Optimized Random Features for the Neural Tangent Kernel (Student Abstract)

Shrutimoy Das, Binita Maity

Indian Institute of Technology, Gandhinagar
shrutimoydas@iitgn.ac.in, binitamaity@iitgn.ac.in

Abstract

The neural tangent kernel (NTK) has emerged as an important tool in recent years, both for developing a theoretical understanding of deep learning as well as for various applications. Even though recursive closed form expressions have been derived for computing the NTK, these become computationally expensive as the complexity of a network increases. Recent papers have looked at reducing this complexity using various sketching techniques along with random features. Building on these techniques, we propose an additional optimization step which results in better approximation of the NTK.

Introduction

Attempts to develop a theoretical understanding of deep learning algorithms led to the discovery of the correspondence between neural networks and kernel methods. This facilitated the application of kernel methods as proxies for neural networks. This requires the computation of the so called neural tangent kernel (NTK) which can be used for tasks such as kernel ridge regression (KRR) as an alternative to neural network training. However, computing the NTK becomes costly as the size of the dataset as well as the depth and breadth of the network increases. Recently, Han et al. (2021) and Zandieh et al. (2021) proposed using random features based techniques along with sketching for computing the NTK efficiently. The random features and sketching algorithms involve randomizations at various stages, which affects the quality of approximation of the NTK. In this paper, motivated by the results of (Sinha and Duchi 2016), we show that with an additional data-dependent optimization step, the NTK approximation can be improved further. Empirically, we show that the number of random features required is reduced after the optimization step while also leading to better mean squared errors (MSEs) for a regression task on two datasets.

Neural Tangent Kernel

Jacot, Gabriel, and Hongler (2018) showed that for an L-layer fully connected network, as the depth of the hidden layers go to infinity, training a network by gradient descent under least-squares loss with a small step-size is equivalent to kernel regression. The kernel corresponding to this regression is the *Neural Tangent Kernel*, $K^{(L)}_{NTK}$.

Constructing the NTK

For a dataset $X \in \mathbb{R}^{n\times d}$, the NTK of a fully connected network with ReLU activation can be computed by the recursive equations given in Equation (1).

$$\begin{aligned} K^{(0)}_{NTK} &= K^{(0)}_{NNGP} = XX^T, \\ K^{(l)}_{NNGP} &= A_1(K^{(l-1)}_{NNGP}), \\ K^{(l)}_{NTK} &= K^{(l)}_{NNGP} + K^{(l-1)}_{NTK} \odot A_0(K^{(l-1)}_{NNGP}), \end{aligned} \tag{1}$$

where A_0 and A_1 are the arc-cosine kernels of order 0 and 1, respectively and $\odot$ is the elementwise product.

Random Features for the NTK

Computing the NTK using Equation (1) requires $O(n^2(d + L))$ time and $O(n(n+d))$ space complexity. Thus, Han et al. (2021) and Zandieh et al. (2021) proposed using the random features of the arc-cosine kernels for computing an approximation of the NTK. The random features for A_0 and A_1 are given as

$$z_0(x) = \sqrt{\frac{2}{m_0}}\text{Step}\Big([w_1, \ldots, w_{m_0}]^T x\Big), \tag{2}$$

$$z_1(x) = \sqrt{\frac{2}{m_1}}\text{ReLU}\Big([w'_1, \ldots, w'_{m_1}]^T x\Big) \tag{3}$$

where $w_1, \ldots, w_{m_0}, w'_1, \ldots, w'_{m_1}$ are drawn i.i.d from $\mathcal{N}(0, \mathbf{I}_d)$. Here, m_0 is the number of random features sampled for the A_0 kernel and m_1 is the number of random features for the A_1 kernel.

Given the random feature mappings, the approximate NTK can be computed as

$$\begin{aligned} \psi^{(0)}(x) &= \phi^{(0)}(x) = x, \\ \psi^{(l+1)}(x) &= z_1(\psi^{(l)}(x)) \\ \phi^{(l+1)}(x) &= \Big[\psi^{(l+1)}(x), z_0(\psi^{(l)}(x)) \otimes \phi^{(l)}(x)\Big] \end{aligned}$$

Then, the NTK for the L-layer network is $K^{(L)}_{NTK}(x, x') \approx \langle \phi^{(L)}(x), \phi^{(L)}(x')\rangle$. Here, $\otimes$ denotes a tensor product. The

Dataset	Method	num_layers	m_0	m_1	m_s	MSE	Time
CT (N = 53500, d = 379)	Exact	1	-	-	262960	64.74	1366.63
	NTK-RF	1	692	692	8192	216.74	14.05
	OPT NTK-RF	**1**	**∼350**	**∼395**	**∼7895**	**199.15**	**74.74**
	OPT NTK-RF(no GS)	**1**	**∼363**	**∼422**	**∼7922**	**183.63**	**16.9**
WorkLoads (N = 199843, d = 6)	Exact	2	-	-	-	OOM	-
	NTK-RF	2	3192	3192	8192	32662.96	115.09
	OPT NTK-RF	**2**	**Layer 0: ∼274 Layer 1: ∼347**	**Layer 0: ∼309 Layer 1: ∼361**	**∼5386**	**28362.16**	**107.55**
	OPT NTK-RF(no GS)	**2**	**Layer 0 : ∼280 Layer 1: ∼350**	**Layer 0: ∼318 Layer 1: ∼408**	**∼5408**	**30052.59**	**57.04**

Table 1: In this table, num_layers is the number of hidden layers in the network, Exact refers to kernel regression with full NTK matrix, NTK-RF refers to kernel regression with NTK computed from unoptimized arc-cosine kernel random features, OPT NTK-RF and OPT NTK-RF(no GS) refers to our methods. The results are averaged after a 4-fold validation. The time is reported in seconds. 'OOM' refers to Out of Memory error.

tensor product in the last recursive equation results in a large number of features, which is exponential in the number of layers. Zandieh et al. (2021) proposed using sketching schemes, such as SRHT, for approximating the tensor product, so as to limit the number of features to m_s.

Proposed Method

Since the approximation method discussed in the previous section is based on sampling random features from some distribution, these features may not be the optimal set of features for approximating the kernel. In this paper, once we sample an initial set of random features for the A_0 and the A_1 kernels (the computations for both of the kernels are done independently), we compute an optimal distribution over these random features such that the approximate kernel computed using these random features aligns with the label space of the points. This has been motivated by the approaches in Sinha and Duchi (2016).

Let $D_f(P||Q)$ be an f-divergence between distributions P and Q, where f is a convex function such that $f(1) = 0$. Suppose we sample N_w random features from $P = \mathbf{1}/N_w$. Then, we want to learn a distribution Q such that $D_f(Q||\frac{\mathbf{1}}{N_w}) < \rho$. If the support of the optimal distribution Q is sparse, we need only those random features for which the probabilities are non-zero, in order to maximize the kernel alignment. It was shown in (Sinha and Duchi 2016) that the distribution Q has a closed form solution and can be computed in time $O(N_w \log(1/\epsilon))$, for an ϵ-approximation to the solution of the kernel alignment problem. We then sample the random features from this optimal distribution and use it for approximating the kernel, say $\tilde{A}_0$ and $\tilde{A}_1$. These approximations are then used for computing the NTK. For our experiments, we consider $N_w = 20000$ and f to be the χ^2-divergence.

Results and Discussions

We test our approach for a kernel regression problem on two datasets, with the kernels as described in Table 1. In Table 1, we compare the performance of our methods with the exact NTK and NTK computed using unoptimized random features for kernel regression. In (Han et al. 2021), it was shown that to guarantee that the approximate NTK is "close" to the exact NTK, the random features for the A_1 kernel must be sampled using a Gibbs sampling (GS) scheme. In our experiments, we tried two variants: one with the A_1 random features from a GS scheme and the other without GS (OPT NTK-RF(no GS)). We wanted to check whether the learned distribution would make the GS scheme redundant. We only considered the datasets and architectures used in (Zandieh et al. 2021) for comparison.

From the table, it can be seen that the mean squared error (MSE) with the NTKs computed using optimized (kernel-aligned) random features, is less than that of NTK-RF. However, it is not close to the MSE for the exact NTK. The number of random features, i.e., m_0 and m_1, are those features which are a support for the distribution Q, for our method. That is, out of $N_w = 20000$ random features for our methods, only ~ 350 random features have non-zero probabilities in the distribution Q. It can be observed that the number of kernel-aligned random features required is less than the number of features for NTK-RF, in order to achieve and a lower MSE. Also, if the random features for the A_1 kernel are not sampled using GS scheme, the number of random features required for kernel alignment is more. This suggests that the modified distribution for A_1 random features does give good random features, at the expense of a few seconds. Overall, it can be seen that aligning the arc-cosine kernels at each layer with the output does help in achieving good performance for the kernel ridge regression problem.

Conclusion and Future Work

We observed that optimizing the random features of the arc-cosine kernels leads to better results for the kernel regression problem. One direction of future work is to find a distribution of the random features of the NTK such that the tensoring step is not required. We also aim to explore whether we can find asymptotic upper bounds on the number of random features required for the intermediate arc-cosine kernels. This work can be further applied for speeding up downstream tasks such as data distillation methods.

Ethical Statement

This work focuses on speeding up the computation of an approximation to the NTK. The datasets were taken from publicly available repositories and does not have any ethical conflicts.

Acknowledgements

The authors would like to thank Prof. Anirban Dasgupta for the helpful discussions regarding this project. Shrutimoy is supported by the Prime Minister's Research Fellowship (PMRF) awarded by the Government of India. The authors also acknowledge the support received from IIT Gandhinagar.

References

Han, I.; Avron, H.; Shoham, N.; Kim, C.; and Shin, J. 2021. Random Features for the Neural Tangent Kernel. *arxiv*.

Jacot, A.; Gabriel, F.; and Hongler, C. 2018. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in Neural Information Processing Systems*, 2018-Decem(5): 8571–8580.

Sinha, A.; and Duchi, J. 2016. Learning kernels with random features. *Advances in Neural Information Processing Systems*, 1306–1314.

Zandieh, A.; Han, I.; Avron, H.; Shoham, N.; Kim, C.; and Shin, J. 2021. Scaling Neural Tangent Kernels via Sketching and Random Features. *Advances in Neural Information Processing Systems*, 2: 1062–1073.