dots,B_\varepsilon(U_{N_\varepsilon})$.

Finally, let f be a model returned by RHSS. We will analyze its error on a labeled set $(x_1,y_1),\dots,(x_n,y_n)$, defined as

$$er_n(f) = \frac{1}{n}\sum_{i=1}^n [f(x_i) - y_i]^2. \tag{9}$$

We will also analyze its error on the population, defined as

$$er(f) = \mathbb{E}[f(x) - y]^2, \tag{10}$$

where (x,y) denotes a random instance.

4.2. Theoretical Analysis of RHSS

Our main result is stated as follows.

Theorem 4.6. *Suppose $\tilde{H}$ in (7) satisfy the conditions in Lemma 4.4 and Assumption 4.5. Then, there exist constants $a, b, c > 0$ such that any model f returned by Framework 1 satisfies $er_n(f) \le \varepsilon$ with probability at least $1 - \delta_1 - \delta_2$ (over the random choice of hypotheses), where $\delta_1 = 2\exp(-b(\sqrt{k} - a\sqrt{n})^2)$ and $\delta_2 = \exp(-k(\frac{\sqrt{n\varepsilon}}{2cr})^{(n-1)})$.*

Proof. Our proof has four steps: (i) show the vectorized hypotheses are linearly independent with high probability; (ii) show $er_n(f)$ is bounded by some distance d_S; (iii) upper bound the introduced distance; (iv) specify the probability for the upper bound to hold. Details are elaborated below.

Step (i): Probability of Linear Independence

Consider an event that $\tilde{h}_1, \ldots, \tilde{h}_k$ are linearly independent. Let P_1 be the probability this event does not occur. Then, Lemma 4.4 suggests there exists constants a, b such that

$$P_1 \leq 2\exp(-b(\sqrt{k} - a\sqrt{n})^2). \tag{11}$$

The rest of the analysis will be based on the assumption that the event of linear independence occurs.

Step (ii): Bound $er_n(f)$ by $d_S(\tilde{Y}, \tilde{V}_i)$.

Let ℓ be a proper number (to be picked later) and evenly divide $\tilde{h}_i$'s into $m = k/\ell$ groups. By the assumption in Step (i), each group spans an ℓ-subspace. Let $\tilde{V}_1, \ldots, \tilde{V}_m \in G_{n,l}$ be the ℓ-subspaces spanned by the m groups, respectively.

Then, RHSS can be viewed as using one $\tilde{V}_i$ to approximate the target set $\tilde{Y} = \{\tilde{y}\}$, as it finds a model whose vectorized representation $\tilde{f} = [f(x_1), \ldots, f(x_n)]^T \in \tilde{V}_i$ has the smallest distance to $\tilde{y} \in \tilde{Y}$, i.e.,

$$||\tilde{f} - \tilde{y}|| = \sup_{\tilde{y}\in\tilde{Y}} \min_{\tilde{h}\in\tilde{V}_i} ||\tilde{h} - \tilde{y}|| = d_S(\tilde{Y}, \tilde{V}_i). \tag{12}$$

Moreover, it is easy to verify (by definition) that

$$er_n(f) = [d_S(\tilde{Y}, \tilde{V}_i)]^2/n. \tag{13}$$

Thus to bound $er_n(f)$, it suffices to bound $d_S(\tilde{Y}, \tilde{V}_i)$.

Step (iii): Bound $d_S(\tilde{Y}, \tilde{V}_i)$.

To bound $d_S(\tilde{Y}, \tilde{V}_i)$, we first apply the developed pseudo-triangular inequality. Let $V_* = \arg\min_{V\in G_{n,\ell}} d_S(\tilde{Y}, V)$ be a subspace that best approximates $\tilde{Y}$ and $r = ||\tilde{y}||$. Then, Lemma 4.2 and the remark of (5) suggest that

$$\begin{aligned} d_S(\tilde{Y}, \tilde{V}_i) &\leq d_S(\tilde{Y}, V_*) + d_{G,r}(V_*, \tilde{V}_i) \\ &= d_{G,r}(V_*, \tilde{V}_i). \end{aligned} \tag{14}$$

To bound $d_{G,r}(V_*, \tilde{V}_i)$, we apply results on Grassmannian. Recall N_ε is the covering number of $G_{n,\ell}$, and let $B_\varepsilon(U_1), \ldots, B_\varepsilon(U_{N_\varepsilon})$ be the ε-covering in assumption 4.5.

By definition, V_* must fall in one of the balls – without loss of generality, assume $V_* \in B_\varepsilon(U_1)$. Now, if $\tilde{V}_i$ also falls in $B_\varepsilon(U_1)$, by the triangular inequality for metric d_G,

$$d_{G,r}(\tilde{V}_i, V_*) \leq d_{G,r}(\tilde{V}_i, U_1) + d_{G,r}(U_1, V_*) \leq 2\varepsilon. \tag{15}$$

Plugging (14) (15) back to (13), we have

$$er_n(f) \leq (4\varepsilon^2)/n. \tag{16}$$

We are interested in quantity $\varepsilon' = (4\varepsilon^2)/n$, which implies

$$\varepsilon = \sqrt{\varepsilon' n}/2. \tag{17}$$

Step (iv): Specify the Probability for the Bound

It remains to specify the probability for (16). By the uniform assumption, the probability for one $\tilde{V}_i$ to fall outside $B_\varepsilon(U_1)$ is $1 - 1/N_\varepsilon$, and for all $\tilde{V}_1, \ldots, \tilde{V}_m$ to fall outside $B_\varepsilon(U_1)$ is $P_2 = (1 - 1/N_\varepsilon)^m$. By Lemma 4.1 and (17), we have

$$\begin{aligned} P_2 \leq \exp(-\frac{m}{N_\varepsilon}) &\leq \exp(-\frac{k}{\ell}\left(\frac{\varepsilon}{cr}\right)^{\ell(n-\ell)}) \\ &\leq \exp(-\frac{k}{\ell}(\frac{\sqrt{n\varepsilon'}}{2cr})^{\ell(n-\ell)}), \end{aligned} \tag{18}$$

for some constant c. Now we pick ℓ. Simple analysis shows the right side of (18) is minimum when $\ell = 1$. Thus

$$P_2 \leq \exp(-k(\frac{\sqrt{n\varepsilon'}}{2cr})^{(n-1)}). \tag{19}$$

Finally, combining all by a union bound and replacing ε' with ε proves the theorem. □

4.2.1. Implications of Theorem 4.6

In Theorem 4.6, the probability for RHSS to have guaranteed performance is determined by δ_1 and δ_2, where the former determines how likely the sampled hypotheses are linearly independent, and the latter determines how likely the output model performs well. In the following, we will focus on discussing the impact of k on both terms, since it is the major hyper-parameter of RHSS.

For δ_1, the impact of k is not monotonic based on $\delta_1 = 2\exp(-b(\sqrt{k} - a\sqrt{n})^2)$. When $\sqrt{k} < a\sqrt{n}$, increasing k will increase δ_1 and generate a weaker guarantee; otherwise, increasing k will decrease δ_1. This implies if one wants all sampled hypotheses to be linearly independent, one could sample either very few or a lot. (Fortunately, in experiment we see a few is sufficient for good performance.)

For δ_2, it monotonically decreases as k increases based on $\delta_2 = \exp(-k(\frac{\sqrt{n}\varepsilon}{2c||\tilde{y}||})^{n-1})$, giving a higher probabilistic guarantee. When δ_2 is held constant, we see that increasing k allows one to pick a smaller ε. This implies sampling more hypotheses allows one to have a smaller error guarantee.

Since both δ_1 and δ_2 have the form $\exp(-c_n k)$ for some c_n, together they provide a strong guarantee that $er_n(f) > \varepsilon$ with probability at most $\exp(-ck)$, which drops exponentially fast as k increases.

It is worth mentioning that, δ_2 often dominates δ_1 in practice, especially for large n and small k. Moreover, it is easy to show $\delta_1 = 0$ if more ideal sampling distributions can be assumed, such as those remarked in following corollary.

Corollary 4.7. *If $\tilde{H}_1, \ldots, \tilde{H}_k$, defined in (7), are independently and uniformly distributed, then there exists constant $c > 0$ as in Lemma 4.1 such that any model f output from Framework 1 satisfies $er_n(f) \leq \varepsilon$ with probability at least $1 - \delta$, where $\delta = \exp(-k(\frac{\sqrt{n}\varepsilon}{2cr})^{(n-1)})$.*

4.2.2. Extension to Generalization Error

In this section, we extend the above result from a data set to the population using Rademacher complexity. Recall such complexity[1] of a hypothesis class H w.r.t. random inputs $x_1, \ldots, x_n$ is defined as

$$\mathcal{R}_n(H) = \mathbb{E}_x \mathbb{E}_t \sup_{h \in H} \frac{1}{n} \left| \sum\nolimits_{i=1}^{n} t_i h(x_i) \right|, \qquad (20)$$

where $t_1, \ldots, t_n$ are independent random variables uniformly picked from $\{-1, +1\}$. Let k, M be two constants and define the following hypothesis class

$$F = \left\{ \sum\nolimits_{i=1}^{k} \alpha_i h_i \mid h_i \in H, \alpha_i \in \mathbb{R}. |\alpha_i| \leq M \right\}, \quad (21)$$

We develop the following relation between $\mathcal{R}_n(F)$ and $\mathcal{R}_n(H)$. Its proof is in Appendix C.

Theorem 4.8. *For any finite $K, M, n > 0$,*

$$\mathcal{R}_n(F) = Mk \cdot \mathcal{R}_n(H). \qquad (22)$$

Combining this with standard generalization arguments e.g., Theorem 3.3 in (Mohri et al., 2018) and Theorem 4.8, we obtain the following generalization error bound for RHSS.

Theorem 4.9. *In Framework 1, suppose all hypotheses are bounded by a constant $T > 0$ and the n instances are sampled i.i.d.. Then there exists a constant M depending on λ such that, with probability at least $1 - \delta$, any output model f satisfies $er(f) \leq er_n(f) + 8TMk\,\mathcal{R}_n(H) + T^2\sqrt{\frac{8 \log \frac{1}{\delta}}{n}}$.*

To better interpret the bound, consider the scenario in Corollary 4.7 which implies $er_n(f) \leq \frac{4c^2r^2}{n}(\frac{1}{k}\log\frac{1}{\delta})^{\frac{1}{n-1}}$. Plugging this into Theorem 4.9, the error bound becomes

$$\frac{4c^2r^2}{n}(\frac{1}{k}\log\frac{1}{\delta})^{\frac{1}{n-1}} + 8TMk\,\mathcal{R}_n(H) + O(\frac{1}{\sqrt{n}}). \quad (23)$$

We see k balances the first two terms, as increasing it will decrease the 1st term (approximation error) but increase the 2nd term (complexity of F). This makes perfect sense.

Interestingly, the balance suggests an optimal k that could minimize the error bound. Let $J(k)$ be the sum of the first two terms in (23). Solving $\frac{\partial J(k)}{\partial k} = 0$ gives this optimal $k = \left(\frac{c^2 r^2 (\log \frac{1}{\delta})^{\frac{2-n}{n-1}}}{2TM\mathcal{R}_n(H)n(n-1)} \right)^{\frac{n-1}{n}}$. Plugging this back to (23) and assuming n is sufficiently large, it is easy to show the 1st term is in $O(\frac{1}{n})$ and the 2nd term is in $O(\frac{1}{n^2})$.

Note that both terms are much smaller than the 3rd term, which is in $O(\frac{1}{\sqrt{n}})$ and induced solely from generalization. Then, Theorem 4.9 suggests the error induced from approximation is negligible compared to the error induced from generalization. This presents a theoretical justification on the effectiveness of RHSS, and is also consistent with our experimental results.

[1]Here we adopt the version with absolute value.

5. Applications of RHSS

Applying RHSS is straightforward: sample some hypotheses, get their predictions and learn their best combination. In practice, the design of hypothesis sampling depends on the model. We give three examples in this section.

5.1. Kernel Ridge Regression (KRR)

Let ϕ be an (implicit) feature mapping and H_ϕ be the set of all linear hypotheses in the mapped space. Standard KRR learns a model in H_ϕ in $O(n^3)$ time.

We propose RHSS based KRR (RHSS-KRR), which randomly samples hypotheses from

$$H_\phi = \left\{ \sum_{i=1}^{n} \beta_i \phi(x_i) \mid \beta_i \in \mathbb{R} \right\}. \qquad (24)$$

Specifically, a hypothesis is sampled by independently sampling its β_i's from a proper distribution such as Gaussian.

Suppose the j_{th} hypothesis $h_j = \sum_{i=1}^{n} \beta_i^j \phi(x_i)$ is sampled. Its prediction vector can be evaluated as

$$\tilde{h}_j = [h_j(\phi(x_1)), \ldots, h_j(\phi(x_n))]^T = K \cdot \vec{\beta}_j, \qquad (25)$$

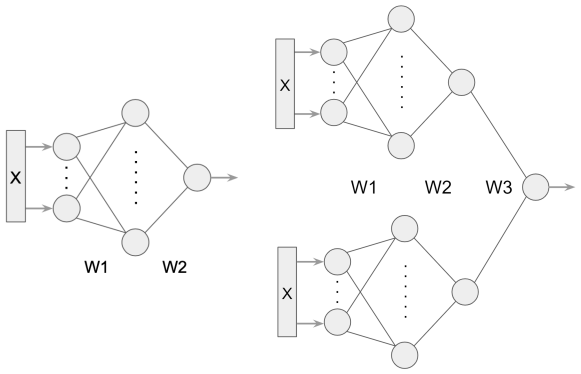
where $\vec{\beta}_j = [\beta_1^j, \ldots, \beta_n^j]^T$ and K is the Gram matrix.

In terms of time complexity, RHSS-KRR takes $O(nk^2)$ to learn the output model, which is more efficient than the standard KRR that takes $O(n^3)$, and equally efficient as random Fourier Feature that takes $O(nk^2)$ with k random features. The process of sampling and evaluating $\tilde{h}_1, \ldots, \tilde{h}_k$ takes $O(n^2k)$, which is less efficient than RFF which takes $O(nk)$ but still more efficient than standard KRR.

5.2. Multi-Layer Perceptron (MLP)

Let H_τ be the set of MLPs with the same architecture τ which specifies the number of hidden layers, number of neurons per layer and activation functions. Standard MLP learning is done through back-propagation.

We propose RHSS based MLP (RHSS-MLP), which randomly samples a network in H_τ by independently sampling all its weights from a proper distribution such as Gaussian.



(a) MLP/RVFL (b) RHSS-MLP (k = 2)

Figure 1: Architectures of MLP, RVFL and RHSS-RVFL

RHSS-MLP takes $O(nk^2)$ to learn the output model, while RVFL takes $O(nm_\tau^2)$ with m_τ being the number of neurons in the last hidden layer. Interestingly, we can view RHSS-MLP as applying the RVFL principle on a network with special architecture, as illustrated in Figure 1.

Figure 1(a) shows an MLP with a single hidden layer. Let W1 be the set of weights between the input and hidden layers, and W2 be the set of weights between the hidden and output layers. Back-propagation optimizes W1 and W2, while RVFL randomly sets W1 and only optimizes W2.

Figure 1(b) shows the corresponding network of RHSS-MLP. It has k blocks of MLP's, as k sampled hypotheses, and combines them at the end. Let W3 be the set of weights between the output and the last hidden layer. RHSS-MLP randomly sets W1 and W2, and only optimizes W3.

5.3. Decision Tree

Let H_τ be the set of decision trees that can be generated based on a feature set τ. Standard tree learning algorithms find optimal features to split tree nodes.

We propose RHSS based tree (RHSS-Tree), which randomly samples trees in H_τ by applying the extra tree generation technique (Geurts et al., 2006) on bootstrap samples. More specifically, we sample a tree by randomly selecting features to split its nodes. Bootstrapping is necessary in this application, since different trees generated by an extra tree will have the same predictions on the training set, making the optimization in (1) useless (all α_i's are identical).

Standard tree learning and extra tree learning have the same time complexity, although the latter is faster since it avoids the time of finding optimal features for node split. RHSS-Tree has the same time complexity as it applies extra tree.

6. Experiments

We compare the performance of the proposed RHSS-KRR, RHSS-MLP and RHSS-Tree with their existing randomized counterparts on three public real-world data sets, namely, Crime and Community, Adult and COMPAS. On each data set, we use the first half of the instances for training and the other half for testing. To account for the randomness in randomized learners, we run each learner for 20 times and report its average performance and standard deviation. We focus on reporting accuracy of the trained models (measured by their rooted mean square errors) versus hyper-parameter k. In all figures, the k values are log-scaled.

In the following, we first present three sets of experiments, each comparing one RHSS based learner with its existing randomized counterpart. Then, we perform a set of sensitivity analysis including the impact of RP on RHSS. The codes of all experiments are available at https://github.com/yxc827/RHSS.

6.1. Comparisons with Existing Randomized Learners

We design three sets of comparisons, each based on one proposed application of RHSS in Section 5.

The first set compares RHSS-KRR with standard KRR and Random Fourier Feature based KRR (RFF-KRR). We use RBF kernel with width optimized to 1e-3. For RHSS-KRR, the hypothesis coefficients are sampled independently from $N(0, 1)$. We chose this distribution simply because it is common, but RHSS actually seems fairly robust across different sampling distributions. (See results in the next section.) Both KRR and RFF-KRR apply ridge regression, and the regularization coefficient is optimized to 0.1 on Crime and 0.001 on the other two data sets. In this experiment, k is the number of sampled hypotheses for RHSS-KRR and the number of random features for RFF-KRR. Results are shown in Figure 2(a) 2(b) 2(c). We see RHSS-KRR converges slightly faster than RFF-KRR, and converges to KRR at around $k = 100$.

The second set compares RHSS-MLP with standard MLP and RVFL. Since RVFL is mainly designed for the single-hidden-layer architecture, we apply this architecture with 20 hidden neurons and ReLU activation function. For MLP and RVFL, the regularization coefficient is optimized to 10. For RHSS-MLP and RVFL, all non-optimized parameters are independently sampled from $N(0, 1)$. Results are shown in Figure 2(d) 2(e) 2(f). We see RHSS-MLP converges to MLP when around $k = 100$ and offers a better approximation of MLP than RVFL at that point on two data sets.

The third set compares RHSS-Tree with decision tree, extra tree and random forest. In this experiment, k is the number of sampled hypotheses for RHSS-Tree and the number of trees for extra tree and random forest. For RHSS-Tree,

bootstrap sample size is set to 80% of the original training set. The configurations of all other methods are set as default. Results are shown in Figure 2(g) 2(h) 2(i). We see RHSS-Tree and extra tree both outperform decision tree as k increases to a small number, and can well approximate the powerful random forest on two data sets. On COMPAS, RHSS-Tree slightly outperforms random forest.

Figures 2(j) 2(k) 2(l) show the training time (in terms of seconds) of all methods on Crime. We see those popular randomized learners RFF-KRR, RVFL and extra tree are indeed extremely efficient, followed by RHSS based learners. They are all a lot faster than standard learners.

Overall, we see RHSS provides an efficient and effective randomized learning framework for different models.

6.2. Experiment with RP and Sensitivity Analysis

In this section, we evaluate the performance of random projection (RP) for KRR on the Crime and Community data set. We experiment two methods: (i) RP-KRR first applies RP and then applies KRR; (ii) RP-RHSS-KRR: first applies RP and then applies RHSS-KRR. In both methods, k is the projected dimension of RP, and we fix the number of sampled hypotheses to 100 for RHSS-KRR. For RP, all projection entries are independently sampled from $N(0, 0.01)$. All other configurations are the same as before. Results are shown in Figure 2(m). We see RP-KRR also offers a good approximation to KRR as k increases. However, it does not speed up learning as much as RHSS-KRR, as shown in Figure 2(j). We also see RP-RHSS-KRR is not as efficient as other methods, leaving how to effectively combine randomized dimensionality reduction method and randomized learning method an open question.

Next, we evaluate the performance of RHSS-KRR on Crime with different sampling distributions. Keeping all other configurations, we experiment four distributions: (i) Gaussian $N(0, 1)$; (iii) uniform in $[-\sqrt{3}, \sqrt{3}]$, (iii) Laplace with zero mean and unit scale, and (iv) symmetric Bernoulli with $p = 0.5$. Results are shown in Figure 2(o). We see RHSS is fairly robust across the different sampling distributions.

Finally, we evaluate the performance of RVFL and RHSS-MLP when the network architecture varies. Specifically, we increase the number of hidden neurons, with k fixed to 100 for RHSS-MLP, and report results in Figure 2(n). We see RVFL improves as more hidden neurons are added, which is a known result. The impact of hidden neurons on RHSS-KRR is limited, however. Our general observation is that RHSS based methods are mainly affected by k.

7. Conclusion

This paper presents a model-agnostic randomized learning framework based on Random Hypothesis Subspace Sampling (RHSS), which ties the popular model-specific randomized learners and provides a more unified base for the future developments of randomized machine learning.

The proposed RHSS framework is simple and easy to apply, and cast learning for any hypothesis class as a linear least square problem solvable in $O(nk^2)$ time with n training instances and k sampled hypotheses. On the theory side, we derive error bounds for RHSS and show the approximation error is negligible compared to the generalization error, which theoretically justifies its effectiveness. On the practical side, we demonstrate the applications of RHSS on kernel, neural network and tree based models. In experiments, we show the proposed RHSS-based learners converge efficiently to standard learners and often outperform their model-specific randomized counterparts, including random Fourier feature, RVFL and extra tree, on real-world data sets. Our results suggest a strong practical value of the proposed unifying framework.

Acknowledgements

We thank all anonymous reviewers for their insightful feedback for enhancing the quality of the paper. This paper is based upon work supported by the US National Science Foundation under Grant No. 2101936.

References

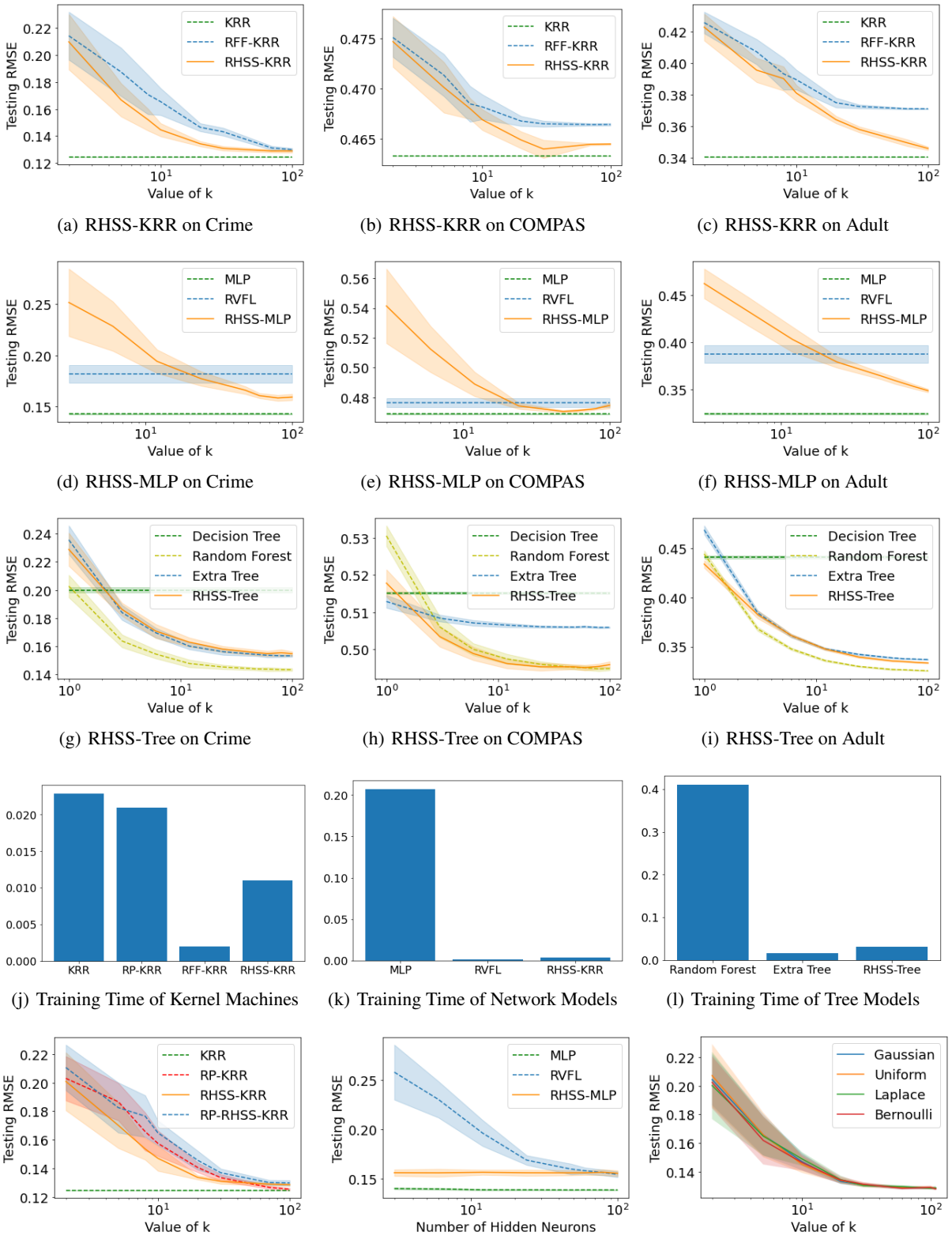
Arriaga, R. I. and Vempala, S. An algorithmic theory of learning: Robust concepts and random projection. *Machine learning*, 63(2):161–182, 2006.

Avron, H., Kapralov, M., Musco, C., Musco, C., Velingker, A., and Zandieh, A. Random fourier features for kernel ridge regression: Approximation bounds and statistical guarantees. In *International Conference on Machine Learning*, pp. 253–262. PMLR, 2017.

Desir, C., Petitjean, C., Heutte, L., Salaun, M., and Thiberville, L. Classification of endomicroscopic images of the lung based on random subwindows and extra-trees. *IEEE Transactions on Biomedical Engineering*, 59(9): 2677–2683, 2012. doi: 10.1109/TBME.2012.2204747.

Durrant, R. and Kabán, A. Sharp generalization error bounds for randomly-projected classifiers. In *International Conference on Machine Learning*, pp. 693–701, 2013.

Fradkin, D. and Madigan, D. Experiments with random projections for machine learning. In *Proceedings of the ninth*



(a) RHSS-KRR on Crime (b) RHSS-KRR on COMPAS (c) RHSS-KRR on Adult

(d) RHSS-MLP on Crime (e) RHSS-MLP on COMPAS (f) RHSS-MLP on Adult

(g) RHSS-Tree on Crime (h) RHSS-Tree on COMPAS (i) RHSS-Tree on Adult

(j) Training Time of Kernel Machines (k) Training Time of Network Models (l) Training Time of Tree Models

(m) Performance of Random Projection (n) Performance of RVFL (o) Sampling Distributions of RHSS-KRR

Figure 2: Performance of RHSS based Learning Algorithms on Three Data Sets

ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 517–522, 2003.

Gallicchio, C. and Scardapane, S. Deep randomized neural networks. *Recent Trends in Learning From Data*, pp. 43–68, 2020.

Garg, S., Galstyan, A., Ver Steeg, G., and Cecchi, G. A. Nearly-unsupervised hashcode representations for biomedical relation extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4026–4036, 2019a.

Garg, S., Galstyan, A., Ver Steeg, G., Rish, I., Cecchi, G., and Gao, S. Kernelized hashcode representations for relation extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 6431–6440, 2019b.

Geurts, P., Ernst, D., and Wehenkel, L. Extremely randomized trees. *Machine learning*, 63(1):3–42, 2006.

Igelnik, B. and Pao, Y.-H. Stochastic choice of basis functions in adaptive function approximation and the functional-link net. *IEEE transactions on Neural Networks*, 6(6):1320–1329, 1995.

Li, P. Linearized gmm kernels and normalized random fourier features. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 315–324, 2017.

Li, Z., Ton, J.-F., Oglic, D., and Sejdinovic, D. Towards a unified analysis of random fourier features. In *International Conference on Machine Learning*, pp. 3905–3914. PMLR, 2019.

Maier, O., Wilms, M., von der Gablentz, J., Krämer, U. M., Münte, T. F., and Handels, H. Extra tree forests for sub-acute ischemic stroke lesion segmentation in mr sequences. *Journal of neuroscience methods*, 240:89–100, 2015.

Maillard, O. and Munos, R. Linear regression with random projections. *Journal of Machine Learning Research*, 13: 2735–2772, 2012.

Mohri, M., Rostamizadeh, A., and Talwalkar, A. *Foundations of machine learning*. MIT press, 2018.

Needell, D., Nelson, A. A., Saab, R., and Salanevich, P. Random vector functional link networks for function approximation on manifolds. *arXiv preprint arXiv:2007.15776*, 2020.

Pao, Y.-H., Park, G.-H., and Sobajic, D. J. Learning and generalization characteristics of the random vector functional-link net. *Neurocomputing*, 6(2):163–180, 1994.

Pinkus, A. *N-widths in Approximation Theory*, volume 7. Springer Science & Business Media, 2012.

Rahimi, A. and Recht, B. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pp. 1177–1184, 2008.

Rudi, A. and Rosasco, L. Generalization properties of learning with random features. *Advances in neural information processing systems*, 30:3215–3225, 2017.

Szabó, Z. Optimal rates for random fourier features. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, pp. 1144–1152. 2015.

Szarek, S. Metric entropy of homogeneous spaces, quantum probability,(gdensk 1977).

Szarek, S. J. Nets of grassmann manifold and orthogonal group. In *Proceedings of research workshop on Banach space theory (Iowa City, Iowa, 1981)*, volume 169, pp. 185, 1982.

Vempala, S. S. *The random projection method*, volume 65. American Mathematical Soc., 2005.

Vershynin, R. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

Yang, T., Li, Y.-F., Mahdavi, M., Jin, R., and Zhou, Z.-H. Nyström method vs random fourier features: A theoretical and empirical comparison. *Advances in neural information processing systems*, 25:476–484, 2012.

Zhang, L. and Suganthan, P. N. A comprehensive evaluation of random vector functional link networks. *Information sciences*, 367:1094–1105, 2016.

A. Proof of Lemma 4.2

Lemma 4.2. For any $U, V \in G_{\ell,n}$ and finite $A \subseteq S_r$,

$$d_S(A, U) \leq d_S(A, V) + d_{G,r}(V, U). \tag{26}$$

Proof. Let $a \in A$, $u \in U$ and $v \in V$. There is

$$||a - u|| \leq ||a - v|| + ||v - u||. \tag{27}$$

Taking infimum of $u \in U$ on both sides of (27) gives

$$\inf_{u\in U} ||a - u|| \leq ||a - v|| + \inf_{u\in U} ||v - u||. \tag{28}$$

Let $v_a = \arg\inf_{v\in V} ||a - v||$. Taking infimum of $v \in V$ on both sides of (28) gives

$$\begin{aligned} \inf_{u\in U} ||a - u|| &\leq \inf_{v\in V}(||a - v|| + \inf_{u\in U} ||v - u||) \\ &\leq ||a - v_a|| + \inf_{u\in U} ||v_a - u|| \\ &= \inf_{v\in V} ||a - v|| + \inf_{u\in U} ||v_a - u||. \end{aligned} \tag{29}$$

Since $A \subseteq S_r$, we have $||a|| = r$ and thus $||v_a|| = r$. Then

$$\inf_{u\in U} ||v_a - u|| \leq \inf_{u\in U\cap S_r} ||v_a - u|| \leq \sup_{v\in V\cap S_r} \inf_{u\in U\cap S_r} ||v - u|| \leq D_{G,r}(V, U). \tag{30}$$

Plugging (30) back to (29) and taking supremum of $a \in A$ on both sides of the inequality proves the lemma. □

B. Proof of Lemma 4.4

Lemma 4.4. For random matrix $\tilde{H}$ in (7), if $H_1, \ldots, H_k$ are i.i.d. and each $\tilde{H}_{i:}$ follows a sub-Gaussian distribution and has an invertible expected outer product, then

(i) $\tilde{H}_{1:}, \ldots, \tilde{H}_{k:}$ are i.i.d..

(ii) There exist constants a, b depending on the largest sub-Gaussian norm and expected outer product of $\tilde{H}_{i:}$, such that a sample of $\tilde{H}$ has linearly independent rows with probability at least $1 - 2\exp(-b(\sqrt{k} - a\sqrt{n})^2)$.

Proof. We first prove (i). To show any two rows have identical distribution is trivial. To show they are independent, let $\rho(x; E) = \{h \in H; h(x) \in E\}$. Then, for any two H_i, H_j, fixed inputs x, z and sets E_1, E_2, we have

$$\begin{aligned} \Pr\{H_i(x) \in E_1, H_j(z) \in E_2\} &= \Pr\{H_i \in \rho(x; E_1), H_j \in \rho(z; E_2)\} \\ &= \Pr\{H_i \in \rho(x; E_1)\} \cdot \Pr\{H_j \in \rho(z; E_2)\} \\ &= \Pr\{H_i(x) \in E_1\} \cdot \Pr\{H_j(z) \in E_2\}, \end{aligned} \tag{31}$$

where the third line is by the independence assumption. The argument can be readily generalized to all inputs which implies $[H_i(x_1), \ldots, H_i(x_n)]$ and $[H_j(x_1), \ldots, H_j(x_n)]$ are independent vectors. This proves claim (i).

Now we prove (ii). Let $\Sigma = E[\tilde{H}_{i:}^T \tilde{H}_{i:}]$. Let $\tilde{H}'$ be an k-by-n matrix whose i_{th} row is

$$\tilde{H}'_{i:} = \Sigma^{-1/2} \tilde{H}_{i:}. \tag{32}$$

It is easy to show $\tilde{H}'_{i:}$ has i.i.d. sub-Gaussian isotropic rows.

Let $\sigma'_{\min}$ be the least singular value of $\tilde{H}'$. By Theorem 4.3,

$$\Pr\{\sigma'_{\min} < \sqrt{k} - a'\sqrt{n} - t\} \leq 2\exp(-bt^2), \tag{33}$$

for some constants a', b.

Now, pick an arbitrarily small $\varepsilon > 0$ and set $t = \sqrt{k} - (a' + \varepsilon)\sqrt{n}$, we have $\Pr\{\sigma'_{\min} = 0\} \leq \Pr\{\sigma'_{\min} < \varepsilon\sqrt{n}\} \leq 2\exp(-b[k + (a')^2 n - 2a'\sqrt{nk}])$. Further, let $\sigma_{\min}$ be the least singular value of $\tilde{H}$. Then $\Pr\{\sigma_{\min} = 0\} \leq \Pr\{\sigma'_{\min} = 0\}$, since a sample of $\tilde{H}$ with linearly dependent rows implies the existence of a sample of $\tilde{H}'$ with linearly dependent rows constructed through (32). Setting $a = (a')^2$ and putting all together prove claim (ii). □

C. Proof of Theorem 4.8

Theorem 4.8. For any finite $K, M, n > 0$,

$$\mathcal{R}_n(F) = MK \cdot \mathcal{R}_n(H). \tag{34}$$

Proof. For compact presentation, we will omit subscripts in the expectation in $\mathcal{R}_n$, and use α and h to denote the set of $\alpha_1, \ldots, \alpha_K$ and set of $h_1, \ldots, h_K$ respectively.

We first prove $\mathcal{R}_n(F) \leq MK\mathcal{R}_n(H)$. This is true because

$$\begin{aligned}
\mathcal{R}_n(F) &= \frac{1}{n}\mathbb{E}\sup_{\alpha,h}\left|\sum\nolimits_{i=1}^{n} t_i \cdot \left(\sum\nolimits_{j=1}^{K} \alpha_j h_j(x_i)\right)\right| \\
&= \frac{1}{n}\mathbb{E}\sup_{\alpha,h}\left|\sum\nolimits_{j=1}^{K} \alpha_j \cdot \left(\sum\nolimits_{i=1}^{n} t_i h_j(x_i)\right)\right| \\
&\leq \frac{1}{n}\mathbb{E}\sup_{\alpha,h}\sum\nolimits_{j=1}^{K} |\alpha_j| \cdot \left|\sum\nolimits_{i=1}^{n} t_i h_j(x_i)\right| \\
&= \frac{1}{n}\mathbb{E}\sum\nolimits_{j=1}^{K} M \cdot \sup_{h_j}\left|\sum\nolimits_{i=1}^{n} t_i h_j(x_i)\right| \\
&= \sum\nolimits_{j=1}^{K} M \cdot \frac{1}{n}\mathbb{E}\sup_{h_j}\left|\sum\nolimits_{i=1}^{n} t_i h_j(x_i)\right| \\
&= MK \cdot \mathcal{R}_n(H),
\end{aligned} \tag{35}$$

where the third line is by the triangular inequality (for any fixed α, h), the fourth line is by the definition of supremum so that the sum and product of non-negative variables are maximized when these variables are maximized; the fifth line is by the linearity of expectation.

Next we prove $\mathcal{R}_n(F) \geq MK\mathcal{R}_n(H)$. This is true because

$$\begin{aligned}
\mathcal{R}_n(F) &= \frac{1}{n}\mathbb{E}\sup_{\alpha,h}\left|\sum\nolimits_{j=1}^{K} \alpha_j \cdot \left(\sum\nolimits_{i=1}^{n} t_i h_j(x_i)\right)\right| \\
&\geq \frac{1}{n}\mathbb{E}\sup_{h}\left|\sum\nolimits_{j=1}^{K} M \cdot \left(\sum\nolimits_{i=1}^{n} t_i h_j(x_i)\right)\right| \\
&\geq \frac{1}{n}\mathbb{E}\sup_{h_j=h'}\left|\sum\nolimits_{j=1}^{K} M \cdot \left(\sum\nolimits_{i=1}^{n} t_i h'(x_i)\right)\right| \\
&= MK \cdot \frac{1}{n}\mathbb{E}\sup_{h'}\left|\sum\nolimits_{i=1}^{n} t_i h'(x_i)\right| \\
&= MK \cdot \mathcal{R}_n(H),
\end{aligned} \tag{36}$$

where the third line is obtained by setting $\alpha_1, \ldots, \alpha_K$ to M; the fourth line is obtained by adding a constraint $h_1 = \ldots = h_K$ when taking the supremum. Combining (35) and (36) proves the theorem. □