Random Features for the Neural Tangent Kernel

Insu Han [*] Haim Avron [†] Neta Shoham [‡] Chaewon Kim [§] Jinwoo Shin [¶]

April 6, 2021

Abstract

The Neural Tangent Kernel (NTK) has discovered connections between deep neural networks and kernel methods with insights of optimization and generalization. Motivated by this, recent works report that NTK can achieve better performances compared to training neural networks on small-scale datasets. However, results under large-scale settings are hardly studied due to the computational limitation of kernel methods. In this work, we propose an efficient feature map construction of the NTK of fully-connected ReLU network which enables us to apply it to large-scale datasets. We combine random features of the arc-cosine kernels with a sketching-based algorithm which can run in linear with respect to both the number of data points and input dimension. We show that dimension of the resulting features is much smaller than other baseline feature map constructions to achieve comparable error bounds both in theory and practice. We additionally utilize the leverage score based sampling for improved bounds of arc-cosine random features and prove a spectral approximation guarantee of the proposed feature map to the NTK matrix of two-layer neural network. We benchmark a variety of machine learning tasks to demonstrate the superiority of the proposed scheme. In particular, our algorithm can run tens of magnitude faster than the exact kernel methods for large-scale settings without performance loss.

1 Introduction

Recent literature have shown that trained overparameterized Deep Neural Networks (DNNs), i.e., neural networks with substantially more parameters than training data points, generalize surprisingly well. In an effort to understand this phenomena, recently researchers have studied the infinite width limit of DNNs (i.e., the number of neurons in each hidden layer goes to infinity) and has shown that in that limit, deep learning is equivalent to kernel regression where the kernel is the so-called Neural Tangent Kernel (NTK) of the network (Arora et al., 2019b; Chizat et al., 2019; Jacot et al., 2018; Lee et al., 2020). This connection has been used to shed light on the ability of DNNs to generalize (Cao & Gu, 2019; Neyshabur et al., 2019) and the ability to optimize (train) their parameters efficiently (Allen-Zhu et al., 2019; Arora et al., 2019a; Du et al., 2019).

Beyond the aforementioned theoretical purpose, several papers have explored the algorithmic use of the NTK. Arora et al. (2019c) and Geifman et al. (2020) showed that NTK based kernel models can sometimes perform better than trained neural networks. The NTK has also been used in experimental design for neural networks (Shoham & Avron, 2020) and predicting training time (Zancato et al., 2020).

Although the NTK of a given network can sometimes be expressed as closed-form formulas (Arora et al., 2019b; Novak et al., 2020), actually using it to learn kernel models encounters the computational bottlenecks

[*] School of Electrical Engineering, Korea Advanced Institute of Science and Technology. Email: insu.han@kaist.ac.kr

[†] Department of Applied Mathematics, Tel Aviv University. Email: haimav@tauex.tau.ac.il

[‡] Department of Applied Mathematics, Tel Aviv University. Email: shohamne@gmail.com

[§] Graduate School of AI, Korea Advanced Institute of Science and Technology. Email: chaewonk@kaist.ac.kr

[¶] Graduate School of AI, Korea Advanced Institute of Science and Technology. Email: jinwoos@kaist.ac.kr

of kernel learning, e.g., $\mathcal{O}(n^3)$ time and $\mathcal{O}(n^2)$ space complexity for kernel ridge regression with n data points. With the NTK, the situation is much worse since the cost required to compute the kernel matrix can be huge (Novak et al., 2020). This makes exact kernel learning with NTK infeasible under large-scale setups.

There is a rich literature on using kernel approximations in order to enable large-scale learning. One of the most popular approaches is the *random features* approach, originally due to Rahimi & Recht (2009). Following the seminal work, many random feature constructions have been suggested for a variety of kernels, e.g., arc-cosine kernels (Cho & Saul, 2009), polynomial kernels (Pham & Pagh, 2013; Pennington et al., 2015), general dot product kernels (Han et al., 2020), just to name a few. These low-dimensional features enable us to apply them to fast linear methods, saving time and space complexity drastically. Furthermore, their performances are similar or sometimes better than the exact kernel methods due to implicit regularization effect (Rahimi & Recht, 2009; Rudi & Rosasco, 2016; Jacot et al., 2020).

In this paper, we propose an efficient random features construction for the NTK of fully-connected neural networks with ReLU activations. Our starting point is the explicit feature map for the NTK suggested by Bietti & Mairal (2019). That feature map uses known explicit feature maps for the arc-cosine kernel. By replacing the explicit feature map of the arc-cosine kernel with a random feature map for the same kernel (Cho & Saul, 2009), we obtain a random feature map for the NTK. However, the size of that feature map for that construction can be even larger than the number of input data points n. The underlying reason is from the tensor products between features generated in consecutive layers. To avoid the issue, we utilize an efficient sketching algorithm known as TensorSketch transform (Pham & Pagh, 2013; Ahle et al., 2020; Woodruff & Zandieh, 2020) which can effectively approximate the tensor products of vectors while preserving their inner products. We provide a rigorous error analysis of the proposed scheme. The resulting random features have smaller dimension than the previous NTK feature map constructions.

Furthermore, in order to approximate the NTK with less features, we improve the underlying existing random feature map of the arc-cosine kernel. Our construction is based on a modified leverage score based sampling. Recent literature has shown that random features that use leverage score sampling entertain better convergence bounds (Avron et al., 2017b; Lee et al., 2020). However, computing the exact leverage scores requires the inversion of a n-by-n matrix which is equivalent to the cost for solving the kernel method exactly. Luckily, Avron et al. (2017b); Lee et al. (2020) showed that sampling from the upper bound of leverage score is enough to provide tight error bounds. Motivated by these results, we propose simple and closed-form upper bounds of leverage scores regarding to arc-cosine kernels. For further efficiency, we make use of Gibbs sampling to generate random features from the proposed modified distribution.

To theoretically justify our construction, we provide a spectral approximation guarantee for the proposed random features for two-layer neural network. Recent literature has advocated the use of such spectral bounds as a more general metric to measure kernel approximation quality as it pertains to many downstream tasks (Avron et al., 2017b).

Finally, we empirically benchmark the proposed random feature methods under machine learning tasks including classification/regression under UCI datasets, and active learning under MNIST datasets. We demonstrate that our random features method can perform similar to or better than the kernel method with NTK. We further show that the random features approach can run up to 17 times much faster with tested large-scale datasets, without loss on performance.

Related works. Many literature have studied a variety of NTK properties including optimization (Allen-Zhu et al., 2019; Du et al., 2019), generalization (Cao & Gu, 2019), loss surface (Mei et al., 2018) and so on. Recent works have more focused on NTK kernel itself. Geifman et al. (2020); Chen & Xu (2021) have discovered that NTK are similar to the Laplace kernel in term of spectral information when data points are in hypersphere. (Fan & Wang, 2020) studied eigenvalue distributions of NNGP and NTK and showed that they converge to deterministic distribution. However, to the best of our knowledge, not many works focus on approximating the NTK. Arora et al. (2019b) studied that gradient of randomly initialized network with finite widths can approximate the NTK in theory. However, they report that practical performances of random gradients are worse than that of the exact NTK by a large margin. Another line of work on NTK approximation is an explicit feature map construction via tensor product proposed by Bietti & Mairal (2019). These explicit features can have infinite dimension in general. Hence, it is impossible to use their features in practice. Even though one can use a finite-dimensional feature map, the computational gain of random features can be lost due to expensive tensor product operations.

2 Preliminaries

Notations. We use $[n] := \{1, \dots, n\}$. We denote $\otimes$ by the tensor (a.k.a. Kronecker) product and $\odot$ by the element-wise (a.k.a. Hadamard) product of two matrices. For square matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, we write $\boldsymbol{A} \preceq \boldsymbol{B}$ if $\boldsymbol{B} - \boldsymbol{A}$ is positive semi-definite. We write $[\boldsymbol{A}]_{i,j}$ as an entry of $\boldsymbol{A}$ in i-th row and j-th column. Similarly, $[\boldsymbol{v}]_i$ is used for an i-th entry of vector $\boldsymbol{v}$. We also denote $\mathrm{ReLU}(x) = \max(x, 0)$ and consider this element-wise operation when the input is a matrix. Given a positive semidefinite matrix $\boldsymbol{K}$ and $\lambda > 0$, the statistical dimension of $\boldsymbol{K}$ with λ is defined as $s_\lambda(\boldsymbol{K}) := \mathtt{tr}(\boldsymbol{K}(\boldsymbol{K} + \lambda \boldsymbol{I})^{-1})$.

2.1 NTK of Fully-connected Deep Neural Networks

Given an input $\boldsymbol{x} \in \mathbb{R}^d$, consider a fully-connected ReLU network with input dimension d, hidden layer dimensions $d_1, \dots, d_L$ as

$$f(\boldsymbol{x}, \boldsymbol{\theta}) = \boldsymbol{h}_L^\top \boldsymbol{w}, \quad \boldsymbol{h}_0 = \boldsymbol{x}, \quad \boldsymbol{h}_\ell = \sqrt{\frac{2}{d_\ell}} \, \mathrm{ReLU}\left(\boldsymbol{h}_{\ell-1}^\top \boldsymbol{W}_\ell\right), \tag{1}$$

where $\boldsymbol{\theta} := (\boldsymbol{W}_1, \dots, \boldsymbol{W}_L, \boldsymbol{w})$ for $\boldsymbol{W}_\ell \in \mathbb{R}^{d_{\ell-1} \times d_\ell}$, $\boldsymbol{w} \in \mathbb{R}^{d_L}$, $\ell \in [L]$ represents the trainable parameters and $d_0 = d$. The *Neural Tangent Kernel* (NTK) is defined as

$$K_{\mathrm{NTK}}^{(L)}(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}_{\boldsymbol{\theta}} \left[\left\langle \frac{\partial f(\boldsymbol{x}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}, \frac{\partial f(\boldsymbol{x}', \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right\rangle \right] \tag{2}$$

where $\boldsymbol{\theta}$ is from the standard Gaussian distribution. Given n data points $\boldsymbol{X} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_n]^\top \in \mathbb{R}^{n \times d}$, we will write $f(\boldsymbol{X}, \boldsymbol{\theta}) := [f(\boldsymbol{x}_1, \boldsymbol{\theta}), \dots, f(\boldsymbol{x}_n, \boldsymbol{\theta})]^\top \in \mathbb{R}^n$ and the NTK matrix as $\boldsymbol{K}_{\mathrm{NTK}}^{(L)} \in \mathbb{R}^{n \times n}$ whose (i, j)-th entry is $K_{\mathrm{NTK}}^{(L)}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ for $i, j \in [n]$.

The motivation for the definition of the NTK is as follows. Consider learning the network parameters $\boldsymbol{\theta}$ by minimizing the squared loss $\frac{1}{2} \left\| f(\boldsymbol{X}, \boldsymbol{\theta}) - \boldsymbol{y} \right\|_2^2$ for the target $\boldsymbol{y} \in \mathbb{R}^n$ using gradient descent with infinitesimally small learning rate. Regard the parameters as a time-evolving continuous variable $\boldsymbol{\theta}_t$ for $t \geq 0$ that develops in the course of the optimization. Then, Arora et al. (2019b) showed that

$$\frac{d}{dt} f\left(\boldsymbol{X}, \theta_t\right) = -\boldsymbol{K}_t \cdot \left(f(\boldsymbol{X}, \theta_t) - \boldsymbol{y}\right) \tag{3}$$

where $\boldsymbol{K}_t := \frac{\partial f(\boldsymbol{X}, \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}} \left(\frac{\partial f(\boldsymbol{X}, \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}} \right)^\top \in \mathbb{R}^{n \times n}$. In the infinite width limit, i.e., $d_1, \dots, d_L \to \infty$, recent works analyzed that $\boldsymbol{\theta}_t$ remains constant during optimization, i.e., equals to $\boldsymbol{\theta}_0$ (Chizat et al., 2019; Allen-Zhu et al., 2019; Du et al., 2019) and $\boldsymbol{K}_t = \boldsymbol{K}_0$. Furthermore, under a certain random initialization, in the same infinite width limit, $\boldsymbol{K}_0$ converges in probability to $\boldsymbol{K}_{\mathrm{NTK}}^{(L)}$. This implies equivalence between the prediction of neural network under the some initialization and kernel regression with NTK (Arora et al., 2019b).

In addition, when the parameters of only last layer are updated, the network prediction corresponds to the kernel so-called the neural network Gaussian Process (NNGP):

$$K_{\mathrm{NNGP}}^{(L)}(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}_{\boldsymbol{\theta}} \left[f(\boldsymbol{x}, \boldsymbol{\theta}) \cdot f(\boldsymbol{x}', \boldsymbol{\theta}) \right] \tag{4}$$

where the expectation is same as the NTK and we denote $[\boldsymbol{K}_{\mathrm{NNGP}}^{(L)}]_{i,j} := K_{\mathrm{NNGP}}^{(L)}(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

NTK computation. The NTK matrix of a fully-connected ReLU network can be computed by the following recursive relation (Jacot et al., 2018; Chizat et al., 2019; Arora et al., 2019b):

$$\begin{aligned}
&\boldsymbol{K}_{\mathrm{NTK}}^{(0)} = \boldsymbol{K}_{\mathrm{NNGP}}^{(0)} = \boldsymbol{X}\boldsymbol{X}^\top, \\
&\boldsymbol{K}_{\mathrm{NNGP}}^{(\ell)} = F_1\left(\boldsymbol{K}_{\mathrm{NNGP}}^{(\ell-1)}\right), \\
&\boldsymbol{K}_{\mathrm{NTK}}^{(\ell)} = \boldsymbol{K}_{\mathrm{NNGP}}^{(\ell)} + \boldsymbol{K}_{\mathrm{NTK}}^{(\ell-1)} \odot F_0\left(\boldsymbol{K}_{\mathrm{NNGP}}^{(\ell-1)}\right),
\end{aligned} \tag{5}$$

where $F_0, F_1 : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$ are defined as

$$\begin{aligned}
[F_0(\boldsymbol{K})]_{i,j} &:= 1 - \frac{1}{\pi}\cos^{-1}\left(\frac{[\boldsymbol{K}]_{i,j}}{\sqrt{[\boldsymbol{K}]_{i,i}[\boldsymbol{K}]_{j,j}}}\right), \\
[F_1(\boldsymbol{K})]_{i,j} &:= \sqrt{[\boldsymbol{K}]_{i,i}[\boldsymbol{K}]_{j,j}} \cdot f\left(\frac{[\boldsymbol{K}]_{i,j}}{\sqrt{[\boldsymbol{K}]_{i,i}[\boldsymbol{K}]_{j,j}}}\right)
\end{aligned}$$

where $f(x) = \frac{1}{\pi}\left(\sqrt{1-x^2} + (\pi - \cos^{-1}(x))x\right)$ for $x \in [-1, 1]$ and $\boldsymbol{K}$ is an arbitrary positive semidefinite matrix. Note that these matrix functions are derived from arc-cosine kernels (Cho & Saul, 2009):

$$\begin{aligned}
A_0(\boldsymbol{x}, \boldsymbol{x}') &:= 1 - \frac{1}{\pi}\cos^{-1}\left(\frac{\langle \boldsymbol{x}, \boldsymbol{x}'\rangle}{\|\boldsymbol{x}\|_2 \|\boldsymbol{x}'\|_2}\right), \\
A_1(\boldsymbol{x}, \boldsymbol{x}') &:= \|\boldsymbol{x}\|_2 \|\boldsymbol{x}'\|_2 \, f\left(\frac{\langle \boldsymbol{x}, \boldsymbol{x}'\rangle}{\|\boldsymbol{x}\|_2 \|\boldsymbol{x}'\|_2}\right).
\end{aligned} \tag{6}$$

Computing NTK of the network with L layers takes $\mathcal{O}(n^2(d+L))$ time and $\mathcal{O}(n(n+d))$ space complexity.

2.2 Random Features and Spectral Approximation

Random features (Rahimi & Recht, 2009) is a methodology for scaling kernel methods that saves both time and storage. In most general terms, the random features model targets kernels $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ that can be written as $K(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}_{\boldsymbol{v}\sim p}\left[\boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{v}) \cdot \boldsymbol{\Phi}(\boldsymbol{x}', \boldsymbol{v})\right]$ for some distribution p and a function $\boldsymbol{\Phi} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. The random features approximations works as follows. First, we generate m vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m \in \mathbb{R}^d$ sampled from p. We then define the feature map as

$$\boldsymbol{\Phi}_m(\boldsymbol{x}) := \frac{1}{\sqrt{m}}\left[\boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{v}_1), \ldots, \boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{v}_m)\right]^\top \in \mathbb{R}^m$$

and the approximate kernel is $K'(\boldsymbol{x}, \boldsymbol{x}') = \langle \boldsymbol{\Phi}_m(\boldsymbol{x}), \boldsymbol{\Phi}_m(\boldsymbol{x}')\rangle$.

The main utility of the random feature map is due to the fact that the kernel matrix $\boldsymbol{K}'$ associated with K' is a low-rank matrix with a known factorization. In particular, let $\boldsymbol{\Phi} := \left[\boldsymbol{\Phi}_m(\boldsymbol{x}_1), \ldots, \boldsymbol{\Phi}_m(\boldsymbol{x}_n)\right]^\top \in \mathbb{R}^{n\times m}$, then $\boldsymbol{K}' = \boldsymbol{\Phi}\boldsymbol{\Phi}^\top \approx \boldsymbol{K}$. The rank of the approximate kernel matrix is m, which allows faster computation and less storage. The parameter m trades between computational complexity and approximation quality. A small m results in faster speedup but less accurate kernel approximation.

Although the random features can approximate well the kernel function itself, it is still questionable how it affects the performance of downstream tasks. Several works on kernel approximation adopt *spectral approximation* bound with regularization $\lambda > 0$, that is,

$$(1-\varepsilon)\left(\boldsymbol{K} + \lambda \boldsymbol{I}\right) \preceq \boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \lambda \boldsymbol{I} \preceq (1+\varepsilon)\left(\boldsymbol{K} + \lambda \boldsymbol{I}\right)$$

for $\varepsilon > 0$ and show that it can provide rigorous guarantees of downstream applications including kernel ridge regression (Avron et al., 2017b), clustering and PCA (Musco & Musco, 2017).

2.3 CountSketch and TensorSketch Transforms

The CountSketch transform is a norm-preserving dimensionality reduction technique (Charikar et al., 2002). Formally, let $h : [d] \to [m]$ be a pairwise independent hash function whose bins are chosen uniformly at random and $s : [d] \to \{+1, -1\}$ be a pairwise independent sign function where signs are chosen uniformly at random. Given $\boldsymbol{x} \in \mathbb{R}^d$ and $m \in \mathbb{N}$, we define $\mathcal{C} : \mathbb{R}^d \to \mathbb{R}^m$ such that for $i \in [m]$

$$[\mathcal{C}(\boldsymbol{x})]_i = \sum_{j : h(j) = i} s(j)[\boldsymbol{x}]_j \tag{7}$$

and it is well-known that $\mathbb{E}[\langle \mathcal{C}(\boldsymbol{x}), \mathcal{C}(\boldsymbol{y}) \rangle] = \langle \boldsymbol{x}, \boldsymbol{y} \rangle$. Observe that it requires a single pass over the input, hence the running time becomes $\mathcal{O}(d)$.

Pham & Pagh (2013) proposed an efficient algorithm to apply CountSketch to the tensor product of vectors and referred to this as TensorSketch. Let $h_1 : [d_1] \to [m]$, $h_2 : [d_2] \to [m]$ be pairwise independent random hash functions and $s_1 : [d_1] \to \{-1, 1\}$, $s_2 : [d_2] \to \{-1, 1\}$ be pairwise independent sign functions. We denote the corresponding CountSketches by $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively. Now consider a new transform $\mathcal{C} : \mathbb{R}^{d_1 d_2} \to \mathbb{R}^m$ whose hash and sign functions are defined as $H(j_1, j_2) \equiv h_1(j_1) + h_2(j_2) \pmod{m}$ and $S(j_1, j_2) = s_1(j_1) \cdot s_2(j_2)$ for $j_1 \in [d_1], j_2 \in [d_2]$. Given $\boldsymbol{x} \in \mathbb{R}^{d_1}, \boldsymbol{y} \in \mathbb{R}^{d_2}$, Pham & Pagh (2013) showed that $\mathcal{C}\left(\boldsymbol{x} \otimes \boldsymbol{y}\right)$ equals the convolution between $\mathcal{C}_1(\boldsymbol{x})$ and $\mathcal{C}_2(\boldsymbol{y})$ and its computation can be amortized as

$$\mathcal{C}\left(\boldsymbol{x} \otimes \boldsymbol{y}\right) = \mathrm{FFT}^{-1}\left(\mathrm{FFT}(\mathcal{C}_1(\boldsymbol{x})) \odot \mathrm{FFT}(\mathcal{C}_2(\boldsymbol{y}))\right) \tag{8}$$

where FFT and FFT^{-1} are the Fast Fourier Transform and its inverse. By the inner product preserving property, TensorSketch also can be used as a low-rank approximation of element-wise product between two Gramian matrices. More specifically, given $\boldsymbol{X} \in \mathbb{R}^{n \times d_1}, \boldsymbol{Y} \in \mathbb{R}^{n \times d_2}$, it holds

$$\left(\boldsymbol{X}\boldsymbol{X}^\top\right) \odot \left(\boldsymbol{Y}\boldsymbol{Y}^\top\right) = \left(\boldsymbol{X} \otimes \boldsymbol{Y}\right)\left(\boldsymbol{X} \otimes \boldsymbol{Y}\right)^\top = \mathbb{E}\left[\mathcal{C}\left(\boldsymbol{X} \otimes \boldsymbol{Y}\right)\mathcal{C}\left(\boldsymbol{X} \otimes \boldsymbol{Y}\right)^\top\right] \tag{9}$$

where $\otimes$ and $\mathcal{C}$ are performed in a row-wise manner. Note that $\mathcal{C}\left(\boldsymbol{X} \otimes \boldsymbol{Y}\right) \in \mathbb{R}^{n \times m}$ can be computed in $\mathcal{O}(n(d_1 + d_2 + m \log m))$ time using Equation (8). This is much cheaper than that of computing n-by-n dense matrix when $n \gg d_1, d_2, m$. A larger m guarantees better approximation quality but also increases its running time. Avron et al. (2017a); Ahle et al. (2020); Woodruff & Zandieh (2020) analyzed a spectral approximation guarantee of TensorSketch transform.

For simplicity, we presented TensorSketch for vectors that are the tensor product of only two vectors. This is enough for our needs, though we mention that the TensorSketch transform can be defined for an arbitrary number of tensor products.

3 Efficient NTK Random Features via Sketching Method

Our goal is to design efficient such random features for the NTK. Seemingly, one can obtain such random features from definition of NTK (i.e., Equation (2)) by using gradients of the randomly initialized networks as features. However, the NTK is the infinite-width limit, while in practice we need to fix some finite width, which will introduce a bias. Moreover, Arora et al. (2019b) showed that the gradient features from a network with finite widths degrade the practical performance by a huge gap. Instead, we focus on the closed-form expression of NTK with ReLU activations.

We begin by introducing random features of arc-cosine kernels A_0 and A_1 originally due to Cho & Saul

Algorithm 1 Random Features for NTK of ReLU network via CountSketch

```
Input: $\boldsymbol{x} \in \mathbb{R}^d$, network depth $L$, feature dimensions $m_0$, $m_1$ and $m_{\texttt{cs}}$
$\boldsymbol{\Phi}^{(0)}(\boldsymbol{x}) \leftarrow \boldsymbol{x}, \boldsymbol{\Psi}^{(0)}(\boldsymbol{x}) \leftarrow \boldsymbol{x}$, and $m \leftarrow d$
for $\ell = 1$ to $L$ do
    Draw i.i.d. $\boldsymbol{w}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_m)$ for $i \in [m_0]$ and
        $\boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}) \leftarrow \sqrt{\frac{2}{m_0}}\, \mathrm{Step}\left(\left[\boldsymbol{w}_1, \ldots, \boldsymbol{w}_{m_0}\right]^\top \boldsymbol{\Psi}^{(\ell-1)}(\boldsymbol{x})\right)$
    Draw i.i.d. $\boldsymbol{w}'_j \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_m)$ for $j \in [m_1]$ and
        $\boldsymbol{\Psi}^{(\ell)}(\boldsymbol{x}) \leftarrow \sqrt{\frac{2}{m_1}}\, \mathrm{ReLU}\left(\left[\boldsymbol{w}'_1, \ldots, \boldsymbol{w}'_{m_1}\right]^\top \boldsymbol{\Psi}^{(\ell-1)}(\boldsymbol{x})\right)$
    Draw two independent CountSketch transforms $\mathcal{C}_0^{(\ell)}$ and $\mathcal{C}_1^{(\ell)}$ that map to $\mathbb{R}^{m_{\texttt{cs}}}$ and
        $\boldsymbol{\Gamma}^{(\ell)}(\boldsymbol{x}) \leftarrow \mathrm{FFT}^{-1}\left(\mathrm{FFT}(\mathcal{C}_0^{(\ell)}(\boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}))) \odot \mathrm{FFT}(\mathcal{C}_1^{(\ell)}(\boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x})))\right)$
    $\boldsymbol{\Phi}^{(\ell)}(\boldsymbol{x}) \leftarrow \left[\boldsymbol{\Psi}^{(\ell)}(\boldsymbol{x}),\ \boldsymbol{\Gamma}^{(\ell)}(\boldsymbol{x})\right]$, $m \leftarrow m_1$
end for
return $\boldsymbol{\Phi}^{(L)}(\boldsymbol{x})$
```

(2009):

$$a_0(\boldsymbol{x}) = \sqrt{\frac{2}{m_0}}\, \mathrm{Step}\left(\left[\boldsymbol{w}_1, \ldots, \boldsymbol{w}_{m_0}\right]^\top \boldsymbol{x}\right), \tag{10}$$

$$a_1(\boldsymbol{x}) = \sqrt{\frac{2}{m_1}}\, \mathrm{ReLU}\left(\left[\boldsymbol{w}'_1, \ldots, \boldsymbol{w}'_{m_1}\right]^\top \boldsymbol{x}\right) \tag{11}$$

where $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_{m_0}, \boldsymbol{w}'_1, \ldots, \boldsymbol{w}'_{m_1} \in \mathbb{R}^d$ are sampled from $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$. It is known that $\mathbb{E}[\langle a_0(\boldsymbol{x}), a_0(\boldsymbol{x}')\rangle] = A_0(\boldsymbol{x}, \boldsymbol{x}')$ and $\mathbb{E}[\langle a_1(\boldsymbol{x}), a_1(\boldsymbol{x}')\rangle] = A_1(\boldsymbol{x}, \boldsymbol{x}')$ for $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^d$.

Recently, Bietti & Mairal (2019) presented an explicit infinite-dimensional feature map for the NTK of ReLU networks by using recursive tensoring of explicit feature maps for the arc-cosine kernel. Replacing each explicit feature map with a random feature map for the corresponding kernel we can obtain a random feature map for the NTK. The resulting construction is:

$$\begin{aligned} \boldsymbol{\Psi}^{(\ell+1)}(\boldsymbol{x}) &= a_1\left(\boldsymbol{\Psi}^{(\ell)}(\boldsymbol{x})\right), \quad \boldsymbol{\Phi}^{(0)}(\boldsymbol{x}) = \boldsymbol{\Psi}^{(0)}(\boldsymbol{x}) = \boldsymbol{x}, \\ \boldsymbol{\Phi}^{(\ell+1)}(\boldsymbol{x}) &= \left[\boldsymbol{\Psi}^{(\ell+1)}(\boldsymbol{x}), \quad a_0\left(\boldsymbol{\Psi}^{(\ell)}(\boldsymbol{x})\right) \otimes \boldsymbol{\Phi}^{(\ell)}(\boldsymbol{x})\right], \end{aligned} \tag{12}$$

for $\ell = 0, \ldots, L-1$. These features can be used for approximating both NTK and NNGP as $K_{\mathrm{NTK}}^{(\ell)}(\boldsymbol{x}, \boldsymbol{x}') \approx \left\langle \boldsymbol{\Phi}^{(\ell)}(\boldsymbol{x}), \boldsymbol{\Phi}^{(\ell)}(\boldsymbol{x}')\right\rangle$ and $K_{\mathrm{NNGP}}^{(\ell)}(\boldsymbol{x}, \boldsymbol{x}') \approx \left\langle \boldsymbol{\Psi}^{(\ell)}(\boldsymbol{x}), \boldsymbol{\Psi}^{(\ell)}(\boldsymbol{x}')\right\rangle$.

However, one major drawback of the last construction is that the number of features is exponential in the depth. Indeed, the dimension of output features $\boldsymbol{\Phi}^{(L)}(\boldsymbol{x})$ is $\left(\sum_{k=0}^{L-1} m_0^k\right) m_1 + m_0^L d = \mathcal{O}(m_0^L(m_1 + d))$. This also leads to $\mathcal{O}(m_0^L(m_1 + d) + m_1^2)$ time complexity. The exponential growth in depth L is due to the tensor product $\otimes$ in Equation (12). For a large L, the number of features can easily be larger than the number of data points n and any computational saving is hopeless.

In order to make the feature map more compact, we utilize a TensorSketch to reduce the dimension of $a_0\left(\mathbf{\Psi}^{(\ell)}(\boldsymbol{x})\right) \otimes \mathbf{\Phi}^{(\ell)}(\boldsymbol{x})$. We do so by replacing it with

$$\mathbf{\Gamma}^{(\ell)}(\boldsymbol{x}) := \mathrm{FFT}^{-1}\left(\mathrm{FFT}\left(\mathcal{C}_1(a_0(\mathbf{\Psi}^{(\ell)}(\boldsymbol{x})))\right) \odot \mathrm{FFT}\left(\mathcal{C}_2(\mathbf{\Phi}^{(\ell)}(\boldsymbol{x}))\right)\right)$$

where $\mathcal{C}_1$ and $\mathcal{C}_2$ are independent[1] CountSketch transforms that map to $\mathbb{R}^{m_{\mathsf{cs}}}$. Denote $\widehat{\mathbf{\Phi}}^{(\ell)}(\boldsymbol{x}) := \left[\mathbf{\Psi}^{(\ell)}(\boldsymbol{x}),\ \mathbf{\Gamma}^{(\ell)}(\boldsymbol{x})\right]$ and one can expect that

$$\left\langle \mathbf{\Phi}^{(\ell)}(\boldsymbol{x}), \mathbf{\Phi}^{(\ell)}(\boldsymbol{x}')\right\rangle \approx \left\langle \widehat{\mathbf{\Phi}}^{(\ell)}(\boldsymbol{x}), \widehat{\mathbf{\Phi}}^{(\ell)}(\boldsymbol{x}')\right\rangle \tag{13}$$

from the property in Equation (9). The process is repeated for every layer. A pseudo-code for the proposed feature construction is described in Algorithm 1.

We now provide that the approximation error bound of generated features from Algorithm 1.

Theorem 1. *Given $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$ such that $\|\boldsymbol{x}\|_2 = \|\boldsymbol{x}'\|_2 = 1$ and $L \geq 1$, let $K_{\mathrm{NTK}}^{(L)}$ the NTK of L-layer fully-connected ReLU network. Given $\delta, \in (0,1)$, $\varepsilon \in (0, 1/L)$, there exist constants $C_0, C_1, C_2 > 0$ such that*

$$m_0 \geq C_0 \frac{L^2}{\varepsilon^2} \log\left(\frac{L}{\delta}\right), \quad m_1 \geq C_1 \frac{L^6}{\varepsilon^4} \log\left(\frac{L}{\delta}\right), \quad m_{\mathsf{cs}} \geq C_2 \frac{L^3}{\varepsilon^2 \delta}$$

and

$$\Pr\left(\left|\left\langle \mathbf{\Phi}^{(L)}(\boldsymbol{x}), \mathbf{\Phi}^{(L)}(\boldsymbol{x}')\right\rangle - K_{\mathrm{NTK}}^{(L)}(\boldsymbol{x}, \boldsymbol{x}')\right| \leq L\varepsilon\left(1 + \frac{\varepsilon}{2}\right)^2 + \varepsilon\right) \geq 1 - \delta$$

where $\mathbf{\Phi}^{(L)}(\boldsymbol{x}), \mathbf{\Phi}^{(L)}(\boldsymbol{x}') \in \mathbb{R}^{m_1 + m_{\mathsf{cs}}}$ be the output of Algorithm 1 of $\boldsymbol{x}, \boldsymbol{x}'$, respectively, using the same CountSketch transforms.

The proof of Theorem 1 is provided in Appendix A.1. We note that the restriction of inputs to the hypersphere (i.e., $\|\boldsymbol{x}_i\|_2 = 1$) is a common assumption used in the NTK analysis (Bietti & Mairal, 2019; Geifman et al., 2020). This can be easily achieved by normalizing input data points. From Theorem 1, the dimension of the proposed random features can be $\mathcal{O}\left(\frac{L^6}{\varepsilon^4}\log\left(\frac{L}{\delta}\right) + \frac{L^3}{\varepsilon^2\delta}\right)$, which gets rid of the exponential dependency on L, to guarantee the above error bound. Furthermore, Arora et al. (2019b) studied that the gradient of randomly initialized ReLU network with finite width can approximate the NTK, but their feature dimension should be $\Omega\left(\frac{L^{13}}{\varepsilon^8}\log^2\left(\frac{L}{\delta}\right) + \frac{L^6}{\varepsilon^4}\log\left(\frac{L}{\delta}\right) d\right)$ to guarantee an approximation error of $(L+1)\varepsilon$ with probability at least $1 - \delta$. This error bound is smaller than that in Theorem 1 by a factor of $\left(1 + \frac{\varepsilon}{2}\right)^2$, but their feature dimension is much larger by a factor of $\mathcal{O}\left(\frac{L^7}{\varepsilon^4}\log\left(\frac{L}{\delta}\right)\right)$. We empirically observe that Algorithm 1 requires much fewer dimension than both the random gradient and the naïve feature map construction in Equation (12) to achieve the same error and provide these results in Section 5.1.

4 Spectral Approximation for the NTK

Our ultimate goal is to provide lower bounds on the parameter $m_0, m_1, m_{\mathsf{cs}}$ to achieve tight error bound in terms of spectral approximation of the NTK, i.e.,

$$(1 - \varepsilon)\left(\boldsymbol{K}_{\mathrm{NTK}}^{(L)} + \lambda \boldsymbol{I}\right) \preceq \mathbf{\Phi}^{(L)}(\mathbf{\Phi}^{(L)})^\top + \lambda \boldsymbol{I} \preceq (1 + \varepsilon)\left(\boldsymbol{K}_{\mathrm{NTK}}^{(L)} + \lambda \boldsymbol{I}\right),$$

[1] i.e., hash and sign functions of $\mathcal{C}_1$ and $\mathcal{C}_2$ are independent.

where $\mathbf{\Phi}^{(L)} := \left[\mathbf{\Phi}^{(L)}(\boldsymbol{x}_1), \dots, \mathbf{\Phi}^{(L)}(\boldsymbol{x}_n)\right]^\top$. We first provide spectral bounds of the arc-cosine kernels in Equation (6), which are necessary prerequisites of our analysis on the NTK random features. Based on these results, we present spectral bound of a two-layer ReLU network (i.e., $L = 1$ in Equation (1)) and discuss hardness on generalizing this result to networks with deeper layer. To the best of our knowledge, this has not been studied in previous literature.

4.1 Spectral Approximation for Arc-cosine Kernels

Recently, Avron et al. (2017b); Lee et al. (2020) proposed that random features with sampling from a modified distribution can give better approximation guarantee. More precisely, suppose $\mathbf{\Phi} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a function for random features of kernel K with distribution p. Consider random vectors $\boldsymbol{z}_1, \dots, \boldsymbol{z}_m \in \mathbb{R}^d$ sampled from some distribution q. Denote that

$$\overline{\mathbf{\Phi}}_m(\boldsymbol{x}) := \frac{1}{\sqrt{m}} \left[\sqrt{\frac{p(\boldsymbol{z}_1)}{q(\boldsymbol{z}_1)}} \mathbf{\Phi}(\boldsymbol{x}, \boldsymbol{z}_1), \dots \sqrt{\frac{p(\boldsymbol{z}_m)}{q(\boldsymbol{z}_m)}} \mathbf{\Phi}(\boldsymbol{x}, \boldsymbol{z}_m) \right]^\top \in \mathbb{R}^m$$

then one can verify that $\mathbb{E}_{\boldsymbol{z} \sim q}\left[\langle \overline{\mathbf{\Phi}}_m(\boldsymbol{x}), \overline{\mathbf{\Phi}}_m(\boldsymbol{x}') \rangle\right] = K(\boldsymbol{x}, \boldsymbol{x}')$ for all $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^d$. Now assume that the distribution q is defined by a measurable function $q_\lambda : \mathbb{R}^d \to \mathbb{R}$ satisfies that $q(\boldsymbol{v}) = q_\lambda(\boldsymbol{v}) / \int_{\mathbb{R}^d} q_\lambda(\boldsymbol{v}) d\boldsymbol{v}$ and

$$q_\lambda(\boldsymbol{v}) \geq p(\boldsymbol{v}) \cdot \mathbf{\Phi}(\boldsymbol{X}, \boldsymbol{v})^\top (\boldsymbol{K} + \lambda \boldsymbol{I})^{-1} \mathbf{\Phi}(\boldsymbol{X}, \boldsymbol{v}) \tag{14}$$

for $\lambda > 0$ where $\mathbf{\Phi}(\boldsymbol{X}, \boldsymbol{v}) := [\mathbf{\Phi}(\boldsymbol{x}_1, \boldsymbol{v}), \dots, \mathbf{\Phi}(\boldsymbol{x}_n, \boldsymbol{v})]^\top \in \mathbb{R}^n$. Then, they proved that with high probability it holds

$$(1 - \varepsilon)(\boldsymbol{K} + \lambda \boldsymbol{I}) \preceq \overline{\mathbf{\Phi}}_m \overline{\mathbf{\Phi}}_m^\top + \lambda \boldsymbol{I} \preceq (1 + \varepsilon)(\boldsymbol{K} + \lambda \boldsymbol{I})$$

where $\overline{\mathbf{\Phi}}_m = \left[\overline{\mathbf{\Phi}}_m(\boldsymbol{x}_1), \dots, \overline{\mathbf{\Phi}}_m(\boldsymbol{x}_n)\right]^\top \in \mathbb{R}^{n \times m}$ and $\varepsilon \in (0, 1)$ is a given parameter.

Observe that the lower bound in Equation (14) requires $\mathcal{O}(n^3)$ operations to compute due to the matrix inverse. This can hurt the computational advantage of random features. Hence, it is important to find such distribution q that is easy to sample while holding the Equation (14).

In what follows, we provide that the original arc-cosine random features of 0-th order in Equation (10) can indeed guarantee a spectral approximation bound.

Theorem 2. *Given $\boldsymbol{X} \in \mathbb{R}^{n \times d}$, let $\boldsymbol{A}_0 \in \mathbb{R}^{n \times n}$ be the arc-cosine kernel matrix of 0-th order with $\boldsymbol{X}$ and denote $\mathbf{\Phi}_0 := \sqrt{\frac{2}{m}} \mathrm{Step}(\boldsymbol{X}\boldsymbol{W}) \in \mathbb{R}^{n \times m}$ where each entry in $\boldsymbol{W} \in \mathbb{R}^{d \times m}$ is an i.i.d. sample from $\mathcal{N}(0, 1)$. Let s_λ be the statistical dimension of $\boldsymbol{A}_0$. Given $\lambda \in (0, \|\boldsymbol{A}_0\|_2)$, $\varepsilon \in (0, 1/2)$ and $\delta \in (0, 1)$, if $m \geq \frac{8}{3} \frac{n}{\lambda \varepsilon^2} \log\left(\frac{16 s_\lambda}{\delta}\right)$, then it holds*

$$(1 - \varepsilon)(\boldsymbol{A}_0 + \lambda \boldsymbol{I}) \preceq \mathbf{\Phi}_0 \mathbf{\Phi}_0^\top + \lambda \boldsymbol{I} \preceq (1 + \varepsilon)(\boldsymbol{A}_0 + \lambda \boldsymbol{I})$$

with probability at least $1 - \delta$.

The proof of Theorem 2 is provided in Appendix A.2. The analysis is similar to that studied by Avron et al. (2017a), i.e., $q_\lambda(\boldsymbol{v}) = (n/\lambda) p(\boldsymbol{v})$, which implies that the modified distribution is identical to the original one.

Next, we present our result on spectral approximation for arc-cosine random features of 1-st order. Unlike the previous case, sampling vectors from the modified distribution in the form of the Gaussian scaled by squared ℓ_2-norm is required. The formal statement is provided in Theorem 3.

Algorithm 2 Modified Random Arc-cosine Features of 1-st order Equation (16) via Gibbs Sampling

```
Input: $\boldsymbol{X} \in \mathbb{R}^{n \times d}$, feature dimension $m_1$, Gibbs iterations $T$
Draw i.i.d. $\boldsymbol{v}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$ for $i \in [m_1]$
for $i = 1$ to $m_1$ do
    $q(x, z) \leftarrow$ inverse of $\frac{\mathrm{erf}(x/\sqrt{2})+1}{2} - \frac{x \exp(-x^2/2)}{\sqrt{2\pi}(z+1)}$ (corresponds to the CDF of $\Pr([\boldsymbol{v}_i]_j | [\boldsymbol{v}_i]_{\setminus \{j\}})$)
    for $t = 1$ to $T$ do
        for $j = 1$ to $d$ do
            $u \leftarrow$ sample from $[0, 1]$ at uniformly random
            $[\boldsymbol{v}_i]_j \leftarrow q\left(u, \sum_{k \in [d] \setminus \{j\}} [\boldsymbol{v}_i]_k^2\right)$
        end for
    end for
end for
return $\sqrt{\frac{2d}{m}} \left[\frac{\mathrm{ReLU}(\boldsymbol{X}\boldsymbol{v}_1)}{\|\boldsymbol{v}_1\|_2}, \ \dots \ , \frac{\mathrm{ReLU}(\boldsymbol{X}\boldsymbol{v}_m)}{\|\boldsymbol{v}_m\|_2}\right]$
```

Theorem 3. *Given $\boldsymbol{X} \in \mathbb{R}^{n \times d}$, let $\boldsymbol{A}_1 \in \mathbb{R}^{n \times n}$ be the arc-cosine kernel matrix of 1-th order with $\boldsymbol{X}$ and $\boldsymbol{v}_1, \dots, \boldsymbol{v}_m \in \mathbb{R}^d$ be i.i.d. random vectors from probability distribution*

$$q(\boldsymbol{v}) = \frac{1}{(2\pi)^{d/2} d} \|\boldsymbol{v}\|_2^2 \exp\left(-\frac{1}{2} \|\boldsymbol{v}\|_2^2\right). \tag{15}$$

Denote

$$\boldsymbol{\Phi}_1 := \sqrt{\frac{2d}{m}} \left[\frac{\mathrm{ReLU}(\boldsymbol{X}\boldsymbol{v}_1)}{\|\boldsymbol{v}_1\|_2}, \ \dots \ , \frac{\mathrm{ReLU}(\boldsymbol{X}\boldsymbol{v}_m)}{\|\boldsymbol{v}_m\|_2}\right] \tag{16}$$

and let s_λ be the statistical dimension of $\boldsymbol{A}_1$. Given $\lambda \in (0, \|\boldsymbol{A}_1\|_2)$, $\varepsilon \in (0, 1/2)$ and $\delta \in (0, 1)$, if $m \geq \frac{8}{3} \frac{d \|\boldsymbol{X}\|_2^2}{\lambda \varepsilon^2} \log\left(\frac{16 s_\lambda}{\delta}\right)$, then it holds that

$$(1 - \varepsilon)(\boldsymbol{A}_1 + \lambda I) \preceq \boldsymbol{\Phi}_1 \boldsymbol{\Phi}_1^\top + \lambda \boldsymbol{I} \preceq (1 + \varepsilon)(\boldsymbol{A}_1 + \lambda I)$$

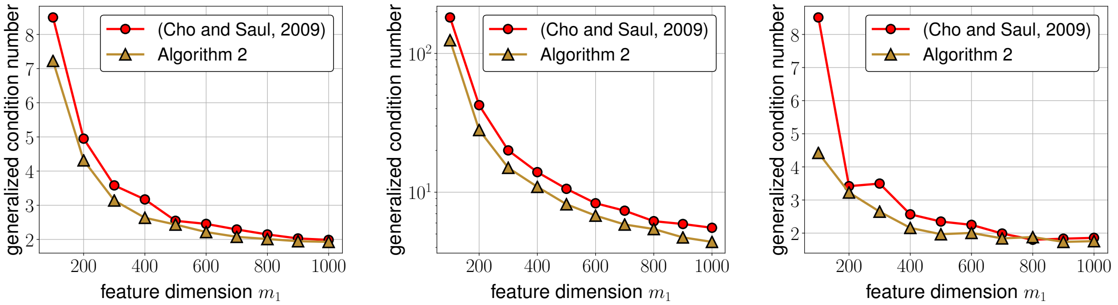
with probability at least $1 - \delta$.

The proof of Theorem 3 is provided in Appendix A.3. We note that the modified distribution can be expressed as a closed-form formula as in Equation (15). Once random vectors are sampled from this distribution, the modified random features in Equation (16) can be computed at the same cost of the original features in Equation (11). In addition, the lower bound on feature dimension depends on the square of the spectral norm of input.

Approximate sampling. It is not trivial to sample a vector $\boldsymbol{v} \in \mathbb{R}^d$ from the distribution $q(\cdot)$ defined in Equation (15). Thus, we suggest to perform an approximate sampling via Gibbs sampling. The algorithm starts with a random initialized vector $\boldsymbol{v}$ and then iteratively replaces $[\boldsymbol{v}]_i$ with a sample from $q([\boldsymbol{v}]_i | [\boldsymbol{v}]_{\setminus i})$ for $i \in [d]$ and repeat this process for T iterations. One can derive the conditional distribution

$$q([\boldsymbol{v}]_i | [\boldsymbol{v}]_{\setminus \{i\}}) \propto \frac{\|\boldsymbol{v}\|_2^2}{1 + \|\boldsymbol{v}\|_2^2 - [\boldsymbol{v}]_i^2} \exp\left(-\frac{[\boldsymbol{v}]_i^2}{2}\right) \tag{17}$$

and sampling a single random variable from Equation (17) can be done via the inverse transformation method.[2] We empirically verify that $T = 1$ is enough for promising performances. The running time of Gibbs sampling becomes $\mathcal{O}(m_1 d T)$ where m_1 corresponds to the number of independent samples from



(a) `abalone` (b) `ecoli` (c) `titanic`

Figure 1: Generalized condition numbers of arc-cosine random features of (Cho & Saul, 2009) and the proposed approach (Algorithm 2) under real-world UCI datasets.

$q(\boldsymbol{v})$. This is negligible compared to the feature map construction of CountSketch for $T = \mathcal{O}(1)$. The pseudo-code for the modified random features of A_1 using Gibbs sampling is outlined in Algorithm 2.

We evaluate approximation quality of the proposed method (Algorithm 2) to that of the random features (Cho & Saul, 2009) in Figure 1. In particular, we compute the condition number (i.e., ratio of the largest and smallest eigenvalues) of $(\boldsymbol{A}_1 + \lambda \boldsymbol{I})^{-1/2} \left(\boldsymbol{\Phi}_1 \boldsymbol{\Phi}_1^\top + \lambda \boldsymbol{I}\right) (\boldsymbol{A}_1 + \lambda \boldsymbol{I})^{-1/2}$. If $(\boldsymbol{\Phi}_1 \boldsymbol{\Phi}_1^\top + \lambda \boldsymbol{I})$ is spectrally close to $(\boldsymbol{A}_1 + \lambda \boldsymbol{I})$, then the corresponding condition number will be close to 1. We evaluate the condition numbers of those random features using 3 UCI datasets and set $\lambda = 10^{-4} \cdot n$ when n data points are given. For each dataset, we increase m_1 from 100 to 1,000. Observe that the proposed random features for arc-cosine features have smaller condition numbers than the previous method for all datasets. We provide more experimental results that the modified random features can improve performance on downstream tasks in Section 5.

4.2 Spectral Approximation for the NTK of Two-layer ReLU Network

We are now ready to state a spectral approximation bound for our NTK random features of a two-layer ReLU network, i.e., $L = 1$.

Theorem 4. *Given $\boldsymbol{X} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_n]^\top \in \mathbb{R}^{n \times d}$, assume that $\|\boldsymbol{x}_i\|_2 = 1$ for $i \in [n]$. Let $\boldsymbol{K}_{\mathrm{NTK}}$ be the NTK of two-layer ReLU network, i.e., $L = 1$ in Equation (1), and $\boldsymbol{A}_0, \boldsymbol{A}_1$ denote the arc-cosine kernels of 0-th, 1-st order with $\boldsymbol{X}$, respectively, as in Equation (6). For any $\lambda \in (0, 2\min(\|\boldsymbol{A}_0\|_2, \|\boldsymbol{A}_1\|_2)]$, suppose s_λ is an upper bound of statistical dimensions of both $\boldsymbol{A}_0, \boldsymbol{A}_1$. Given $\varepsilon \in (0, 1/2)$, $\delta \in (0,1)$, let $\boldsymbol{\Phi} \in \mathbb{R}^{n \times (m_1 + m_{\mathtt{cs}})}$ be the first output of Algorithm 1 with $L = 1$ and*

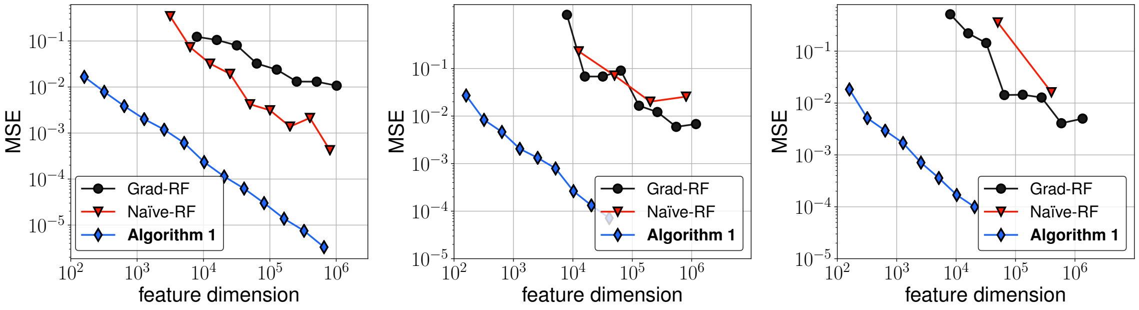
$$m_0 \geq \frac{48n}{\varepsilon^2 \lambda} \log\left(\frac{48 s_\lambda}{\delta}\right), \quad m_1 \geq \frac{16}{3} \frac{d \, \|\boldsymbol{X}\|_2^2}{\varepsilon^2 \lambda} \log\left(\frac{48 s_\lambda}{\delta}\right), \quad m_{\mathtt{cs}} \geq \frac{297}{\varepsilon^2 \delta} \left(\frac{n}{\lambda + 1}\right)^2$$

Then, with probability at least $1 - \delta$, it holds that

$$(1 - \varepsilon)\left(\boldsymbol{K}_{\mathrm{NTK}} + \lambda \boldsymbol{I}\right) \preceq \boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \lambda \boldsymbol{I} \preceq (1 + \varepsilon)\left(\boldsymbol{K}_{\mathrm{NTK}} + \lambda \boldsymbol{I}\right). \tag{18}$$

The proof of Theorem 4 is provided in Appendix A.4. We note that the ridge regularization parameter typically set to $\lambda = \lambda' n$ where λ' is a small constant, e.g., 10^{-4} (Rudi & Rosasco, 2016; Avron et al., 2017b; Geifman et al., 2020). Combining this setting with the fact that $\|\boldsymbol{x}_i\|_2 = 1$ yields that $\boldsymbol{s}_\lambda = \mathcal{O}(1)$. Hence, it is enough to choose $m_0 = \mathcal{O}\left(\frac{1}{\varepsilon^2} \log\left(\frac{1}{\delta}\right)\right), m_{\mathtt{cs}} = \mathcal{O}\left(\frac{1}{\varepsilon^2 \delta}\right)$ and $m_1 = \mathcal{O}\left(\frac{d}{\varepsilon^2} \log\left(\frac{1}{\delta}\right)\right)$ to achieve the spectral approximation in Equation (18) since $\|\boldsymbol{X}\|_2^2 \leq \|\boldsymbol{X}\|_F^2 = n$. This leads us to $\mathcal{O}\left(\frac{d}{\varepsilon^2} \log\left(\frac{1}{\delta}\right) + \frac{1}{\varepsilon^2 \delta}\right)$ feature dimension which is nearly linear in the input dimension d.

[2] It requires the CDF of $q([\boldsymbol{v}]_i | [\boldsymbol{v}]_{\setminus \{i\}})$ which is equivalent to $\frac{\mathrm{erf}\left([\boldsymbol{v}]_i / \sqrt{2}\right) + 1}{2} - \frac{[\boldsymbol{v}]_i \exp\left(-[\boldsymbol{v}]_i^2 / 2\right)}{\sqrt{2\pi}\left(1 + \|\boldsymbol{v}\|_2^2 - [\boldsymbol{v}]_i^2\right)}$.



(a) $L = 1$ (b) $L = 2$ (c) $L = 3$

Figure 2: Mean squared error of entries in NTK matrix approximated by (1) gradient of randomly initialized networks (Grad-RF), (2) the naive feature map construction in (Bietti & Mairal, 2019) (Naïve-RF) and our algorithm based on a sketching method under subsampled MNIST dataset.

The current proof technique cannot be used to generalize the result in Theorem 4 to deeper networks (i.e., $L \geq 2$). For the proof to work, one needs a monotone property of arc-cosine kernels, i.e., $F_1(\boldsymbol{X}) \preceq F_1(\boldsymbol{Y})$ for $\boldsymbol{X} \preceq \boldsymbol{Y}$. However, this property does not hold in general. Thus, we leave the extension to deeper networks to future work.

5 Experiments

In this section, we provide experimental results of our method on kernel approximation and various kernel learning tasks including classification, regression and active learning.

5.1 Kernel Approximation on MNIST Dataset

We first explore Algorithm 1 for approximating the NTK matrices. We compare to gradient-based NTK random features (Arora et al., 2019b) (Grad-RF) and the naïve random features without sketching (Bietti & Mairal, 2019) (Naïve-RF) as baseline methods. To compute the exact NTK, we randomly choose $n = 10{,}000$ data samples from MNIST dataset and evaluate the mean squared error (MSE) of all approximate entries in NTK. We use the ReLU network with depths $L = 1, 2, 4$. For Grad-RF, we use an implementation proposed by Novak et al. (2020).[3] In particular, it returns an approximate NTK matrix rather than random features because the dimension of features can be larger than n which loses the computational gain of random features. For example, gradient of a two-layer, 16-width ReLU network for MNIST has 127,040 dimension. We vary the network width in $\{2^1 \ldots, 2^7\}$. For fair dimension comparisons, we report the expected dimension of Grad-RF. For Naïve-RF, we set $m_0 = m_1$ and $m_0 \in \{2^1, \ldots, 2^7\}$. For our method, we set $m_0 = m_1 = m - m_{\texttt{cs}}$ and $m_{\texttt{cs}} \in \{\frac{m}{10}, \ldots, \frac{9m}{10}\}$ for each $m \in \{10 \cdot 2^4, \ldots, 10 \cdot 2^{16}\}$ and report the average MSE of 9 different values of $m_{\texttt{cs}}$. We omit to report the result when memory overflow causes.

In Figure 2, we observe that our random features achieves the lowest MSE for the same dimension compared to other competitors. The Grad-RF is the worst method and this corresponds to observations reported in Arora et al. (2019b), i.e., gradient features from a finite width network can degrade practical performances. As the number of layers L increases, the performance gaps between Naïve-RF and other methods become large because its dimension grows exponential in L.

[3] https://github.com/google/neural-tangents

Table 1: Results of average test accuracy, P90, P95 and PMA (percentage of the maximum accuracy) on 90 UCI classification datasets. **Bold entries** indicate the best results.

Method	Test Accuracy (%)	P90	P95	PMA
AdaBoost	76.32 ± 3.56	66.67	37.78	89.44
Random Forest	77.46 ± 3.75	79.31	57.47	90.84
k-Nearest Neighbors	76.95 ± 3.42	72.22	43.33	90.31
Fully-connected ReLU Network	81.10 ± 3.11	85.56	78.89	95.33
Polynomial Kernel	79.54 ± 3.41	79.31	66.67	93.31
RBF Kernel	81.79 ± 2.95	91.11	75.56	95.97
Random Fourier Features	81.61 ± 2.98	88.89	71.11	95.74
NTK	**82.24 ± 2.94**	**92.22**	**80.00**	**96.53**
NTK Random Features	81.84 ± 2.89	**92.22**	75.56	96.16
NTK Random Features with GS	81.85 ± 2.98	**92.22**	75.56	96.05

5.2 Classification on Small-scale UCI Datasets

Next, we run our algorithm under 90 small-scale UCI classification datasets. The number of data points n varies from 10 to 5,000. We choose hyperparameters using validation data and evaluate the test accuracy using 4-fold cross-validation provided in Fernández-Delgado et al. (2014).[4] We also consider the following additional metrics used in Arora et al. (2019c); Geifman et al. (2020); P90 and P95 are the ratios of datasets where a classifier reaches at least 90% and 95% of the maximum accuracy, and PMA (percentage of the maximum accuracy) is the average ratio its accuracy to the maximum among 90 datasets.

We run Algorithm 1 with and without Gibbs sampling (GS) (i.e., Algorithm 2) where the number of Gibbs iteration is set to $T = 1$ throughout all experiments. We also execute various classifiers including AdaBoost, random forest, k-nearest neighbors and support vector classifier (SVC). For methods running with SVC, we search the cost value C in $\{2^{-19}, \ldots, 2^{-3}, 10^{-7}, \ldots, 10^{3}\}$ and choose the best one that achieves the best validation accuracy. We use the support vector classifier (SVC) for random features methods (ours, RFF). For methods using SVC, the cost value C is chosen by searching in $\{2^{-19}, \ldots, 2^{20}\}$ that achieves the best validation accuracy. For our algorithm and RFF, we consider the output dimension m as a hyperparameter. We search m in $\{10, 20, \ldots, 100\}$ for datasets with $n \leq 600$ and explore m in $\{20, 40, \ldots, 200\}$ for datasets with $n > 600$ that achieves the best validation accuracy. For NTK, the network depth L changes from 1 to 5 which is the same setup in (Arora et al., 2019c; Shankar et al., 2020). We also compare test accuracy of fully-connected ReLU network. We explore the network depth in $\{1, 2, 3, 4, 5\}$ and width in $\{2^{6}, \ldots, 2^{11}\}$. The ReLU network is trained by Adam optimizer for 100 epochs with an initial learning rate 0.1 and cosine annealing is used to schedule learning rate.

In Table 1, the average test accuracy with 95% confidence interval, P90/95 and PMA are reported. Observe that the NTK achieves the best results while the NTK Random Features with GS is the second best. The NTK Random Features performs better than the Random Fourier Features because the NTK is more appropriate choice compared to the RBF kernel. Finally, our method with GS shows higher accuracy than that without GS.

5.3 Regression on Large-scale UCI Datasets

We also demonstrate the computational efficiency of our method using 4 large-scale UCI regression datasets. In particular, we consider kernel ridge regression (KRR) problem. For a kernel function $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, KRR problem can be formulated as

$$y_{\text{test}} = K(\boldsymbol{x}_{\text{test}}, \boldsymbol{X}) \left(\boldsymbol{K} + \lambda n \boldsymbol{I}\right)^{-1} \boldsymbol{y}. \tag{19}$$

[4] http://persoal.citius.usc.es/manuel.fernandez.delgado/papers/jmlr/data.tar.gz

Table 2: Results of the mean squared errors (MSE) and wall-clock time (sec) on large-scale UCI regression datasets. We measure the entire time for solving the kernel ridge regression. **Bold entries** indicate the best MSE or time for each dataset. (-) means the Out-of-Memory error.

	MillionSongs		WorkLoads		Protein		SuperConduct	
# of Training Data n	467,315		179,585		39,617		19,077	
	MSE	Time (s)	MSE	Time (s)	MSE	Time (s)	MSE	Time (s)
RBF Kernel	(-)	(-)	(-)	(-)	112.82	110.2	2239.83	19.5
Random Fourier Features	108.50	159	7.05×10^4	63.7	**81.98**	14.6	1175.13	**7.1**
NTK	(-)	(-)	(-)	(-)	90.03	243	513.25	51.9
NTK Random Features	**80.77**	**149.7**	2.43×10^4	**50.9**	90.28	**13.9**	527.58	7.9
NTK Random Features with GS	80.94	168.5	$\mathbf{2.38\times10^4}$	53.0	85.99	16.1	**492.78**	12.7

where $\boldsymbol{X} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_n]^\top \in \mathbb{R}^{n \times d}$ is training data, $\boldsymbol{y} \in \mathbb{R}^n$ is training label, $\boldsymbol{x}_{\text{test}} \in \mathbb{R}^d$ is a test data, y_{test} is a predicted label, $[\boldsymbol{K}]_{ij} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and $K(\boldsymbol{x}_{\text{test}}, \boldsymbol{X}) = [K(\boldsymbol{x}_{\text{test}}, \boldsymbol{x}_1), \dots, K(\boldsymbol{x}_{\text{test}}, \boldsymbol{x}_n)] \in \mathbb{R}^{1 \times n}$. Note that solving the problem can require $\mathcal{O}(n^3)$ time complexity due to the matrix inversion in general. Consider a feature map $\boldsymbol{\Phi} : \mathbb{R}^d \to \mathbb{R}^m$ can approximate the kernel K such that $K(\boldsymbol{x}, \boldsymbol{x}) \approx \hat{K}(\boldsymbol{x}, \boldsymbol{x}) = \langle \boldsymbol{\Phi}(\boldsymbol{x}), \boldsymbol{\Phi x})\rangle$. Then, the computation can be amortized as $y_{\text{test}} = \boldsymbol{\Phi}(\boldsymbol{x}_{\text{test}})^\top \left(\boldsymbol{\Phi}(\boldsymbol{X})^\top \boldsymbol{\Phi}(\boldsymbol{X}) + \lambda n \boldsymbol{I}_m\right)^{-1} \boldsymbol{\Phi}(\boldsymbol{X})^\top \boldsymbol{y}$ which requires $\mathcal{O}(nm^2 + m^3)$ time to run. When $n \gg m$, this is much efficient than solving the problem with the exact kernel.

We compare our methods to NTK, RBF and RFF. For ours and RFF, we choose the output dimension to $m = 10{,}000$ for all datasets, which is much smaller than the number of data samples n. In Table 2, we report the wall-clock times and mean squared errors (MSE) of test prediction. We face Out-of-Memory errors when running kernel methods using MillionSongs and WorkLoads datasets. Observe that our random features are significantly faster than NTK, e.g., up to ×17 speedup for Protein dataset, without performance loss. We also verify that the NTK features achieve lower MSE than RFF only for Protein but it outperforms with a huge gap for the rest of the datasets.

5.4 Active Learning on MNIST Dataset

We finally apply the proposed method to *active learning* using MNIST dataset. The goal is to select training data of fixed size k that maximizes the performance. Recently, Shoham & Avron (2020) suggested an active learning strategy based on the NTK. They propose a novel criteria that can be an upper bound of statistical risk for general kernel learning and present an algorithm that greedily minimizes this criteria with the NTK. Their greedy process begins with an empty set and iteratively appends singleton that minimizes the proposed risk bound. It takes $\mathcal{O}(n^2k^2)$ time to obtain k data points which equals to the budget size to acquire labels which can be prohibitive if n is large.

Motivated by this, we apply the proposed NTK random features to their greedy algorithm that can improve the running time. Recall that our random features builds an approximation $\boldsymbol{K}' = \boldsymbol{\Phi}\boldsymbol{\Phi}^\top \approx \boldsymbol{K}$ where $\boldsymbol{\Phi} \in \mathbb{R}^{n \times m}$. Under certain parameter regimes, the low rank structure of $\boldsymbol{K}'$ can be used to implement a faster version of the greedy algorithm. Specifically, after $\mathcal{O}(nm^2)$ preprocessing, the cost per iteration of the greedy algorithm can be reduced to $\mathcal{O}(nmj^2)$, and the cost of finding the design of size k to $\mathcal{O}(nm(k^3 + m))$. We provide more details in the supplementary material.

Figure 3 illustrates performance of greedy minimization using our NTK random features compared to randomly generated designs under MNIST dataset. We use a 4-layer fully-connected ReLU network with width 1,000 and the dimension of NTK random features is $m = 10{,}000$. We clearly see that using the random NTK features we can generate much better designs than randomly chosen data points. It justifies that our random features plays a crucial role for active learning tasks. We expect that the proposed method can be applied to various machine learning applications with remarkable performance and computation gains.

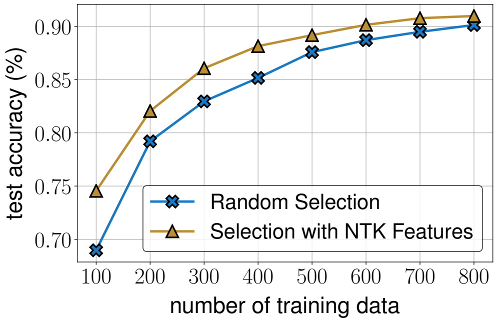


Figure 3: Test accuracy (%) of random selection versus greedy minimization of statistical bound using NTK random features.

6 Conclusion

In this work, we propose an efficient algorithm for generating random features of the Neural Tangent Kernel (NTK). We utilize TensorSketch transform combined with the arc-cosine random features with an importance sampling. We also provide spectral approximation bound to the NTK with layer 2. Our experiments validate the effectiveness of the proposed methods. We believe that our method would be a broad interest both in theoretical and practical domains.

References

Ahle, T. D., Kapralov, M., Knudsen, J. B., Pagh, R., Velingker, A., Woodruff, D. P., and Zandieh, A. Oblivious sketching of high-degree polynomial kernels. In *Symposium on Discrete Algorithms (SODA)*, 2020.

Allen-Zhu, Z., Li, Y., and Song, Z. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning (ICML)*, 2019.

Arora, S., Du, S., Hu, W., Li, Z., and Wang, R. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning (ICML)*, 2019a.

Arora, S., Du, S. S., Hu, W., Li, Z., Salakhutdinov, R. R., and Wang, R. On exact computation with an infinitely wide neural net. In *Neural Information Processing Systems (NeurIPS)*, 2019b.

Arora, S., Du, S. S., Li, Z., Salakhutdinov, R., Wang, R., and Yu, D. Harnessing the Power of Infinitely Wide Deep Nets on Small-data Tasks. In *International Conference on Learning Representations (ICLR)*, 2019c.

Avron, H., Nguyen, H., and Woodruff, D. Subspace embeddings for the polynomial kernel. In *Neural Information Processing Systems (NeurIPS)*, 2014.

Avron, H., Clarkson, K. L., and Woodruff, D. P. Faster Kernel Ridge Regression Using Sketching and Preconditioning. In *SIAM Journal on Matrix Analysis and Applications (SIMAX)*, 2017a.

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 (ICML)*, 2017b.

Bietti, A. and Mairal, J. On the inductive bias of neural tangent kernels. In *Neural Information Processing Systems (NeurIPS)*, 2019.

Bossard, L., Guillaumin, M., and Van Gool, L. Food-101–mining discriminative components with random forests. In *Proceedings of the European Conference on Computer Vision(ECCV)*, 2014.

Cao, Y. and Gu, Q. Generalization bounds of stochastic gradient descent for wide and deep neural networks. In *Neural Information Processing Systems (NeurIPS)*, 2019.

Charikar, M., Chen, K., and Farach-Colton, M. Finding frequent items in data streams. In *International Colloquium on Automata, Languages, and Programming (ICAMP)*, 2002.

Chen, L. and Xu, S. Deep neural tangent kernel and laplace kernel have the same RKHS. In *International Conference on Learning Representations (ICLR)*, 2021.

Chizat, L., Oyallon, E., and Bach, F. On lazy training in differentiable programming. In *Neural Information Processing Systems (NeurIPS)*, 2019.

Cho, Y. and Saul, L. Kernel methods for deep learning. In *Neural Information Processing Systems (NeurIPS)*, 2009.

Daniely, A., Frostig, R., and Singer, Y. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *Neural Information Processing Systems (NeurIPS)*, 2016.

Du, S. S., Zhai, X., Poczos, B., and Singh, A. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations (ICLR)*, 2019.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 2010.

Fan, Z. and Wang, Z. Spectra of the conjugate kernel and neural tangent kernel for linear-width neural networks. In *Neural Information Processing Systems (NeurIPS)*, 2020.

Fei-Fei, L., Fergus, R., and Perona, P. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop*, 2004.

Fernández-Delgado, M., Cernadas, E., Barro, S., and Amorim, D. Do we need hundreds of classifiers to solve real world classification problems? In *Journal of Machine Learning Research (JMLR)*, 2014.

Geifman, A., Yadav, A., Kasten, Y., Galun, M., Jacobs, D., and Basri, R. On the similarity between the laplace and neural tangent kernels. In *Neural Information Processing Systems (NeurIPS)*, 2020.

Goyal, P., Mahajan, D., Gupta, A., and Misra, I. Scaling and benchmarking self-supervised visual representation learning. In *International Conference on Computer Vision (ICCV)*, 2019.

Han, I., Avron, H., and Shin, J. Polynomial Tensor Sketch for Element-wise Function of Low-Rank Matrix. In *International Conference on Machine Learning (ICML)*, 2020.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. In *Neural Information Processing Systems (NeurIPS)*, 2018.

Jacot, A., Simsek, B., Spadaro, F., Hongler, C., and Gabriel, F. Implicit regularization of random feature models. In *International Conference on Machine Learning (ICML)*, 2020.

Khosla, A., Jayadevaprakash, N., Yao, B., and Li, F.-F. Novel dataset for fine-grained image categorization: Stanford dogs. In *Proceedings of CVPR Workshop on Fine-Grained Visual Categorization (FGVC)*, 2011.

Krizhevsky, A. Learning multiple layers of features from tiny images. 2009.

Lee, J., Shen, R., Song, Z., Wang, M., and Yu, Z. Generalized Leverage Score Sampling for Neural Networks. In *Neural Information Processing Systems (NeurIPS)*, 2020.

Mei, S., Montanari, A., and Nguyen, P.-M. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 2018.

Musco, C. and Musco, C. Recursive Sampling for the Nyström Method. In *Neural Information Processing Systems (NeurIPS)*, 2017.

Neyshabur, B., Tomioka, R., and Srebro, N. In search of the real inductive bias: On the role of implicit regularization in deep learning. In *International Conference on Learning Representations (ICLR)*, 2019.

Nilsback, M.-E. and Zisserman, A. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, 2008.

Novak, R., Xiao, L., Hron, J., Lee, J., Alemi, A. A., Sohl-Dickstein, J., and Schoenholz, S. S. Neural tangents: Fast and easy infinite neural networks in python. In *International Conference on Learning Representations (ICLR)*, 2020.

Pennington, J., Yu, F. X. X., and Kumar, S. Spherical random features for polynomial kernels. In *Neural Information Processing Systems (NeurIPS)*, 2015.

Pham, N. and Pagh, R. Fast and scalable polynomial kernels via explicit feature maps. In *Conference on Knowledge Discovery and Data Mining (KDD)*, 2013.

Rahimi, A. and Recht, B. Random Features for Large-Scale Kernel Machines. In *Neural Information Processing Systems (NeurIPS)*, 2009.

Rudi, A. and Rosasco, L. Generalization properties of learning with random features. In *Neural Information Processing Systems (NeurIPS)*, 2016.

Shankar, V., Fang, A., Guo, W., Fridovich-Keil, S., Ragan-Kelley, J., Schmidt, L., and Recht, B. Neural kernels without tangents. In *International Conference on Machine Learning (ICML)*, 2020.

Shoham, N. and Avron, H. Experimental Design for Overparameterized Learning with Application to Single Shot Deep Active Learning. In *arXiv preprint arXiv:2009.12820*, 2020.

Welinder, P., Branson, S., Mita, T., Wah, C., Schroff, F., Belongie, S., and Perona, P. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

Woodruff, D. P. and Zandieh, A. Near Input Sparsity Time Kernel Embeddings via Adaptive Sampling. In *International Conference on Machine Learning (ICML)*, 2020.

Zancato, L., Achille, A., Ravichandran, A., Bhotika, R., and Soatto, S. Predicting training time without training. In *Neural Information Processing Systems (NeurIPS)*, 2020.

A Proof of Theorems

A.1 Proof of Theorem 1

Theorem 1. *Given $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$ such that $\left\|\boldsymbol{x}\right\|_2 = \left\|\boldsymbol{x}'\right\|_2 = 1$ and $L \geq 1$, let $K_{\mathrm{NTK}}^{(L)}$ the NTK of L-layer fully-connected ReLU network. Given $\delta, \in (0,1)$, $\varepsilon \in (0, 1/L)$, there exist constants $C_0, C_1, C_2 > 0$ such that*

$$m_0 \geq C_0 \frac{L^2}{\varepsilon^2} \log\left(\frac{L}{\delta}\right), \quad m_1 \geq C_1 \frac{L^6}{\varepsilon^4} \log\left(\frac{L}{\delta}\right), \quad m_{\mathsf{cs}} \geq C_2 \frac{L^3}{\varepsilon^2 \delta}$$

and

$$\Pr\left(\left|\left\langle \mathbf{\Phi}^{(L)}(\boldsymbol{x}), \mathbf{\Phi}^{(L)}(\boldsymbol{x}')\right\rangle - K_{\mathrm{NTK}}^{(L)}(\boldsymbol{x},\boldsymbol{x}')\right| \le L\varepsilon\left(1+\frac{\varepsilon}{2}\right)^2 + \varepsilon\right) \ge 1-\delta$$

where $\mathbf{\Phi}^{(L)}(\boldsymbol{x}), \mathbf{\Phi}^{(L)}(\boldsymbol{x}') \in \mathbb{R}^{m_1+m_{\mathrm{cs}}}$ *be the output of Algorithm 1 of* $\boldsymbol{x}, \boldsymbol{x}'$*, respectively, using the same* CountSketch *transforms.*

Proof of Theorem 1. For fixed $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{S}^{d-1}$ and $\ell = 0, \dots, L$, we denote the estimate error as

$$\Delta_\ell := \max_{(\boldsymbol{x}_1,\boldsymbol{x}_2)\in\{(\boldsymbol{x},\boldsymbol{x}'),(\boldsymbol{x},\boldsymbol{x}),(\boldsymbol{x}',\boldsymbol{x}')\}} \left|\left\langle \mathbf{\Phi}^{(\ell)}(\boldsymbol{x}_1), \mathbf{\Phi}^{(\ell)}(\boldsymbol{x}_2)\right\rangle - K_{\mathrm{NTK}}^{(\ell)}(\boldsymbol{x}_1,\boldsymbol{x}_2)\right|$$

and note that $\Delta_0 = 0$. Recall that

$$K_{\mathrm{NTK}}^{(\ell)}(\boldsymbol{x},\boldsymbol{x}') = K_{\mathrm{NNGP}}^{(\ell)}(\boldsymbol{x},\boldsymbol{x}') + \dot{K}_{\mathrm{NNGP}}^{(\ell)}(\boldsymbol{x},\boldsymbol{x}') \cdot K_{\mathrm{NTK}}^{(\ell-1)}(\boldsymbol{x},\boldsymbol{x}') \tag{20}$$

where

$$\dot{K}_{\mathrm{NTK}}^{(\ell)}(\boldsymbol{x},\boldsymbol{x}') := 1 - \frac{1}{\pi}\cos^{-1}\left(\frac{K_{\mathrm{NNGP}}^{(\ell-1)}(\boldsymbol{x},\boldsymbol{x}')}{\sqrt{K_{\mathrm{NNGP}}^{(\ell-1)}(\boldsymbol{x},\boldsymbol{x})K_{\mathrm{NNGP}}^{(\ell-1)}(\boldsymbol{x}',\boldsymbol{x}')}}\right) \tag{21}$$

$$\dot{K}_{\mathrm{NNGP}}^{(\ell)}(\boldsymbol{x},\boldsymbol{x}') := f\left(\frac{K_{\mathrm{NNGP}}^{(\ell-1)}(\boldsymbol{x},\boldsymbol{x}')}{\sqrt{K_{\mathrm{NNGP}}^{(\ell-1)}(\boldsymbol{x},\boldsymbol{x}')K_{\mathrm{NNGP}}^{(\ell-1)}(\boldsymbol{x}',\boldsymbol{x}')}}\right) \tag{22}$$

and $f(x) = \frac{1}{\pi}\left(\sqrt{1-x^2} + (\pi - \cos^{-1}(x))x\right)$ for $x \in [-1,1]$.

We use the recursive relation to approximate:

$$\begin{aligned}\left\langle \mathbf{\Phi}^{(\ell)}(\boldsymbol{x}), \mathbf{\Phi}^{(\ell)}(\boldsymbol{x}')\right\rangle &= \left\langle \mathbf{\Psi}^{(\ell)}(\boldsymbol{x}), \mathbf{\Psi}^{(\ell)}(\boldsymbol{x}')\right\rangle + \left\langle \mathbf{\Gamma}^{(\ell)}(\boldsymbol{x}), \mathbf{\Gamma}^{(\ell)}(\boldsymbol{x}')\right\rangle \\ &\approx \left\langle \mathbf{\Psi}^{(\ell)}(\boldsymbol{x}), \mathbf{\Psi}^{(\ell)}(\boldsymbol{x}')\right\rangle + \left\langle \mathbf{\Lambda}^{(\ell)}(\boldsymbol{x}) \otimes \mathbf{\Phi}^{(\ell-1)}(\boldsymbol{x}), \mathbf{\Lambda}^{(\ell)}(\boldsymbol{x}') \otimes \mathbf{\Phi}^{(\ell-1)}(\boldsymbol{x}')\right\rangle \\ &\approx K_{\mathrm{NNGP}}^{(\ell)}(\boldsymbol{x},\boldsymbol{x}') + \dot{K}_{\mathrm{NNGP}}^{(\ell)}(\boldsymbol{x},\boldsymbol{x}') \cdot K_{\mathrm{NTK}}^{(\ell-1)}(\boldsymbol{x},\boldsymbol{x}') = K_{\mathrm{NTK}}^{(\ell)}(\boldsymbol{x},\boldsymbol{x}').\end{aligned}$$

For notational simplicity, we define the following events:

$$\mathcal{E}_{\mathbf{\Psi}}^{(\ell)}(\boldsymbol{x},\boldsymbol{x}',\varepsilon) := \left\{\left|\left\langle \mathbf{\Psi}^{(\ell)}(\boldsymbol{x}), \mathbf{\Psi}^{(\ell)}(\boldsymbol{x}')\right\rangle - K_{\mathrm{NNGP}}^{(\ell)}(\boldsymbol{x},\boldsymbol{x}')\right| \le \varepsilon\right\}, \tag{23}$$

$$\mathcal{E}_{\mathbf{\Lambda}}^{(\ell)}(\boldsymbol{x},\boldsymbol{x}',\varepsilon) := \left\{\left|\left\langle \mathbf{\Lambda}^{(\ell)}(\boldsymbol{x}), \mathbf{\Lambda}^{(\ell)}(\boldsymbol{x}')\right\rangle - \dot{K}_{\mathrm{NNGP}}^{(\ell)}(\boldsymbol{x},\boldsymbol{x}')\right| \le \varepsilon\right\}, \tag{24}$$

$$\mathcal{E}_{\mathbf{\Gamma}}^{(\ell)}(\boldsymbol{x},\boldsymbol{x}',\varepsilon) := \left\{\left|\left\langle \mathbf{\Gamma}^{(\ell)}(\boldsymbol{x}), \mathbf{\Gamma}^{(\ell)}(\boldsymbol{x}')\right\rangle - \left\langle \mathbf{\Lambda}^{(\ell)}(\boldsymbol{x}) \otimes \mathbf{\Phi}^{(\ell-1)}(\boldsymbol{x}), \mathbf{\Lambda}^{(\ell)}(\boldsymbol{x}') \otimes \mathbf{\Phi}^{(\ell-1)}(\boldsymbol{x}')\right\rangle\right| \le \varepsilon\right\} \tag{25}$$

and $\mathcal{E}_{\mathbf{\Omega}}^{(\ell)}(\varepsilon) := \mathcal{E}_{\mathbf{\Omega}}^{(\ell)}(\boldsymbol{x},\boldsymbol{x},\varepsilon) \bigcap \mathcal{E}_{\mathbf{\Omega}}^{(\ell)}(\boldsymbol{x},\boldsymbol{x}',\varepsilon) \bigcap \mathcal{E}_{\mathbf{\Omega}}^{(\ell)}(\boldsymbol{x}',\boldsymbol{x}',\varepsilon)$ for $\mathbf{\Omega} = \{\mathbf{\Psi}, \mathbf{\Lambda}, \mathbf{\Gamma}\}$. Our proof is based on the following claims:

Claim 1. *There exists a constant* $C_1 > 0$ *such that if* $m_1 \ge C_1 \frac{L^6}{\varepsilon^4} \log\left(\frac{L}{\delta}\right)$ *then*

$$\Pr\left(\mathcal{E}_{\mathbf{\Psi}}^{(\ell)}\left(\frac{\varepsilon^2}{32L^2}\right)\right) \ge 1 - \frac{\delta}{3L}.$$

The proof of Claim 1 is provided in Appendix A.1.1.

Claim 2. *There exists a constant $C_0 > 0$ such that if $m_0 \geq C_0 \frac{L^2}{\varepsilon^2} \log\left(\frac{L}{\delta}\right)$ then*

$$\Pr\left(\mathcal{E}_{\mathbf{\Lambda}}^{(\ell)}\left(\frac{3\varepsilon}{8L}\right) \middle| \mathcal{E}_{\mathbf{\Psi}}^{(\ell)}\left(\frac{\varepsilon^2}{32L^2}\right)\right) \geq 1 - \frac{\delta}{3L}.$$

The proof of Claim 2 is provided in Appendix A.1.2.

Claim 3. *There exists a constant $C_2 > 0$ such that if $m_{\mathbf{cs}} \geq C_2 \frac{L^3}{\varepsilon^2 \delta}$ then*

$$\Pr\left(\mathcal{E}_{\mathbf{\Gamma}}^{(\ell)}\left(\frac{\varepsilon}{8L}\left(\ell + \Delta_{\ell-1}\right)\right) \middle| \mathcal{E}_{\mathbf{\Lambda}}^{(\ell)}\left(\frac{3\varepsilon}{8L}\right) \bigcap \mathcal{E}_{\mathbf{\Psi}}^{(\ell)}\left(\frac{\varepsilon^2}{32L^2}\right)\right) \geq 1 - \frac{\delta}{3L}.$$

The proof of Claim 3 is provided in Appendix A.1.3.

Combining Claim 1, Claim 2 and Claim 3, we have

$$\Pr\left(\mathcal{E}_{\mathbf{\Gamma}}^{(\ell)}\left(\frac{\varepsilon}{8L}\left(\ell + \Delta_{\ell}\right)\right) \ \bigcap \ \mathcal{E}_{\mathbf{\Lambda}}^{(\ell)}\left(\frac{3\varepsilon}{8L}\right) \ \bigcap \ \mathcal{E}_{\mathbf{\Psi}}^{(\ell)}\left(\frac{\varepsilon^2}{32L^2}\right)\right) \geq 1 - \frac{\delta}{L}.$$

Next, we claim that the above event can provide the recurrence relation of Δ_ℓ as described in below.

Claim 4. *For $\varepsilon \in (0, 1)$, $L \geq 1$, if event*

$$\mathcal{E}_{\mathbf{\Gamma}}^{(\ell)}\left(\frac{\varepsilon}{8L}\left(\ell + \Delta_{\ell}\right)\right) \ \bigcap \ \mathcal{E}_{\mathbf{\Lambda}}^{(\ell)}\left(\frac{3\varepsilon}{8L}\right) \ \bigcap \ \mathcal{E}_{\mathbf{\Psi}}^{(\ell)}\left(\frac{\varepsilon^2}{32L^2}\right)$$

holds for $\ell \in [L]$ then

$$\Delta_\ell \leq \left(1 + \frac{\varepsilon}{2L}\right)\Delta_{\ell-1} + \frac{\varepsilon}{2L}\ell + \frac{\varepsilon^2}{32L^2}. \tag{26}$$

The proof of Claim 4 is provided in Appendix A.1.4.

Applying union bound on Equation (26) for all $\ell \in [L]$ and solving the recurrence, we obtain that with probabilty at least $1 - \delta$

$$\Delta_L \leq \left(1 + \frac{\varepsilon}{2L} + \frac{\varepsilon^2}{32L^2}\right) \frac{\left(1 + \frac{\varepsilon}{2L}\right)^L - 1}{\frac{\varepsilon}{2L}} - L. \tag{27}$$

When $L = 1$, the statement in Theorem 1 holds since $\Delta_1 \leq \frac{\varepsilon}{2} + \frac{\varepsilon^2}{32} \leq \varepsilon(1 + \frac{\varepsilon}{2})^2 + \varepsilon$ for $\varepsilon \in (0, 1)$. Assume that $L \geq 2$, we obtain

$$\begin{aligned}
\Delta_L &\leq \left(1 + \frac{\varepsilon}{2L} + \frac{\varepsilon^2}{32L^2}\right) L \left(1 + \frac{\varepsilon}{2L}\right)^{L-1} - L && (28)\\
&\leq L\left(\left(1 + \frac{\varepsilon}{L}\right)\left(1 + \frac{\varepsilon}{2L}\right)^{L-1} - 1\right) && (29)\\
&\leq L\left(\left(1 + \frac{\varepsilon}{L}\right)\left(\frac{\varepsilon}{2L}(L-1)\left(1 + \frac{\varepsilon}{2L}\right)^{L-2} + 1\right) - 1\right) && (30)\\
&= L\left(1 + \frac{\varepsilon}{L}\right)\frac{\varepsilon}{2L}(L-1)\left(1 + \frac{\varepsilon}{2L}\right)^{L-2} + \varepsilon && (31)\\
&\leq L\left(1 + \frac{\varepsilon}{2}\right)\frac{\varepsilon}{2}(1 + \varepsilon) + \varepsilon && (32)\\
&\leq L\varepsilon\left(1 + \frac{\varepsilon}{2}\right)^2 + \varepsilon && (33)
\end{aligned}$$

where inequalities in the first and third line are from the fact that $\frac{(1+x)^k-1}{x} \leq kx(1+x)^{k-1}$ for $x > 0, k \geq 1$, the fifth line follows from that

$$\left(1+\frac{\varepsilon}{2L}\right)^{L-2} \leq \exp\left(\frac{\varepsilon}{2L}(L-2)\right) \leq \exp\left(\frac{\varepsilon}{2}\right) \leq 1+\varepsilon.$$

Hence, we conclude that

$$\Pr\left(\Delta_L \leq L\varepsilon\left(1+\frac{\varepsilon}{2}\right)^2+\varepsilon\right) \geq 1-\delta. \tag{34}$$

This completes the proof of Theorem 1. □

A.1.1 Proof of Claim 1

Claim 1. *There exists a constant $C_1 > 0$ such that if $m_1 \geq C_1 \frac{L^6}{\varepsilon^4} \log\left(\frac{L}{\delta}\right)$ then*

$$\Pr\left(\mathcal{E}_{\mathbf{\Psi}}^{(\ell)}\left(\frac{\varepsilon^2}{32L^2}\right)\right) \geq 1-\frac{\delta}{3L}.$$

Proof of Claim 1. Recall that $\mathcal{E}_{\mathbf{\Psi}}^{(\ell)}\left(\frac{\varepsilon^2}{32L^2}\right)$ is equivalent to

$$\left|\left\langle \mathbf{\Psi}^{(\ell)}(\boldsymbol{x}_1), \mathbf{\Psi}^{(\ell)}(\boldsymbol{x}_2)\right\rangle - K_{\text{NNGP}}^{(\ell)}(\boldsymbol{x}_1,\boldsymbol{x}_2)\right| \leq \frac{\varepsilon^2}{32L^2}$$

for $(\boldsymbol{x}_1,\boldsymbol{x}_2) \in \{(\boldsymbol{x},\boldsymbol{x}'),(\boldsymbol{x},\boldsymbol{x}),(\boldsymbol{x}',\boldsymbol{x}')\}$. The proof is directly followed by below lemma.

Lemma 1 (Corollary 16 in (Daniely et al., 2016))**.** *Given $\boldsymbol{x},\boldsymbol{x}' \in \mathbb{R}^d$ such that $\|\boldsymbol{x}\| = \|\boldsymbol{x}'\| = 1$, consider a ReLU network with L layers. For $\delta,\varepsilon \in (0,1)$, there exist constants $C_1, C_2 > 0$ such that $m_1 \geq C_1 \frac{L^2}{\varepsilon_1^2}\log\left(\frac{L}{\delta}\right)$, $\varepsilon_1 \leq \min(c_2, \frac{1}{L})$ and for $(\boldsymbol{x}_1,\boldsymbol{x}_2) \in \{(\boldsymbol{x},\boldsymbol{x}'),(\boldsymbol{x},\boldsymbol{x}),(\boldsymbol{x}',\boldsymbol{x}')\}$ and $\ell \in [L]$ it holds that with probability at least $1-\delta$*

$$\left|\left\langle \mathbf{\Psi}^{(\ell)}(\boldsymbol{x}_1), \mathbf{\Psi}^{(\ell)}(\boldsymbol{x}_2)\right\rangle - K_{\text{NNGP}}^{(\ell)}(\boldsymbol{x}_1,\boldsymbol{x}_2)\right| \leq \varepsilon_1. \tag{35}$$

In Lemma 1, setting $\varepsilon_1 = \frac{\varepsilon^2}{32L^2}$ for $\varepsilon \in (0,1)$ provides the result. This completes the proof of Claim 1. □

A.1.2 Proof of Claim 2

Claim 2. *There exists a constant $C_0 > 0$ such that if $m_0 \geq C_0 \frac{L^2}{\varepsilon^2} \log\left(\frac{L}{\delta}\right)$ then*

$$\Pr\left(\mathcal{E}_{\mathbf{\Lambda}}^{(\ell)}\left(\frac{3\varepsilon}{8L}\right)\middle|\mathcal{E}_{\mathbf{\Psi}}^{(\ell)}\left(\frac{\varepsilon^2}{32L^2}\right)\right) \geq 1-\frac{\delta}{3L}.$$

Proof of Claim 2. Recall that

$$\mathcal{E}_{\mathbf{\Lambda}}^{(\ell)}\left(\frac{3\varepsilon}{8L}\right) = \left\{\left|\left\langle \mathbf{\Lambda}^{(\ell)}(\boldsymbol{x}_1), \mathbf{\Lambda}^{(\ell)}(\boldsymbol{x}_2)\right\rangle - \dot{K}_{\text{NNGP}}^{(\ell)}(\boldsymbol{x}_1,\boldsymbol{x}_2)\right| \leq \frac{3\varepsilon}{8L}\right\}$$
$$\mathcal{E}_{\mathbf{\Psi}}^{(\ell)}\left(\frac{\varepsilon^2}{32L^2}\right) = \left\{\left|\left\langle \mathbf{\Psi}^{(\ell)}(\boldsymbol{x}_1), \mathbf{\Psi}^{(\ell)}(\boldsymbol{x}_2)\right\rangle - K_{\text{NNGP}}^{(\ell)}(\boldsymbol{x}_1,\boldsymbol{x}_2)\right| \leq \frac{\varepsilon^2}{32L^2}\right\}$$

for $(\boldsymbol{x}_1,\boldsymbol{x}_2) \in \{(\boldsymbol{x},\boldsymbol{x}'),(\boldsymbol{x},\boldsymbol{x}),(\boldsymbol{x}',\boldsymbol{x}')\}$. The proof is a direct consequence of the following lemma.

Lemma 2 (Lemma E.5 in (Arora et al., 2019b)). *Given $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^d$, $\ell \in [L]$ and $\varepsilon_2 \in (0,1)$, assume that*

$$\left| \left\langle \boldsymbol{\Psi}^{(\ell)}(\boldsymbol{x}), \boldsymbol{\Psi}^{(\ell)}(\boldsymbol{x}') \right\rangle - K_{\mathrm{NNGP}}^{(\ell)}(\boldsymbol{x}, \boldsymbol{x}') \right| \le \frac{\varepsilon_2^2}{2}. \tag{36}$$

Then, it holds that with probability at least $1 - \delta_2$

$$\left| \left\langle \boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}), \boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}') \right\rangle - \dot{K}_{\mathrm{NNGP}}^{(\ell)}(\boldsymbol{x}, \boldsymbol{x}') \right| \le \varepsilon_2 + \sqrt{\frac{2}{m_0} \log\left(\frac{6}{\delta_2}\right)}. \tag{37}$$

In Lemma 2, we choose $\varepsilon_2 = \frac{\varepsilon}{4L}, \delta_2 = \frac{\delta}{3L}$ and $m_0 \ge \frac{128L^2}{\varepsilon^2} \log\left(\frac{18L}{\delta}\right)$ for $\varepsilon, \delta \in (0,1)$ to obtain that

$$\Pr\left(\left| \left\langle \boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}), \boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}') \right\rangle - \dot{K}_{\mathrm{NNGP}}^{(\ell)}(\boldsymbol{x}, \boldsymbol{x}') \right| \le \frac{3\varepsilon}{8L} \;\middle|\; \left| \left\langle \boldsymbol{\Psi}^{(\ell)}(\boldsymbol{x}), \boldsymbol{\Psi}^{(\ell)}(\boldsymbol{x}') \right\rangle - K_{\mathrm{NNGP}}^{(\ell)}(\boldsymbol{x}, \boldsymbol{x}') \right| \le \frac{\varepsilon^2}{32L^2} \right) \ge 1 - \frac{\delta}{3L}.$$

This completes the proof of Claim 2. □

A.1.3 Proof of Claim 3

Claim 3. *There exists a constant $C_2 > 0$ such that if $m_{\mathsf{cs}} \ge C_2 \frac{L^3}{\varepsilon^2 \delta}$ then*

$$\Pr\left(\mathcal{E}_{\boldsymbol{\Gamma}}^{(\ell)}\left(\frac{\varepsilon}{8L}(\ell + \Delta_{\ell-1})\right) \middle| \mathcal{E}_{\boldsymbol{\Lambda}}^{(\ell)}\left(\frac{3\varepsilon}{8L}\right) \bigcap \mathcal{E}_{\boldsymbol{\Psi}}^{(\ell)}\left(\frac{\varepsilon^2}{32L^2}\right) \right) \ge 1 - \frac{\delta}{3L}.$$

Proof of Claim 3. Recall that $\mathcal{E}_{\boldsymbol{\Gamma}}^{(\ell)}\left(\frac{\varepsilon}{8L}(\ell + \Delta_{\ell-1})\right)$ is equivalent to

$$\left| \left\langle \boldsymbol{\Gamma}^{(\ell)}(\boldsymbol{x}_1), \boldsymbol{\Gamma}^{(\ell)}(\boldsymbol{x}_2) \right\rangle - \left\langle \boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}_1) \otimes \boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_1), \boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}_2) \otimes \boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_2) \right\rangle \right| \le \frac{\varepsilon}{8L}(\ell + \Delta_{\ell-1})$$

for $(\boldsymbol{x}_1, \boldsymbol{x}_2) \in \{(\boldsymbol{x}, \boldsymbol{x}'), (\boldsymbol{x}, \boldsymbol{x}), (\boldsymbol{x}', \boldsymbol{x}')\}$. The proof is based on the following lemma that provides an upper bound on variance of the CountSketch transform.

Lemma 3. *Given $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^m$ and $\boldsymbol{y}, \boldsymbol{y}' \in \mathbb{R}^{m'}$, let $\mathcal{C}_1 : \mathbb{R}^m \to \mathbb{R}^{m_{\mathsf{cs}}}, \mathcal{C}_2 : \mathbb{R}^{m'} \to \mathbb{R}^{m_{\mathsf{cs}}}$ be two independent* CountSketch *transforms for some $m_{\mathsf{cs}} > 0$. Denote*

$$\boldsymbol{\Gamma} := \mathrm{FFT}^{-1}\left(\mathrm{FFT}\left(\mathcal{C}_1(\boldsymbol{x})\right) \odot \mathrm{FFT}\left(\mathcal{C}_2(\boldsymbol{y})\right)\right), \quad \boldsymbol{\Gamma}' := \mathrm{FFT}^{-1}\left(\mathrm{FFT}\left(\mathcal{C}_1(\boldsymbol{x}')\right) \odot \mathrm{FFT}\left(\mathcal{C}_2(\boldsymbol{y}')\right)\right). \tag{38}$$

Then, it holds that with probability at least $1 - \delta_3$

$$\left| \langle \boldsymbol{\Gamma}, \boldsymbol{\Gamma}' \rangle - \langle \boldsymbol{x} \otimes \boldsymbol{y}, \boldsymbol{x}' \otimes \boldsymbol{y}' \rangle \right| \le \sqrt{\frac{11}{\delta_3 m_{\mathsf{cs}}}} \left\| \boldsymbol{x} \right\|_2 \left\| \boldsymbol{x}' \right\|_2 \left\| \boldsymbol{y} \right\|_2 \left\| \boldsymbol{y}' \right\|_2. \tag{39}$$

The proof of Lemma 3 is provided in Appendix B.1. In Lemma 3, we choose $\delta_3 = \frac{\delta}{3L}$, $m_{\mathsf{cs}} \ge \frac{33L(8L+3\varepsilon)^2}{\varepsilon^2 \delta}$ for $\varepsilon, \delta \in (0,1)$ to satisfies that

$$\sqrt{\frac{11}{\delta_3 m_{\mathsf{cs}}}} \le \frac{\frac{\varepsilon}{8L}}{1 + \frac{3\varepsilon}{8L}}. \tag{40}$$

Then, with probability at least $1 - \frac{\delta}{3L}$ we have

$$\left| \left\langle \boldsymbol{\Gamma}^{(\ell)}(\boldsymbol{x}_1), \boldsymbol{\Gamma}^{(\ell)}(\boldsymbol{x}_2) \right\rangle - \left\langle \boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}_1) \otimes \boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_1), \boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}_2) \otimes \boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_2) \right\rangle \right| \tag{41}$$

$$\le \frac{\frac{\varepsilon}{8L}}{1 + \frac{3\varepsilon}{8L}} \left\| \boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}_1) \right\|_2 \left\| \boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}_2) \right\|_2 \left\| \boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_1) \right\|_2 \left\| \boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_2) \right\|_2 \tag{42}$$

$$\le \frac{\frac{\varepsilon}{8L}}{1 + \frac{3\varepsilon}{8L}} \left(1 + \frac{3\varepsilon}{8L}\right) \left\| \boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_1) \right\|_2 \left\| \boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_2) \right\|_2 \tag{43}$$

$$\le \frac{\frac{\varepsilon}{8L}}{1 + \frac{3\varepsilon}{8L}} \left(1 + \frac{3\varepsilon}{8L}\right) (\ell + \Delta_{\ell-1}) = \frac{\varepsilon}{8L} (\ell + \Delta_{\ell-1}) \tag{44}$$

where the second inequlity holds from the fact that $\mathcal{E}_{\boldsymbol{\Lambda}}^{(\ell)}\left(\frac{3\varepsilon}{8L}\right)$ implies that for $\boldsymbol{x}' \in \{\boldsymbol{x}_1, \boldsymbol{x}_2\}$

$$\left\|\boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}')\right\|_2^2 \leq \dot{K}_{\mathrm{NNGP}}^{(\ell-1)}(\boldsymbol{x}', \boldsymbol{x}') + \frac{3\varepsilon}{8L} = 1 + \frac{3\varepsilon}{8L} \tag{45}$$

since $\dot{K}_{\mathrm{NNGP}}^{(\ell-1)}(\boldsymbol{x}', \boldsymbol{x}') = 1$ and the third one follows from that $K_{\mathrm{NTK}}^{(\ell-1)}(\boldsymbol{x}, \boldsymbol{y}) \leq \ell$ for $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{S}^{d-1}, \ell \geq 1$ and

$$\left\|\boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}')\right\|_2^2 \leq K_{\mathrm{NTK}}^{(\ell-1)}(\boldsymbol{x}', \boldsymbol{x}') + \Delta_{\ell-1} \leq \ell + \Delta_{\ell-1}. \tag{46}$$

This completes the proof of Claim 3. □

A.1.4 Proof of Claim 4

Claim 4. *For $\varepsilon \in (0,1)$, $L \geq 1$, if event*

$$\mathcal{E}_{\boldsymbol{\Gamma}}^{(\ell)}\left(\frac{\varepsilon}{8L}(\ell + \Delta_\ell)\right) \bigcap \mathcal{E}_{\boldsymbol{\Lambda}}^{(\ell)}\left(\frac{3\varepsilon}{8L}\right) \bigcap \mathcal{E}_{\boldsymbol{\Psi}}^{(\ell)}\left(\frac{\varepsilon^2}{32L^2}\right)$$

holds for $\ell \in [L]$ then

$$\Delta_\ell \leq \left(1 + \frac{\varepsilon}{2L}\right)\Delta_{\ell-1} + \frac{\varepsilon}{2L}\ell + \frac{\varepsilon^2}{32L^2}. \tag{26}$$

Proof of Claim 4. Recall that

$$\Delta_\ell := \max_{(\boldsymbol{x}_1, \boldsymbol{x}_2) \in \{(\boldsymbol{x}, \boldsymbol{x}'), (\boldsymbol{x}, \boldsymbol{x}), (\boldsymbol{x}', \boldsymbol{x}')\}} \left|\left\langle \boldsymbol{\Phi}^{(\ell)}(\boldsymbol{x}_1), \boldsymbol{\Phi}^{(\ell)}(\boldsymbol{x}_2)\right\rangle - K_{\mathrm{NTK}}^{(\ell)}(\boldsymbol{x}_1, \boldsymbol{x}_2)\right|.$$

Observe that the estimate error Δ_ℓ can be decomposed into three parts:

$$\begin{aligned}\left|\left\langle \boldsymbol{\Phi}^{(\ell)}(\boldsymbol{x}_1), \boldsymbol{\Phi}^{(\ell)}(\boldsymbol{x}_2)\right\rangle - K_{\mathrm{NTK}}^{(\ell)}(\boldsymbol{x}_1, \boldsymbol{x}_2)\right| &\leq \left|\left\langle \boldsymbol{\Psi}^{(\ell)}(\boldsymbol{x}_1), \boldsymbol{\Psi}^{(\ell)}(\boldsymbol{x}_2)\right\rangle - K_{\mathrm{NNGP}}^{(\ell)}(\boldsymbol{x}_1, \boldsymbol{x}_2)\right| \\ &+ \left|\left\langle \boldsymbol{\Gamma}^{(\ell)}(\boldsymbol{x}_1), \boldsymbol{\Gamma}^{(\ell)}(\boldsymbol{x}_2)\right\rangle - \left\langle \boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}_1) \otimes \boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_1), \boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}_2) \otimes \boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_2)\right\rangle\right| \\ &+ \left|\left\langle \boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}_1), \boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}_2)\right\rangle \left\langle \boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_1), \boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_2)\right\rangle - \dot{K}_{\mathrm{NNGP}}^{(\ell)}(\boldsymbol{x}_1, \boldsymbol{x}_2) K_{\mathrm{NTK}}^{(\ell-1)}(\boldsymbol{x}_1, \boldsymbol{x}_2)\right|.\end{aligned} \tag{47}$$

for $(\boldsymbol{x}_1, \boldsymbol{x}_2) \in \{(\boldsymbol{x}, \boldsymbol{x}'), (\boldsymbol{x}, \boldsymbol{x}), (\boldsymbol{x}', \boldsymbol{x}')\}$. By definition, the event $\mathcal{E}_{\boldsymbol{\Psi}}^{(\ell)}\left(\frac{\varepsilon^2}{32L^2}\right)$ implies that

$$\left|\left\langle \boldsymbol{\Phi}^{(\ell)}(\boldsymbol{x}_1), \boldsymbol{\Phi}^{(\ell)}(\boldsymbol{x}_1)\right\rangle - \boldsymbol{\Theta}^{(\ell)}(\boldsymbol{x}_1, \boldsymbol{x}_2)\right| \leq \frac{\varepsilon^2}{32L^2} \tag{48}$$

and the event $\mathcal{E}_{\boldsymbol{\Gamma}}^{(\ell)}\left(\frac{\varepsilon}{8L}(\ell + \Delta_\ell)\right)$ implies that

$$\left|\left\langle \boldsymbol{\Gamma}^{(\ell)}(\boldsymbol{x}_1), \boldsymbol{\Gamma}^{(\ell)}(\boldsymbol{x}_2)\right\rangle - \left\langle \boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}_1) \otimes \boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_1), \boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}_2) \otimes \boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_2)\right\rangle\right| \leq \frac{\varepsilon}{8L}(\ell + \Delta_\ell). \tag{49}$$

For the third part in Equation (47), we observe that

$$\begin{aligned}&\left|\left\langle \boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}_1), \boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}_2)\right\rangle \left\langle \boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_1), \boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_2)\right\rangle - \dot{K}_{\mathrm{NNGP}}^{(\ell)}(\boldsymbol{x}_1, \boldsymbol{x}_2) K_{\mathrm{NTK}}^{(\ell-1)}(\boldsymbol{x}_1, \boldsymbol{x}_2)\right| \\ &\leq \left|\left\langle \boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_1), \boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_2)\right\rangle\right| \left|\left\langle \boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}_1), \boldsymbol{\Lambda}^{(\ell)}(\boldsymbol{x}_2)\right\rangle - \dot{K}_{\mathrm{NNGP}}^{(\ell)}(\boldsymbol{x}_1, \boldsymbol{x}_2)\right| \\ &\qquad + \left|\dot{K}_{\mathrm{NNGP}}^{(\ell)}(\boldsymbol{x}_1, \boldsymbol{x}_2)\right| \left|\left\langle \boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_1), \boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_2)\right\rangle - K_{\mathrm{NTK}}^{(\ell-1)}(\boldsymbol{x}_1, \boldsymbol{x}_2)\right| \\ &\leq \left\|\boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_1)\right\|_2 \left\|\boldsymbol{\Phi}^{(\ell-1)}(\boldsymbol{x}_2)\right\|_2 \frac{3\varepsilon}{8L} + \Delta_{\ell-1} \\ &\leq (\ell + \Delta_{\ell-1})\frac{3\varepsilon}{8L} + \Delta_{\ell-1}\end{aligned} \tag{50}$$

where the second inequality comes from that the event $\mathcal{E}^{(\ell)}_{\boldsymbol{\Lambda}}\left(\frac{3\varepsilon}{8L}\right)$ and $\left|\dot{K}^{(\ell)}_{\mathrm{NNGP}}(\boldsymbol{x}_1, \boldsymbol{x}_2)\right| \leq 1$ and the last one follows from Equation (46). Putting Equation (48), Equation (49) and Equation (50) into Equation (47), we have

$$\Delta_\ell \leq \frac{\varepsilon^2}{32L^2} + \frac{\varepsilon}{8L}\left(\ell + \Delta_\ell\right) + \left(\ell + \Delta_{\ell-1}\right)\frac{3\varepsilon}{8L} + \Delta_{\ell-1} = \left(1 + \frac{\varepsilon}{2L}\right)\Delta_{\ell-1} + \frac{\varepsilon}{2L}\ell + \frac{\varepsilon^2}{32L^2}. \tag{51}$$

This completes the proof of Claim 4. □

A.2 Proof of Theorem 2

The proofs here rely on Theorem 3.3 in (Lee et al., 2020) which states spectral approximation bounds of random features for general kernels equipped with the leverage score sampling. This result is a generalization of (Avron et al., 2017b) working on the Random Fourier Features.

Theorem 5 (Theorem 3.3 in (Lee et al., 2020))**.** *Suppose $\boldsymbol{K} \in \mathbb{R}^{n \times n}$ is a kernel matrix with statistical dimension s_λ for some $\lambda \in (0, \|\boldsymbol{K}\|_2)$. Let $\mathbf{\Phi}(\boldsymbol{w}) \in \mathbb{R}^n$ be a feature map with a random vector $\boldsymbol{w} \sim p(\boldsymbol{w})$ satisfying that $\boldsymbol{K} = \mathbb{E}_{\boldsymbol{w}}\left[\mathbf{\Phi}(\boldsymbol{w})\mathbf{\Phi}(\boldsymbol{w})^\top\right]$. Define $\tau_\lambda(\boldsymbol{w}) := p(\boldsymbol{w}) \cdot \mathbf{\Phi}(\boldsymbol{w})^\top(\boldsymbol{K} + \lambda\boldsymbol{I})^{-1}\mathbf{\Phi}(\boldsymbol{w})$. Let $\widetilde{\tau}(\boldsymbol{w})$ be any measurable function such that $\widetilde{\tau}(\boldsymbol{w}) \geq \tau_\lambda(\boldsymbol{w})$ for all $\boldsymbol{w}$. Assume that $s_{\widetilde{\tau}} := \int \widetilde{\tau}(\boldsymbol{w})d\boldsymbol{w}$ is finite. Consider random vectors $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_m$ sampled from $q(\boldsymbol{w}) := \widetilde{\tau}(\boldsymbol{w})/s_{\widetilde{\tau}}$ and define that*

$$\overline{\mathbf{\Phi}} := \frac{1}{\sqrt{m}}\left[\sqrt{\frac{p(\boldsymbol{w}_1)}{q(\boldsymbol{w}_1)}}\mathbf{\Phi}(\boldsymbol{w}_1), \ \ldots \ , \sqrt{\frac{p(\boldsymbol{w}_m)}{q(\boldsymbol{w}_m)}}\mathbf{\Phi}(\boldsymbol{w}_m)\right]. \tag{52}$$

If $m \geq \frac{8}{3}\varepsilon^{-2}s_{\widetilde{\tau}}\log\left(16 s_\lambda/\delta\right)$ then

$$(1-\varepsilon)\left(\boldsymbol{K} + \lambda\boldsymbol{I}\right) \preceq \overline{\mathbf{\Phi}\mathbf{\Phi}}^\top + \lambda\boldsymbol{I} \preceq (1+\varepsilon)\left(\boldsymbol{K} + \lambda\boldsymbol{I}\right) \tag{53}$$

holds with probability at least $1-\delta$.

We now ready to provide proofs of Theorem 2.

Theorem 2. *Given $\boldsymbol{X} \in \mathbb{R}^{n \times d}$, let $\boldsymbol{A}_0 \in \mathbb{R}^{n \times n}$ be the arc-cosine kernel matrix of 0-th order with $\boldsymbol{X}$ and denote $\mathbf{\Phi}_0 := \sqrt{\frac{2}{m}}\mathrm{Step}(\boldsymbol{X}\boldsymbol{W}) \in \mathbb{R}^{n \times m}$ where each entry in $\boldsymbol{W} \in \mathbb{R}^{d \times m}$ is an i.i.d. sample from $\mathcal{N}(0,1)$. Let s_λ be the statistical dimension of $\boldsymbol{A}_0$. Given $\lambda \in (0, \|\boldsymbol{A}_0\|_2)$, $\varepsilon \in (0, 1/2)$ and $\delta \in (0,1)$, if $m \geq \frac{8}{3}\frac{n}{\lambda\varepsilon^2}\log\left(\frac{16 s_\lambda}{\delta}\right)$, then it holds*

$$(1-\varepsilon)(\boldsymbol{A}_0 + \lambda\boldsymbol{I}) \preceq \mathbf{\Phi}_0\mathbf{\Phi}_0^\top + \lambda\boldsymbol{I} \preceq (1+\varepsilon)(\boldsymbol{A}_0 + \lambda\boldsymbol{I})$$

with probability at least $1-\delta$.

Proof of Theorem 2. Let $\mathbf{\Phi}_0(\boldsymbol{w}) := \sqrt{2}\ \mathrm{Step}(\boldsymbol{X}\boldsymbol{w}) \in \mathbb{R}^n$ for $\boldsymbol{w} \in \mathbb{R}^d$ and $p(\boldsymbol{w})$ be the probability density function of the standard normal distribution. As studied in (Cho & Saul, 2009), $\mathbf{\Phi}_0(\boldsymbol{w})$ is a random feature of $\boldsymbol{A}_0$ such that

$$\boldsymbol{A}_0 = \mathbb{E}_{\boldsymbol{w} \sim p(\boldsymbol{w})}\left[\mathbf{\Phi}_0(\boldsymbol{w})\mathbf{\Phi}_0(\boldsymbol{w})^\top\right]. \tag{54}$$

In order to utilize Theorem 5, we need an upper bound of $\tau_\lambda(\boldsymbol{w})$ as below:

$$\begin{aligned}
\tau_\lambda(\boldsymbol{w}) &:= p(\boldsymbol{w}) \cdot \mathbf{\Phi}_0(\boldsymbol{w})^\top\left(\boldsymbol{A}_0 + \lambda\boldsymbol{I}\right)^{-1}\mathbf{\Phi}_0(\boldsymbol{w}) && (55)\\
&\leq p(\boldsymbol{w})\left\|\left(\boldsymbol{A}_0 + \lambda\boldsymbol{I}\right)^{-1}\right\|_2\|\mathbf{\Phi}_0(\boldsymbol{w})\|_2^2 && (56)\\
&\leq p(\boldsymbol{w})\frac{\|\mathbf{\Phi}_0(\boldsymbol{w})\|_2^2}{\lambda} && (57)\\
&\leq p(\boldsymbol{w})\frac{2n}{\lambda} && (58)
\end{aligned}$$

where the inequality in second line holds from the definition of matrix operator norm and the inequality in third line follows from the fact that smallest eigenvalue of $\boldsymbol{A}_0 + \lambda \boldsymbol{I}$ is equal to or greater than λ. The last inequality is from that $\|\mathrm{Step}(\boldsymbol{x})\|_2^2 \le n$ for any $\boldsymbol{x} \in \mathbb{R}^n$. Note that $\int_{\mathbb{R}^d} p(\boldsymbol{w}) \frac{2n}{\lambda} d\boldsymbol{w} = \frac{2n}{\lambda}$ and since it is constant the modified random features correspond to the original ones. Putting all together into Theorem 5, we can obtain the result. This completes the proof of Theorem 2. □

A.3 Proof of Theorem 3

Theorem 3. *Given $\boldsymbol{X} \in \mathbb{R}^{n \times d}$, let $\boldsymbol{A}_1 \in \mathbb{R}^{n \times n}$ be the arc-cosine kernel matrix of 1-th order with $\boldsymbol{X}$ and $\boldsymbol{v}_1, \dots, \boldsymbol{v}_m \in \mathbb{R}^d$ be i.i.d. random vectors from probability distribution*

$$q(\boldsymbol{v}) = \frac{1}{(2\pi)^{d/2} d} \|\boldsymbol{v}\|_2^2 \exp\left(-\frac{1}{2}\|\boldsymbol{v}\|_2^2\right). \tag{15}$$

Denote

$$\boldsymbol{\Phi}_1 := \sqrt{\frac{2d}{m}} \left[\frac{\mathrm{ReLU}(\boldsymbol{X}\boldsymbol{v}_1)}{\|\boldsymbol{v}_1\|_2}, \dots, \frac{\mathrm{ReLU}(\boldsymbol{X}\boldsymbol{v}_m)}{\|\boldsymbol{v}_m\|_2}\right] \tag{16}$$

and let s_λ be the statistical dimension of $\boldsymbol{A}_1$. Given $\lambda \in (0, \|\boldsymbol{A}_1\|_2)$, $\varepsilon \in (0, 1/2)$ and $\delta \in (0,1)$, if $m \ge \frac{8}{3}\frac{d\|\boldsymbol{X}\|_2^2}{\lambda\varepsilon^2} \log\left(\frac{16 s_\lambda}{\delta}\right)$, then it holds that

$$(1-\varepsilon)(\boldsymbol{A}_1 + \lambda I) \preceq \boldsymbol{\Phi}_1\boldsymbol{\Phi}_1^\top + \lambda \boldsymbol{I} \preceq (1+\varepsilon)(\boldsymbol{A}_1 + \lambda I)$$

with probability at least $1-\delta$.

Proof of Theorem 3. Let $\boldsymbol{\Phi}_1(\boldsymbol{w}) := \sqrt{2}\ \mathrm{ReLU}(\boldsymbol{X}\boldsymbol{w}) \in \mathbb{R}^n$ for $\boldsymbol{w} \in \mathbb{R}^d$ and $p(\boldsymbol{w})$ be the probability density function of standard normal distribution. Cho & Saul (2009) also showed that $\boldsymbol{\Phi}_1(\boldsymbol{w})$ is a random feature of $\boldsymbol{A}_1$ such that

$$\boldsymbol{A}_1 = \mathbb{E}_{\boldsymbol{w}\sim p(\boldsymbol{w})}\left[\boldsymbol{\Phi}_1(\boldsymbol{w})\boldsymbol{\Phi}_1(\boldsymbol{w})^\top\right]. \tag{59}$$

Again, we use the below upper bound as follow:

$$\begin{aligned}
\tau_\lambda(\boldsymbol{w}) &:= p(\boldsymbol{w}) \cdot \boldsymbol{\Phi}_1(\boldsymbol{w})^\top (\boldsymbol{A}_1 + \lambda \boldsymbol{I})^{-1} \boldsymbol{\Phi}_1(\boldsymbol{w}) && (60)\\
&\le p(\boldsymbol{w}) \left\|(\boldsymbol{A}_1 + \lambda \boldsymbol{I})^{-1}\right\|_2 \|\boldsymbol{\Phi}_1(\boldsymbol{w})\|_2^2 && (61)\\
&= 2\ p(\boldsymbol{w}) \frac{\|\mathrm{ReLU}(\boldsymbol{X}\boldsymbol{w})\|_2^2}{\lambda} && (62)\\
&\le 2\ p(\boldsymbol{w}) \frac{\|\boldsymbol{X}\boldsymbol{w}\|_2^2}{\lambda} && (63)\\
&\le 2\ p(\boldsymbol{w}) \|\boldsymbol{w}\|_2^2 \frac{\|\boldsymbol{X}\|_2^2}{\lambda} && (64)
\end{aligned}$$

where the inequality in fourth line holds from that $\|\mathrm{ReLU}(\boldsymbol{x})\|_2^2 \le \|\boldsymbol{x}\|_2^2$ for any vector $\boldsymbol{x}$. Denote $\widetilde{\tau}(\boldsymbol{w}) := 2\ p(\boldsymbol{w}) \|\boldsymbol{w}\|_2^2 \frac{\|\boldsymbol{X}\|_2^2}{\lambda}$ and it holds that

$$\int_{\mathbb{R}^d} \widetilde{\tau}(\boldsymbol{w}) d\boldsymbol{w} = \int_{\mathbb{R}^d} 2\ p(\boldsymbol{w}) \|\boldsymbol{w}\|_2^2 \frac{\|\boldsymbol{X}\|_2^2}{\lambda} d\boldsymbol{w} = 2d \frac{\|\boldsymbol{X}\|_2^2}{\lambda} \tag{65}$$

since $\int_{\mathbb{R}^d} p(\boldsymbol{w}) \|\boldsymbol{w}\|_2^2 = \mathtt{tr}(\boldsymbol{I}_d) = d$ for $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$. We define the modified distribution as

$$q(\boldsymbol{w}) := \frac{\widetilde{\tau}(\boldsymbol{w})}{\int_{\mathbb{R}^d} \widetilde{\tau}(\boldsymbol{w}) d\boldsymbol{w}} = p(\boldsymbol{w}) \frac{\|\boldsymbol{w}\|_2^2}{d} = \frac{1}{(2\pi)^{d/2} d} \|\boldsymbol{w}\|_2^2 \exp\left(-\frac{1}{2}\|\boldsymbol{w}\|_2^2\right) \tag{66}$$

and recall the modified random features as

$$\boldsymbol{\Phi}_1 = \frac{1}{\sqrt{m}} \left[\sqrt{\frac{p(\boldsymbol{w}_1)}{q(\boldsymbol{w}_1)}}\ \boldsymbol{\Phi}_1(\boldsymbol{w}_1),\ \dots\ , \sqrt{\frac{p(\boldsymbol{w}_m)}{q(\boldsymbol{w}_m)}}\ \boldsymbol{\Phi}_1(\boldsymbol{w}_m) \right] \tag{67}$$

$$= \sqrt{\frac{2d}{m}} \left[\frac{\mathrm{ReLU}(\boldsymbol{X}\boldsymbol{w}_1)}{\|\boldsymbol{w}_1\|_2},\ \dots\ , \frac{\mathrm{ReLU}(\boldsymbol{X}\boldsymbol{w}_m)}{\|\boldsymbol{w}_m\|_2} \right]. \tag{68}$$

Putting all together into Theorem 5, we derive the result. This completes the proof of Theorem 3. □

A.4 Proof of Theorem 4

Before diving into detailed algorithmic analysis, we introduce spectral approximation bounds of CountSketch when it applies to Hadamard product of two PSD matrices. Recall that the CountSketch plays a key role for reducing the feature map dimensionality and below theorem is used in the proof of Theorem 4.

Lemma 4. *Given $\boldsymbol{X} \in \mathbb{R}^{n \times d_1}$ and $\boldsymbol{Y} \in \mathbb{R}^{n \times d_2}$, let $\mathcal{C}_1, \mathcal{C}_2$ be the two independent* CountSketch *transforms from $\mathbb{R}^{d_1}, \mathbb{R}^{d_2}$ to $\mathbb{R}^m$, respectively. Denote that*

$$\boldsymbol{\Gamma} := \mathrm{FFT}^{-1}(\mathrm{FFT}(\mathcal{C}_1(\boldsymbol{X})) \odot \mathrm{FFT}(\mathcal{C}_2(\boldsymbol{Y}))). \tag{69}$$

Given $\varepsilon, \delta \in (0,1)$, $\lambda \geq 0$ and $m \geq \frac{11}{\varepsilon^2 \delta} \left(\frac{\mathtt{tr}(\boldsymbol{X}\boldsymbol{X}^\top \odot \boldsymbol{Y}\boldsymbol{Y}^\top)}{\mathtt{tr}(\boldsymbol{X}\boldsymbol{X}^\top \odot \boldsymbol{Y}\boldsymbol{Y}^\top)/n + \lambda} \right)^2$, then it holds

$$(1-\varepsilon)\left(\boldsymbol{X}\boldsymbol{X}^\top \odot \boldsymbol{Y}\boldsymbol{Y}^\top + \lambda \boldsymbol{I}\right) \preceq \boldsymbol{\Gamma}\boldsymbol{\Gamma}^\top + \lambda \boldsymbol{I} \preceq (1+\varepsilon)\left(\boldsymbol{X}\boldsymbol{X}^\top \odot \boldsymbol{Y}\boldsymbol{Y}^\top + \lambda \boldsymbol{I}\right) \tag{70}$$

with probability at least $1-\delta$.

The proof of Lemma 4 is provided in Appendix B.2.

Theorem 4. *Given $\boldsymbol{X} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_n]^\top \in \mathbb{R}^{n \times d}$, assume that $\|\boldsymbol{x}_i\|_2 = 1$ for $i \in [n]$. Let $\boldsymbol{K}_{\mathrm{NTK}}$ be the NTK of two-layer ReLU network, i.e., $L = 1$ in Equation (1), and $\boldsymbol{A}_0, \boldsymbol{A}_1$ denote the arc-cosine kernels of 0-th, 1-st order with $\boldsymbol{X}$, respectively, as in Equation (6). For any $\lambda \in (0, 2\min(\|\boldsymbol{A}_0\|_2, \|\boldsymbol{A}_1\|_2)]$, suppose s_λ is an upper bound of statistical dimensions of both $\boldsymbol{A}_0, \boldsymbol{A}_1$. Given $\varepsilon \in (0, 1/2)$, $\delta \in (0,1)$, let $\boldsymbol{\Phi} \in \mathbb{R}^{n \times (m_1 + m_{\mathtt{cs}})}$ be the first output of Algorithm 1 with $L = 1$ and*

$$m_0 \geq \frac{48n}{\varepsilon^2 \lambda} \log\left(\frac{48 s_\lambda}{\delta}\right), \quad m_1 \geq \frac{16}{3} \frac{d\,\|\boldsymbol{X}\|_2^2}{\varepsilon^2 \lambda} \log\left(\frac{48 s_\lambda}{\delta}\right), \quad m_{\mathtt{cs}} \geq \frac{297}{\varepsilon^2 \delta} \left(\frac{n}{\lambda + 1}\right)^2$$

Then, with probability at least $1-\delta$, it holds that

$$(1-\varepsilon)\left(\boldsymbol{K}_{\mathrm{NTK}} + \lambda \boldsymbol{I}\right) \preceq \boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \lambda \boldsymbol{I} \preceq (1+\varepsilon)\left(\boldsymbol{K}_{\mathrm{NTK}} + \lambda \boldsymbol{I}\right). \tag{18}$$

Proof of Theorem 4. Note that the NTK of two-layer ReLU network can be formulated as

$$\boldsymbol{K}_{\mathrm{NTK}} = \boldsymbol{A}_1 + \boldsymbol{A}_0 \odot (\boldsymbol{X}\boldsymbol{X}^\top) \tag{71}$$

where $\boldsymbol{A}_0$ and $\boldsymbol{A}_1$ are the arc-cosine kernel matrices of order 0 and 1 with $\boldsymbol{X}$, respectively. Let $\boldsymbol{\Phi}_0$ and $\boldsymbol{\Phi}_1$ be the random features of $\boldsymbol{A}_0$ and $\boldsymbol{A}_1$, respectively, satisfying that $\boldsymbol{A}_0 = \mathbb{E}\left[\boldsymbol{\Phi}_0\boldsymbol{\Phi}_0^\top\right]$ and $\boldsymbol{A}_1 = \mathbb{E}\left[\boldsymbol{\Phi}_1\boldsymbol{\Phi}_1^\top\right]$. Based on the property of CountSketch , one can check that $\boldsymbol{\Phi}_0\boldsymbol{\Phi}_0^\top \odot \boldsymbol{X}\boldsymbol{X}^\top = \mathbb{E}[\boldsymbol{\Gamma}\boldsymbol{\Gamma}^\top]$ where we recall that

$$\boldsymbol{\Gamma} := \mathrm{FFT}^{-1}(\mathrm{FFT}(\mathcal{C}_1(\boldsymbol{\Phi}_0)) \odot \mathrm{FFT}(\mathcal{C}_2(\boldsymbol{X}))). \tag{72}$$

Our proof is a combination of spectral analysis of $\mathbf{\Phi}_0\mathbf{\Phi}_0^\top, \mathbf{\Phi}_1\mathbf{\Phi}_1^\top$ and $\mathbf{\Gamma}\mathbf{\Gamma}^\top$ which are stated in Theorem 2, Theorem 3 and Lemma 4, respectively.

From Theorem 3, if $m_1 \geq \frac{16}{3}\frac{d\|\boldsymbol{X}\|_2^2}{\lambda\varepsilon^2}\log\left(\frac{48 s_\lambda}{\delta}\right)$ then with probability at least $1-\frac{\delta}{3}$ it holds

$$(1-\varepsilon)\left(\boldsymbol{A}_1+\frac{\lambda}{2}\boldsymbol{I}\right) \preceq \mathbf{\Phi}_1\mathbf{\Phi}_1^\top+\frac{\lambda}{2}\boldsymbol{I} \preceq (1+\varepsilon)\left(\boldsymbol{A}_1+\frac{\lambda}{2}\boldsymbol{I}\right). \tag{73}$$

From Theorem 2, if $m_0 \geq 48\frac{n}{\lambda\varepsilon^2}\log\left(\frac{48 s_\lambda}{\delta}\right)$ then with probability at least $1-\frac{\delta}{3}$ it holds

$$\left(1-\frac{\varepsilon}{3}\right)\left(\boldsymbol{A}_0+\frac{\lambda}{2}\boldsymbol{I}\right) \preceq \mathbf{\Phi}_0\mathbf{\Phi}_0^\top+\frac{\lambda}{2}\boldsymbol{I} \preceq \left(1+\frac{\varepsilon}{3}\right)\left(\boldsymbol{A}_0+\frac{\lambda}{2}\boldsymbol{I}\right) \tag{74}$$

Rearranging Equation (74), we get

$$\mathbf{\Phi}_0\mathbf{\Phi}_0^\top \preceq \left(1+\frac{\varepsilon}{3}\right)\boldsymbol{A}_0+\frac{\varepsilon}{6}\lambda\boldsymbol{I}. \tag{75}$$

To guarantee spectral approximation of $\mathbf{\Gamma}\mathbf{\Gamma}^\top$, we will use the result of Lemma 4. Before applying it, we provide an upper bound of the trace of $\mathbf{\Phi}_0\mathbf{\Phi}_0^\top \odot \boldsymbol{X}\boldsymbol{X}^\top$. Consider $\mathbf{\Phi}_0 = \sqrt{\frac{2}{m_0}}\left[\mathrm{Step}(\boldsymbol{Z}\boldsymbol{w}_1),\ldots,\mathrm{Step}(\boldsymbol{Z}\boldsymbol{w}_{m_0})\right]$ for some $\boldsymbol{Z}=[\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n] \in \mathbb{R}^{n\times d}$. Then, we have

$$\begin{aligned}
\mathtt{tr}\left(\mathbf{\Phi}_0\mathbf{\Phi}_0^\top \odot \boldsymbol{X}\boldsymbol{X}^\top\right) &= \sum_{j=1}^n \left[\mathbf{\Phi}_0\mathbf{\Phi}_0^\top\right]_{jj}\cdot\left[\boldsymbol{X}\boldsymbol{X}^\top\right]_{jj} && (76)\\
&= \sum_{j=1}^n \left(\frac{2}{m_0}\sum_{i=1}^{m_0}\mathrm{Step}\left(\langle \boldsymbol{z}_j,\boldsymbol{w}_i\rangle\right)^2\right)\cdot\|\boldsymbol{x}_j\|_2^2 && (77)\\
&\leq \sum_{j=1}^n 1\cdot\|\boldsymbol{x}_j\|_2^2 = n && (78)
\end{aligned}$$

where the inequality in third line holds from that $\mathrm{Step}(x) \leq 1$ for any $x \in \mathbb{R}$ and the last equality follows from the assumption that $\|\boldsymbol{x}_j\|_2 = 1$ for all $j \in [n]$.

Hence, using Lemma 4 with the fact that $m_{\mathtt{cs}} \geq \frac{297}{\varepsilon^2\delta}\left(\frac{n}{1+\lambda}\right)^2 \geq \frac{297}{\varepsilon^2\delta}\left(\frac{\mathtt{tr}(\mathbf{\Phi}_0\mathbf{\Phi}_0^\top\odot\boldsymbol{X}\boldsymbol{X}^\top)}{\mathtt{tr}(\mathbf{\Phi}_0\mathbf{\Phi}_0^\top\odot\boldsymbol{X}\boldsymbol{X}^\top)/n+\lambda}\right)^2$, we have

$$\left(1+\frac{\varepsilon}{3}\right)\left(\mathbf{\Phi}_0\mathbf{\Phi}_0^\top\odot\boldsymbol{X}\boldsymbol{X}^\top+\frac{\lambda}{2}\boldsymbol{I}\right) \preceq \mathbf{\Gamma}\mathbf{\Gamma}^\top+\frac{\lambda}{2}\boldsymbol{I} \preceq \left(1+\frac{\varepsilon}{3}\right)\left(\mathbf{\Phi}_0\mathbf{\Phi}_0^\top\odot\boldsymbol{X}\boldsymbol{X}^\top+\frac{\lambda}{2}\boldsymbol{I}\right) \tag{79}$$

with probability at least $1-\frac{\delta}{3}$. Combining Equation (79) with Equation (75), with probability at least $1-\frac{2}{3}\delta$, we get that

$$\begin{aligned}
\mathbf{\Gamma}\mathbf{\Gamma}^\top+\frac{\lambda}{2}\boldsymbol{I} &\preceq \left(1+\frac{\varepsilon}{3}\right)\left(\mathbf{\Phi}_0\mathbf{\Phi}_0^\top\odot\boldsymbol{X}\boldsymbol{X}^\top+\frac{\lambda}{2}I\right) && (80)\\
&\preceq \left(1+\frac{\varepsilon}{3}\right)\left(\left[\left(1+\frac{\varepsilon}{3}\right)\boldsymbol{A}_0+\frac{\varepsilon}{6}\lambda\boldsymbol{I}\right]\odot\boldsymbol{X}\boldsymbol{X}^\top+\frac{\lambda}{2}\boldsymbol{I}\right) && (81)\\
&= \left(1+\frac{\varepsilon}{3}\right)\left(\left(1+\frac{\varepsilon}{3}\right)\left(\boldsymbol{A}_0\odot\boldsymbol{X}\boldsymbol{X}^\top\right)+\frac{\varepsilon}{6}\lambda(\boldsymbol{I}\odot\boldsymbol{X}\boldsymbol{X}^\top)+\frac{\lambda}{2}\boldsymbol{I}\right) && (82)\\
&= \left(1+\frac{\varepsilon}{3}\right)\left(1+\frac{\varepsilon}{3}\right)\left(\boldsymbol{A}_0\odot\boldsymbol{X}\boldsymbol{X}^\top+\frac{\lambda}{2}\boldsymbol{I}\right) && (83)\\
&\preceq (1+\varepsilon)\left(\boldsymbol{A}_0\odot\boldsymbol{X}\boldsymbol{X}^\top+\frac{\lambda}{2}\boldsymbol{I}\right) && (84)
\end{aligned}$$

where the equality in second line follows from the fact that $\boldsymbol{A} \odot \boldsymbol{C} \preceq \boldsymbol{B} \odot \boldsymbol{C}$ holds if $\boldsymbol{A} \preceq \boldsymbol{B}$ for positive semidefinite matrices $\boldsymbol{A}, \boldsymbol{B}$ and $\boldsymbol{C}$ [5] and the fourth equality is from the assumption $\left\|\boldsymbol{x}_i\right\|_2 = 1$ for all $i \in [n]$ which leads that $\boldsymbol{I} \odot (\boldsymbol{X}\boldsymbol{X}^\top) = \boldsymbol{I}$. The last inequality holds since $\varepsilon \in (0, 1/2)$.

Similarly, we can obtain the following lower bound:

$$\boldsymbol{\Gamma}\boldsymbol{\Gamma}^\top + \frac{\lambda}{2}\boldsymbol{I} \succeq (1-\varepsilon)\left(\boldsymbol{A}_0 \odot \boldsymbol{X}\boldsymbol{X}^\top + \frac{\lambda}{2}\boldsymbol{I}\right). \tag{85}$$

Combining Equation (73), Equation (84) with Equation (85) gives

$$(1-\varepsilon)\left(\boldsymbol{A}_1 + \boldsymbol{A}_0 \odot \boldsymbol{X}\boldsymbol{X}^\top + \lambda\boldsymbol{I}\right) \preceq \boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \lambda\boldsymbol{I} \preceq (1+\varepsilon)\left(\boldsymbol{A}_1 + \boldsymbol{A}_0 \odot \boldsymbol{X}\boldsymbol{X}^\top + \lambda\boldsymbol{I}\right). \tag{86}$$

Furthermore, by taking a union bound over all events, Equation (86) holds with probability at least $1-\delta$. This completes the proof of Theorem 4. □

B Proof of Lemmas

The proofs of Lemma 3 and Lemma 4 are obtained from Lemma 2 in (Avron et al., 2014) that provides an upper bound on variance of TensorSketch transform of order $q \geq 2$.

Lemma 5 (Lemma 2 in (Avron et al., 2014)). *For $q \geq 2$, consider q of 3-wise independent hash functions $h_1, \dots h_q : [d] \to [m]$ and q of 4-wise independent sign functions $s_1, \dots, s_q : [d] \to \{+1, -1\}$. Define the hash function $H : [d^q] \to [m]$ and the sign function $S : [d^q] \to \{-1, +1\}$ such that*

$$\begin{aligned} H(j) &\equiv h_1(i_1) + h_2(i_2) + \cdots + h_q(i_q) \pmod{m}, \\ S(j) &= s_1(i_1) \cdot s_2(i_2) \cdot \cdots \cdot s_q(i_q). \end{aligned}$$

where $j \in [d^q]$ and $i_1, \dots, i_q \in [d]$ such that $j = i_q d^{q-1} + \cdots + i_2 d + i_1$. Denote sketch matrix $\boldsymbol{S} \in \{-1, 0, +1\}^{d^q \times m}$ satisfying that $\boldsymbol{S}_{j,H(j)} = S(j)$ for $j \in [d^q]$ and other entries are set to zero. For any $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n \times d^q}$, it holds

$$\mathbb{E}_{\boldsymbol{S}}\left[\left\|\boldsymbol{A}\boldsymbol{S}\boldsymbol{S}^\top\boldsymbol{B}^\top - \boldsymbol{A}\boldsymbol{B}^\top\right\|_F^2\right] \leq \frac{(2+3^q)}{m}\left\|\boldsymbol{A}\right\|_F^2 \left\|\boldsymbol{B}\right\|_F^2. \tag{87}$$

B.1 Proof of Lemma 3

Lemma 3. *Given $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^m$ and $\boldsymbol{y}, \boldsymbol{y}' \in \mathbb{R}^{m'}$, let $\mathcal{C}_1 : \mathbb{R}^m \to \mathbb{R}^{m_{\mathsf{cs}}}, \mathcal{C}_2 : \mathbb{R}^{m'} \to \mathbb{R}^{m_{\mathsf{cs}}}$ be two independent* CountSketch *transforms for some $m_{\mathsf{cs}} > 0$. Denote*

$$\boldsymbol{\Gamma} := \mathrm{FFT}^{-1}\left(\mathrm{FFT}\left(\mathcal{C}_1(\boldsymbol{x})\right) \odot \mathrm{FFT}\left(\mathcal{C}_2(\boldsymbol{y})\right)\right), \quad \boldsymbol{\Gamma}' := \mathrm{FFT}^{-1}\left(\mathrm{FFT}\left(\mathcal{C}_1(\boldsymbol{x}')\right) \odot \mathrm{FFT}\left(\mathcal{C}_2(\boldsymbol{y}')\right)\right). \tag{38}$$

Then, it holds that with probability at least $1-\delta_3$

$$\left|\langle \boldsymbol{\Gamma}, \boldsymbol{\Gamma}' \rangle - \langle \boldsymbol{x} \otimes \boldsymbol{y}, \boldsymbol{x}' \otimes \boldsymbol{y}' \rangle\right| \leq \sqrt{\frac{11}{\delta_3 m_{\mathsf{cs}}}} \left\|\boldsymbol{x}\right\|_2 \left\|\boldsymbol{x}'\right\|_2 \left\|\boldsymbol{y}\right\|_2 \left\|\boldsymbol{y}'\right\|_2. \tag{39}$$

Proof of Lemma 3. By the Markov's inequality, we have

$$\Pr\left(\left|\langle \boldsymbol{\Gamma}, \boldsymbol{\Gamma}' \rangle - \langle \boldsymbol{x} \otimes \boldsymbol{y}, \boldsymbol{x}' \otimes \boldsymbol{y}' \rangle\right| \geq \varepsilon\right) \leq \frac{1}{\varepsilon^2}\mathbb{E}\left[\left|\langle \boldsymbol{x} \otimes \boldsymbol{y}, \boldsymbol{x}' \otimes \boldsymbol{y}' \rangle - \langle \boldsymbol{\Gamma}, \boldsymbol{\Gamma}' \rangle\right|^2\right] \leq \frac{11 \left\|\boldsymbol{x}\right\|_2^2 \left\|\boldsymbol{x}'\right\|_2^2 \left\|\boldsymbol{y}\right\|_2^2 \left\|\boldsymbol{y}'\right\|_2^2}{\varepsilon^2 m_{\mathsf{cs}}} \tag{88}$$

where the last inequality follows from Lemma 5 with $q = 2$. This completes the proof of Lemma 3. □

[5] It is enough to show that $(\boldsymbol{B} - \boldsymbol{A}) \odot \boldsymbol{C} \succeq 0$. Since $\boldsymbol{B} - \boldsymbol{A}, \boldsymbol{C} \succeq 0$, this holds from the Schur product theorem.

B.2 Proof of Lemma 4

Lemma 4. *Given* $\boldsymbol{X} \in \mathbb{R}^{n\times d_1}$ *and* $\boldsymbol{Y} \in \mathbb{R}^{n \times d_2}$*, let* $\mathcal{C}_1, \mathcal{C}_2$ *be the two independent* CountSketch *transforms from* $\mathbb{R}^{d_1}, \mathbb{R}^{d_2}$ *to* $\mathbb{R}^m$*, respectively. Denote that*

$$\boldsymbol{\Gamma} := \mathrm{FFT}^{-1}(\mathrm{FFT}(\mathcal{C}_1(\boldsymbol{X})) \odot \mathrm{FFT}(\mathcal{C}_2(\boldsymbol{Y}))). \tag{69}$$

Given $\varepsilon, \delta \in (0,1)$*,* $\lambda \geq 0$ *and* $m \geq \frac{11}{\varepsilon^2\delta}\left(\frac{\mathtt{tr}(\boldsymbol{X}\boldsymbol{X}^\top \odot \boldsymbol{Y}\boldsymbol{Y}^\top)}{\mathtt{tr}(\boldsymbol{X}\boldsymbol{X}^\top \odot \boldsymbol{Y}\boldsymbol{Y}^\top)/n+\lambda}\right)^2$*, then it holds*

$$(1-\varepsilon)\left(\boldsymbol{X}\boldsymbol{X}^\top \odot \boldsymbol{Y}\boldsymbol{Y}^\top + \lambda \boldsymbol{I}\right) \preceq \boldsymbol{\Gamma}\boldsymbol{\Gamma}^\top + \lambda\boldsymbol{I} \preceq (1+\varepsilon)\left(\boldsymbol{X}\boldsymbol{X}^\top \odot \boldsymbol{Y}\boldsymbol{Y}^\top + \lambda \boldsymbol{I}\right) \tag{70}$$

with probability at least $1-\delta$*.*

Proof of Lemma 4. Let $s_1 : [d_1] \to \{-1,+1\}$ and $h_1 : [d_1] \to [m]$ be the random sign and hash function of $\mathcal{C}_1$, respectively. Similarly, denote s_2 and h_2 by that of $\mathcal{C}_2$, respectively. Then, $\boldsymbol{\Gamma}$ is the output of Count Sketch $\mathcal{C} : \mathbb{R}^{d_1 d_2} \to \mathbb{R}^m$ applying to $\boldsymbol{X} \otimes \boldsymbol{Y}$ whose sign and hash functions are defined as

$$s(i,j) = s_1(i) \cdot s_2(j) \tag{89}$$

$$h(i,j) \equiv h_1(i) + h_2(j) \pmod{m} \tag{90}$$

for $i \in [d_1], j \in [d_2]$. Here, index (i,j) can be considered as some $k \in [d_1 d_2]$ by transforming $i = \lfloor k/d_2 \rfloor$ and $j \equiv k \pmod{d_2}$.

Let $\boldsymbol{S} \in \{-1,0,+1\}^{d_1 d_2 \times m}$ be the sketch matrix of $\mathcal{C}$ and we write $\boldsymbol{Z} := \boldsymbol{X} \otimes \boldsymbol{Y}$ for notational simplicity. As shown in (Avron et al., 2014), it is easy to check that $\boldsymbol{\Gamma} = \boldsymbol{Z}\boldsymbol{S}$ and we have

$$\begin{aligned}
\mathbb{E}[\boldsymbol{\Gamma}\boldsymbol{\Gamma}^\top] &= \mathbb{E}[(\boldsymbol{X}\otimes\boldsymbol{Y})\boldsymbol{S}\boldsymbol{S}^\top(\boldsymbol{X}\otimes\boldsymbol{Y})^\top] &(91)\\
&= (\boldsymbol{X}\otimes\boldsymbol{Y})\mathbb{E}[\boldsymbol{S}\boldsymbol{S}^\top](\boldsymbol{X}\otimes\boldsymbol{Y})^\top &(92)\\
&= (\boldsymbol{X}\otimes\boldsymbol{Y})(\boldsymbol{X}\otimes\boldsymbol{Y})^\top &(93)\\
&= \boldsymbol{X}\boldsymbol{X}^\top \odot \boldsymbol{Y}\boldsymbol{Y}^\top = \boldsymbol{Z}\boldsymbol{Z}^\top. &(94)
\end{aligned}$$

Rearranging Equation (70), we have

$$-\varepsilon\left(\boldsymbol{Z}\boldsymbol{Z}^\top + \lambda\boldsymbol{I}\right) \preceq \boldsymbol{Z}\boldsymbol{S}\boldsymbol{S}^\top\boldsymbol{Z}^\top - \boldsymbol{Z}\boldsymbol{Z}^\top \preceq \varepsilon\left(\boldsymbol{Z}\boldsymbol{Z}^\top + \lambda\boldsymbol{I}\right). \tag{95}$$

By multiplying $(\boldsymbol{Z}\boldsymbol{Z}^\top + \lambda\boldsymbol{I})^{-1/2}$ by both left and right sides in (95), it is enough to show that

$$\left\|(\boldsymbol{Z}\boldsymbol{Z}^\top + \lambda\boldsymbol{I})^{-1/2}\boldsymbol{Z}\boldsymbol{S}\boldsymbol{S}^\top\boldsymbol{Z}^\top(\boldsymbol{Z}\boldsymbol{Z}^\top + \lambda\boldsymbol{I})^{-1/2} - (\boldsymbol{Z}\boldsymbol{Z}^\top + \lambda\boldsymbol{I})^{-1/2}\boldsymbol{Z}\boldsymbol{Z}^\top(\boldsymbol{Z}\boldsymbol{Z}^\top + \lambda\boldsymbol{I})^{-1/2}\right\|_2 \leq \varepsilon$$

By denoting $\boldsymbol{A} := (\boldsymbol{Z}\boldsymbol{Z}^\top + \lambda\boldsymbol{I})^{-1/2}\boldsymbol{Z}$, it is equivalent to prove that

$$\left\|\boldsymbol{A}\boldsymbol{S}\boldsymbol{S}^\top\boldsymbol{A}^\top - \boldsymbol{A}\boldsymbol{A}^\top\right\|_2 \leq \varepsilon. \tag{96}$$

By Markov's inequality, we have

$$\begin{aligned}
\Pr\left[\left\|\boldsymbol{A}\boldsymbol{S}\boldsymbol{S}^\top\boldsymbol{A}^\top - \boldsymbol{A}\boldsymbol{A}^\top\right\|_2 \geq \varepsilon\right] &\leq \Pr\left[\left\|\boldsymbol{A}\boldsymbol{S}\boldsymbol{S}^\top\boldsymbol{A}^\top - \boldsymbol{A}\boldsymbol{A}^\top\right\|_F \geq \varepsilon\right]\\
&\leq \varepsilon^{-2}\mathbb{E}\left[\left\|\boldsymbol{A}\boldsymbol{S}\boldsymbol{S}^\top\boldsymbol{A}^\top - \boldsymbol{A}\boldsymbol{A}^\top\right\|_F^2\right]\\
&\leq \frac{11}{\varepsilon^2 m}\left\|\boldsymbol{A}\right\|_F^4\\
&= \frac{11}{\varepsilon^2 m}\left[\mathtt{tr}(\boldsymbol{A}^\top\boldsymbol{A})\right]^2\\
&= \frac{11}{\varepsilon^2 m}\left[\mathtt{tr}\left((\boldsymbol{Z}\boldsymbol{Z}^\top + \boldsymbol{I})^{-1}\boldsymbol{Z}\boldsymbol{Z}^\top\right)\right]^2\\
&\leq \frac{11}{\varepsilon^2 m}\left(\frac{\mathtt{tr}(\boldsymbol{X}\boldsymbol{X}^\top \odot \boldsymbol{Y}\boldsymbol{Y}^\top)}{\mathtt{tr}(\boldsymbol{X}\boldsymbol{X}^\top \odot \boldsymbol{Y}\boldsymbol{Y}^\top)/n+\lambda}\right)^2
\end{aligned}$$

where the inequality in third line holds from Lemma 5 with $q = 2$ and the last inequality follows from Lemma 6. Taking $m \geq \frac{11}{\varepsilon^2 \delta} \left(\frac{\mathtt{tr}(\boldsymbol{X}\boldsymbol{X}^\top \odot \boldsymbol{Y}\boldsymbol{Y}^\top)}{\mathtt{tr}(\boldsymbol{X}\boldsymbol{X}^\top \odot \boldsymbol{Y}\boldsymbol{Y}^\top)/n + \lambda} \right)^2$, Equation (96) holds with probability at least $1 - \delta$. This completes the proof of Lemma 4. □

Lemma 6. *Let $\boldsymbol{A}, \boldsymbol{B}$ be positive semidefinite matrices and let s_λ be the statistical dimension of $\boldsymbol{A} \odot \boldsymbol{B}$ for any $\lambda > 0$, i.e., $s_\lambda := \mathtt{tr}\left((\boldsymbol{A} \odot \boldsymbol{B})(\boldsymbol{A} \odot \boldsymbol{B} + \lambda \boldsymbol{I})^{-1}\right)$. Then, it holds that $s_\lambda \leq \frac{\mathtt{tr}(\boldsymbol{A} \odot \boldsymbol{B})}{\mathtt{tr}(\boldsymbol{A} \odot \boldsymbol{B})/n + \lambda}$.*

Proof of Lemma 6. By the Schur product theorem, $\boldsymbol{A} \odot \boldsymbol{B}$ is a positive semidefinite matrix. Let $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n \geq 0$ be the eigenvalues of $\boldsymbol{A} \odot \boldsymbol{B}$. By the definition of statistical dimension, it holds that

$$s_\lambda = \mathtt{tr}\left((\boldsymbol{A} \odot \boldsymbol{B}) (\boldsymbol{A} \odot \boldsymbol{B} + \lambda \boldsymbol{I})^{-1} \right) = \sum_{i=1}^{n} \frac{\lambda_i}{\lambda_i + \lambda} \tag{97}$$

$$\leq n \frac{\left(\sum_i \lambda_i\right)/n}{\left(\sum_i \lambda_i\right)/n + \lambda} = \frac{\sum_i \lambda_i}{(\sum_i \lambda_i)/n + \lambda} = \frac{\mathtt{tr}(\boldsymbol{A} \odot \boldsymbol{B})}{\mathtt{tr}(\boldsymbol{A} \odot \boldsymbol{B})/n + \lambda} \tag{98}$$

where the inequality holds from the Jensen's inequality. This completes the proof of Lemma 6. □

C Mathematical Details on Deep Active Learning

In this section, we give additional details on how NTK random features can be used to accelerate the greedy selection algorithm in (Shoham & Avron, 2020). In actuality, the improvement is not specific to NTK features, but works for every kernel for which have a low-rank factorization. It applies to NTK random features by virtue of the low-rank factorization the approximate kernel induces.

For a matrix $\boldsymbol{A}$ and index sets $\mathcal{S}$ and $\mathcal{T}$, let $\boldsymbol{A}_{\mathcal{S},\mathcal{T}}$ denote the matrix obtained by restricting to the rows whose index is in $\mathcal{S}$ and the columns whose index is in $\mathcal{T}$. Using : as the index set denotes the entire relevant index set. Consider kernel ridge regression, and assume the kernel matrix is $\boldsymbol{K}$. One important variant of the criteria developed in (Shoham & Avron, 2020) is the minimization of

$$J(\mathcal{S}) = \mathtt{tr}(-\boldsymbol{K}_{:,\mathcal{S}}(\boldsymbol{K}_{\mathcal{S},\mathcal{S}} + \lambda \boldsymbol{I}_{|\mathcal{S}|})^{-1} \boldsymbol{K}_{:,\mathcal{S}}^\top)$$

In order to perform greedy minimization of $J(\mathcal{S})$ we need to be able to evaluate $J(\mathcal{S})$ quickly for a given $\mathcal{S}$.

Assume that $\boldsymbol{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^\top$ where $\boldsymbol{\Phi}$ has m columns. We now show how after preprocessing $J(\mathcal{S})$ can be computed in $\mathcal{O}(|\mathcal{S}|^2 m)$ time as long as $|\mathcal{S}| \leq m$. First, notice that

$$\begin{aligned} J(\mathcal{S}) &= \mathtt{tr}(-\boldsymbol{K}_{:,\mathcal{S}}(\boldsymbol{K}_{\mathcal{S},\mathcal{S}} + \lambda \boldsymbol{I}_{|\mathcal{S}|})^{-1} \boldsymbol{K}_{:,\mathcal{S}}^\top) \\ &= \mathtt{tr}(-(\boldsymbol{K}_{\mathcal{S},\mathcal{S}} + \lambda \boldsymbol{I}_{|\mathcal{S}|})^{-1} \boldsymbol{K}_{:,\mathcal{S}}^\top \boldsymbol{K}_{:,\mathcal{S}}) \\ &= \mathtt{tr}(-\underbrace{(\boldsymbol{K}_{\mathcal{S},\mathcal{S}} + \lambda \boldsymbol{I}_{|\mathcal{S}|})^{-1} \boldsymbol{\Phi}_{\mathcal{S},:}}_{\boldsymbol{A}(\mathcal{S})} \cdot \underbrace{\boldsymbol{\Phi}^\top \boldsymbol{\Phi} \boldsymbol{\Phi}_{\mathcal{S},:}^\top}_{\boldsymbol{B}(\mathcal{S})}) \end{aligned}$$

Now, notice that $\boldsymbol{A}(\mathcal{S})$ can be computed in $\mathcal{O}(|\mathcal{S}|^2 m)$ time if we assume that $|\mathcal{S}| \leq m$. As for $\boldsymbol{B}(\mathcal{S})$, this matrix consists exactly of the columns in $\mathcal{S}$ of $\boldsymbol{\Phi}^\top \boldsymbol{\Phi} \boldsymbol{\Phi}^\top$. To take advantage of that we precompute $\boldsymbol{\Phi}^\top \boldsymbol{\Phi} \boldsymbol{\Phi}^\top$ in $\mathcal{O}(nm^2)$. Finally, note that we are only interested in the trace of $\boldsymbol{A}(\mathcal{S}) \cdot \boldsymbol{B}(\mathcal{S})$. There is no need to compute the entire product; we can compute only the diagonal elements. We see that after the $\mathcal{O}(nm^2)$ preprocessing, we can compute $J(\mathcal{S})$ in $\mathcal{O}(|\mathcal{S}|^2 m)$ time.

In order to greedily minimize $J(\mathcal{S})$, we start with $\mathcal{S} = \emptyset$, and and add at each iteration the index that will minimize $J(\mathcal{S})$. To do so, we scan the entire index set, evaluating $J(\mathcal{T})$ for each candidate $\mathcal{T}$ that consists of the current $\mathcal{S}$ and the addition index. Since there are n data-points, the scan takes $\mathcal{O}(n|\mathcal{S}|^2 m)$ which is the cost per iteration. Including preprocessing time, the cost of finding a design of size k is $\mathcal{O}(nm(k^3 + m))$.

D Additional Experiments on Image Classification

D.1 Classification on Image Datasets

We additionally conduct experiments on image classification using 8 fine-grained datasets: `CIFAR`-10/100 (Krizhevsky, 2009), `VOC07` (Everingham et al., 2010), `Caltech`-101 (Fei-Fei et al., 2004), `CUB`-200 (Welinder et al., 2010), `Dog`-120 (Khosla et al., 2011), `Flower`-102 (Nilsback & Zisserman, 2008), `Food`-101 (Bossard et al., 2014). In particular, we follow the transfer learning mechanism (Goyal et al., 2019) where we extract image features from the penultimate layer of the pretrained ResNet18 (He et al., 2016) with dimension $d = 512$. These features are then leveraged as inputs to be transformed to random features. In particular, we follow low-shot setting used in (Arora et al., 2019c); we randomly choose 5 image data from each class of training set, and use the whole test set for evaluation. We repeat sampling training images 50 times for `CIFAR`-10 and `VOC07`, 10 times for other datasets. This is because these datasets have relatively small classes, i.e., 10 and 20, respectively.

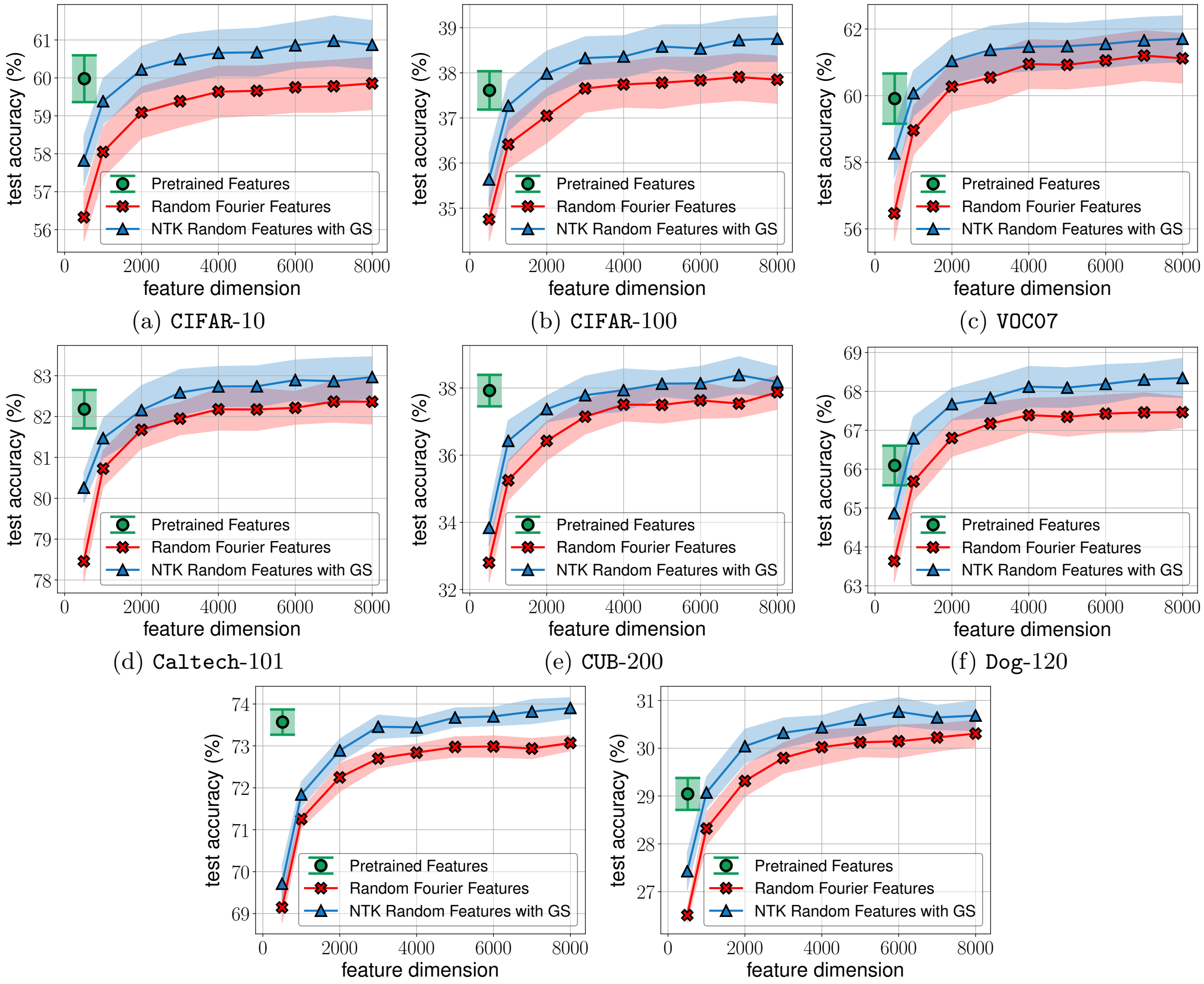
We run Algorithm 1 with and without Gibbs sampling (GS) (i.e., Algorithm 2). The number of Gibbs iteration is set to $T = 1$. The output dimension m is fixed to 4,000 and CountSketch dimension is considered $m_{\texttt{cs}}$ as a hyperparameter. We set $m_0 = m_1 = m - m_{\texttt{cs}}$ and choose $m_{\texttt{cs}} \in \{0, \frac{m}{10}, \dots, \frac{9m}{10}, m\}$ for the best validation accuracy. We normalize the output of Algorithm 1 so that the corresponding features lie in $\mathbb{S}^{d-1}$. We also benchmark the Random Fourier Features (RFF) with the same dimension $m = 4{,}000$. Once features are generated, we train a linear classifier with SGD optimizer for 500 epochs where inputs can be the pretrained features or random features. We perform grid search for finding the best learning rate in $\{0.01, 0.02, 0.1, 0.2\}$ and momentum in $\{0, 0.9, 0.99\}$ and report the best test accuracy among 12 different setups. We also execute various classifiers including AdaBoost, random forest, k-nearest neighbors and support vector classifier (SVC). For methods running with SVC, we search the cost value C in $\{2^{-19}, \dots, 2^{-3}, 10^{-7}, \dots, 10^{3}\}$ and choose the best one that achieves the globally maximum accuracy. Hence, a single cost value is globally used for all iterations. The number of network depth for NTK is chosen by $L = 2$. For RBF kernel (i.e., $K(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\gamma \left\| \boldsymbol{x} - \boldsymbol{y} \right\|_2^2)$), we choose the best γ in $\{\frac{1}{d}, \frac{1}{d\sigma}\}$ that achieves the globally maximum accuracy where σ is the variance of training data. For k-nearest neighbor classifier, we search the best k in $\{1, 2, \dots, \min(20, n/2)\}$ where n is the number of training data. This allows the number of training instance per class to be roughly larger than 2. For AdaBoost and Random Forest, we search the number of ensembles in $\{50, 100, 200, 500\}$.

In Table 3, the average test accuracy with 95% confidence interval is reported. Observe that NTK and the corresponding random features show better performance than other competitors for all datasets. These observations match with the previous result (Arora et al., 2019c) that the NTK can outperform on small-scale datasets. We additionally verify that NTK random features can perform similar or even better than NTK for most datasets, especially with Gibbs sampling. Such performance gaps are also observed between the Random Fourier Features and RBF kernel. This is likely due to the fact that random features has an implicit regularization effect which can lead to better generalization ability.

We also investigate the effect of feature dimension m to image classification performance. In Figure 4, we plot the test accuracy of the proposed NTK random features with Gibbs sampling (blue, triangle), Random Fourier Features (red, cross) and features from the pretrained ResNet-18 (green, circle) when feature dimension m changes from 500 to 8,000. Note that the pretrained features has a fixed dimension $d = 512$. The hyperparameters are chosen by the same approach as described above. We observe that a larger m can lead to higher test accuracy for both ours and RFF. It suffices to set $m = 4{,}000$ for the NTK features to achieve higher accuracy than the pretrained features. We also verify that the proposed NTK random features shows better performance than RFF for the same dimension m. This justifies that ours is more effective for fine-grained image classifications.

Table 3: Results of average test accuracy for image classification using features from the pretrained ResNet-18. We measure the 95% confidence interval across 50 iterations for `CIFAR-10` and `VOC07`, and 10 iterations for the rest. **Bold entries** indicate the best mean accuracy for each dataset.

Method	`CIFAR-10`	`CIFAR-100`	`VOC07`	`Caltech-101`	`CUB-200`	`Dog-120`	`Flower-102`	`Food-101`
Linear Classifier	59.97 ± 0.61	37.61 ± 0.42	59.91 ± 0.75	82.41 ± 0.39	37.92 ± 0.46	66.09 ± 0.51	73.56 ± 0.30	29.04 ± 0.33
k-Nearest Neighbors	47.64 ± 0.77	26.09 ± 0.50	50.32 ± 0.89	73.37 ± 0.46	24.78 ± 0.48	54.33 ± 0.59	56.36 ± 0.49	18.59 ± 0.43
AdaBoost	32.09 ± 0.90	9.56 ± 0.57	31.56 ± 0.99	26.41 ± 2.34	7.01 ± 0.87	22.72 ± 1.95	20.87 ± 1.81	6.20 ± 0.45
Random Forest	56.79 ± 0.63	30.09 ± 0.43	56.16 ± 0.75	75.70 ± 0.74	29.71 ± 0.42	61.31 ± 0.45	64.49 ± 0.37	22.47 ± 0.39
Linear SVM	58.59 ± 0.63	35.74 ± 0.48	60.63 ± 0.79	81.39 ± 0.55	36.68 ± 0.33	66.32 ± 0.49	70.94 ± 0.25	28.44 ± 0.37
RBF Kernel SVM	59.16 ± 0.63	36.42 ± 0.54	60.79 ± 0.76	82.14 ± 0.45	36.67 ± 0.44	66.49 ± 0.57	71.41 ± 0.14	29.03 ± 0.34
Random Fourier Features	59.63 ± 0.68	37.74 ± 0.50	60.95 ± 0.73	82.17 ± 0.51	37.50 ± 0.48	67.38 ± 0.43	72.83 ± 0.20	30.02 ± 0.36
NTK SVM	60.48 ± 0.60	37.53 ± 0.55	61.19 ± 0.70	**82.83 ± 0.41**	37.95 ± 0.40	67.72 ± 0.49	72.32 ± 0.23	29.63 ± 0.37
NTK Random Features	60.63 ± 0.62	**38.53 ± 0.50**	61.44 ± 0.67	82.65 ± 0.53	38.11 ± 0.56	68.06 ± 0.54	73.62 ± 0.27	30.49 ± 0.34
NTK Random Features with GS	**60.66 ± 0.60**	38.49 ± 0.51	**61.48 ± 0.68**	82.67 ± 0.51	**38.22 ± 0.51**	**68.07 ± 0.55**	**73.67 ± 0.28**	**30.50 ± 0.32**



(a) `CIFAR-10`

(b) `CIFAR-100`

(c) `VOC07`

(d) `Caltech-101`

(e) `CUB-200`

(f) `Dog-120`

(g) `Flower-102`

(h) `Food-101`

Figure 4: Test accuracy of the proposed NTK random features with Gibbs sampling and Random Fourier Features when the number of feature dimension m changes from 500 to 8,000.