Provably and Practically Efficient Neural Contextual Bandits

Sudeep Salgia [1]

Abstract

We consider the neural contextual bandit problem. In contrast to the existing work which primarily focuses on ReLU neural nets, we consider a general set of smooth activation functions. Under this more general setting, (i) we derive non-asymptotic error bounds on the difference between an over-parameterized neural net and its corresponding neural tangent kernel, (ii) we propose an algorithm with a provable sublinear regret bound that is also efficient in the finite regime as demonstrated by empirical studies. The non-asymptotic error bounds may be of broader interests as a tool to establish the relation between the smoothness of the activation functions in neural contextual bandits and the smoothness of the kernels in kernel bandits.

1. Introduction

The stochastic contextual bandit has been extensively studied in the literature (see, Langford & Zhang, 2007; Lattimore & Szepesvári, 2020, and references therein). In this problem, at each discrete sequential step, a *context* is revealed by the environment. Then, a bandit agent chooses one of the K available actions. Choosing each action yields a random reward, the distribution of which is determined by the context. The problem was primarily studied under a linear setting where the expected reward of each arm is a linear function of the context (Chu et al., 2011; Li et al., 2019b; Chen et al., 2021). It was later extended to kernel-based models (Valko et al., 2013), where the expected reward function associated with the context-action pairs belongs to the reproducing kernel Hilbert space (RKHS) determined by a known kernel. Recently, a variation of the problem has been proposed where the expected reward function associated with the context-action pairs is modeled using a neural net (Zhou et al., 2020; Zhang et al., 2021; Gu et al., 2021; Kassraie & Krause, 2022). The three contextual bandit settings mentioned above are increasingly more complex in the following order: *Linear* $\rightarrow$ *Kernel-based* $\rightarrow$ *Neural*.

Contextual bandits find application in recommender systems, information retrieval, healthcare and finance (see a survey on applications in Bouneffouf & Rish (2019)). The problem can also be seen as a middle step from classic stochastic bandits (Auer et al., 2002) and (non-contextual) linear bandits (Abbasi-Yadkori et al., 2011) towards a reinforcement learning (RL) problem on a Markov decision process (MDP), where the contexts resemble the states of the MDP. One of the first formulations of linear contextual bandits was referred to as *associative RL* with linear value function (Auer, 2002). Nonetheless, contextual bandit is different from RL in that the contexts are determined arbitrarily (adversarially) rather than following a stochastic Markovian process.

1.1. Existing Results on Neural Contextual Bandits

With neural nets demonstrating a great representation power (much more general than linear models), there has been an increasing interest in modeling bandit and RL problems based on neural nets. This setting can be implemented using the typical neural net toolboxes. The neural contextual bandit has been considered in several works: (Zhou et al., 2020) and (Zhang et al., 2021), respectively, adopted the upper confidence bound (UCB) and Thompson sampling (TS) algorithms to the neural bandit setting. The algorithms are, respectively, referred to as NeuralUCB and NeuralTS. (Gu et al., 2021) considered a batch observation setting, where the reward function can be evaluated only on batches of data. (Kassraie & Krause, 2022) provided analysis for a variation of NeuralUCB algorithm with diminishing instantaneous regret.

The analysis of neural bandits has been enabled through

[1]Department of Electrical and Computer Engineering, Cornell University, Ithaca, NY, USA. This is a joint work with Sattar Vakili at MediaTek Research, Cambridge, UK, and Qing Zhao at Cornell University. Due to an error at the time of submission, their names were not included in the authorship. The original preprint of this work with full authorship can be found at `http://arxiv.org/abs/2206.00099`. Correspondence to: Sudeep Salgia <ss3827@cornell.edu>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

the theory of neural tangent (NT) kernel (Jacot et al., 2018) which approximates an overparameterized neural net with the kernel corresponding to its infinite width, and the bounds on the approximation error (e.g., see, Arora et al. (2019)). Zhou et al. (2020), Zhang et al. (2021) and Gu et al. (2021) proved $\tilde{\mathcal{O}}(\Gamma_k(T)\sqrt{T})$[1] bounds on cumulative regret, where T is the number of steps and $\Gamma_k(T)$ is a complexity term determined by the neural tangent kernel k associated with the particular neural network. Specifically, $\Gamma_k(T)$ is the information gain corresponding to k for datasets of size T. For the ReLU neural nets on d-dimensional domains considered in Zhou et al. (2020), Zhang et al. (2021) and Gu et al. (2021), $\Gamma_k(T) = \tilde{\mathcal{O}}(T^{\frac{d-1}{d}})$ is the best upper bound known for the information gain (Kassraie & Krause, 2022; Vakili et al., 2021b). Inserting this bound on $\Gamma_k(T)$, the aforementioned regret bounds become trivial (superlinear) generally when $d > 1$, unless further restrictive assumptions are made on the contexts. Kassraie & Krause (2022) addressed this issue by considering the *Sup* variant of NeuralUCB (referred to as SupNeuralUCB). The Sup variant is an adoption of SupLinUCB, which was initially introduced in Chu et al. (2011) for linear contextual bandits, to the neural setting. Kassraie & Krause (2022) proved an $\tilde{\mathcal{O}}(\sqrt{\Gamma_k(T)T})$ regret bound for SupNeuralUCB in the case of ReLU neural nets, which solved the issue of superlinear regret bound (non-diminishing instantaneous regret).

1.2. Contribution

All the existing works mentioned above, are limited to neural nets with ReLU activation functions. This limitation is rooted in the fact that the existing error bound between an overparameterized neural net and the corresponding NT kernel (Arora et al., 2019, Theorem 3.2) holds only for the ReLU neural nets. In Theorem 1, which may be of broader interest, we extend this result by providing error bounds for neural nets with arbitrarily smooth activation functions. Using Theorem 1, together with the recently established bounds on $\Gamma_k(T)$ corresponding to arbitrarily smooth neural nets (Vakili et al., 2021b), we provide regret bounds for neural contextual bandits under a more general setting than only ReLU activation function. In particular, in Theorem 3, we prove an $\tilde{\mathcal{O}}(T^{\frac{2d+2s-3}{2d+4s-4}})$ upper bound on regret, where s is the smoothness parameter of the activation functions ($s = 1$ in the case of ReLU, and s grows larger as the activations become smoother). Our regret bounds recover that of Kassraie & Krause (2022) for ReLU activations, and can become as small as $\tilde{\mathcal{O}}(\sqrt{T})$ when the activations become infinitely smooth.

Broadening the scope of neural bandits to general smooth activations is of interest in two aspects. Firstly, as shown in (Li et al., 2019a; Opschoor et al., 2022), deep neural nets constructed using smoother activation functions like Rectified Power Units or RePUs provide better approximation guarantees than those with ReLUs. This is particularly true when the function to be approximated is smooth, which is generally the case in practical applications. Additionally, smoother activation functions have also been shown to more suitable for certain applications like implicit representations (Sitzmann et al., 2020). We explore whether such advantages of smoother activations reflect in the neural bandit settings. In particular, the regret bounds with ReLU neural nets, although sublinear, grow at a fast $\tilde{\mathcal{O}}(T^{\frac{2d-1}{2d}})$ rate, that quickly approaches the linear rate as d grows. We show that using neural nets with smoother activations can significantly reduce rate and thereby affirmatively establish the benefits of employing smoother activations in neural nets. Secondly, our results help establish connections between varying smoothness of activations in neural bandits and varying smoothness of kernels in kernel-based bandits . Kernel-based bandits (Srinivas et al., 2010) and Kernel-based contextual bandits (Valko et al., 2013) are well studied problems with regret bounds reported for popular kernels such as Matérn family, which is perhaps the most commonly used family of kernels in practice (Snoek et al., 2012; Shahriari et al., 2015). Our regret bound for a neural contextual bandit problem with activations with smoothness parameter s is equivalent to the regret bound in the case of kernel-based bandits with a Matérn kernel (Valko et al., 2013; Li & Scarlett, 2022) with smoothness parameter $s - \frac{1}{2}$. This connection may be of general interest in terms of contrasting neural nets and kernel-based models through NT kernel theory.

In Theorem 2, we prove confidence intervals for overparameterized neural nets with smooth activation functions, which are later used in the analysis of our algorithm. The confidence intervals are a consequence of our Theorem 1 and those for kernel regression. In contrast to prior work, our confidence intervals are tighter and hold for a general set of smooth activation functions (see Sec. 3).

In addition to the above analytical results for neural bandits with general smooth activation functions, we propose a practically efficient algorithm. The Sup variants of UCB algorithms are known to perform poorly in experiments (Calandriello et al., 2019; Li & Scarlett, 2022), including SupNeuralUCB presented in Kassraie & Krause (2022), due to over-exploration. We propose NeuralGCB, a neural contextual bandit algorithm guided by both upper and lower confidence bounds to provide a finer control of exploration-exploitation trade-off to avoid over-exploration. In the experiments, we show that NeuralGCB outperforms all other

[1]The notations $\mathcal{O}$ and $\tilde{\mathcal{O}}$ are used for mathematical order and that up to logarithmic factors, respectively.

existing algorithms, namely NeuralUCB (Zhou et al., 2020), NeuralTS (Zhang et al., 2021) and SupNeuralUCB (Kassraie & Krause, 2022).

1.3. Other Related Work

Linear bandits have been considered also in a non-contextual setting, where the expected rewards are linear in the actions in a fixed domain (e.g., see the seminal work of Abbasi-Yadkori et al., 2011). The kernel-based bandits (also referred to as Gaussian process bandits) have been extensively studied under a non-contextual setting (Srinivas et al., 2010; Chowdhury & Gopalan, 2017). The GP-UCB and GP-TS algorithms proposed in this setting also show suboptimal regret bounds (see Vakili et al. (2021c) for details). Recently there have been several works offering optimal order regret bounds for kernel-based (non-contextual) bandits (Salgia et al., 2021; Camilleri et al., 2021; Li & Scarlett, 2022). These results however do not address the additional difficulties in the contextual setting.

There is a large body of work that studies the performance of overparametrized neural nets (see, for example, Cao & Gu, 2019; 2020; Allen-Zhu et al., 2019) which is closely related to the NTK approximation studied in this work. While there are similarities, our proposed technical approach introduces non-trivial novel analysis over all these studies. Specifically, all of them consider neural nets with ReLU activation functions. In our work, we derive new results and lemmas that hold for a larger class of activation functions as opposed to only the ReLU activation function. A notable exception to activation function being ReLU is the work by Du et al. (2019) where the authors also consider "smooth" activation functions. However, the activation functions considered in that work are "smooth" in the sense that the derivative of the activation function is globally Lipschitz. This is not necessarily true for the class of activation functions considered in this work, especially for $s \geq 3$. Consequently, our work extends the results in Du et al. (2019) to a new class of activation functions.

2. Preliminaries and Problem Formulation

In the stochastic contextual bandit problem, at each discrete sequential step $t = 1, 2, \ldots, T$, for each action $a \in [K] := \{1, 2, \ldots, K\}$, a context vector $\mathbf{x}_t \in \mathcal{X}$ is revealed, where $\mathcal{X}$ is a compact subset of $\mathbb{R}^d$. Then, a bandit agent chooses an action $a_t \in [K]$ and receives the corresponding reward $r_t = h(\mathbf{x}_{t,a_t}) + \xi_t$, where $h : \mathbb{R}^d \to \mathbb{R}$ is the underlying reward function and $\xi_t \in \mathbb{R}$ is a zero-mean martingale noise process. The goal is to choose actions which maximize the cumulative reward over T steps. This is equivalent to minimizing the cumulative *regret*, that is defined as the total difference between the maximum possible context-dependent reward and the actually received reward

$$R(T) = \sum_{t=1}^{T} (h(\mathbf{x}_{t,a_t^*}) - h(\mathbf{x}_{t,a_t})), \tag{1}$$

where $a_t^* \in \arg\max_{a\in[K]} h(\mathbf{x}_{t,a})$ is the context-dependent maximizer of the reward function at step t. Following is a formal statement of the assumptions on $\mathcal{X}$, f and ξ_t. These are mild assumptions which are shared with other works on neural bandits and NT kernel.

Assumption 1. (i) *The input space is the d dimensional hypersphere: $\mathcal{X} = \mathbb{S}^{d-1}$.* (ii) *The reward function h belongs to the RKHS induced by the NT kernel, and $h(\mathbf{x}) \in [0, 1]$, $\forall\ \mathbf{x} \in \mathcal{X}$.* (iii) *$\xi_t$ are assumed to be conditionally ν-sub-Gaussian, i.e., for any $\zeta \in \mathbb{R}$, $\ln(\mathbb{E}[e^{\zeta\xi_t}|\xi_1, \ldots, \xi_{t-1}]) \leq \zeta^2\nu^2/2$.*

2.1. The Neural Net Model and Corresponding NT Kernel

In this section, we briefly outline the neural net model and the associated NT kernel. Let $f(\mathbf{x}; \mathbf{W})$ be a fully connected feedforward neural net with L hidden layers of equal width m. We can express f using the following set of recursive equations:

$$f(\mathbf{x}; \mathbf{W}) = \sqrt{\frac{c_s}{m}} \mathbf{W}^{(L+1)} \sigma_s(f^{(L)}(\mathbf{x})), \tag{2}$$

$$f^{(1)}(\mathbf{x}) = \mathbf{W}^{(1)}\mathbf{x},\ \ f^{(l)}(\mathbf{x}) = \sqrt{\frac{c_s}{m}} \mathbf{W}^{(l)} \sigma_s(f^{(l-1)}(\mathbf{x})) \quad \text{for } 1 < l \leq L$$

Let $\mathbf{W} = (\mathbf{W}^l)_{1\leq l\leq L+1}$ denote the collection of all weights. The dimensions of weight matrices satisfy: $\mathbf{W}^{(1)} \in \mathbb{R}^{m\times d}$; for $1 < l \leq L$, $\mathbf{W}^{(l)} \in \mathbb{R}^{m\times m}$; $\mathbf{W}^{(L+1)} \in \mathbb{R}^{1\times m}$. All the weights $\mathbf{W}^{(l)}_{i,j}$, $\forall\ l, i, j$, are initialized to $\mathcal{N}(0, 1)$ and $c_s := 2/(2s-1)!!$. With an abuse of notation, $\sigma_s : \mathbb{R}^m \to \mathbb{R}^m$ is used for coordinate-wise application the activations of the form $\sigma_s(u) = (\max(0, u))^s$ to the outputs of each layer. Note that σ_s is $s-1$ times differentiable everywhere which gives our results a general interpretability across smoothness of activations and the resulting function classes and allows us to draw parallels between our results and well established results for kernel-based bandits.

It has been shown that gradient based training of the neural net described above reaches a global minimum where the weights remain within a close vicinity of random initialization. As a result the model can be approximated with its linear projection on the tangent space at random initialization $\mathbf{W}_0$:

$$f(\mathbf{x}; \mathbf{W}) \approx f(\mathbf{x}; \mathbf{W}_0) + (\mathbf{W} - \mathbf{W}_0)^\top \mathbf{g}(\mathbf{x}; \mathbf{W}_0),$$

where the notation $\mathbf{g}(\mathbf{x};\mathbf{W})=\nabla_{\mathbf{W}}f(\mathbf{x};\mathbf{W})$ is used for the gradient of f with respect to the parameters. The approximation error can be bounded by the second order term $\mathcal{O}(\|\mathbf{W}-\mathbf{W}_0\|^2)$, and shown to be diminishing as the width m grows, where the implied constant depends on the spectral norm of the Hessian matrix (Liu et al., 2020). The NT kernel corresponding to this neural net when m grows to infinity is defined as the kernel in the feature space determined by the gradient at initialization:

$$k(\mathbf{x},\mathbf{x}')=\lim_{m\to\infty}\mathbf{g}^\top(\mathbf{x};\mathbf{W}_0)\mathbf{g}(\mathbf{x};\mathbf{W}_0).$$

The particular form on the NT kernel depends on the activation function and the number of layers. For example, for a two layer neural net with ReLU activations, we have

$$k(\mathbf{x},\mathbf{x}')=\frac{1}{\pi}\left(2u(\pi-\arccos(u))+\sqrt{1-u^2}\right)$$

where $u=\mathbf{x}^\top\mathbf{x}'$. For the closed form derivation of NT kernel for other values of s and L, see Vakili et al. (2021b).

2.2. Assumptions on Neural Net and NT Kernel

The following technical assumptions are mild assumptions which are used in the handling of the overparameterized neural nets using NT kernel. These assumptions are common in the literature and often are fulfilled without loss of generality (Arora et al., 2019; Zhou et al., 2020; Zhang et al., 2021; Gu et al., 2021; Kassraie & Krause, 2022).

Assumption 2. (i) *Consider* $\mathbf{H}\in\mathbb{R}^{KT\times KT}$ *such that* $[\mathbf{H}]_{i,j}=k(\mathbf{z}_i,\mathbf{z}_j)$ *for all pairs in* $\mathcal{Z}\times\mathcal{Z}$*, where* $\mathcal{Z}=\{\{\mathbf{x}_{t,a}\}_{a=1}^K\}_{t=1}^T$*. We assume* $\mathbf{H}\succcurlyeq\lambda_0\mathbb{I}$*, where* $\lambda_0>0$. (ii) $f(\mathbf{x};\mathbf{W}_0)$ *is assumed to be* 0. (iii) *The number of epochs during training,* J*, and the learning rate,* η*, satisfy* $J=\frac{2}{c\lambda}T\log(T)$ *and* $\eta=c'/T$ *for some constants* $c,c'>0$ *and* $\lambda>0$ *is the regularization parameter.*

2.3. Information Gain

The regret bounds for neural bandits are typically given in terms of maximal information gain (or the effective dimension) of the NT kernel. The maximal information gain is defined as the maximum log determinant of the kernel matrix (Srinivas et al., 2010):

$$\Gamma_k(t)=\sup_{\mathbf{X}_t\subseteq\mathcal{X}}\log\left(\det\left(\mathbb{I}+\frac{1}{\lambda}k_{\mathbf{X}_t,\mathbf{X}_t}\right)\right). \tag{3}$$

It is closely related to the effective dimension of the kernel, which denotes the number of features with a significant impact on the regression model and can be finite for a finite dataset even when the feature space of the kernel is infinite dimensional (Zhang, 2005; Valko et al., 2013). It is defined as

$$\tilde{d}_k(t)=\mathrm{tr}\left(k_{\mathbf{X}_t,\mathbf{X}_t}(k_{\mathbf{X}_t,\mathbf{X}_t}+\lambda\mathbb{I})^{-1}\right). \tag{4}$$

It is known that the information gain and the effective dimension are the same up to logarithmic factors (Calandriello et al., 2019). We give our results in terms of information gain; nonetheless, that can be replaced by effective dimension.

3. Approximation Error and Confidence Bounds

As discussed earlier, the error between an overparameterized neural net and the associated NT kernel can be quantified and shown to be small for large m. To formalize this, we recall the technique of kernel ridge regression. Given a dataset $\mathcal{D}_t=\{(\mathbf{x}_i,y_i)\}_{i=1}^t$, let f_{NTK} denote the regressor obtained by kernel ridge regression using NT kernel. In particular, we have

$$f_{\text{NTK}}(\mathbf{x})=k_{\mathbf{X}_t}^\top(\mathbf{x})(\lambda\mathbb{I}_t+k_{\mathbf{X}_t,\mathbf{X}_t})^{-1}\mathbf{Y}_t, \tag{5}$$

where $k_{\mathbf{X}_t}(\mathbf{x})=[k(\mathbf{x}_1,\mathbf{x}),\dots,k(\mathbf{x}_t,\mathbf{x})]^\top$ is the NT kernel evaluated between $\mathbf{x}$ and t data points, $k_{\mathbf{X}_t,\mathbf{X}_t}=[k(\mathbf{x}_i,\mathbf{x}_j)]_{i,j=1}^t$ is the NT kernel matrix evaluated on the data, $\mathbf{Y}_t=[y_1,\dots,y_t]^\top$ is the vector of output values, and $\lambda\geq0$ is a free parameter. Note that f_{NTK} can be computed in closed form (without training a neural net). In addition, let f_{NN} be prediction of the neural net at the end of training using $\mathcal{D}_t$. Theorem 3.2 of Arora et al. (2019) states that, when the activations are ReLU and m is sufficiently large, we have, with probability at least $1-\delta$, that $|f_{\text{NN}}(\mathbf{x})-f_{\text{NTK}}(\mathbf{x})|\leq\epsilon$. In this theorem, the value of m should be sufficiently large in terms of t,L,δ,ϵ, and λ_0. We now present a bound on the error between the neural net and the associated NT kernel for smooth activations.

Theorem 1. *Consider the neural net* $f(\mathbf{x};\mathbf{W})$ *defined in* (2) *consisting of* L *hidden layers each of width* $m\geq\mathrm{poly}(1/\varepsilon,s,1/\lambda_0,|\mathcal{D}|,\log(1/\delta))$ *with activations* σ_s*. Let* f_{NTK} *and* f_{NN} *be the kernel ridge regressor and trained neural net using a dataset* $\mathcal{D}$*. Then for any* $\mathbf{x}\in\mathcal{X}$*, with probability at least* $1-\delta$ *over the random initialization of the neural net, we have,*

$$|f_{\text{NTK}}(\mathbf{x})-f_{\text{NN}}(\mathbf{x})|\leq\varepsilon.$$

Theorem 1 is an generalization of Theorem 3.2 in Arora et al. (2019) to the class of smooth activation functions $\{\sigma_s,s\in\mathbb{N}\}$. To prove Theorem 1, we show the following bound on the error between NT kernel and the kernel induced by the finite width neural net at initialization. This result is an generalization of Theorem 3.1 in Arora et al. (2019) to smooth activation functions.

Lemma 1. *Consider the neural net* $f(\mathbf{x};\mathbf{W})$ *defined in* (2) *consisting of* L *hidden layers each of width* $m\geq\mathcal{O}(\frac{s^LL^2c_s^2}{\varepsilon^2}\log^2(sL/\delta\varepsilon))$ *with activations* σ_s*. Let*

$\mathbf{g}(\mathbf{x};\mathbf{W}) = \nabla_{\mathbf{W}} f(\mathbf{x};\mathbf{W})$ *and k be the associated NT kernel, as defined in Section 2.1. Fix $\varepsilon > 0$ and $\delta \in (0,1)$. Then for any $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$, with probability at least $1-\delta$ over the initialization of the network, we have,*

$$\left|\mathbf{g}^\top(\mathbf{x};\mathbf{W}_0)\mathbf{g}(\mathbf{x}';\mathbf{W}_0) - k(\mathbf{x},\mathbf{x}')\right| \leq (L+1)\varepsilon.$$

The proof entails a non-trivial generalization of various lemmas used in the proof of Arora et al. (2019, Theorem 3.2) to the class of smooth activation functions. We defer the detailed proof of the lemma and theorem to supplementary material.

Confidence Intervals: The analysis of bandit problems classically builds on confidence intervals applicable to the values of the reward function. The NT kernel allows us to use the confidence intervals for kernel ridge regression, in building confidence intervals for overparameterized neural nets. In particular, given a dataset $\mathcal{D}_t$, let

$$\mathbf{V}_t = \lambda\mathbb{I} + \sum_{i=1}^{t} \mathbf{g}(\mathbf{x}_i;\mathbf{W}_0)\mathbf{g}^\top(\mathbf{x}_i;\mathbf{W}_0) \tag{6}$$

be the Gram matrix in the feature space determined by the gradient. Taking into account the linearity of the model in the feature space, we can define a (surrogate) posterior variance as follows,

$$\hat{\sigma}_t^2(\mathbf{x}) = \mathbf{g}^\top(\mathbf{x})\mathbf{V}_t^{-1}\mathbf{g}(\mathbf{x}), \tag{7}$$

that can be used as a measure of uncertainty in the prediction provided by the trained neural net. Using Theorem 1 in conjunction with the confidence intervals for kernel ridge regression, we prove the following confidence intervals for a sufficiently wide neural net.

Theorem 2. *Let $\mathcal{D}_t = \{(\mathbf{x}_i, r_i)\}_{i=1}^t$ denote a dataset obtained under the observation model described in Section 2 such that the points $\{\mathbf{x}_i\}_{i=1}^t$ are independent of the noise sequence $\{\xi_i\}_{i=1}^t$. Suppose Assumptions 1 and 2 hold. Suppose the neural net defined in (2) consisting of L hidden layers each of width $m \geq \mathrm{poly}(T, s, L, K, \lambda^{-1}, \lambda_0^{-1}, \log(1/\delta))$ is trained using this dataset. Then, with probability at least $1-\delta$, the following relation holds for any $\mathbf{x} \in \mathcal{X}$:*

$$|h(\mathbf{x}) - f(\mathbf{x};\mathbf{W}_t)| \leq \frac{C_{s,L}t}{\lambda m} + \beta_t\hat{\sigma}_t(\mathbf{x}), \tag{8}$$

where $\mathbf{W}_t$ denotes the parameters of the trained model, $\beta_t = S + 2\nu\sqrt{\log(1/\delta)} + C'_{s,L}t(1-\eta\lambda)^{J/2}\sqrt{t/\lambda} + C''_{s,L}\sqrt{t^3/\lambda m}$, S is the RKHS (corresponding to the NT kernel) norm of h and $C_{s,L}, C'_{s,L}, C''_{s,L}$ are constants depending only on s and L.

Both results presented in this section may be of broader interest in neural net literature. We would like to emphasize a couple of points here. Similar to (Vakili et al., 2021a; Kassraie & Krause, 2022), the independence of query points from the noise sequence is fundamental to obtain these tighter bounds and hence cannot be directly used to improve the regret bounds of adaptive algorithms like NeuralUCB. Secondly, these bounds hold for all activation functions $\{\sigma_s : s \in \mathbb{N}\}$ improving upon the existing results which hold only for ReLU activation function. Furthermore, this bound is tighter than the one derived in Kassraie & Krause (2022), even for the case of ReLU activation function. The confidence intervals are important building blocks in the analysis of NeuralGCB in Section 4. Please refer to the supplementary material for a detailed proof of the theorem.

4. Algorithm

The UCB family of algorithms achieve optimal regret in classic stochastic finite action bandits (Auer et al., 2002). The optimal regret, however, hinges on the statistical independence of the actions. In more complex settings such as kernel-based and neural bandits, the inherent statistical dependence across adaptively chosen actions leads to skewed posterior distributions, resulting in sub-optimal and potentially trivial (i.e., superlinear) regret guarantees of UCB based algorithms (Vakili et al., 2021c). To address this issue, one approach adopted in the literature is to use samples with limited adaptivity. These algorithms are typically referred to as Sup variant of UCB algorithms, and have been developed for linear (Chu et al., 2011), kernel-based (Valko et al., 2013) and neural (Kassraie & Krause, 2022) contextual bandits. These Sup variants, however, are known to perform poorly in practice due to their tendency to overly explore suboptimal arms.

We propose NeuralGCB, a neural bandit algorithm guided by both upper and lower confidence bounds to avoid over-exploring, leading to superior performance in practice while preserving the nearly optimal regret guarantee. The key idea of NeuralGCB is a finer control of the exploration-exploitation tradeoff based on the predictive standard deviation of past actions. This finer control encourages more exploitation and reduces unnecessary exploration. Specifically, NeuralGCB partitions past action-reward pairs into $R = \log T$ subsets (some subsets may be empty at the beginning of the learning horizon). Let $\{\Psi^{(r)}\}_{r=1}^R$ denote these subsets of action-reward pairs where the index r represents the level of uncertainty about the data points in that set (with $r = 1$ representing the highest uncertainty). Upon seeing a new context at time t, NeuralGCB traverses down the subsets starting from $r = 1$. At each level r, it evaluates the largest predictive standard deviation among current set of candidate actions, A_r, and compares it with 2^{-r}.

Algorithm 1 NeuralGCB

```
1: Require: Time horizon $T$, maximum initial variance $\sigma_0$, error probability $\delta$
2: Initialize: $R \leftarrow \lceil \log_2 T \rceil$, Ensemble of $R$ Neural Nets with $\mathbf{W}_0^{(r)} = \mathbf{W}_0$, $\Psi_0^{(r)} \leftarrow \emptyset$, $\forall\, r \in [R]$, $\mathcal{H} \leftarrow \emptyset$, arrays ctr, max_mu, fb of size $R$ with all elements set to 0, batch sizes $\{q_r\}_{r=1}^{R}$
3: for $t = 1, 2, 3, \dots T$ do
4:   $r \leftarrow 1, \hat{A}_r(t) = [K]$
5:   while True do
6:     Receive the context-action pairs $\{\mathbf{x}_{t,a}\}_{a=1}^{K}$
7:     $\{f(\mathbf{x}_{t,a}; \mathbf{W}_t^{(r)}), \hat{\sigma}_{t-1}^{(r)}(\mathbf{x}_{t,a})\}_{a=1}^{K}, \mathbf{W}_t^{(r)}, \texttt{fb}[r] \leftarrow$ GetPredictions$\Big(\mathcal{H}, \Psi_{t-1}^{(r)}, \{\mathbf{x}_{t,a}\}_{a=1}^{K}, \texttt{fb}[r], \mathbf{W}_{t-1}^{(r)}, q_r\Big)$
8:     $\tilde{\sigma}_{t-1}^{(r)} \leftarrow \max_{a \in \hat{A}_r(t)} \hat{\sigma}_{t-1}^{(r)}(x_{t,a})$
9:     $\texttt{max\_mu}[r] \leftarrow \arg\max_{a \in \hat{A}_r(t)} f(x_{t,a}; \mathbf{W}_{t-1}^{(r)})$
10:    if $\tilde{\sigma}_{t-1}^{(r)} \leq \sigma_0 2^{-r}$ then
11:      $a_{\text{UCB}} \leftarrow \arg\max_{a \in \hat{A}_r(t)} \text{UCB}_{t-1}^{(r)}(\mathbf{x}_{t,a})$
12:      if $\sigma_{t-1}^{(r)}(x_{t,a_{\text{UCB}}}) \leq \eta_0/\sqrt{t}$ then
13:        Choose $a_t \leftarrow a_{\text{UCB}}$ and set $\upsilon_t \leftarrow 1$
14:        Receive $y_t = h(\mathbf{x}_{t,a_t}) + \xi_t$ and update $\mathcal{H} \leftarrow \mathcal{H} \cup \{(x_{t,a_t}, y_t)\}$
15:        Set $\Psi_t^{(r+1)} \leftarrow \Psi_{t-1}^{(r+1)} \cup \{(t, \upsilon_t)\}$ and $\Psi_t^{(r')} \leftarrow \Psi_{t-1}^{(r')}$ for all $r' \in [R] \setminus \{r+1\}$
16:        break
17:      else
18:        $\hat{A}_{r+1}(t) \leftarrow \{a \in \hat{A}_r(t) : \text{UCB}_{t-1}^{(r)}(\mathbf{x}_{t,a}) \geq \max_{a' \in \hat{A}_r(t)} \text{LCB}_{t-1}^{(r)}(\mathbf{x}_{t,a'})\}$, $r \leftarrow r + 1$
19:      end if
20:    else
21:      if $r = 1$ or $\texttt{ctr}[r] > \alpha_0 4^r$ then
22:        Choose any $a_t \in \hat{A}_r(t)$ such that $\hat{\sigma}_{t-1}^{(r)}(x_{t,a}) > \sigma_0 2^{-r}$ and set $\upsilon_t \leftarrow 2$
23:      else
24:        Choose $a_t \leftarrow \texttt{max\_mu}[r-1]$ and set $\upsilon_t \leftarrow 3$
25:      end if
26:      Receive $y_t = h(\mathbf{x}_{t,a_t}) + \xi_t$ and update $\mathcal{H} \leftarrow \mathcal{H} \cup \{(x_{t,a_t}, y_t)\}$, $\texttt{ctr}[r] \leftarrow \texttt{ctr}[r] + 1$
27:      Set $\Psi_t^{(r)} \leftarrow \Psi_{t-1}^{(r)} \cup \{(t, \upsilon_t)\}$ and $\Psi_t^{(r')} \leftarrow \Psi_{t-1}^{(r')}$ for all $r' \in [R] \setminus \{r\}$
28:      break
29:    end if
30:  end while
31: end for
```

If this value is smaller than 2^{-r}, NeuralGCB checks if it can exploit based on predictions at the r^{th} level. Specifically, if the predictive standard deviation of the action that maximizes the UCB score is $\mathcal{O}(1/\sqrt{t})$ (indicating a high reward with reasonably high confidence), then NeuralGCB plays that action. Otherwise, it updates A_r by eliminating actions with small UCB score and moves to the next level. This encourages exploitation of less certain actions in a controlled manner as opposed to SupNeuralUCB which exploits only actions with much higher certainty.

On the other hand, if the value is greater than 2^{-r}, NeuralGCB directly exploits the maximizer of the mean computed using data points in $\Psi^{(r-1)}$ until the allocated budget for exploitation at level r is exhausted. It then resorts to more exploratory actions by choosing actions with large values of $\hat{\sigma}_{t-1}^{(r)}$. The exploitation budget is set to match the length of exploration sequence at the corresponding level to ensure an optimal balance of exploitation and exploration and preserve the optimal regret order while boosting empirical performance.

In addition to improved empirical performance, NeuralGCB takes into consideration the practical requirements for training the neural nets. Specifically, as pointed out in Gu et al. (2021), it is practically more efficient to train the neural net over batches of observations, rather than sequentially at each step. Consequently, before evaluating the mean and variance at any level r, NeuralGCB retrains the neural net corresponding to that level only if q_r samples have been added to that level since it was last trained. This index-dependent choice of batch size, q_r lends a natural adaptivity to the retraining frequency by reducing it as time progresses.

A pseudo code of the algorithm is outlined in Algorithm 1. In the pseudo code, $\text{UCB}_t^{(r)}$ and $\text{LCB}_t^{(r)}$ refer to the upper and lower confidence scores respectively at time t corresponding to index r and are defined as $\text{UCB}_t^{(r)}(\cdot) = f(\cdot; \mathbf{W}_t^{(r)}) + \beta\hat{\sigma}_t^{(r)}(\cdot)$ and $\text{LCB}_t^{(r)}(\cdot) = f(\cdot; \mathbf{W}_t^{(r)}) - \beta\hat{\sigma}_t^{(r)}(\cdot)$. GetPredictions is a local routine that calculates the predictive mean and variance after appropriately retraining the neural net (see supplementary for a pseudo code). Lastly, the arrays ctr, max_mu and fb are used to store the exploitation count, maximizer of the neural net output and feedback time instants. We now formally state the regret guarantees for NeuralGCB in the following theorem.

Theorem 3. *Suppose Assumptions 1 and 2 hold. Consider NeuralGCB given in Algorithm 1, with R neural nets, as defined in (2) with L hidden layers each of width $m \geq poly(T, L, K, s, \lambda^{-1}, \lambda_0^{-1}, S^{-1}, \log(1/\delta))$. Suppose NeuralGCB is run with a fixed batch size for each group, then the regret defined in (1) satisfies*

$$R(T) = \tilde{\mathcal{O}}\big(\sqrt{T\Gamma_k(T)} + \sqrt{T\Gamma_k(T)\log(1/\delta)} + \max_r q_r \Gamma_k(T)\big)$$

As suggested by the above theorem, NeuralGCB preserves the regret guarantees of SupNeuralUCB which are much tighter than those of NeuralUCB. Moreover, these guarantees hold even for smooth activation functions as opposed to just for ReLU activation. In particular, on plugging in the upper bound on $\Gamma_k(T)$ obtained in Vakili et al. (2021b), we obtain that the regret incurred by NeuralGCB satisfies $\tilde{\mathcal{O}}(T^{\frac{2d+2s-3}{2d+4s-4}})$, which is a sublinear function of T, implying convergence to the maximizer. For $s = 1$, i.e., the ReLU activation function, it reduces to $\tilde{\mathcal{O}}(T^{\frac{2d-1}{d}})$, matching the known regret bound for ReLU neural nets (Kassraie & Krause, 2022). Moreover, the regret bound for neural nets with activation functions with smoothness parameter s matches the optimal order of regret for optimization of a function in an RKHS corresponding to the Matérn kernel with smoothness $s - 1/2$. This equivalence paves a path to enable us extend known results for Matérn kernels for NTK theory. Furthermore, the above regret guarantees also show that the retraining neural nets only at the end of batches of observations increases the regret at most with an additive term in the batch size. Our results also hold with an adaptive batch size (see Appendix D).

5. Empirical Studies

In this section, we provide numerical experiments on comparing NeuralGCB with several representative baselines, namely, LinUCB (Chu et al., 2011), NeuralUCB (Zhou et al., 2020), NeuralTS (Zhang et al., 2021), SupNeuralUCB (Kassraie & Krause, 2022) and Batched NeuralUCB (Gu et al., 2021).

We perform the empirical studies on three synthetic and two real-world datasets. We first compare NeuralGCB with the fully sequential algorithms (LinUCB, NeuralUCB, NeuralTS ans SupNeuralUCB). We then compare the regret incurred and the time taken by NeuralGCB, NeuralUCB and Batch NeuralUCB. We perform both set of experiments with two different activation functions. The construction of the synthetic datasets and the experimental settings are described below.

5.1. Datasets

For each of the synthetic datasets, we construct a contextual bandit problem with a feature dimension of $d = 10$ and $K = 4$ actions per context running over a time horizon of $T = 2000$ rounds. The set of context vectors $\{\{\mathbf{x}_{t,a}\}_{a=1}^{K}\}_{t=1}^{T}$ are drawn uniformly from the unit sphere. Similar to Zhou et al. (2020), we consider the following three reward functions:

$$h_1(\mathbf{x}) = 4|\mathbf{a}^\top\mathbf{x}|^2; \quad h_2(\mathbf{x}) = 4\sin^2(\mathbf{a}^\top\mathbf{x}); \quad h_3(\mathbf{x}) = \|\mathbf{A}\mathbf{x}\|_2. \tag{9}$$

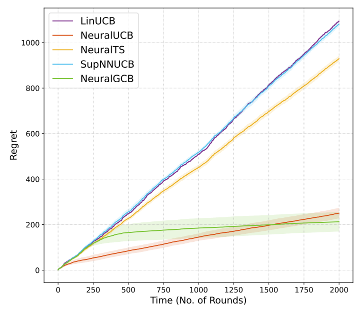
For the above functions, the vector $\mathbf{a}$ is drawn uniformly from the unit sphere and each entry of matrix $\mathbf{A}$ is randomly generated from $\mathcal{N}(0, 0.25)$.

We also consider two real datasets for classification namely Mushroom and Statlog (Shuttle), both of which are available on the UCI repository (Dua & Graff, 2017). The classification problem is then converted into a contextual bandit problem using techniques outlined in Li et al. (2010). Each datapoint in the dataset $(\mathbf{x}, y) \in \mathbb{R}^d \times \mathbb{R}$ is transformed into K vectors of the form $\mathbf{x}^{(1)} = (\mathbf{x}, \mathbf{0}, \dots, \mathbf{0}), \dots, \mathbf{x}^{(K)} = (\mathbf{0}, \dots, \mathbf{0}, \mathbf{x}) \in \mathbb{R}^{Kd}$ corresponding to the K actions associated with context $\mathbf{x}$. Here K denotes the number of classes in the original classification problem. The reward function is set to $h(\mathbf{x}^{(k)}) = \mathbb{1}\{y = k\}$, that is, the agent receives a reward of 1 if they classify the context correctly and 0 otherwise. We present the results for $h_1(x), h_2(x)$ and the Mushroom dataset in the main paper, and the additional results in the supplementary material.

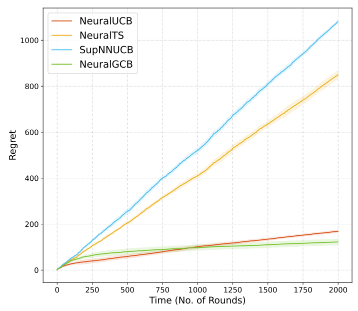
5.2. Experimental Setting

For all the experiments, the rewards are generated by adding zero mean Gaussian noise with a standard deviation of 0.1 to the reward function. All the experiments are run for a time horizon of $T = 2000$. We report the regret averaged over 10 Monte Carlo runs with different random seeds. For the real datasets, we shuffle the context vectors for each Monte Carlo run. For all the algorithms, we set the parameter ν to 0.1, and S, the RKHS norm of the reward, to 4 for synthetic functions, and 1 for real datasets. The exploration parameter β_t is set to the value prescribed by each algorithm.

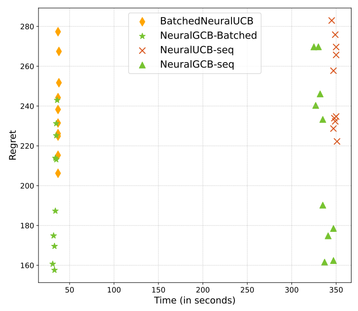
We consider a 2 layered neural net for all the experiments as described in Equation (2). We carry two sets of experiments each with different activation functions, namely σ_1 (or equivalently, ReLU) and σ_2. For the experiments with σ_1 as the activation, we set the number of hidden neurons to $m = 20$ and $m = 50$ for synthetic and real datasets respectively. Similarly, for σ_2, m is set to 30 and 80 for synthetic and real datasets, respectively. For all the experiments, we perform a grid search for λ and η over $\{0.05, 0.1, 0.5\}$ and $\{0.001, 0.01, 0.1\}$, respectively, and choose the best ones for each algorithm. The number of epochs is set to 200 for synthetic datasets and Mushroom, and to 400 for Statlog. For the experiments with sequential algorithms, we retrain the neural nets at every step, including NeuralGCB. For Batched NeuralUCB, we use a fixed batch size of 10 for synthetic datasets, and 20 for Mushroom. For NeuralGCB we set batch size to $q_r = 5 \cdot 2^{r-1}$ for synthetic datasets, and $q_r = 5 \cdot 2^{r+1}$ for Mushroom. More details and additional experiments are given in the supplementary material.



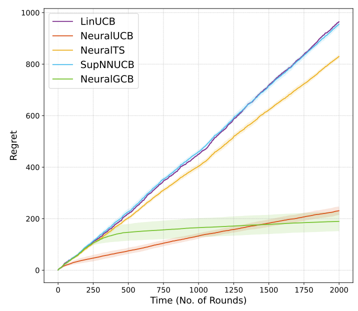
(a) $h_1(x)$ with $\sigma_1(x)$



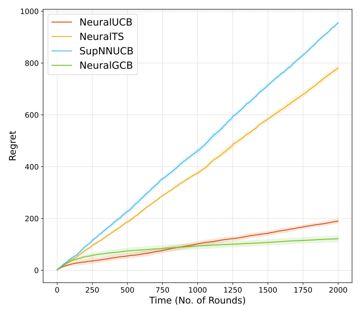
(b) $h_1(x)$ with $\sigma_2(x)$



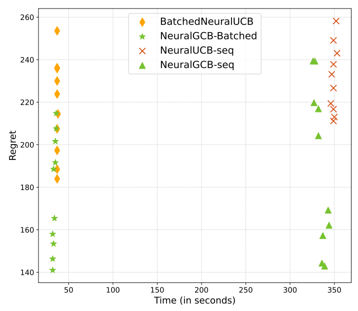
(c) Batched algorithms on $h_1(x)$



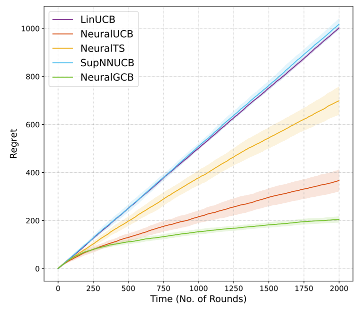
(d) $h_2(x)$ with $\sigma_1(x)$



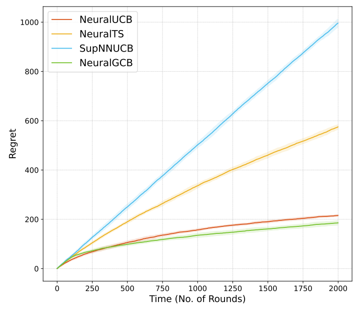
(e) $h_2(x)$ with $\sigma_2(x)$



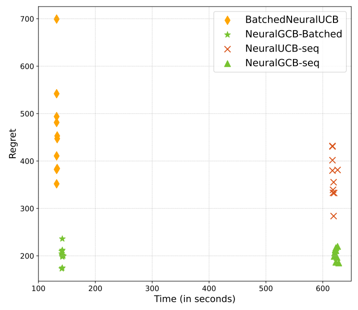
(f) Batched algorithms on $h_2(x)$



(g) Mushroom with $\sigma_1(x)$



(h) Mushroom with $\sigma_2(x)$



(i) Batched algorithms on Mushroom

Figure 1: First, second and third rows correspond to the reward functions $h_1(x)$, $h_2(x)$ and the Mushroom dataset, respectively. The two leftmost columns show the cumulative regret incurred by the algorithms against number of steps, with σ_1 activation functions for the first column and σ_2 for the second. The rightmost column compares the regret incurred and the time taken (in seconds) for batched and sequential versions of NeuralUCB and NeuralGCB.

5.3. Results

The two leftmost columns in Fig. 1show that NeuralGCB outperforms other algorithms in the case of both synthetic and real world datasets, corroborating the theoretical claims. That holds true for both set of experiments with different activation functions, further bolstering the practical efficiency of the proposed algorithm. In addition, the regret incurred by the algorithms in experiments with σ_2 as the activation is less than that for the experiments with σ_1 as the activation, demonstrating the effect of smooth kernels on the performance of the algorithms in practice.

In the third column, we compare the regret and running time between the batched and the sequential versions of NeuralUCB and NeuralGCB. We plot the regret incurred against time taken for different training schedules for 10 different runs. For all functions, the regret incurred by the batched version is comparable to that of the sequential version while having a significantly less running time. Furthermore, NeuralGCB has a smaller regret compared to Batched NeuralUCB for comparable running times.

6. Conclusion

In this work, we considered the problem of neural contextual bandits with general set of smooth activation functions. We established non-asymptotic error bounds on the difference between an overparametrized neural net and its corresponding NT kernel along with confidence bounds for prediction using neural nets under this general setting. Furthermore, we proposed a new algorithm that incurs sublinear regret under this general setting is also efficient in practice, as demonstrated by extensive empirical studies.

7. Acknowledgements

This work was supported by the Government of Israel - Israeli Ministry of Defense, Mission to the USA.

References

Abbasi-Yadkori, Y., Pál, D., and Szepesvári, C. Improved algorithms for linear stochastic bandits. *Advances in neural information processing systems*, 24, 2011.

Allen-Zhu, Z., Li, Y., and Song, Z. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning*, pp. 242–252. PMLR, 2019.

Arora, S., Du, S. S., Hu, W., Li, Z., Salakhutdinov, R. R., and Wang, R. On exact computation with an infinitely wide neural net. *Advances in Neural Information Processing Systems*, 32, 2019.

Auer, P. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3(Nov):397–422, 2002.

Auer, P., Cesa-Bianchi, N., and Fischer, P. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2):235–256, 2002.

Boucheron, S., Lugosi, G., and Massart, P. *Concentration Inequalities - A Nonasymptotic Theory of Independence*. Oxford University Press, 2013. ISBN 978-0-19-953525-5. doi: 10.1093/acprof:oso/9780199535255.001.0001. URL https://doi.org/10.1093/acprof:oso/9780199535255.001.0001.

Bouneffouf, D. and Rish, I. A survey on practical applications of multi-armed and contextual bandits. *arXiv preprint arXiv:1904.10040*, 2019.

Calandriello, D., Carratino, L., Lazaric, A., Valko, M., and Rosasco, L. Gaussian process optimization with adaptive sketching: Scalable and no regret. In *Conference on Learning Theory*, pp. 533–557. PMLR, 2019.

Camilleri, R., Jamieson, K., and Katz-Samuels, J. High-dimensional experimental design and kernel bandits. In *International Conference on Machine Learning*, pp. 1227–1237. PMLR, 2021.

Cao, Y. and Gu, Q. Generalization bounds of stochastic gradient descent for wide and deep neural networks. *Advances in neural information processing systems*, 32, 2019.

Cao, Y. and Gu, Q. Generalization error bounds of gradient descent for learning over-parameterized deep relu networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 3349–3356, 2020.

Chen, C., Luo, L., Zhang, W., Yu, Y., and Lian, Y. Efficient and robust high-dimensional linear contextual bandits. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pp. 4259–4265, 2021.

Chowdhury, S. R. and Gopalan, A. On kernelized multi-armed bandits. In *International Conference on Machine Learning*, pp. 844–853, 2017.

Chu, W., Li, L., Reyzin, L., and Schapire, R. Contextual bandits with linear payoff functions. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 208–214. JMLR Workshop and Conference Proceedings, 2011.

Cramér, H. Sur un nouveau théorème-limite de la théorie des probabilités. Actual. sci. industr. 736, 5-23. (Confér. internat. Sci. math. Univ. Genève. Théorie des probabilités. III: Les sommes et les fonctions de variables aléatoires.) (1938)., 1938.

Daniely, A., Frostig, R., and Singer, Y. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *Advances in Neural Information Processing Systems*, pp. 2261–2269, 2016.

Du, S., Lee, J., Li, H., Wang, L., and Zhai, X. Gradient descent finds global minima of deep neural networks. In *International conference on machine learning*, pp. 1675–1685. PMLR, 2019.

Dua, D. and Graff, C. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Gantert, N., Ramanan, K., and Rembart, F. Large deviations for weighted sums of stretched exponential random variables. *Electronic Communications in Probability*, 19, 2014. ISSN 1083589X. doi: 10.1214/ECP.v19-3266.

Gu, Q., Karbasi, A., Khosravi, K., Mirrokni, V., and Zhou, D. Batched neural bandits. *arXiv preprint arXiv:2102.13028*, 2021.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. *arXiv preprint arXiv:1806.07572*, 2018.

Kassraie, P. and Krause, A. Neural contextual bandits without regret. In *AISTATS*, 2022.

Langford, J. and Zhang, T. The epoch-greedy algorithm for multi-armed bandits with side information. *Advances in neural information processing systems*, 20, 2007.

Lattimore, T. and Szepesvári, C. *Bandit algorithms*. Cambridge University Press, 2020.

Li, B., Tang, S., and Yu, H. Better approximations of high dimensional smooth functions by deep neural networks with rectified power units. *arXiv preprint arXiv:1903.05858*, 2019a.

Li, L., Chu, W., Langford, J., and Schapire, R. E. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web, WWW '10*, pp. 661–670, feb 2010. ISBN 9781605587998. doi: 10.1145/1772690.1772758. URL http://arxiv.org/abs/1003.0146http://dx.doi.org/10.1145/1772690.1772758.

Li, Y., Wang, Y., and Zhou, Y. Nearly minimax-optimal regret for linearly parameterized bandits. In *Conference on Learning Theory*, pp. 2173–2174. PMLR, 2019b.

Li, Z. and Scarlett, J. Gaussian process bandit optimization with few batches. In *AISTATS*, 2022.

Liu, C., Zhu, L., and Belkin, M. On the linearity of large non-linear models: when and why the tangent kernel is constant. *Advances in Neural Information Processing Systems*, 33:15954–15964, 2020.

Opschoor, J., Schwab, C., and Zech, J. Exponential relu dnn expression of holomorphic maps in high dimension. *Constructive Approximation*, 55:1–46, 02 2022. doi: 10.1007/s00365-021-09542-5.

Riquelme, C., Tucker, G., and Snoek, J. Deep Bayesian bandits showdown: An empirical comparison of Bayesian deep networks for Thompson sampling. In *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, 2018. URL https://sites.google.com/site/deepbayesianbandits/.

Salgia, S., Vakili, S., and Zhao, Q. A domain-shrinking based Bayesian optimization algorithm with order-optimal regret performance. *Conference on Neural Information Processing Systems*, 34, 2021.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., and De Freitas, N. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104 (1):148–175, 2015.

Sitzmann, V., Martel, J., Bergman, A., Lindell, D., and Wetzstein, G. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33:7462–7473, 2020.

Snoek, J., Larochelle, H., and Adams, R. P. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25, 2012.

Srinivas, N., Krause, A., Kakade, S., and Seeger, M. Gaussian process optimization in the bandit setting: no regret and experimental design. In *International Conference on Machine Learning*, pp. 1015–1022. Omnipress, 2010.

Vakili, S., Bouziani, N., Jalali, S., Bernacchia, A., and Shiu, D.-s. Optimal order simple regret for Gaussian process bandits. In *Conference on Neural Information Processing Systems*, 2021a.

Vakili, S., Bromberg, M., Garcia, J., Shiu, D.-s., and Bernacchia, A. Uniform generalization bounds for overparameterized neural networks. *arXiv preprint arXiv:2109.06099*, 2021b.

Vakili, S., Scarlett, J., and Javidi, T. Open problem: Tight online confidence intervals for RKHS elements. In *Conference on Learning Theory*, pp. 4647–4652. PMLR, 2021c.

Valko, M., Korda, N., Munos, R., Flaounas, I., and Cristianini, N. Finite-time analysis of kernelised contextual bandits. *arXiv preprint arXiv:1309.6869*, 2013.

Wang, T., Zhou, D., and Gu, Q. Provably Efficient Reinforcement Learning with Linear Function Approximation Under Adaptivity Constraints, 2021. URL http://arxiv.org/abs/2101.02195.

Zhang, T. Learning bounds for kernel regression using effective data dimensionality. *Neural Computation*, 17 (9):2077–2098, 2005.

Zhang, W., Zhou, D., Li, L., and Gu, Q. Neural Thompson sampling. In *International Conference on Learning Representations*, 2021.

Zhou, D., Li, L., and Gu, Q. Neural contextual bandits with UCB-based exploration. In *International Conference on Machine Learning*, pp. 11492–11502. PMLR, 2020.

Appendix

We present the proofs of the theorems and lemmas stated in the main paper, as well as additional empirical results, in the appendix. The appendix is structured as follows: In Appendix A, we provide the proof of Theorem 1, while we defer the proof of auxiliary lemmas to Appendix B. Proof of Theorem 2 is given in Appendix C. Further details on NeuralGCB and proof of Theorem 3 are provided in Appendix D. Lastly, the details on empirical studies and further results are reported in Appendix E.

A. Proof of Theorem 1

Before we provide the proof of Theorem 1, we first set up some preliminaries and notation that would be useful throughout the proof.

A.1. Proof Preliminaries

A.1.1. Notation

For any $n \in \mathbb{N}$, $[n]$ denotes the set $\{1, 2, \dots, n\}$. For any vector $\mathbf{v} \in \mathbb{R}^n$, $\mathrm{diag}(\mathbf{v})$ is the $\mathbb{R}^{n \times n}$ diagonal matrix with the elements on $\mathbf{v}$ on its diagonal entries. $\|\mathbf{v}\|$ denotes the L_2 norm of the vector $\mathbf{v}$. $\|\mathbf{M}\|_2$ and $\|\mathbf{M}\|_F$ denotes the spectral and Frobenius norms respectively of a matrix $\mathbf{M}$. For events A and B, we define $A \Rightarrow B := \neg A \vee B$. For matrix $\mathbf{A}$, we denote the projection matrix for the column space of $\mathbf{A}$ by $\mathbf{\Pi_A} := \mathbf{A}\mathbf{A}^\dagger$, where $\mathbf{A}^\dagger$ denotes the pseudo-inverse of $\mathbf{A}$. Similarly, we denote the orthogonal projection matrix as $\mathbf{\Pi}_{\mathbf{A}}^{\perp} := \mathbf{I} - \mathbf{A}\mathbf{A}^\dagger$. For two random variables, X and Y, $X \overset{d}{=}_A Y$ means X is equal to Y in distribution conditioned on the σ-algebra generated by the event A. For any $\rho \in [-1, 1]$, we use Σ_ρ to denote the matrix $\begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$. Lastly, for $n \in \mathbb{N}$, we define $(2n-1)!! = \prod_{i=1}^{n}(2i-1)$. For example $3!! = 3$ and $5!! = 15$.

A.1.2. Fully connected Neural Network

In this proof, we consider a general fully connected neural net consisting of L hidden layers defined recursively as follows:

$$\mathbf{f}^{(l)}(\mathbf{x}) = \mathbf{W}^{(l)}\mathbf{h}^{(h-1)}(\mathbf{x}) \in \mathbb{R}^{d_l}, \quad \mathbf{h}^{(l)}(\mathbf{x}) = \sqrt{\frac{c_\sigma}{d_l}}\sigma\left(\mathbf{f}^{(l)}(\mathbf{x})\right) \in \mathbb{R}^{d_l}, \quad l = 1, 2, \dots, L, \tag{10}$$

where $\mathbf{h}^{(0)}(\mathbf{x}) = \mathbf{x}$ and $\mathbf{x} \in \mathcal{X}$ is the input to the network. In the above expression, $\mathbf{W}^{(l)} \in \mathbb{R}^{d_l \times d_{l-1}}$ is the weight matrix in the l^{th} layer for $l \in [L]$ and $d_0 = d$, $\sigma(\cdot) : \mathbb{R} \to \mathbb{R}$ is a coordinate-wise activation function and the constant $c_\sigma := \left(\mathbb{E}_{z \sim \mathcal{N}(0,1)}[\sigma(z)^2]\right)^{-1}$. The output of the neural network is given by its last layer defined as follows:

$$f(\mathbf{x}; \mathbf{W}) = \mathbf{f}^{(L+1)}(\mathbf{x}) = \mathbf{W}^{(L+1)}\mathbf{h}^{(L)}(\mathbf{x}), \tag{11}$$

where $\mathbf{W}^{(L+1)} \in \mathbb{R}^{1 \times d_L}$ is the weight matrix in the final layer, and $\mathbf{W} = (\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \dots, \mathbf{W}^{(L+1)})$ represents all the parameters of the network. Recall that the domain is assumed to be the hypersphere $\mathbb{S}^{d-1}$. Consequently, $\|\mathbf{x}\| = 1$ for all $\mathbf{x} \in \mathcal{X}$. This is just the generalization of the of the neural net defined in eqn. (2) with possibly different width in each layer.

The partial derivative of the output of the neural network with respect to a particular weight matrix is given as

$$\frac{\partial f(\mathbf{x}; \mathbf{W})}{\partial \mathbf{W}^{(l)}} = \mathbf{b}^{(l)}(\mathbf{x}) \cdot \left(\mathbf{h}^{(l-1)}(\mathbf{x})\right)^\top, \quad l = 1, 2, \dots, L+1, \tag{12}$$

where $\mathbf{b}^{(l)}(\mathbf{x})$ is defined recursively as

$$\mathbf{b}^{(l)}(\mathbf{x}) = \begin{cases} 1 & l = L+1, \\ \sqrt{\frac{c_\sigma}{d_l}}\mathbf{D}^{(l)}(\mathbf{x})\left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)}(\mathbf{x}) & l = 1, 2, \dots, L. \end{cases} \tag{13}$$

In the above definition,

$$\mathbf{D}^{(l)}(\mathbf{x}) := \mathrm{diag}\left(\sigma'\left(\mathbf{f}^{(l)}(\mathbf{x})\right)\right) \in \mathbb{R}^{d_l \times d_l}, \tag{14}$$

is a diagonal matrix, where $\sigma'(\cdot)$ is the derivative of the activation function σ and is also applied coordinate wise. Note that $\mathbf{b}^{(l)}(\mathbf{x}) \in \mathbb{R}^{d_l}$ for $l = 1, 2, \ldots, L$ and $\mathbf{b}^{(L+1)}(\mathbf{x}) \in \mathbb{R}$. In other words, $\mathbf{b}^{(l)}(\mathbf{x})$ is the gradient of output of the neural network $f(\mathbf{x}; \mathbf{W})$ with respect to $\mathbf{f}^{(l)}$, the pre-activation of layer l.

A.2. Neural Tangent Kernel

In the infinite width limit, the pre-activation functions $\mathbf{f}^{(l)}$ at every hidden layer $l \in [L]$ has all its coordinates tending to i.i.d. centered Gaussian Processes with covariance matrix $\Sigma^{(l-1)} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ defined recursively for $l \in [L]$ as:

$$\begin{aligned}
&\Sigma^{(0)}(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{x}', \\
&\Lambda^{(l)}(\mathbf{x}, \mathbf{x}') = \begin{bmatrix} \Sigma^{(l-1)}(\mathbf{x}, \mathbf{x}) & \Sigma^{(l-1)}(\mathbf{x}, \mathbf{x}') \\ \Sigma^{(l-1)}(\mathbf{x}', \mathbf{x}) & \Sigma^{(l-1)}(\mathbf{x}', \mathbf{x}') \end{bmatrix}, \\
&\Sigma^{(l)}(\mathbf{x}, \mathbf{x}') = c_\sigma \mathbb{E}_{(u,v)\sim\mathcal{N}(0,\Lambda^{(l)}(\mathbf{x},\mathbf{x}'))}[\sigma(u)\sigma(v)].
\end{aligned} \tag{15}$$

Similar to $\Sigma^{(l)}(\mathbf{x}, \mathbf{x}')$, we also define $\dot{\Sigma}^{(l)}(\mathbf{x}, \mathbf{x}')$ as follows:

$$\dot{\Sigma}^{(l)}(\mathbf{x}, \mathbf{x}') = c_\sigma \mathbb{E}_{(u,v)\sim\mathcal{N}(0,\Lambda^{(l)}(\mathbf{x},\mathbf{x}'))}[\sigma'(u)\sigma'(v)], \tag{16}$$

for $l \in [L]$ and $\dot{\Sigma}^{(L+1)}(\mathbf{x}, \mathbf{x}') = 1$ for $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$. The final NTK expression for the fully-connected network is given as

$$\Theta^{(L)}(\mathbf{x}, \mathbf{x}') = \sum_{l=1}^{L+1} \left(\Sigma^{(l-1)}(\mathbf{x}, \mathbf{x}') \prod_{j=l}^{L+1} \dot{\Sigma}^{(j)}(\mathbf{x}, \mathbf{x}') \right). \tag{17}$$

Note that this is same as $k(\mathbf{x}, \mathbf{x}')$ referred to in the main text.

A.3. The Activation Function

In this work, we assume that the activation function $\sigma \in \mathcal{A}_\sigma$, where $\mathcal{A}_\sigma = \{\sigma_s(x) : s \in \mathbb{N}\}$ and $\sigma_s(x) = (\max(0, x))^s$. Note that $\sigma_1(x)$ corresponds to the popular ReLU activation function and existing results hold only for $\sigma_1(x)$.

We state some definitions which we will be later used to establish certain properties of activation functions in $\mathcal{A}$.

Fact 1. *(Vakili et al. (2021b)) The normalizing constant* $c_{\sigma_s} = \dfrac{2}{(2s-1)!!}$ *for all* $\sigma_s \in \mathcal{A}$.

For simplicity of notation we use c_s instead of c_{σ_s} for the rest of the proof.

Definition 1. *A function* $f : \mathbb{R} \to \mathbb{R}$ *is said to be* α*-homogeneous, if* $f(\lambda x) = \lambda^\alpha f(x)$ *for all* $x \in \mathbb{R}$ *and* $\lambda > 0$.

It is straightforward to note that $\sigma_s(x)$ is s-homogeneous.

Let $\mathcal{M}_+(d)$ denote the set of positive semi-definite matrices of dimension d, that is, $\mathcal{M}_+ = \{\mathbf{M} \in \mathbb{R}^{d\times d} : \mathbf{x}^\top \mathbf{M} \mathbf{x} \geq 0 \ \forall\, \mathbf{x} \in \mathbb{R}^d\}$. Similarly, we use $\mathcal{M}_{++}(d)$ to denote the class of positive definite matrices of dimension d. For $\gamma \in [0, 1]$, we denote

$$\mathcal{M}_+^\gamma = \left\{ \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{12} & \Sigma_{22} \end{pmatrix} \in \mathcal{M}_+(2) \middle| 1 - \gamma \leq \Sigma_{11}, \Sigma_{22} \leq 1 + \gamma \right\}. \tag{18}$$

Definition 2. *The dual of an activation function* $\sigma(\cdot)$ *is the function* $\bar{\sigma} : [-1, 1] \to \mathbb{R}$ *defined as*

$$\bar{\sigma}(\rho) = c_\sigma \mathbb{E}_{(X,Y)\sim\mathcal{N}(0,\Sigma_\rho)}[\sigma(X)\sigma(Y)]. \tag{19}$$

Note that from definition of c_σ, $\bar{\sigma}(\rho) \in [-1, 1]$ and $\bar{\sigma}(1) = 1$.

Fact 2 (Daniely et al. (2016), Lemma 11)**.** *The dual* $\bar{\sigma}$ *is continuous in* $[-1, 1]$*, smooth in* $(-1, 1)$*, convex in* $[0, 1]$ *and is non-decreasing.*

The dual of an activation function can be extended to $\breve{\sigma} : \mathcal{M}_+(2) \to \mathbb{R}$ as $\breve{\sigma}(\Sigma) = c_\sigma \mathbb{E}_{(X,Y)\sim\mathcal{N}(0,\Sigma)}[\sigma(X)\sigma(Y)]$. Note that if $\Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{12} & \Sigma_{22} \end{pmatrix}$ and σ is k-homogeneous, we have,

$$\breve{\sigma}(\Sigma) = (\Sigma_{11}\Sigma_{22})^{k/2} \bar{\sigma}\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right).$$

Let $\bar{\sigma}_s$ be the dual function of $\sigma_s \in \mathcal{A}_\sigma$ for all $s \in \mathbb{N}$ and $\bar{\mathcal{A}}_\sigma = \{\bar{\sigma}_s : s \in \mathbb{N}\}$ denote the set of dual functions. It is not difficult to note that $\bar{\sigma}_s(-1) = 0$ and $\bar{\sigma}_s(1) = 1$ for all $s \in \mathbb{N}$. Since $\bar{\sigma}_s$ is non-decreasing, $\bar{\sigma}_s(\rho) \in [0,1]$ for all $\rho \in [-1,1]$.

Fact 3 (Vakili et al. (2021b), Lemma 1)**.** *The functions in $\bar{\mathcal{A}}_\sigma$ satisfy,*

$$\bar{\sigma}_s'(\rho) = \frac{s^2}{2s-1}\bar{\sigma}_{s-1}(\rho) \tag{20}$$

for $s > 1$. Here $\bar{\sigma}_s'$ denotes the derivative of $\bar{\sigma}_s$.

Consequently, $|\bar{\sigma}_s'(\rho)| \leq \frac{s^2}{2s-1}|\bar{\sigma}_{s-1}(\rho)| \leq \frac{s^2}{2s-1}$. Thus, $\bar{\sigma}_s$ is $s^2/(2s-1)$ Lipschitz.

Definition 3. *For any function $\sigma_s \in \mathcal{A}_\sigma$, we define $\mu_{s,\rho} := \mathbb{E}_{(X,Y)\sim\mathcal{N}(0,\Sigma_\rho)}[\sigma_s(X)\sigma_s(Y)] = \frac{\bar{\sigma}_s}{c_s}$.*

A.4. Proof of Lemma 1

The central piece in the proof of Theorem 1 is Lemma 1. We focus our attention on first proving Lemma 1. Since the proof is involved, we first provide an outline of the proof to give the reader an overview of the approach before delving into the technical details.

Informally, Lemma 1 states that for sufficiently large network widths, the following relation holds with probability of at least $1-\delta$ over the random initialization of the network weights.

$$\left|\left\langle \frac{\partial f(\mathbf{x};\mathbf{W})}{\partial \mathbf{W}}, \frac{\partial f(\mathbf{x}';\mathbf{W})}{\partial \mathbf{W}} \right\rangle - \Theta^{(L)}(\mathbf{x},\mathbf{x}')\right| \leq (L+1)\varepsilon.$$

Firstly, note that

$$\left\langle \frac{\partial f(\mathbf{x};\mathbf{W})}{\partial \mathbf{W}}, \frac{\partial f(\mathbf{x}';\mathbf{W})}{\partial \mathbf{W}} \right\rangle = \sum_{l=1}^{L+1} \left\langle \frac{\partial f(\mathbf{x};\mathbf{W})}{\partial \mathbf{W}^{(l)}}, \frac{\partial f(\mathbf{x}';\mathbf{W})}{\partial \mathbf{W}^{(l)}} \right\rangle$$

and recall that

$$\Theta^{(L)}(\mathbf{x},\mathbf{x}') = \sum_{l=1}^{L+1}\left(\Sigma^{(l-1)}(\mathbf{x},\mathbf{x}') \prod_{j=l}^{L+1} \dot{\Sigma}^{(j)}(\mathbf{x},\mathbf{x}')\right).$$

Using these relations, note that it is sufficient to show that,

$$\left|\left\langle \frac{\partial f(\mathbf{x};\mathbf{W})}{\partial \mathbf{W}^{(l)}}, \frac{\partial f(\mathbf{x}';\mathbf{W})}{\partial \mathbf{W}^{(l)}} \right\rangle - \Sigma^{(l-1)}(\mathbf{x},\mathbf{x}') \prod_{j=l}^{L+1} \dot{\Sigma}^{(j)}(\mathbf{x},\mathbf{x}')\right| \leq \varepsilon \tag{21}$$

holds for all $l \in [L]$ with probability $1-\delta$. Furthermore, we have,

$$\begin{aligned}\left\langle \frac{\partial f(\mathbf{x};\mathbf{W})}{\partial \mathbf{W}^{(l)}}, \frac{\partial f(\mathbf{x}';\mathbf{W})}{\partial \mathbf{W}^{(l)}} \right\rangle &= \left\langle \mathbf{b}^{(l)}(\mathbf{x}) \cdot \left(\mathbf{h}^{(l-1)}(\mathbf{x})\right)^\top, \mathbf{b}^{(l)}(\mathbf{x}') \cdot \left(\mathbf{h}^{(l-1)}(\mathbf{x}')\right)^\top \right\rangle \\ &= \left\langle \mathbf{h}^{(l-1)}(\mathbf{x}), \mathbf{h}^{(l-1)}(\mathbf{x})' \right\rangle \left\langle \mathbf{b}^{(l)}(\mathbf{x}), \mathbf{b}^{(l)}(\mathbf{x}') \right\rangle.\end{aligned}$$

The proof revolves around establishing $\langle \mathbf{h}^{(l-1)}(\mathbf{x}), \mathbf{h}^{(l-1)}(\mathbf{x}')\rangle$ is close to $\Sigma^{(l-1)}(\mathbf{x},\mathbf{x}')$ while $\langle \mathbf{b}^{(l)}(\mathbf{x}), \mathbf{b}^{(l)}(\mathbf{x}')\rangle$ is close to $\prod_{j=l}^{L+1} \dot{\Sigma}^{(j)}(\mathbf{x},\mathbf{x}')$. On combining these two relations, we obtain the result in (21) and consequently prove the theorem.

Throughout the proof, we fix some $s \in \mathbb{N}$ and hence $\sigma = \sigma_s$. Recall from equation (15), we have,

$$\begin{aligned}
\Sigma^{(0)}(\mathbf{x},\mathbf{x}') &= \mathbf{x}^\top \mathbf{x}', \\
\Lambda^{(l)}(\mathbf{x},\mathbf{x}') &= \begin{bmatrix} \Sigma^{(l-1)}(\mathbf{x},\mathbf{x}) & \Sigma^{(l-1)}(\mathbf{x},\mathbf{x}') \\ \Sigma^{(l-1)}(\mathbf{x}',\mathbf{x}) & \Sigma^{(l-1)}(\mathbf{x}',\mathbf{x}') \end{bmatrix}, \\
\Sigma^{(l)}(\mathbf{x},\mathbf{x}') &= c_s \mathbb{E}_{(u,v)\sim\mathcal{N}(0,\Lambda^{(l)}(\mathbf{x},\mathbf{x}'))}[\sigma_s(u)\sigma_s(v)].
\end{aligned}$$

Since $\|\mathbf{x}\| = 1$ for all $x \in \mathcal{X}$, $\Sigma^{(0)}(\mathbf{x},\mathbf{x}) = 1$ for all $x \in \mathcal{X}$. Using induction, we can establish that $\Sigma^{(l)}(\mathbf{x},\mathbf{x}) = 1$ for all $x \in \mathcal{X}$, for all $l \in \{0,1,2,\dots,L\}$. The base case follows immediately as $\Sigma^{(0)}(\mathbf{x},\mathbf{x}) = 1$ for all $x \in \mathcal{X}$. Assume true for $l-1$. Consequently, we have, $\Lambda^{(l)}(\mathbf{x},\mathbf{x}) = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$. On plugging this value in the definition of $\Sigma^{(l)}(\mathbf{x},\mathbf{x})$, we obtain $\Sigma^{(l)}(\mathbf{x},\mathbf{x}) = 1$, completing the inductive step. As a result, $|\Sigma^{(l)}(\mathbf{x},\mathbf{x}')| \leq 1$ for all $\mathbf{x},\mathbf{x}' \in \mathcal{X}$ and hence we can write $\Sigma^{(l)}(\mathbf{x},\mathbf{x}') = \bar{\sigma}_s(\Sigma^{(l-1)}(\mathbf{x},\mathbf{x}'))$.

As the final step before the proof, we define a sequence of events which will be used throughout the proof. Let $\mathbf{\Delta}^{(l)}(\mathbf{x},\mathbf{x}') := \mathbf{D}^{(l)}(\mathbf{x})\mathbf{D}^{(l)}(\mathbf{x}')$. We define the following events:

- $\mathcal{A}^l(\mathbf{x},\mathbf{x}',\varepsilon_1) = \left\{ \left| \left(\mathbf{h}^{(l)}(\mathbf{x})\right)^\top \mathbf{h}^{(l)}(\mathbf{x}') - \Sigma^{(l)}(\mathbf{x},\mathbf{x}') \right| \leq \varepsilon_1 \right\}$,
- $\bar{\mathcal{A}}^l(\mathbf{x},\mathbf{x}',\varepsilon_1) = \mathcal{A}^l(\mathbf{x},\mathbf{x}',\varepsilon_1) \cap \mathcal{A}^l(\mathbf{x},\mathbf{x},\varepsilon_1) \cap \mathcal{A}^l(\mathbf{x}',\mathbf{x}',\varepsilon_1)$
- $\bar{\mathcal{A}}(\mathbf{x},\mathbf{x}',\varepsilon_1) = \bigcap_{l=0}^{L} \bar{\mathcal{A}}^l(\mathbf{x},\mathbf{x}',\varepsilon_1)$
- $\mathcal{B}^l(\mathbf{x},\mathbf{x}',\varepsilon_2) = \left\{ \left| \left\langle \mathbf{b}^{(l)}(\mathbf{x}), \mathbf{b}^{(l)}(\mathbf{x}') \right\rangle - \prod_{j=l}^{L+1} \dot{\Sigma}^{(j)}(\mathbf{x},\mathbf{x}') \right| \leq \varepsilon_2 \right\}$
- $\bar{\mathcal{B}}^l(\mathbf{x},\mathbf{x}',\varepsilon_2) = \mathcal{B}^l(\mathbf{x},\mathbf{x}',\varepsilon_2) \cap \mathcal{B}^l(\mathbf{x},\mathbf{x},\varepsilon_2) \cap \mathcal{B}^l(\mathbf{x}',\mathbf{x}',\varepsilon_2)$
- $\bar{\mathcal{B}}(\mathbf{x},\mathbf{x}',\varepsilon_2) = \bigcap_{l=1}^{L+1} \bar{\mathcal{B}}^l(\mathbf{x},\mathbf{x}',\varepsilon_2)$
- $\bar{\mathcal{C}}(\mathbf{x},\mathbf{x}',\varepsilon_3) = \{|f(\mathbf{x};\mathbf{W})| \leq \varepsilon_3, |f(\mathbf{x}';\mathbf{W})| \leq \varepsilon_3\}$
- $\mathcal{D}^l(\mathbf{x},\mathbf{x}',\varepsilon_4) = \left\{ \left| c_s \frac{\mathrm{tr}(\mathbf{\Delta}^{(l)}(\mathbf{x},\mathbf{x}'))}{d_l} - \dot{\Sigma}^{(l)}(\mathbf{x},\mathbf{x}') \right| < \varepsilon_4 \right\}$
- $\bar{\mathcal{D}}^l(\mathbf{x},\mathbf{x}',\varepsilon_4) = \mathcal{D}^l(\mathbf{x},\mathbf{x}',\varepsilon_4) \cap \mathcal{D}^l(\mathbf{x},\mathbf{x},\varepsilon_4) \cap \mathcal{D}^l(\mathbf{x}',\mathbf{x}',\varepsilon_4)$
- $\bar{\mathcal{D}}(\mathbf{x},\mathbf{x}',\varepsilon_4) = \bigcap_{l=1}^{L+1} \bar{\mathcal{D}}^l(\mathbf{x},\mathbf{x}',\varepsilon_4)$
- $\mathcal{E}^l(\mathbf{x},\mathbf{x}',\varepsilon_5) = \left\{ \|\mathbf{\Delta}^{(l)}(\mathbf{x},\mathbf{x}')\|_2 < \varepsilon_5 \right\}$
- $\bar{\mathcal{E}}^l(\mathbf{x},\mathbf{x}',\varepsilon_5) = \mathcal{E}^l(\mathbf{x},\mathbf{x}',\varepsilon_4) \cap \mathcal{E}^l(\mathbf{x},\mathbf{x},\varepsilon_4) \cap \mathcal{E}^l(\mathbf{x}',\mathbf{x}',\varepsilon_5)$
- $\bar{\mathcal{E}}(\mathbf{x},\mathbf{x}',\varepsilon_5) = \bigcap_{l=1}^{L+1} \bar{\mathcal{E}}^l(\mathbf{x},\mathbf{x}',\varepsilon_5)$

We also state the following two Lemmas taken from Arora et al. (2019) which are used at several points in the proof before beginning with the first part.

Lemma 2. *For any two events, A and B, $\Pr(A \Rightarrow B) \geq \Pr(B|A)$.*

Lemma 3. *Let $\mathbf{w} \sim \mathcal{N}(0, \mathbf{I}_d)$, $\mathbf{G} \in \mathbb{R}^{d \times k}$ be a fixed matrix and $\mathbf{F} = \mathbf{w}^\top \mathbf{G}$ be a random matrix. Then, conditioned on the value of $\mathbf{F}$, $\mathbf{w}$ remains Gaussian in the null space of the row space of $\mathbf{G}$. Mathematically,*

$$\mathbf{\Pi}_{\mathbf{G}}^{\perp} \mathbf{w} \overset{d}{=}_{\mathbf{F} = \mathbf{w}^\top \mathbf{G}} \mathbf{\Pi}_{\mathbf{G}}^{\perp} \tilde{\mathbf{w}},$$

where $\tilde{\mathbf{w}}$ is a i.i.d. copy of $\mathbf{w}$.

A.4.1. Pre-activations are close to the NTK matrix

In the first step, we show that $\left\langle \mathbf{h}^{(l-1)}(\mathbf{x}), \mathbf{h}^{(l-1)}(\mathbf{x}') \right\rangle$ is close to $\Sigma^{(l-1)}(\mathbf{x}, \mathbf{x}')$. We formalize this idea in the following lemma.

Lemma 4. *For $\sigma(z) = \sigma_s(z)$ and $[\mathbf{W}^{(l)}]_{ij} \overset{i.i.d.}{\sim} \mathcal{N}(0, 1)$ for all $i \in [d_{l+1}], j \in [d_l]$ and $l \in \{0, 1, 2, \ldots, L\}$. There exist constants $c_1, c_2 > 0$ such that if $\min_l d_l \geq c_1 \frac{s^L L^2 c_s^2}{\varepsilon^2} \log^2(sL/\delta\varepsilon)$ and $\varepsilon \leq c_2/s$, then for fixed $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$,*

$$\left| \left\langle \mathbf{h}^{(l)}(\mathbf{y}), \mathbf{h}^{(l)}(\mathbf{y}') \right\rangle - \Sigma^{(l)}(\mathbf{y}, \mathbf{y}') \right| \leq \varepsilon,$$

holds with probability at least $1 - \delta$ for all $l \in \{0, 1, \ldots, L\}$ and all $(\mathbf{y}, \mathbf{y}') \in \{(\mathbf{x}, \mathbf{x}), (\mathbf{x}, \mathbf{x}'), (\mathbf{x}', \mathbf{x}')\}$. In other words, $\Pr(\bar{\mathcal{A}}(\varepsilon)) \geq 1 - \delta$, for fixed $\mathbf{x}, \mathbf{x}'$.

Proof. The proof of this Lemma relies on following lemmas.

Lemma 5. *Let $\rho \in [-1, 1]$ and $(X, Y) \sim \mathcal{N}(0, \Sigma_\rho)$. If $(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)$ are independent samples from $\mathcal{N}(0, \Sigma_\rho)$ and $S_n = \frac{1}{n} \sum_{i=1}^n \sigma_s(X_i)\sigma_s(Y_i)$ for some $\sigma_s \in \mathcal{A}$, then for any $t \geq 0$,*

$$\Pr(S_n - \mu_s \geq t) \leq \begin{cases} (n+1) \exp\left(-\dfrac{nt^2}{2\sqrt{3\mu_{4s,\rho}}}\right) & \text{if } t \leq t^*(n), \\ (n+1) \exp\left(-\dfrac{(nt)^{1/s}}{8(1+\rho)}\right) & \text{if } t > t^*(n). \end{cases}$$

$$\Pr(S_n - \mu_s \leq -t) \leq \exp\left(-\frac{nt^2}{2\mu_{2s,\rho}}\right),$$

where $t^*(n) = \left(\dfrac{\sqrt{3\mu_{4s,\rho}}}{4(1+\rho)}\right)^{2-1/s} n^{-\left(\frac{s-1}{2s-1}\right)}$. Consequently, if $n \geq n_*(s, \delta, \rho)$, then

$$\Pr\left(|S_n - \mu_{s,\rho}| \geq \sqrt{\frac{2(\sqrt{3\mu_{4s,\rho}} + \mu_{2s,\rho})}{n} \log\left(\frac{n+1}{\delta}\right)}\right) \leq \delta,$$

where $n_*(s, \delta, \rho) := \min\left\{ m \in \mathbb{N} : m \geq \dfrac{(8(1+\rho))^{2s}}{2\sqrt{3\mu_{4s,\rho}}} \left[\log\left(\dfrac{m+1}{\delta}\right)\right]^{2s-1} \right\} + 1$

For simplicity of notation, we define $\varphi_s(n, \delta, \rho) := \sqrt{\frac{2(\sqrt{3\mu_{4s,\rho}} + \mu_{2s,\rho})}{n} \log\left(\frac{n+1}{\delta}\right)}$. Thus, the result of Lemma 5 can be restated as $\Pr(|S_n - \mu_{s,\rho}| \geq \varphi_s(n, \delta, \rho)) \leq \delta$.

Lemma 6. *For a given $s \in \mathbb{N}$, the dual activation function $\breve{\sigma}_s$ is β_s-Lipschitz in $\mathcal{M}_+^{\gamma_s}$ w.r.t. the ∞-norm for $\gamma_s \leq 1/s$ and $\beta_s \leq 6s$.*

Lemma 7 ((Daniely et al., 2016))**.** *For $\sigma(z) = \sigma_s(z)$ and $[\mathbf{W}^{(l)}]_{ij} \overset{i.i.d.}{\sim} \mathcal{N}(0,1)$ for all $i \in [d_{l+1}], j \in [d_l]$ and $l \in \{0,1,2,\ldots,L\}$. If $\min_{l\in[L]} d_l \geq n_s^*\left(\frac{\varepsilon}{B_{L,s}}, \frac{\delta}{8d}\right)$, then for fixed $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$,*

$$\left|\left\langle \mathbf{h}^{(l)}(\mathbf{y}), \mathbf{h}^{(l)}(\mathbf{y}')\right\rangle - \Sigma^{(l)}(\mathbf{y},\mathbf{y}')\right| \leq \varepsilon,$$

holds with probability at least $1-\delta$ for all $l \in \{0,1,\ldots,L\}$ and all $(\mathbf{y},\mathbf{y}') \in \{(\mathbf{x},\mathbf{x}),(\mathbf{x},\mathbf{x}'),(\mathbf{x}',\mathbf{x}')\}$. In the above expression, $B_{L,s} = \sum_{j=0}^{L-1}\beta_s^i$, $\bar{d} = \sum_{l=1}^{L+1} d_l$ and $n_s^*(\varepsilon,\delta) = \min\{n : \varphi_s(n,\delta) \leq \varepsilon\}$.

Lemma 4 follows immediately from Lemma 7 which in turn follows from the previous lemmas. The proofs of Lemma 5 and 6 are provided in Appendix B.

□

A.4.2. Pre-activation gradients are close to NTK derivative matrices

In the second step, we show that $\langle \mathbf{b}^{(l)}(\mathbf{x}), \mathbf{b}^{(l)}(\mathbf{x}')\rangle$ is close to $\prod_{j=l}^{L+1}\dot{\Sigma}^{(j)}(\mathbf{x},\mathbf{x}')$. We formalize this idea in the following lemma.

Lemma 8. *For $\sigma = \sigma_s$ and $[\mathbf{W}^{(l)}]_{ij} \overset{i.i.d.}{\sim} \mathcal{N}(0,1)$ for all $i \in [d_{l+1}], j \in [d_l]$ and $l \in \{0,1,2,\ldots,L\}$. There exist constants $c_1, c_2 > 0$ such that if $\min_l d_l \geq c_1 \frac{s^L L^2 c_s^2}{\varepsilon^2}\log^2(sL/\delta\varepsilon)$ and $\varepsilon \leq c_2 s^{-L+2}/L$, then for fixed $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$,*

$$\left|\left\langle \mathbf{b}^{(l)}(\mathbf{y}), \mathbf{b}^{(l)}(\mathbf{y}')\right\rangle - \prod_{j=l}^{L+1}\dot{\Sigma}^{(j)}(\mathbf{y},\mathbf{y}')\right| \leq L(\beta_s+1)s^{L+1}\varepsilon,$$

holds with probability at least $1-\delta$ for all $l \in \{0,1,\ldots,L\}$ and all $(\mathbf{y},\mathbf{y}') \in \{(\mathbf{x},\mathbf{x}),(\mathbf{x},\mathbf{x}'),(\mathbf{x}',\mathbf{x}')\}$. In other words, if $\min_l d_l \geq c_1\frac{s^L L^2 c_s^2}{\varepsilon_1^2}\log^2(sL/\delta\varepsilon_1)$ and $\varepsilon_1 \leq c_2 s^{-L+2}/L$, then for fixed $\mathbf{x}, \mathbf{x}'$, $\Pr\left(\bar{\mathcal{A}}(\varepsilon_1/2) \wedge \bar{\mathcal{B}}(L(\beta_s+1)s^{L+1}\varepsilon_1)\right) \geq 1-\delta$.

Proof. We first state some helper lemmas that will be used in the proof followed by the proof of the above lemma.

Lemma 9 (Arora et al. (2019), Lemma E.4)**.**

$$\Pr\left[\bar{\mathcal{A}}^L(\varepsilon/2) \Rightarrow \bar{\mathcal{C}}\left(2\sqrt{\log\left(\frac{4}{\delta}\right)}\right)\right] \geq 1-\delta \qquad \forall\, \varepsilon \in [0,1],\ \forall\, \delta \in (0,1).$$

Lemma 10. *If $\bar{\mathcal{A}}^L(\varepsilon_1/2) \wedge \bar{\mathcal{B}}^{l+1}(\varepsilon_2) \wedge \bar{\mathcal{C}}(\varepsilon_3) \wedge \bar{\mathcal{D}}^l(\varepsilon_4) \wedge \bar{\mathcal{E}}^l(\varepsilon_5)$ for any pair of points $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$, then for any $(\mathbf{y},\mathbf{y}') \in \{(\mathbf{x},\mathbf{x}),(\mathbf{x},\mathbf{x}'),(\mathbf{x}',\mathbf{x}')\}$, we have*

$$\left|c_s\frac{\mathrm{tr}(\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}'))}{d_l}\left\langle \mathbf{b}^{(l+1)}(\mathbf{y}), \mathbf{b}^{(l+1)}(\mathbf{y}')\right\rangle - \prod_{j=l}^{L}\dot{\Sigma}^{(l)}(\mathbf{y},\mathbf{y}')\right| \leq s^{L-l}\varepsilon_4 + s\varepsilon_2.$$

Lemma 11. *If $\bar{\mathcal{A}}^L(\varepsilon_1/2) \wedge \bar{\mathcal{B}}^{l+1}(\varepsilon_2) \wedge \bar{\mathcal{C}}(\varepsilon_3) \wedge \bar{\mathcal{D}}^l(\varepsilon_4) \wedge \bar{\mathcal{E}}^l(\varepsilon_5)$ for any pair of points $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$, then for any $(\mathbf{y},\mathbf{y}') \in \{(\mathbf{x},\mathbf{x}),(\mathbf{x},\mathbf{x}'),(\mathbf{x}',\mathbf{x}')\}$*

$$\begin{aligned}
&\left|\frac{c_s}{d_l}\left(\mathbf{b}^{(l+1)}(\mathbf{y})\right)^\top \mathbf{W}^{(l+1)}\mathbf{\Pi}_{\mathbf{G}}^{\perp}\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\mathbf{\Pi}_{\mathbf{G}}^{\perp}\left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)}(\mathbf{y}) - \frac{c_s}{d_l}\mathrm{tr}(\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}'))\left\langle \mathbf{b}^{(l+1)}(\mathbf{y}), \mathbf{b}^{(l+1)}(\mathbf{y}')\right\rangle\right| \\
&\leq c_s\left(\left\langle \mathbf{b}^{(l+1)}(\mathbf{y}), \mathbf{b}^{(l+1)}(\mathbf{y})\right\rangle + \left\langle \mathbf{b}^{(l+1)}(\mathbf{y}'), \mathbf{b}^{(l+1)}(\mathbf{y}')\right\rangle\right)\left(\sqrt{\frac{2}{d_l}\log\left(\frac{3}{\delta}\right)} + \frac{1}{d_l}\log\left(\frac{3}{\delta}\right)\right)\varepsilon_5 \\
&\qquad + \frac{2c_s}{d_l}\left\langle \mathbf{b}^{(l+1)}(\mathbf{y}), \mathbf{b}^{(l+1)}(\mathbf{y}')\right\rangle\varepsilon_5.
\end{aligned}$$

holds with probability at least $1-\delta$. *Consequently, for any* $\mathbf{y} \in \{\mathbf{x}, \mathbf{x}'\}$,

$$\sqrt{\frac{c_s}{d_l}} \left\| \left(\mathbf{b}^{(l+1)}(\mathbf{y})\right)^\top \mathbf{W}^{(l+1)} \mathbf{\Pi}_{\mathbf{G}}^{\perp} \mathbf{D}^{(l)}(\mathbf{y}) \right\| \le \sqrt{2 c_s \left\langle \mathbf{b}^{(l+1)}(\mathbf{y}), \mathbf{b}^{(l+1)}(\mathbf{y}) \right\rangle \varepsilon_5}.$$

Lemma 12. *If* $\bar{\mathcal{A}}^L(\varepsilon_1/2) \wedge \bar{\mathcal{B}}^{l+1}(\varepsilon_2) \wedge \bar{\mathcal{C}}(\varepsilon_3) \wedge \bar{\mathcal{D}}^l(\varepsilon_4) \wedge \bar{\mathcal{E}}^l(\varepsilon_5)$ *for any pair of points* $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$, *then for any* $(\mathbf{y}, \mathbf{y}') \in \{(\mathbf{x}, \mathbf{x}), (\mathbf{x}, \mathbf{x}'), (\mathbf{x}', \mathbf{x}')\}$

$$\begin{aligned}
\frac{c_s}{d_l} \Bigg| & \left(\mathbf{b}^{(l+1)}(\mathbf{y})\right)^\top \mathbf{W}^{(l+1)} \mathbf{\Delta}^{(l)}(\mathbf{y}, \mathbf{y}') \left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)}(\mathbf{y}') \\
& - \left(\mathbf{b}^{(l+1)}(\mathbf{y})\right)^\top \mathbf{W}^{(l+1)} \mathbf{\Pi}_{\mathbf{G}}^{\perp} \mathbf{\Delta}^{(l)}(\mathbf{y}, \mathbf{y}') \mathbf{\Pi}_{\mathbf{G}}^{\perp} \left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)}(\mathbf{y}') \Bigg| \le \\
\frac{c_s \varepsilon_5}{\sqrt{d_l}} & \left[(s^{(L-l)/2} + \varepsilon_2) \sqrt{2 \log\left(\frac{8}{\delta}\right)} + s^{L-l} \varepsilon_3 \sqrt{2} \right] \times \\
& \left\{ \frac{1}{\sqrt{d_l}} \left[(s^{(L-l)/2} + 1) \sqrt{2 \log\left(\frac{8}{\delta}\right)} + s^{L-l} \varepsilon_3 \sqrt{2} \right] + \sqrt{8} (s^{(L-l)/2} + 1) \right\}
\end{aligned}$$

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

Lemma 13. *For any* $\varepsilon_1 \in [0, 1/s]$, $\varepsilon_2, \varepsilon_4 \in [0, 1]$ *and* $\varepsilon_3, \varepsilon_5 \ge 0$,

$$\Pr\left[\bar{\mathcal{A}}^L(\varepsilon_1/2) \wedge \bar{\mathcal{B}}^{l+1}(\varepsilon_2) \wedge \bar{\mathcal{C}}(\varepsilon_3) \wedge \bar{\mathcal{D}}^l(\varepsilon_4) \wedge \bar{\mathcal{E}}^l(\varepsilon_5) \Rightarrow \bar{\mathcal{B}}^l(\varepsilon')\right] \ge 1 - \delta,$$

where

$$\begin{aligned}
\varepsilon' = {} & 2 c_s (s^{L-l} + 1) \left(\sqrt{\frac{2}{d_l} \log\left(\frac{6}{\delta}\right)} + \frac{1}{d_l} \log\left(\frac{6}{\delta}\right) \right) \varepsilon_5 + \frac{2 c_s}{d_l} (s^{L-l} + 1) \varepsilon_5 + s^{L-l} \varepsilon_4 + s \varepsilon_2 \\
& + \frac{c_s \varepsilon_5}{\sqrt{d_l}} \left[(s^{(L-l)/2} + 1) \sqrt{2 \log\left(\frac{16}{\delta}\right)} + s^{L-l} \varepsilon_3 \sqrt{2} \right] \times \\
& \left\{ \frac{1}{\sqrt{d_l}} \left[(s^{(L-l)/2} + 1) \sqrt{2 \log\left(\frac{16}{\delta}\right)} + s^{L-l} \varepsilon_3 \sqrt{2} \right] + \sqrt{8} (s^{(L-l)/2} + 1) \right\}.
\end{aligned}$$

We now prove Lemma 8 by using induction on Lemma 13. Before the inductive step, we note some relations that will be useful to us during the analysis. Firstly, using Lemma 4, we can conclude that if $d_l \ge n_s^*(\varepsilon/B_{L,s}, \delta/32\bar{d}(L+1))$ and $\varepsilon \le c_2/s$, then

$$\Pr\left[\bar{\mathcal{A}}^L(\varepsilon/2)\right] \ge 1 - \delta/4. \tag{22}$$

From Lemma 9, we can conclude that

$$\Pr\left[\bar{\mathcal{A}}^L(\varepsilon/2) \Rightarrow \bar{\mathcal{C}}\left(2\sqrt{\log\left(\frac{16}{\delta}\right)}\right)\right] \ge 1 - \delta/4. \tag{23}$$

If $d_l \ge n_s^*(\varepsilon/3, \delta/24(L+1))$, then we can conclude that $3s^2 \varphi_{s-1}((\min_l d_l), \delta/24(L+1)) + \frac{s^2 \beta_s \varepsilon}{2s-1} \le (\beta+1)s^2\varepsilon$. Consequently for $\varepsilon \le 2/s$,

$$\Pr\left[\bar{\mathcal{A}}(\varepsilon/2) \Rightarrow \bar{\mathcal{D}}((\beta_s + 1)s^2\varepsilon) \wedge \bar{\mathcal{E}}\left(\varepsilon''\right)\right] \ge 1 - \delta/4, \tag{24}$$

where $\varepsilon'' = 3s^2\left[2\log\left(\frac{6(L+1)(\max_l d_l)}{\delta}\right)\right]^{s-1}$. Applying the union bound on (22), (23) and (24), we obtain that

$$\Pr\left[\bar{\mathcal{A}}(\varepsilon/2)\wedge\bar{\mathcal{C}}\left(2\sqrt{\log\left(\frac{16}{\delta}\right)}\right)\wedge\bar{\mathcal{D}}((\beta_s+1)s^2\varepsilon)\wedge\bar{\mathcal{E}}\left(\varepsilon''\right)\right]\geq 1-3\delta/4. \tag{25}$$

Let d_l be chosen large enough to ensure that

$$\begin{aligned}s^{L-l+2}\varepsilon \geq\ & 2c_s(s^{L-l}+1)\left(\sqrt{\frac{2}{d_l}\log\left(\frac{24(L+1)}{\delta}\right)}+\frac{1}{d_l}\left(+\log\left(\frac{24(L+1)}{\delta}\right)\right)\right)\varepsilon''\\ &+c_s\varepsilon''\sqrt{\frac{8}{d_l}}(s^{(L-l)/2}+1)\left[(s^{(L-l)/2}+1)\sqrt{2\log\left(\frac{64(L+1)}{\delta}\right)}+4s^{L-l}\sqrt{\log\left(\frac{16(L+1)}{\delta}\right)}\right]\\ &+\frac{c_s\varepsilon''}{d_l}\left[(s^{(L-l)/2}+1)\sqrt{2\log\left(\frac{64(L+1)}{\delta}\right)}+4s^{L-l}\sqrt{\log\left(\frac{16(L+1)}{\delta}\right)}\right]^2.\end{aligned}$$

Define $\upsilon_l := (L+1-l)(\beta_s+1)s^{L-l+2}\varepsilon$ for $l=0,1,\dots,L+1$. Invoking Lemma 13 with $\varepsilon_1=\varepsilon$, $\varepsilon_2=\upsilon_{l+1}$, $\varepsilon_3 = 2\sqrt{\log\left(\frac{16}{\delta}\right)}$, $\varepsilon_4=(\beta_s+1)s^2\varepsilon$ and $\varepsilon_5=\varepsilon''$ gives us that

$$\Pr\left[\bar{\mathcal{A}}^L(\varepsilon/2)\wedge\bar{\mathcal{B}}^{l+1}(\upsilon_{l+1})\wedge\bar{\mathcal{C}}\left(2\sqrt{\log\left(\frac{16}{\delta}\right)}\right)\wedge\bar{\mathcal{D}}^l((\beta_s+1)s^2\varepsilon)\wedge\bar{\mathcal{E}}^l(\varepsilon'')\Rightarrow\bar{\mathcal{B}}^l(\upsilon_l)\right]\geq 1-\frac{\delta}{4(L+1)}.$$

Thus,

$$\begin{aligned}&\Pr\left[\bar{\mathcal{A}}(\varepsilon/2)\wedge\bar{\mathcal{B}}(\upsilon_1)\right]\\ &\geq\Pr\left[\underbrace{\bar{\mathcal{A}}(\varepsilon/2)\wedge\bar{\mathcal{C}}\left(2\sqrt{\log\left(\frac{16}{\delta}\right)}\right)\wedge\bar{\mathcal{D}}((\beta_s+1)s^2\varepsilon)\wedge\bar{\mathcal{E}}(\varepsilon'')}_{:=\mathcal{F}(\varepsilon)}\wedge\left(\bigcap_{l=1}^{L+1}\bar{\mathcal{B}}^l(\upsilon_l)\right)\right]\\ &\geq\Pr\left[\mathcal{F}(\varepsilon)\wedge\bar{\mathcal{B}}^{L+1}(\upsilon_{L+1})\wedge\bigcap_{l=0}^{L}\left\{\mathcal{F}(\varepsilon)\wedge\left(\bigcap_{j=L+1-l}^{L+1}\bar{\mathcal{B}}^{j+1}(\upsilon_{j+1})\right)\Rightarrow\bar{\mathcal{B}}^{L-l}(\upsilon_{L-l})\right\}\right]\\ &\geq\Pr\Bigg[\mathcal{F}(\varepsilon)\wedge\bar{\mathcal{B}}^{L+1}(\upsilon_{L+1})\wedge\\ &\qquad\bigcap_{l=0}^{L}\left\{\bar{\mathcal{A}}^L(\varepsilon/2)\wedge\bar{\mathcal{C}}\left(2\sqrt{\log\left(\frac{16}{\delta}\right)}\right)\wedge\bar{\mathcal{D}}^l((\beta_s+1)s^2\varepsilon)\wedge\bar{\mathcal{E}}^l(\varepsilon'')\wedge\bar{\mathcal{B}}^{l+1}(\upsilon_{l+1})\Rightarrow\bar{\mathcal{B}}^l(\upsilon_l)\right\}\Bigg]\\ &\geq 1-\Pr\left[\neg\left\{\bar{\mathcal{A}}(\varepsilon/2)\wedge\bar{\mathcal{C}}\left(2\sqrt{\log\left(\frac{16}{\delta}\right)}\right)\wedge\bar{\mathcal{D}}((\beta_s+1)s^2\varepsilon)\wedge\bar{\mathcal{E}}(\varepsilon'')\right\}\right]-\Pr\left[\neg\bar{\mathcal{B}}^{L+1}(\upsilon_{L+1})\right]\\ &\quad-\sum_{l=0}^{L}\Pr\left[\neg\left\{\bar{\mathcal{A}}^L(\varepsilon/2)\wedge\bar{\mathcal{C}}\left(2\sqrt{\log\left(\frac{16}{\delta}\right)}\right)\wedge\bar{\mathcal{D}}^h((\beta_s+1)s^2\varepsilon)\wedge\bar{\mathcal{E}}^l(\varepsilon'')\wedge\bar{\mathcal{B}}^{l+1}(\upsilon_{l+1})\Rightarrow\bar{\mathcal{B}}^l(\upsilon_l)\right\}\right]\\ &\geq 1-\frac{3\delta}{4}-\sum_{l=0}^{L}\frac{\delta}{4(L+1)}\\ &\geq 1-\delta.\end{aligned}$$

Thus, $\Pr\left[\bar{\mathcal{A}}(\varepsilon/2)\wedge\bar{\mathcal{B}}(L(\beta_s+1)s^{L+1}\varepsilon)\right]\geq 1-\delta$, as required. The proof of the helper lemmas are provided in Appendix B.

□

Now, it is straightforward to note that Lemma 1 immediately follows from the Lemma 4 and Lemma 8. With Lemma 1 in place, the rest of the proof of Theorem 1 follows similar to the proof of Theorem 3.2 in Arora et al. (2019). The only step in the proof of Theorem 3.2 that is dependent on kernel being ReLU, is Lemma F.6, where they bound the perturbation in gradient based on the amount of perturbation in the weight matrices. We would like to emphasize that this is only step apart from the ones which use Theorem 3.1. Such steps follow immediately by invoking Lemma 1 proven above, instead of Theorem 3.1. Using the result from Liu et al. (2020, Theorem 3.2), we know that the spectral norm of the Hessian of the neural net is $\mathcal{O}(m^{-1/2})$. Here the implied constant only depends on s and L. Thus for the a perturbation in the weight matrices of the order of $\mathcal{O}(\sqrt{m}\omega)$, the corresponding perturbation in the gradient is $\mathcal{O}(\omega)$, with the implied constant depending only on s and L. On substituting this relation in place of Lemma F.6, the claim of Theorem 1 follows using the same arguments as the ones used in the proof of Theorem 3.2 of Arora et al. (2019).

B. Proof of Helper Lemmas

We first state some additional intermediate lemmas which are used in the proofs several helper lemmas then provide a proof of all the lemmas.

Definition 4.

$$\mathbf{G}^{(l)}(\mathbf{x},\mathbf{x}') = [\mathbf{h}^{(l)}(\mathbf{x}) \;\; \mathbf{h}^{(l)}(\mathbf{x}')] \in \mathbb{R}^{d_l \times 2}$$

$$\tilde{\mathbf{G}}^{(l)}(\mathbf{x},\mathbf{x}') = [\mathbf{G}^{(l)}(\mathbf{x},\mathbf{x}')]^\top \mathbf{G}^{(l)}(\mathbf{x},\mathbf{x}') = \begin{bmatrix} \langle \mathbf{h}^{(l)}(\mathbf{x}), \mathbf{h}^{(l)}(\mathbf{x}) \rangle & \langle \mathbf{h}^{(l)}(\mathbf{x}), \mathbf{h}^{(l)}(\mathbf{x}') \rangle \\ \langle \mathbf{h}^{(l)}(\mathbf{x}'), \mathbf{h}^{(l)}(\mathbf{x}) \rangle & \langle \mathbf{h}^{(l)}(\mathbf{x}'), \mathbf{h}^{(l)}(\mathbf{x}') \rangle \end{bmatrix} \equiv \begin{bmatrix} G_{11} & G_{12} \\ G_{12} & G_{22} \end{bmatrix}$$

Lemma 14.

$$\Pr\left[\bar{\mathcal{A}}^l(\varepsilon) \Rightarrow \bar{\mathcal{D}}^{l+1}\left(3s^2\varphi_{s-1}(d_l, \delta/6, 1) + \frac{2s^2\beta_s\varepsilon}{2s-1}\right) \wedge \bar{\mathcal{E}}^{l+1}\left(3s^2\left[2\log\left(\frac{6d_l}{\delta}\right)\right]^{s-1}\right)\right] \geq 1-\delta$$
$$\forall\, \varepsilon \in [0, 1/s],\ \delta \in (0,1).$$

Lemma 15. *Let*

$$\varepsilon' = 3s^2\varphi_{s-1}((\min_l d_l), \delta/6(L+1), 1) + \frac{2s^2\beta_s\varepsilon}{2s-1}, \quad \varepsilon'' = 3s^2\left[2\log\left(\frac{6(L+1)(\max_l d_l)}{\delta}\right)\right]^{s-1}.$$

Then,

$$\Pr\left[\bar{\mathcal{A}}(\varepsilon) \Rightarrow \bar{\mathcal{D}}(\varepsilon') \wedge \bar{\mathcal{E}}(\varepsilon'')\right] \geq 1-\delta \quad \forall\, \varepsilon \in [0,1/s],\ \delta \in (0,1).$$

Lemma 16. *For any $1 \leq l \leq L$, any fixed $\{\mathbf{W}^{(i)}\}_{i=1}^{l-1}$ and $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$, with probability $1-\delta$ over the randomness of $\mathbf{W}^{(l)}$, we have,*

$$\left| c_s \frac{\operatorname{tr}(\mathbf{\Delta}^{(l)}(\mathbf{x},\mathbf{x}'))}{d_l} - \frac{s^2}{2s-1}\breve{\sigma}_{s-1}\left(\tilde{\mathbf{G}}^{(l)}(\mathbf{x},\mathbf{x}')\right)\right| \leq s^2 (G_{11}G_{22})^{\frac{(s-1)}{2}}\varphi_{s-1}(d_l, \delta/2, \rho_G),$$

and

$$\|\mathbf{\Delta}^{(l)}(\mathbf{x},\mathbf{x}')\|_2 \leq s^2\left[\left(\sqrt{G_{11}G_{22}} + G_{12}\right)\log\left(\frac{2d_l}{\delta}\right)\right]^{s-1},$$

where $\rho_G = \frac{G_{12}}{\sqrt{G_{11}G_{22}}}$.

Lemma 17 ((Boucheron et al., 2013))**.** *Let $\boldsymbol{\xi} \sim \mathcal{N}(0, \mathbf{I}_n)$ be an n-dimensional unit Gaussian random vector and $\mathbf{A} \in \mathbb{R}^{n\times n}$ be a symmetric matrix, then for any $t > 0$,*

$$\Pr\left(\left|\boldsymbol{\xi}^\top \mathbf{A}\boldsymbol{\xi} - \mathbb{E}\left[\boldsymbol{\xi}^\top \mathbf{A}\boldsymbol{\xi}\right]\right| > 2\|\mathbf{A}\|_F\sqrt{t} + 2\|\mathbf{A}\|_2 t\right) \leq \exp(-t),$$

or equivalently,

$$\Pr\left(\left|\boldsymbol{\xi}^\top \mathbf{A}\boldsymbol{\xi} - \mathbb{E}\left[\boldsymbol{\xi}^\top \mathbf{A}\boldsymbol{\xi}\right]\right| > t\right) \leq \exp\left(-\frac{t^2}{4\|\mathbf{A}\|_F^2 + 4\|\mathbf{A}\|_2}\right).$$

Lemma 18. *If $\bar{\mathcal{A}}^L(\varepsilon_1/2) \wedge \bar{\mathcal{B}}^{l+1}(\varepsilon_2) \wedge \bar{\mathcal{C}}(\varepsilon_3) \wedge \bar{\mathcal{D}}^l(\varepsilon_4) \wedge \bar{\mathcal{E}}^l(\varepsilon_5)$ for any pair of points $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$, then for any $\mathbf{y} \in \{\mathbf{x}, \mathbf{x}'\}$*

$$\left\|\boldsymbol{\Pi}_{\mathbf{G}}\left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)}(\mathbf{y})\right\| \leq (s^{(L-l)/2}+1)\sqrt{2\log\left(\frac{4}{\delta}\right)} + s^{L-l}\varepsilon_3\sqrt{2}.$$

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

B.1. Proof of Lemma 5

We fix a particular $s \in \mathbb{N}$. Let $(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)$ be n independent samples from $\mathcal{N}(0, \Sigma_\rho)$ and let $Z_i = \sigma_s(X_i)\sigma_s(Y_i)$. Define $S_n := \frac{1}{n}\sum_{i=1}^n Z_i$. Note that $\mathbb{E}[S_n] = \mathbb{E}[Z_1] = \mu_s$. To prove the bound, we first bound $\Pr(Z_i > t)$ for any $t \geq 0$ and then using techniques borrowed from Gantert et al. (2014), we obtain an upper bound on $\Pr(S_n - \mu_s > t)$. The bound on $\Pr(S_n - \mu_s < -t)$ follows from Cramer's Theorem (Cramér, 1938).

We begin with bounding $\Pr(Z_i > t)$ for any $t \geq 0$. Since $Z_i = (\sigma_1(X_i)\sigma_1(Y_i))^s$, we just focus on bounding $\Pr(\sigma_1(X)\sigma_1(Y) > t)$ for $(X, Y) \sim \mathcal{N}(0, \Sigma_\rho)$. Fix a $t \geq 0$. We have,

$$\begin{aligned}\Pr(\sigma_1(X)\sigma_1(Y) > t) &= \mathbb{E}\left[\mathbb{1}\left\{\sigma_1(X)\sigma_1(Y) > t\right\}\right] \\ &= \mathbb{E}\left[\mathbb{1}\left\{XY > t, X \geq 0, Y \geq 0\right\}\right].\end{aligned}$$

Using a change of variables define $U = (X+Y)/\sqrt{2}$ and $V = (X-Y)/\sqrt{2}$. Note that U and V are zero-mean Gaussian random variables with Cov$(U, V) = 0$ implying that they are independent. Also Var$(U) = 1+\rho$ and Var$(V) = 1-\rho$. Thus,

$$\begin{aligned}\Pr(\sigma_1(X)\sigma_1(Y) > t) &= \mathbb{E}_{X,Y}\left[\mathbb{1}\left\{XY > t, X \geq 0, Y \geq 0\right\}\right] \\ &= \mathbb{E}_{U,V}\left[\mathbb{1}\left\{U^2 - V^2 > 2t, U \geq 0, V \in [-U, U]\right\}\right] \\ &\leq \mathbb{E}_{U,V}\left[\mathbb{1}\left\{U^2 > V^2 + 2t, U \geq 0\right\}\right] \\ &\leq \mathbb{E}_V\left[\Pr\left(U > \sqrt{V^2 + 2t}\right)\right] \\ &\leq \mathbb{E}_V\left[\exp\left(-\frac{V^2 + 2t}{2(1+\rho)}\right)\right] \\ &\leq \frac{1}{\sqrt{2\pi(1-\rho)}}\int_{-\infty}^{\infty}\exp\left(-\frac{v^2+2t}{2(1+\rho)}\right)\exp\left(-\frac{v^2}{2(1-\rho)}\right)\,\mathrm{d}v \\ &\leq \frac{e^{-t/(1+\rho)}}{\sqrt{2\pi(1-\rho)}}\int_{-\infty}^{\infty}\exp\left(-\frac{v^2}{(1-\rho^2)}\right)\,\mathrm{d}v \\ &\leq \exp\left(-\frac{t}{1+\rho}\right)\sqrt{\frac{1+\rho}{2}} \\ &\leq \exp\left(-\frac{t}{1+\rho}\right).\end{aligned}$$

In the sixth line we used the concentration bound for Gaussian random variable and in the eighth line we used the Gaussian integral. Consequently,

$$\Pr(Z_i > t) = \Pr(\sigma_1(X)\sigma_1(Y) > t^{1/s}) \leq \exp\left(-\frac{t^{1/s}}{1+\rho}\right). \tag{26}$$

Using this relation, we now move on to bound $\Pr(S_n \geq x)$ for $x \geq \mu_{s,\rho}$. For the rest of the proof we drop the argument ρ for simplicity. Firstly, note that

$$\Pr(S_n \geq x) \leq \underbrace{\Pr\left(\max_{j\in[n]} Z_j \geq n(x-\mu_s)\right)}_{A_1^n} + \underbrace{\Pr\left(S_n \geq x,\ \max_{j\in[n]} Z_j < n(x-\mu_s)\right)}_{A_2^n}.$$

We begin with the first term.

$$\Pr\left(\max_{j\in[n]} Z_j \geq n(x-\mu_s)\right) \leq n\Pr(Z_i \geq n(x-\mu_s)) \leq n\exp\left(-\frac{n^{1/s}(x-\mu_s)^{1/s}}{1+\rho}\right).$$

Let $\beta_\zeta(n) = \frac{\zeta n^{1/s}}{1+\rho} > 0$ for some $\zeta > 0$ which will be specified later. We have,

$$\begin{aligned}
A_2^n &= \Pr\left(S_n \geq x,\ \max_{j\in[n]} Z_j < n(x-\mu_s)\right) \\
&= \Pr\left(\exp\left(\beta_\zeta(n)S_n\right) \geq \exp\left(\beta_\zeta(n)x\right),\ \max_{j\in[n]} Z_j < n(x-\mu_s)\right) \\
&\leq \exp\left(-\beta_\zeta(n)x\right)\mathbb{E}\left[\exp\left(\beta_\zeta(n)S_n\right)\mathbb{1}\left\{\max_{j\in[n]} Z_j < n(x-\mu_s)\right\}\right] \\
&\leq \exp\left(-\beta_\zeta(n)x\right)\prod_{j=1}^n \mathbb{E}\left[\exp\left(\frac{\beta_\zeta(n)Z_j}{n}\right)\mathbb{1}\left\{Z_j < n(x-\mu_s)\right\}\right] \\
\implies \log\left(A_2^n\right) &\leq -\beta_\zeta(n)x + \sum_{j=1}^n {\Gamma_\zeta}^{(j)}(n),
\end{aligned}$$

where ${\Gamma_\zeta}^{(j)}(n) = \log\left(\mathbb{E}\left[\exp\left(\frac{\beta_\zeta(n){Z_j}^{(n)}}{n}\right)\right]\right)$ and ${Z_j}^{(n)} := Z_j\mathbb{1}\left\{Z_j < n(x-\mu_s)\right\}$. Using the relations $\log(1+x) \leq x$ and $e^x \leq 1 + x + \frac{x^2}{2}e^x$, we have,

$$\begin{aligned}
\sum_{j=1}^n {\Gamma_\zeta}^{(j)}(n) &= \sum_{j=1}^n \log\left(\mathbb{E}\left[\exp\left(\frac{\beta_\zeta(n){Z_j}^{(n)}}{n}\right)\right]\right) \\
&\leq \sum_{j=1}^n \log\left(1 + \mathbb{E}\left[\frac{\beta_\zeta(n){Z_j}^{(n)}}{n}\right] + \frac{1}{2}\mathbb{E}\left[\left(\frac{\beta_\zeta(n){Z_j}^{(n)}}{n}\right)^2\exp\left(\frac{\beta_\zeta(n){Z_j}^{(n)}}{n}\right)\right]\right) \\
&\leq \sum_{j=1}^n \left\{\mathbb{E}\left[\frac{\beta_\zeta(n){Z_j}^{(n)}}{n}\right] + \frac{1}{2}\mathbb{E}\left[\left(\frac{\beta_\zeta(n){Z_j}^{(n)}}{n}\right)^2\exp\left(\frac{\beta_\zeta(n){Z_j}^{(n)}}{n}\right)\right]\right\}.
\end{aligned}$$

Note that,

$$\begin{aligned}
\sum_{j=1}^n \mathbb{E}\left[\frac{\beta_\zeta(n){Z_j}^{(n)}}{n}\right] &= \sum_{j=1}^n \frac{\beta_\zeta(n)}{n}\mathbb{E}\left[{Z_j}^{(n)}\right] \\
&\leq \beta_\zeta(n)\mu_s.
\end{aligned}$$

For the second term we have,

$$\mathbb{E}\left[\left(\frac{\beta_\zeta(n){Z_1}^{(n)}}{n}\right)^2\exp\left(\frac{\beta_\zeta(n){Z_1}^{(n)}}{n}\right)\right] \leq \left(\frac{\beta_\zeta(n)}{n}\right)^2\mathbb{E}\left[\left({Z_1}^{(n)}\right)^{\frac{2(1+\eta)}{\eta}}\right]^{\frac{\eta}{1+\eta}}\mathbb{E}\left[\exp\left(\frac{(1+\eta)\beta_\zeta(n){Z_1}^{(n)}}{n}\right)\right]^{\frac{1}{1+\eta}}$$

for some $\eta > 0$ by using Hölder's inequality. Since $\mathbb{E}\left[\left({Z_1}^{(n)}\right)^{2(1+\eta)/\eta}\right]^{\eta/(1+\eta)} = \left[\mu_{\frac{2s(1+\eta)}{\eta}}\right]^{\frac{\eta}{1+\eta}}$, we focus on the other term. We have,

$$\begin{aligned}
&\mathbb{E}\left[\exp\left(\frac{(1+\eta)\beta_\zeta(n){Z_1}^{(n)}}{n}\right)\right] \\
&= 1 + \int_0^{n(x-\mu_s)} \frac{(1+\eta)\beta_\zeta(n)}{n}\exp\left(\frac{(1+\eta)\beta_\zeta(n)z}{n}\right)\Pr(Z_1 > z)\,\mathrm{d}z \\
&= 1 + n(x-\mu_s)\int_0^1 \frac{(1+\eta)\beta_\zeta(n)}{n}\exp\left(\frac{(1+\eta)\beta_\zeta(n)n(x-\mu_s)y}{n}\right)\Pr(Z_1 > n(x-\mu_s)y)\,\mathrm{d}y \\
&\le 1 + (x-\mu_s)\int_0^1 (1+\eta)\beta_\zeta(n)\exp\left((1+\eta)\beta_\zeta(n)(x-\mu_s)y - \frac{(n(x-\mu_s)y)^{1/s}}{1+\rho}\right)\,\mathrm{d}y \\
&\le 1 + (x-\mu_s)(1+\eta)\beta_\zeta(n)\int_0^1 \exp\left(\beta_\zeta(n)\left[(1+\eta)(x-\mu_s)y - \frac{(x-\mu_s)^{1/s}y^{1/s}}{\zeta}\right]\right)\,\mathrm{d}y.
\end{aligned}$$

For $\zeta \le \dfrac{(x-\mu_s)^{1/s-1}}{2(1+\eta)}$, we have,

$$\begin{aligned}
&\mathbb{E}\left[\exp\left(\frac{(1+\eta)\beta_\zeta(n){Z_1}^{(n)}}{n}\right)\right] \\
&\le 1 + (x-\mu_s)(1+\eta)\beta_\zeta(n)\int_0^1 \exp\left(\beta_\zeta(n)\left[(1+\eta)(x-\mu_s)y - \frac{(x-\mu_s)^{1/s}y^{1/s}}{\zeta}\right]\right)\,\mathrm{d}y \\
&\le 1 + (x-\mu_s)(1+\eta)\beta_\zeta(n)\int_0^1 \exp\left(-\frac{1}{2}\beta_\zeta(n)(1+\eta)(x-\mu_s)y\right)\,\mathrm{d}y \\
&\le 3.
\end{aligned}$$

On combining all the equations, we obtain that for $\zeta \le \frac{(x-\mu_s)^{1/s-1}}{2(1+\eta)}$ and $\eta > 0$

$$\log\left(A_2^n\right) \le -\beta_\zeta(n)(x-\mu_s) + \frac{(\beta_\zeta(n))^2}{2n}\left(\left[\mu_{\frac{2s(1+\eta)}{\eta}}\right]^{\frac{\eta}{1+\eta}} 3^{\frac{1}{1+\eta}}\right)$$

We set $\eta = 1$. Thus for $\zeta \le 0.25(x-\mu_s)^{1/s-1}$, we have,

$$\log\left(A_2^n\right) \le -\frac{\zeta n^{1/s}(x-\mu_s)}{1+\rho} + \frac{\zeta^2 n^{2/s-1}}{2(1+\rho)^2}\sqrt{3\mu_{4s}}.$$

Note that the RHS is minimized at $\zeta_* = \dfrac{n^{1-1/s}(x-\mu_s)(1+\rho)}{\sqrt{3\mu_{4s}}}$. However, $\zeta_* \le 0.25(x-\mu_s)^{1/s-1}$ only for $(x-\mu_s) \le \underbrace{\left(\dfrac{\sqrt{3\mu_{4s}}}{4(1+\rho)}\right)^{2-1/s} n^{-\left(\frac{s-1}{2s-1}\right)}}_{:=t^*(n)}$. Thus, for $(x-\mu_s) \le t^*(n)$, we can plug in $\zeta = \zeta_*$ in the above expression to obtain

$$\log\left(A_2^n\right) \le -\frac{n(x-\mu_s)^2}{2\sqrt{3\mu_{4s}}}.$$

For $(x-\mu_s) > t^*(n)$, we plug in $\zeta = 0.25(x-\mu_s)^{1/s-1}$ to obtain

$$\log\left(A_2^n\right) \le -\frac{n^{1/s}(x-\mu_s)^{1/s}}{8(1+\rho)}.$$

On combining the two, we obtain that for any $t \geq 0$

$$\Pr(S_n - \mu_s \geq t) = \begin{cases} (n+1)\exp\left(-\dfrac{nt^2}{2\sqrt{3\mu_{4s}}}\right) & \text{if } t \leq t^*(n), \\ (n+1)\exp\left(-\dfrac{(nt)^{1/s}}{8(1+\rho)}\right) & \text{if } t > t^*(n). \end{cases}$$

We now consider the other side. Consider any $x \leq \mu_s$ and $\lambda > 0$. We have,

$$\begin{aligned} \Pr(S_n \leq x) &= \Pr\left(\exp(-\lambda S_n) \geq e^{-\lambda x}\right) \\ &\leq \mathbb{E}\left[\exp(-\lambda S_n)\right] e^{\lambda x} \\ &\leq \mathbb{E}\left[1 - \lambda S_n + \frac{\lambda^2 S_n^2}{2}\right] e^{\lambda x} \\ &\leq \left(1 - \lambda \mathbb{E}[S_n] + \frac{\lambda^2 \mathbb{E}[S_n^2]}{2}\right) e^{\lambda x} \\ &\leq \left(1 - \lambda\mu_s + \frac{\lambda^2 \mu_{2s}}{2n}\right) e^{\lambda x} \\ &\leq \exp\left(-\lambda\mu_s + \frac{\lambda^2\mu_{2s}}{2n} + \lambda x\right) \\ &\leq \exp\left(\lambda(x - \mu_s) + \frac{\lambda^2\mu_{2s}}{2n} + \lambda x\right). \end{aligned}$$

Minimizing this over λ yields $\lambda_* = -\dfrac{n(x-\mu_s)}{\mu_{2s}} > 0$. On plugging this value, we obtain,

$$\Pr(S_n \leq x) \leq \exp\left(-\frac{n(x-\mu_s)^2}{2\mu_{2s}}\right).$$

Consequently for any $t \geq 0$, we have,

$$\Pr(S_n - \mu_s \leq -t) \leq \exp\left(-\frac{nt^2}{2\mu_{2s}}\right).$$

On combining the relations, we can conclude that if $n \geq n_*(s, \delta, \rho)$, then

$$\Pr\left(|S_n - \mu_s| \geq \sqrt{\frac{2(\sqrt{3\mu_{4s}} + \mu_{2s})}{n}\log\left(\frac{n+1}{\delta}\right)}\right) \leq \delta.$$

B.2. Proof of Lemma 6

Fix a $s \in \mathbb{N}$. We drop the subscript s for the remainder of the proof for ease of notation. We need to show that the dual activation function $\breve{\sigma}$ is β-Lipschitz in $\mathcal{M}_+^\gamma$ w.r.t. the ∞-norm. Let $\Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{12} & \Sigma_{22} \end{pmatrix} \in \mathcal{M}_+^\gamma$. In order to prove β-Lipschitzness w.r.t. the ∞-norm, it is sufficient to show that,

$$\|\nabla \breve{\sigma}_s\|_1 = \left|\frac{\partial \breve{\sigma}_s}{\partial \Sigma_{12}}\right| + \left|\frac{\partial \breve{\sigma}_s}{\partial \Sigma_{11}}\right| + \left|\frac{\partial \breve{\sigma}_s}{\partial \Sigma_{22}}\right| \leq \beta.$$

Recall that since σ_s is s-homogeneous,

$$\breve{\sigma}_s(\Sigma) = (\Sigma_{11}\Sigma_{22})^{s/2}\bar{\sigma}_s\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right).$$

Using this relation, we have,

$$\begin{aligned}
\frac{\partial \breve{\sigma}_s}{\partial \Sigma_{12}} &= (\Sigma_{11}\Sigma_{22})^{\frac{s-1}{2}} \bar{\sigma}_s'\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right) \\
\frac{\partial \breve{\sigma}_s}{\partial \Sigma_{11}} &= \frac{s}{2}(\Sigma_{11}\Sigma_{22})^{\frac{s}{2}-1}\Sigma_{22}\bar{\sigma}_s\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right) - \frac{1}{2}(\Sigma_{11}\Sigma_{22})^{\frac{s-1}{2}}\bar{\sigma}_s'\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right)\frac{\Sigma_{12}}{\Sigma_{11}} \\
&= \Sigma_{22}\left\{\frac{s}{2}(\Sigma_{11}\Sigma_{22})^{\frac{s}{2}-1}\bar{\sigma}_s\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right) - \frac{1}{2}(\Sigma_{11}\Sigma_{22})^{\frac{s-1}{2}}\bar{\sigma}_s'\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right)\frac{\Sigma_{12}}{\Sigma_{11}\Sigma_{22}}\right\} \\
\frac{\partial \breve{\sigma}_s}{\partial \Sigma_{22}} &= \frac{s}{2}(\Sigma_{11}\Sigma_{22})^{\frac{s}{2}-1}\Sigma_{11}\bar{\sigma}_s\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right) - \frac{1}{2}(\Sigma_{11}\Sigma_{22})^{\frac{s-1}{2}}\bar{\sigma}_s'\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right)\frac{\Sigma_{12}}{\Sigma_{22}} \\
&= \Sigma_{11}\left\{\frac{s}{2}(\Sigma_{11}\Sigma_{22})^{\frac{s}{2}-1}\bar{\sigma}_s\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right) - \frac{1}{2}(\Sigma_{11}\Sigma_{22})^{\frac{s-1}{2}}\bar{\sigma}_s'\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right)\frac{\Sigma_{12}}{\Sigma_{11}\Sigma_{22}}\right\}.
\end{aligned}$$

Consequently,

$$\begin{aligned}
\|\nabla\breve{\sigma}_s\|_1 = (\Sigma_{11}\Sigma_{22})^{\frac{s-1}{2}}&\left|\bar{\sigma}_s'\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right)\right| + \\
&\frac{(\Sigma_{11}+\Sigma_{22})(\Sigma_{11}\Sigma_{22})^{\frac{s}{2}-1}}{2}\left|s\bar{\sigma}_s\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right) - \bar{\sigma}_s'\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right)\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right|.
\end{aligned}$$

For simplicity, let $\tilde{\rho} = \frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}$. For $s = 1$, Arora et al. (2019) showed that the above expression is $1 + o(\gamma)$. We consider the case for $s > 1$. Using (20), we have,

$$\begin{aligned}
\|\nabla\breve{\sigma}_s\|_1 &= \frac{s^2}{2s-1}(\Sigma_{11}\Sigma_{22})^{\frac{s-1}{2}}\left|\bar{\sigma}_{s-1}(\tilde{\rho})\right| + \frac{(\Sigma_{11}+\Sigma_{22})(\Sigma_{11}\Sigma_{22})^{\frac{s}{2}-1}}{2}\left|s\bar{\sigma}_s(\tilde{\rho}) - \frac{s^2}{2s-1}\bar{\sigma}_{s-1}(\tilde{\rho})\,\tilde{\rho}\right| \\
&\leq \frac{2s}{3}(\Sigma_{11}\Sigma_{22})^{\frac{s-1}{2}} + \frac{3s}{4}(\Sigma_{11}+\Sigma_{22})(\Sigma_{11}\Sigma_{22})^{\frac{s}{2}-1} \\
&\leq \frac{13s}{6}(1+\gamma)^{s-1}
\end{aligned}$$

If $\gamma \leq 1/s$, then $\|\nabla\breve{\sigma}_s\|_1 \leq 6s$, as required.

B.3. Proof of Lemma 10

We have,

$$\begin{aligned}
&\left|c_s\frac{\mathrm{tr}(\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}'))}{d_l}\left\langle \mathbf{b}^{(l+1)}(\mathbf{y}),\mathbf{b}^{(l+1)}(\mathbf{y}')\right\rangle - \prod_{j=l}^{L}\dot{\Sigma}^{(j)}(\mathbf{y},\mathbf{y}')\right| \\
&\leq \left|c_s\frac{\mathrm{tr}(\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}'))}{d_l} - \dot{\Sigma}^{(l)}(\mathbf{y},\mathbf{y}')\right|\left|\left\langle \mathbf{b}^{(l+1)}(\mathbf{y}),\mathbf{b}^{(l+1)}(\mathbf{y}')\right\rangle\right| + \\
&\qquad\left|\dot{\Sigma}^{(l)}(\mathbf{y},\mathbf{y}')\right|\left|\left\langle \mathbf{b}^{(l+1)}(\mathbf{y}),\mathbf{b}^{(l+1)}(\mathbf{y}')\right\rangle - \prod_{j=l+1}^{L}\dot{\Sigma}^{(j)}(\mathbf{y},\mathbf{y}')\right| \\
&\leq \left|\left\langle \mathbf{b}^{(l+1)}(\mathbf{y}),\mathbf{b}^{(l+1)}(\mathbf{y}')\right\rangle\right|\varepsilon_4 + \frac{s^2}{2s-1}\breve{\sigma}_{s-1}\left(\Lambda^{(l)}(\mathbf{x},\mathbf{x}')\right)\varepsilon_2
\end{aligned}$$

Since $\bar{\mathscr{B}}^{l+1}(\varepsilon_2)$, $\left\langle \mathbf{b}^{(l+1)}(\mathbf{y}), \mathbf{b}^{(l+1)}(\mathbf{y}')\right\rangle \le \prod_{j=l+1}^{L} \dot{\Sigma}^{(j)}(\mathbf{y},\mathbf{y}') + \varepsilon_2 \le \prod_{j=l+1}^{L} \dot{\Sigma}^{(j)}(\mathbf{y},\mathbf{y}') + 1$. Moreover, since $\dot{\Sigma}^{(l)}(\mathbf{y},\mathbf{y}') = \frac{s^2}{2s-1}\breve{\sigma}_{s-1}\left(\Lambda^{(l)}(\mathbf{y},\mathbf{y}')\right) \le \frac{s^2}{2s-1}$, we have,

$$\left| c_s \frac{\mathrm{tr}(\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}'))}{d_l}\left\langle \mathbf{b}^{(l+1)}(\mathbf{y}), \mathbf{b}^{(l+1)}(\mathbf{y}')\right\rangle - \prod_{j=l+1}^{L} \dot{\Sigma}^{(j)}(\mathbf{y},\mathbf{y}')\right| \le \left(\left(\frac{s^2}{2s-1}\right)^{L-l} + 1\right)\varepsilon_4 + s\varepsilon_2$$
$$\le s^{L-l}\varepsilon_4 + s\varepsilon_2.$$

B.4. Proof of Lemma 11

Fix any $(\mathbf{y},\mathbf{y}') \in \{(\mathbf{x},\mathbf{x}),(\mathbf{x},\mathbf{x}'),(\mathbf{x}',\mathbf{x}')\}$. Recall that $\mathbf{G}^{(l)}(\mathbf{y},\mathbf{y}') = [\mathbf{h}^{(l)}(\mathbf{y}) \;\; \mathbf{h}^{(l)}(\mathbf{y}')] \in \mathbb{R}^{d_l \times 2}$. Similarly define, $\mathbf{F}^{(l+1)}(\mathbf{y},\mathbf{y}') = [\mathbf{f}^{(l+1)}(\mathbf{y}) \;\; \mathbf{f}^{(l+1)}(\mathbf{y}')] \in \mathbb{R}^{d_l \times 2}$. Using Lemma 3 for each row of $\mathbf{W}^{(l+1)}$, we can conclude that

$$\mathbf{W}^{(l+1)}\mathbf{\Pi}_{\mathbf{G}}^{\perp} \overset{d}{=}_{\mathbf{F}^{(l+1)}=\mathbf{W}^{(l+1)}\mathbf{G}^{(l)}} \widetilde{\mathbf{W}}\mathbf{\Pi}_{\mathbf{G}}^{\perp},$$

where $\widetilde{\mathbf{W}}$ is a i.i.d. copy of $\mathbf{W}^{(l+1)}$.

Note that $\left[\left(\mathbf{b}^{(l+1)}(\mathbf{y})\right)^{\top}\widetilde{\mathbf{W}} \left(\mathbf{b}^{(l+1)}(\mathbf{y}')\right)^{\top}\widetilde{\mathbf{W}}\right]^{\top} \in \mathbb{R}^{2d_l} \sim \mathcal{N}(0,\mathbf{\Sigma})$ where $\mathbf{\Sigma} = \begin{bmatrix} b\mathbb{I}_{d_l} & b'\mathbb{I}_{d_l} \\ b'\mathbb{I}_{d_l} & b''\mathbb{I}_{d_l}\end{bmatrix}$. In the definition of $\mathbf{\Sigma}$, $b = \left\langle \mathbf{b}^{(l+1)}(\mathbf{y}), \mathbf{b}^{(l+1)}(\mathbf{y})\right\rangle, b' = \left\langle \mathbf{b}^{(l+1)}(\mathbf{y}), \mathbf{b}^{(l+1)}(\mathbf{y}')\right\rangle$ and $b'' = \left\langle \mathbf{b}^{(l+1)}(\mathbf{y}'), \mathbf{b}^{(l+1)}(\mathbf{y}')\right\rangle$. If $\mathbf{M}$ is such that $\mathbf{\Sigma} = \mathbf{M}\mathbf{M}^{\top}$, then

$$\left[\left(\mathbf{b}^{(l+1)}(\mathbf{y})\right)^{\top}\widetilde{\mathbf{W}} \left(\mathbf{b}^{(l+1)}(\mathbf{y}')\right)^{\top}\widetilde{\mathbf{W}}\right]^{\top} \overset{d}{=} \mathbf{M}\boldsymbol{\xi},$$

where $\boldsymbol{\xi} \sim \mathcal{N}(0,\mathbb{I}_{2d_l})$. Thus, for fixed $\mathbf{G}^{(l)}$ and conditioned on the value of $\{\mathbf{b}^{(l+1)}(\mathbf{y}), \mathbf{b}^{(l+1)}(\mathbf{y}'), \mathbf{h}^{(l)}(\mathbf{y}), \mathbf{h}^{(l)}(\mathbf{y}')\}$, we have,

$$\begin{aligned}
&\left(\mathbf{b}^{(l+1)}(\mathbf{y})\right)^{\top}\mathbf{W}^{(l+1)}\mathbf{\Pi}_{\mathbf{G}}^{\perp}\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\mathbf{\Pi}_{\mathbf{G}}^{\perp}\left(\mathbf{W}^{(l+1)}\right)^{\top}\mathbf{b}^{(l+1)}(\mathbf{y}) \\
&\overset{d}{=} \left(\mathbf{b}^{(l+1)}(\mathbf{y})\right)^{\top}\widetilde{\mathbf{W}}\mathbf{\Pi}_{\mathbf{G}}^{\perp}\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\mathbf{\Pi}_{\mathbf{G}}^{\perp}\left(\widetilde{\mathbf{W}}\right)^{\top}\mathbf{b}^{(l+1)}(\mathbf{y}) \\
&\overset{d}{=} ([\mathbb{I}_{d_l} \;\; 0]\mathbf{M}\boldsymbol{\xi})^{\top}\mathbf{\Pi}_{\mathbf{G}}^{\perp}\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\mathbf{\Pi}_{\mathbf{G}}^{\perp}([0 \;\; \mathbb{I}_{d_l}]\mathbf{M}\boldsymbol{\xi}) \\
&\overset{d}{=} \frac{1}{2}\boldsymbol{\xi}^{\top}\mathbf{M}^{\top}\begin{bmatrix} 0 & \mathbf{\Pi}_{\mathbf{G}}^{\perp}\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\mathbf{\Pi}_{\mathbf{G}}^{\perp} \\ \mathbf{\Pi}_{\mathbf{G}}^{\perp}\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\mathbf{\Pi}_{\mathbf{G}}^{\perp} & 0\end{bmatrix}\mathbf{M}\boldsymbol{\xi}
\end{aligned}$$

Define

$$\mathbf{A} := \frac{1}{2}\mathbf{M}^{\top}\begin{bmatrix} 0 & \mathbf{\Pi}_{\mathbf{G}}^{\perp}\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\mathbf{\Pi}_{\mathbf{G}}^{\perp} \\ \mathbf{\Pi}_{\mathbf{G}}^{\perp}\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\mathbf{\Pi}_{\mathbf{G}}^{\perp} & 0\end{bmatrix}\mathbf{M}.$$

Then we have,

$$\begin{aligned}
\mathbb{E}[\boldsymbol{\xi}^{\top}\mathbf{A}\boldsymbol{\xi}] = \mathrm{tr}(\mathbf{A}) &= \frac{1}{2}\mathrm{tr}\left(\begin{bmatrix} 0 & \mathbf{\Pi}_{\mathbf{G}}^{\perp}\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\mathbf{\Pi}_{\mathbf{G}}^{\perp} \\ \mathbf{\Pi}_{\mathbf{G}}^{\perp}\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\mathbf{\Pi}_{\mathbf{G}}^{\perp} & 0\end{bmatrix}\mathbf{\Sigma}\right) \\
&= b'\mathrm{tr}(\mathbf{\Pi}_{\mathbf{G}}^{\perp}\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\mathbf{\Pi}_{\mathbf{G}}^{\perp}\mathbb{I}_{d_l}) \\
&= b'\mathrm{tr}(\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\mathbf{\Pi}_{\mathbf{G}}^{\perp}) \\
&= b'\mathrm{tr}(\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')(\mathbf{I}_{d_l} - \mathbf{\Pi}_{\mathbf{G}})) \\
&= b'\left[\mathrm{tr}(\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')) - \mathrm{tr}(\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\mathbf{\Pi}_{\mathbf{G}})\right].
\end{aligned}$$

Since $\mathrm{tr}(\mathbf{\Pi}_{\mathbf{G}}) = \mathrm{rank}(\mathbf{\Pi}_{\mathbf{G}})) \le 2$, we have,

$$0 \le \mathrm{tr}(\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\mathbf{\Pi}_{\mathbf{G}}) \le \|\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\|_2\mathrm{tr}(\mathbf{\Pi}_{\mathbf{G}}) \le 2\|\mathbf{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\|_2.$$

Consequently,

$$\left|\mathbb{E}[\boldsymbol{\xi}^\top \mathbf{A}\boldsymbol{\xi}] - b'\mathrm{tr}(\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}'))\right| \leq 2b'\|\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\|_2.$$

For the upper bound on the spectrum, note that $\|\mathbf{M}\|_2^2 = \|\boldsymbol{\Sigma}\|_2 \leq b + b''$. Hence,

$$\begin{aligned}
\|\mathbf{A}\|_2 &\leq \frac{1}{2}\|\mathbf{M}\|_2^2\|\boldsymbol{\Pi}_{\mathbf{G}}^\perp\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\boldsymbol{\Pi}_{\mathbf{G}}^\perp\|_2 \\
&\leq \frac{1}{2}(b+b'')\|\boldsymbol{\Pi}_{\mathbf{G}}^\perp\|_2\|\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\|_2\|\boldsymbol{\Pi}_{\mathbf{G}}^\perp\|_2 \\
&\leq \frac{b+b''}{2}\|\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\|_2.
\end{aligned}$$

Thus, on using Lemma 17 with $t = \log(3/\delta)$, we have with probability at least $1-\delta/3$

$$\begin{aligned}
\left|\boldsymbol{\xi}^\top \mathbf{A}\boldsymbol{\xi} - \mathbb{E}[\boldsymbol{\xi}^\top \mathbf{A}\boldsymbol{\xi}]\right| &\leq 2(\|\mathbf{A}\|_F\sqrt{t} + \|\mathbf{A}\|_2 t) \\
&\leq 2\|\mathbf{A}\|_2(\sqrt{2d_l t} + t) \\
&\leq (b+b'')\|\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\|_2\left(\sqrt{2d_l\log\left(\frac{3}{\delta}\right)} + \log\left(\frac{3}{\delta}\right)\right).
\end{aligned}$$

Consequently,

$$\begin{aligned}
&\left|\frac{c_s}{d_l}\left(\mathbf{b}^{(l+1)}(\mathbf{y})\right)^\top \mathbf{W}^{(l+1)}\boldsymbol{\Pi}_{\mathbf{G}}^\perp\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\boldsymbol{\Pi}_{\mathbf{G}}^\perp\left(\mathbf{W}^{(l+1)}\right)^\top\mathbf{b}^{(l+1)}(\mathbf{y}) - \frac{c_s}{d_l}\mathrm{tr}(\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}'))\left\langle\mathbf{b}^{(l+1)}(\mathbf{y}),\mathbf{b}^{(l+1)}(\mathbf{y}')\right\rangle\right| \\
&\leq \frac{c_s}{d_l}\left|\boldsymbol{\xi}^\top\mathbf{A}\boldsymbol{\xi} - \mathbb{E}[\boldsymbol{\xi}^\top\mathbf{A}\boldsymbol{\xi}]\right| + \frac{c_s}{d_l}\left|\mathbb{E}[\boldsymbol{\xi}^\top\mathbf{A}\boldsymbol{\xi}] - b'\mathrm{tr}(\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}'))\right| \\
&\leq c_s(b+b'')\|\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\|_2\left(\sqrt{\frac{2}{d_l}\log\left(\frac{3}{\delta}\right)} + \frac{1}{d_l}\log\left(\frac{3}{\delta}\right)\right) + \frac{2b'c_s}{d_l}\|\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\|_2,
\end{aligned}$$

holds with probability at least $1-\delta/3$. The lemma follows by taking a union bound over all possible choices of $(\mathbf{y},\mathbf{y}')$.

We can use the above result to obtain the following bound for any $\mathbf{y} \in \{\mathbf{x},\mathbf{x}'\}$

$$\begin{aligned}
&\sqrt{\frac{c_s}{d_l}}\left\|\left(\mathbf{b}^{(l+1)}(\mathbf{y})\right)^\top\mathbf{W}^{(l+1)}\boldsymbol{\Pi}_{\mathbf{G}}^\perp\mathbf{D}^{(l)}(\mathbf{y})\right\| \\
&\leq \left\{\frac{c_s}{d_l}\left(\mathbf{b}^{(l+1)}(\mathbf{y})\right)^\top\mathbf{W}^{(l+1)}\boldsymbol{\Pi}_{\mathbf{G}}^\perp\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y})\boldsymbol{\Pi}_{\mathbf{G}}^\perp\left(\mathbf{W}^{(l+1)}\right)^\top\mathbf{b}^{(l+1)}(\mathbf{y})\right\}^{1/2} \\
&\leq \left\{\frac{c_s}{d_l}\mathrm{tr}(\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}))b + 2c_s b\|\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y})\|_2\left(\sqrt{\frac{2}{d_l}\log\left(\frac{3}{\delta}\right)} + \frac{1}{d_l}\log\left(\frac{3}{\delta}\right)\right) + \frac{2bc_s}{d_l}\|\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\|_2\right\}^{1/2} \\
&\leq \sqrt{bc_s\|\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y})\|_2}\cdot\left\{1 + 2\left(\sqrt{\frac{2}{d_l}\log\left(\frac{3}{\delta}\right)} + \frac{1}{d_l}\left(1+\log\left(\frac{3}{\delta}\right)\right)\right)\right\}^{1/2} \\
&\leq \sqrt{2bc_s\|\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y})\|_2},
\end{aligned}$$

where $b = \left\langle\mathbf{b}^{(l+1)}(\mathbf{y}),\mathbf{b}^{(l+1)}(\mathbf{y})\right\rangle$ and the last step uses the relation the bound on d_l.

B.5. Proof of Lemma 12

Fix any $(\mathbf{y},\mathbf{y}') \in \{(\mathbf{x},\mathbf{x}),(\mathbf{x},\mathbf{x}'),(\mathbf{x}',\mathbf{x}')\}$. For ease of writing let $\mathbf{V}^{(l+1)}(\mathbf{x}) := \left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)}(\mathbf{x})$. We are interested in bounding the following expression.

$$\begin{aligned}
&\left(\mathbf{V}^{(l+1)}(\mathbf{y})\right)^\top \boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\mathbf{V}^{(l+1)}(\mathbf{y}') - \left(\mathbf{V}^{(l+1)}(\mathbf{y})\right)^\top \boldsymbol{\Pi}_{\mathbf{G}}^{\perp}\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\boldsymbol{\Pi}_{\mathbf{G}}^{\perp}\mathbf{V}^{(l+1)}(\mathbf{y}')\\
&= \left(\mathbf{V}^{(l+1)}(\mathbf{y})\right)^\top (\boldsymbol{\Pi}_{\mathbf{G}} + \boldsymbol{\Pi}_{\mathbf{G}}^{\perp})\mathbf{D}^{(l)}(\mathbf{y})\mathbf{D}^{(l)}(\mathbf{y}')(\boldsymbol{\Pi}_{\mathbf{G}} + \boldsymbol{\Pi}_{\mathbf{G}}^{\perp})\mathbf{V}^{(l+1)}(\mathbf{y}') -\\
&\qquad \left(\mathbf{V}^{(l+1)}(\mathbf{y})\right)^\top \boldsymbol{\Pi}_{\mathbf{G}}^{\perp}\mathbf{D}^{(l)}(\mathbf{y})\mathbf{D}^{(l)}(\mathbf{y}')\boldsymbol{\Pi}_{\mathbf{G}}^{\perp}\mathbf{V}^{(l+1)}(\mathbf{y}')\\
&= \left(\mathbf{V}^{(l+1)}(\mathbf{y})\right)^\top \boldsymbol{\Pi}_{\mathbf{G}}\mathbf{D}^{(l)}(\mathbf{y})\mathbf{D}^{(l)}(\mathbf{y}')\boldsymbol{\Pi}_{\mathbf{G}}\mathbf{V}^{(l+1)}(\mathbf{y}') + \left(\mathbf{V}^{(l+1)}(\mathbf{y})\right)^\top \boldsymbol{\Pi}_{\mathbf{G}}\mathbf{D}^{(l)}(\mathbf{y})\mathbf{D}^{(l)}(\mathbf{y}')\boldsymbol{\Pi}_{\mathbf{G}}^{\perp}\mathbf{V}^{(l+1)}(\mathbf{y}') +\\
&\quad \left(\mathbf{V}^{(l+1)}(\mathbf{y})\right)^\top \boldsymbol{\Pi}_{\mathbf{G}}^{\perp}\mathbf{D}^{(l)}(\mathbf{y})\mathbf{D}^{(l)}(\mathbf{y}')\boldsymbol{\Pi}_{\mathbf{G}}\mathbf{V}^{(l+1)}(\mathbf{y}').
\end{aligned}$$

Consequently,

$$\begin{aligned}
&\frac{c_s}{d_l}\left|\left(\mathbf{V}^{(l+1)}(\mathbf{y})\right)^\top \boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\mathbf{V}^{(l+1)}(\mathbf{y}') - \left(\mathbf{V}^{(l+1)}(\mathbf{y})\right)^\top \boldsymbol{\Pi}_{\mathbf{G}}^{\perp}\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\boldsymbol{\Pi}_{\mathbf{G}}^{\perp}\mathbf{V}^{(l+1)}(\mathbf{y}')\right|\\
&= \frac{c_s}{d_l}\Big| \left(\mathbf{V}^{(l+1)}(\mathbf{y})\right)^\top \boldsymbol{\Pi}_{\mathbf{G}}\mathbf{D}^{(l)}(\mathbf{y})\mathbf{D}^{(l)}(\mathbf{y}')\boldsymbol{\Pi}_{\mathbf{G}}\mathbf{V}^{(l+1)}(\mathbf{y}') + \left(\mathbf{V}^{(l+1)}(\mathbf{y})\right)^\top \boldsymbol{\Pi}_{\mathbf{G}}\mathbf{D}^{(l)}(\mathbf{y})\mathbf{D}^{(l)}(\mathbf{y}')\boldsymbol{\Pi}_{\mathbf{G}}^{\perp}\mathbf{V}^{(l+1)}(\mathbf{y}') +\\
&\quad \left(\mathbf{V}^{(l+1)}(\mathbf{y})\right)^\top \boldsymbol{\Pi}_{\mathbf{G}}^{\perp}\mathbf{D}^{(l)}(\mathbf{y})\mathbf{D}^{(l)}(\mathbf{y}')\boldsymbol{\Pi}_{\mathbf{G}}\mathbf{V}^{(l+1)}(\mathbf{y}')\Big|\\
&\leq \frac{c_s}{d_l}\left\|\left(\mathbf{b}^{(l+1)}(\mathbf{y})\right)^\top \mathbf{W}^{(l+1)}\boldsymbol{\Pi}_{\mathbf{G}}\mathbf{D}^{(l)}(\mathbf{y})\right\|\left\|\mathbf{D}^{(l)}(\mathbf{y}')\boldsymbol{\Pi}_{\mathbf{G}}\left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)}(\mathbf{y}')\right\| +\\
&\quad \frac{c_s}{d_l}\left\|\left(\mathbf{b}^{(l+1)}(\mathbf{y})\right)^\top \mathbf{W}^{(l+1)}\boldsymbol{\Pi}_{\mathbf{G}}\mathbf{D}^{(l)}(\mathbf{y})\right\|\left\|\mathbf{D}^{(l)}(\mathbf{y}')\boldsymbol{\Pi}_{\mathbf{G}}^{\perp}\left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)}(\mathbf{y}')\right\| +\\
&\quad \frac{c_s}{d_l}\left\|\left(\mathbf{b}^{(l+1)}(\mathbf{y})\right)^\top \mathbf{W}^{(l+1)}\boldsymbol{\Pi}_{\mathbf{G}}^{\perp}\mathbf{D}^{(l)}(\mathbf{y})\right\|\left\|\mathbf{D}^{(l)}(\mathbf{y}')\boldsymbol{\Pi}_{\mathbf{G}}\left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)}(\mathbf{y}')\right\|\\
&\leq \frac{c_s}{d_l}\left[(s^{(L-l)/2}+1)\sqrt{2\log\left(\frac{8}{\delta}\right)} + s^{L-l}\varepsilon_3\sqrt{2}\right]\left[(s^{(L-l)/2}+1)\sqrt{2\log\left(\frac{8}{\delta}\right)} + s^{L-l}\varepsilon_3\sqrt{2}\right]\|\mathbf{D}^{(l)}(\mathbf{y})\|_2\|\mathbf{D}^{(l)}(\mathbf{y}')\|_2 +\\
&\quad c_s\sqrt{\frac{2}{d_l}}\left[(s^{(L-l)/2}+1)\sqrt{2\log\left(\frac{8}{\delta}\right)} + s^{L-l}\varepsilon_3\sqrt{2}\right]\|\mathbf{D}^{(l)}(\mathbf{y})\|_2\|\mathbf{b}^{(l+1)}(\mathbf{y}')\|\|\mathbf{D}^{(l)}(\mathbf{y}')\|_2 +\\
&\quad c_s\sqrt{\frac{2}{d_l}}\left[(s^{(L-l)/2}+1)\sqrt{2\log\left(\frac{8}{\delta}\right)} + s^{L-l}\varepsilon_3\sqrt{2}\right]\|\mathbf{D}^{(l)}(\mathbf{y})\|_2\|\mathbf{b}^{(l+1)}(\mathbf{y})\|\|\mathbf{D}^{(l)}(\mathbf{y}')\|_2\\
&\leq \frac{c_s}{d_l}\left[(s^{(L-l)/2}+1)\sqrt{2\log\left(\frac{8}{\delta}\right)} + s^{L-l}\varepsilon_3\sqrt{2}\right]^2\|\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\|_2 +\\
&\quad c_s\sqrt{\frac{8}{d_l}}\left[(s^{(L-l)/2}+1)\sqrt{2\log\left(\frac{8}{\delta}\right)} + s^{L-l}\varepsilon_3\sqrt{2}\right](s^{(L-l)/2}+1)\|\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\|_2\\
&\leq \frac{c_s\varepsilon_5}{\sqrt{d_l}}\left[(s^{(L-l)/2}+1)\sqrt{2\log\left(\frac{8}{\delta}\right)} + s^{L-l}\varepsilon_3\sqrt{2}\right]\times\\
&\qquad \left\{\frac{1}{\sqrt{d_l}}\left[(s^{(L-l)/2}+1)\sqrt{2\log\left(\frac{8}{\delta}\right)} + s^{L-l}\varepsilon_3\sqrt{2}\right] + \sqrt{8}(s^{(L-l)/2}+1)\right\},
\end{aligned}$$

where we used Lemma 11 and 18 in fourth step and the condition of $\bar{\mathcal{B}}^{l+1}(\varepsilon_2) \wedge \bar{\mathcal{E}}^{l}(\varepsilon_5)$ in the last step. Both lemmas hold with probability at least $1-\delta/2$, ensuring that the overall expression is true with probability at least $1-\delta$.

B.6. Proof of Lemma 13

Fix any $(\mathbf{y},\mathbf{y}') \in \{(\mathbf{x},\mathbf{x}),(\mathbf{x},\mathbf{x}'),(\mathbf{x}',\mathbf{x}')\}$. Conditioned on $\bar{\mathcal{A}}^L(\varepsilon_1/2) \wedge \bar{\mathcal{B}}^{l+1}(\varepsilon_2) \wedge \bar{\mathcal{C}}(\varepsilon_3) \wedge \bar{\mathcal{D}}^l(\varepsilon_4) \wedge \bar{\mathcal{E}}^l(\varepsilon_5)$, we begin with bounding $\left|\left\langle \mathbf{b}^{(l)}(\mathbf{y}),\mathbf{b}^{(l)}(\mathbf{y}')\right\rangle - \prod_{j=l}^{L+1}\dot{\Sigma}^{(j)}(\mathbf{x},\mathbf{y}')\right|$. Using the definition of $\mathbf{b}^{(l)}(\mathbf{y})$ we have,

$$\begin{aligned}
&\left|\left\langle \mathbf{b}^{(l)}(\mathbf{y}),\mathbf{b}^{(l)}(\mathbf{y}')\right\rangle - \prod_{j=l}^{L+1}\dot{\Sigma}^{(j)}(\mathbf{y},\mathbf{y}')\right| \\
&= \left|\frac{c_s}{d_l}\left(\mathbf{b}^{(l+1)}(\mathbf{y})\right)^\top \mathbf{W}^{(l+1)}\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)}(\mathbf{y}') - \prod_{j=l}^{L+1}\dot{\Sigma}^{(j)}(\mathbf{y},\mathbf{y}')\right| \\
&\leq \frac{c_s}{d_l}\left|\left(\mathbf{b}^{(l+1)}(\mathbf{y})\right)^\top \mathbf{W}^{(l+1)}\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)}(\mathbf{y}')\right. \\
&\qquad \left. - \left(\mathbf{b}^{(l+1)}(\mathbf{y})\right)^\top \mathbf{W}^{(l+1)}\boldsymbol{\Pi}_{\mathbf{G}}^{\perp}\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\boldsymbol{\Pi}_{\mathbf{G}}^{\perp}\left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)}(\mathbf{y}')\right| \\
&+ \left|\frac{c_s}{d_l}\left(\mathbf{b}^{(l+1)}(\mathbf{y})\right)^\top \mathbf{W}^{(l+1)}\boldsymbol{\Pi}_{\mathbf{G}}^{\perp}\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}')\boldsymbol{\Pi}_{\mathbf{G}}^{\perp}\left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)}(\mathbf{y}) - \right. \\
&\qquad\qquad \left.\frac{c_s}{d_l}\mathrm{tr}(\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}'))\left\langle \mathbf{b}^{(l+1)}(\mathbf{y}),\mathbf{b}^{(l+1)}(\mathbf{y}')\right\rangle\right| \\
&+ \left|c_s\frac{\mathrm{tr}(\boldsymbol{\Delta}^{(l)}(\mathbf{y},\mathbf{y}'))}{d_l}\left\langle \mathbf{b}^{(l+1)}(\mathbf{y}),\mathbf{b}^{(l+1)}(\mathbf{y}')\right\rangle - \prod_{j=l}^{L+1}\dot{\Sigma}^{(l)}(\mathbf{y},\mathbf{y}')\right| \\
&\leq \left(\frac{c_s\varepsilon_5}{\sqrt{d_l}}\left[(s^{(L-l)/2}+1)\sqrt{2\log\left(\frac{16}{\delta}\right)} + s^{L-l}\varepsilon_3\sqrt{2}\right]\times\right. \\
&\qquad \left.\left\{\frac{1}{\sqrt{d_l}}\left[(s^{(L-l)/2}+1)\sqrt{2\log\left(\frac{16}{\delta}\right)} + s^{L-l}\varepsilon_3\sqrt{2}\right] + \sqrt{8}(s^{(L-l)/2}+1)\right\}\right) \\
&+ c_s\left(\left\langle \mathbf{b}^{(l+1)}(\mathbf{y}),\mathbf{b}^{(l+1)}(\mathbf{y})\right\rangle + \left\langle \mathbf{b}^{(l+1)}(\mathbf{y}'),\mathbf{b}^{(l+1)}(\mathbf{y}')\right\rangle\right)\left(\sqrt{\frac{2}{d_l}\log\left(\frac{6}{\delta}\right)} + \frac{1}{d_l}\log\left(\frac{6}{\delta}\right)\right)\varepsilon_5 \\
&+ \frac{2c_s}{d_l}\left\langle \mathbf{b}^{(l+1)}(\mathbf{y}),\mathbf{b}^{(l+1)}(\mathbf{y}')\right\rangle\varepsilon_5 + s^{L-l}\varepsilon_4 + s\varepsilon_2 \\
&\leq \left(\frac{c_s\varepsilon_5}{\sqrt{d_l}}\left[(s^{(L-l)/2}+1)\sqrt{2\log\left(\frac{16}{\delta}\right)} + s^{L-l}\varepsilon_3\sqrt{2}\right]\times\right. \\
&\qquad \left.\left\{\frac{1}{\sqrt{d_l}}\left[(s^{(L-l)/2}+1)\sqrt{2\log\left(\frac{16}{\delta}\right)} + s^{L-l}\varepsilon_3\sqrt{2}\right] + \sqrt{8}(s^{(L-l)/2}+1)\right\}\right) \\
&+ 2c_s(s^{L-l}+1)\left(\sqrt{\frac{2}{d_l}\log\left(\frac{6}{\delta}\right)} + \frac{1}{d_l}\log\left(\frac{6}{\delta}\right)\right)\varepsilon_5 + \frac{2c_s}{d_l}(s^{L-l}+1)\varepsilon_5 + s^{L-l}\varepsilon_4 + s\varepsilon_2.
\end{aligned}$$

In the fourth step, we used Lemma 10, 12 and 11 with the last two each holding with probability at least $1-\delta/2$. Consequently, the above expression holds with probability at least $1-\delta$.

B.7. Proof of Lemma 14

We carry out the analysis conditioned on $\bar{\mathcal{A}}^l(\varepsilon)$. Since $\left|\left(\mathbf{h}^{(l)}(\mathbf{y})\right)^\top \mathbf{h}^{(l)}(\mathbf{y}') - \Sigma^{(l)}(\mathbf{y},\mathbf{y}')\right| \leq \varepsilon$ for all $(\mathbf{y},\mathbf{y}') \in \{(\mathbf{x},\mathbf{x}),(\mathbf{x},\mathbf{x}'),(\mathbf{x}',\mathbf{x}')\}$ we can conclude that $\|\tilde{\mathbf{G}}^{(l)} - \Lambda^{(l)}\|_\infty \leq 2\varepsilon$. Since $\breve{\sigma}_{s-1}$ is β_{s-1}-Lipschitz in $\mathcal{M}_+^{\gamma_{s-1}}$ w.r.t.

∞-norm, we have,

$$|\breve{\sigma}_{s-1}(\mathbf{G}^{(l)}) - \breve{\sigma}_{s-1}(\Lambda^{(l+1)})| \leq \beta_{s-1}\|\tilde{\mathbf{G}}^{(l)} - \Lambda^{(l+1)}\|_\infty \leq 2\beta_{s-1}\varepsilon,$$

for $\varepsilon \leq \gamma_{s-1}$. Fix any $(\mathbf{y}, \mathbf{y}') \in \{(\mathbf{x}, \mathbf{x}), (\mathbf{x}, \mathbf{x}'), (\mathbf{x}', \mathbf{x}')\}$. We have that,

$$\begin{aligned}\left|c_s \frac{\mathrm{tr}(\boldsymbol{\Delta}^{(l+1)}(\mathbf{y}, \mathbf{y}'))}{d_{l+1}} - \dot{\Sigma}^{(l+1)}(\mathbf{y}, \mathbf{y}')\right| &\leq \left|c_s \frac{\mathrm{tr}(\boldsymbol{\Delta}^{(l)}(\mathbf{y}, \mathbf{y}'))}{d_l} - \frac{s^2}{2s-1}\breve{\sigma}_{s-1}\left(\Lambda^{(l+1)}(\mathbf{y}, \mathbf{y}')\right)\right| \\ &\leq \left|c_s \frac{\mathrm{tr}(\boldsymbol{\Delta}^{(l)}(\mathbf{y}, \mathbf{y}'))}{d_l} - \frac{s^2}{2s-1}\breve{\sigma}_{s-1}\left(\tilde{\mathbf{G}}^{(l)}(\mathbf{y}, \mathbf{y}')\right)\right| + \\ &\quad \frac{s^2}{2s-1}|\breve{\sigma}_{s-1}(\tilde{\mathbf{G}}^{(l)}) - \breve{\sigma}_{s-1}(\Lambda^{(l+1)})| \\ &\leq s^2(1+\varepsilon)^{s-1}\varphi_{s-1}(d_l, \delta/2, \min\{1, \Sigma^{(l)}(\mathbf{y}, \mathbf{y}') + \varepsilon\}) + \frac{2s^2\beta_s\varepsilon}{2s-1},\end{aligned}$$

holds with probability $1 - \delta$. The last step follows from Lemma 16. On taking a union bound, we can conclude that

$$\left|c_s \frac{\mathrm{tr}(\boldsymbol{\Delta}^{(l+1)}(\mathbf{y}, \mathbf{y}'))}{d_{l+1}} - \dot{\Sigma}^{(l+1)}(\mathbf{y}, \mathbf{y}')\right| \leq s^2(1+\varepsilon)^{s-1}\varphi_{s-1}(d_l, \delta/6, 1) + \frac{2s^2\beta_s\varepsilon}{2s-1},$$

holds for all $(\mathbf{y}, \mathbf{y}') \in \{(\mathbf{x}, \mathbf{x}), (\mathbf{x}, \mathbf{x}'), (\mathbf{x}', \mathbf{x}')\}$ with probability $1 - \delta$. Since we have already invoked Lemma 16, it also gives us that

$$\|\boldsymbol{\Delta}^{(l+1)}(\mathbf{x}, \mathbf{x}')\|_2 \leq s^2\left[2\,(1+\varepsilon)\log\left(\frac{6d_l}{\delta}\right)\right]^{s-1},$$

holds for all $(\mathbf{y}, \mathbf{y}') \in \{(\mathbf{x}, \mathbf{x}), (\mathbf{x}, \mathbf{x}'), (\mathbf{x}', \mathbf{x}')\}$ with probability $1 - \delta$. Using this result along with Lemma 2 and $\varepsilon \leq 1/s$ yields us

$$\Pr\left[\bar{\mathcal{A}}^l(\varepsilon) \Rightarrow \bar{\mathcal{D}}^{l+1}\left(3s^2\varphi_{s-1}(d_l, \delta/6, 1) + \frac{2s^2\beta_s\varepsilon}{2s-1}\right) \wedge \bar{\mathcal{E}}^{l+1}\left(3s^2\left[2\log\left(\frac{6d_l}{\delta}\right)\right]^{s-1}\right)\right] \geq 1 - \delta,$$

as required.

B.8. Proof of Lemma 15

Consider,

$$\begin{aligned}
\Pr\left[\bar{\mathcal{A}}(\varepsilon) \Rightarrow \bar{\mathcal{D}}(\varepsilon') \wedge \bar{\mathcal{E}}(\varepsilon'')\right] &= \Pr\left[\left(\bigcap_{l=0}^{L} \bar{\mathcal{A}}^l(\varepsilon)\right) \Rightarrow \left(\bigcap_{l=0}^{L} \{\bar{\mathcal{D}}^{l+1}(\varepsilon') \wedge \bar{\mathcal{E}}^{l+1}(\varepsilon'')\}\right)\right] \\
&= \Pr\left[\neg\left(\bigcap_{l=0}^{L} \bar{\mathcal{A}}^l(\varepsilon)\right) \vee \left(\bigcap_{l=0}^{L} \{\bar{\mathcal{D}}^{l+1}(\varepsilon') \wedge \bar{\mathcal{E}}^{l+1}(\varepsilon'')\}\right)\right] \\
&= 1 - \Pr\left[\left(\bigcap_{l=0}^{L} \bar{\mathcal{A}}^l(\varepsilon)\right) \wedge \left(\bigcup_{l=0}^{L} \neg\left\{\bar{\mathcal{D}}^{l+1}(\varepsilon') \wedge \bar{\mathcal{E}}^{l+1}(\varepsilon'')\right\}\right)\right] \\
&= 1 - \Pr\left[\left(\bigcup_{l=0}^{L}\left(\bigcap_{l=0}^{L} \bar{\mathcal{A}}^l(\varepsilon)\right) \wedge \neg\left\{\bar{\mathcal{D}}^{l+1}(\varepsilon') \wedge \bar{\mathcal{E}}^{l+1}(\varepsilon'')\right\}\right)\right] \\
&\geq 1 - \sum_{l=0}^{L} \Pr\left[\left(\bigcap_{l=0}^{L} \bar{\mathcal{A}}^l(\varepsilon)\right) \wedge \neg\left\{\bar{\mathcal{D}}^{l+1}(\varepsilon') \wedge \bar{\mathcal{E}}^{l+1}(\varepsilon'')\right\}\right] \\
&\geq 1 - \sum_{l=0}^{L} \Pr\left[\bar{\mathcal{A}}^l(\varepsilon) \wedge \neg\left\{\bar{\mathcal{D}}^{l+1}(\varepsilon') \wedge \bar{\mathcal{E}}^{l+1}(\varepsilon'')\right\}\right] \\
&\geq 1 - \sum_{l=0}^{L} \Pr\left[\neg\left(\bar{\mathcal{A}}^l(\varepsilon) \Rightarrow \bar{\mathcal{D}}^{l+1}(\varepsilon') \wedge \bar{\mathcal{E}}^{l+1}(\varepsilon'')\right)\right] \\
&\geq 1 - \sum_{l=0}^{L} \frac{\delta}{L+1} \\
&\geq 1 - \delta,
\end{aligned}$$

where the eighth line follows from Lemma 14 and the choice of ε' and ε''.

B.9. Proof of Lemma 16

We can rewrite $\operatorname{tr}\left(\mathbf{\Delta}^{(l)}(\mathbf{x}, \mathbf{x}')\right)$ as follows:

$$\begin{aligned}
\operatorname{tr}(\mathbf{\Delta}^{(l)}(\mathbf{x}, \mathbf{x}')) &= \operatorname{tr}\left[\mathbf{D}^{(l)}(\mathbf{x})\mathbf{D}^{(l)}(\mathbf{x}')\right] \\
&= \operatorname{tr}\left[\operatorname{diag}\left(\sigma_s'\left(\mathbf{f}^{(l)}(\mathbf{x})\right)\right)\operatorname{diag}\left(\sigma_s'\left(\mathbf{f}^{(l)}(\mathbf{x}')\right)\right)\right] \\
&= \left\langle \sigma_s'\left(\mathbf{f}^{(l)}(\mathbf{x})\right), \sigma_s'\left(\mathbf{f}^{(l)}(\mathbf{x}')\right)\right\rangle \\
&= s^2\left\langle \sigma_{s-1}\left(\mathbf{W}^{(l)}\mathbf{h}^{(l-1)}(\mathbf{x})\right), \sigma_{s-1}\left(\mathbf{W}^{(l)}\mathbf{h}^{(l-1)}(\mathbf{x}')\right)\right\rangle.
\end{aligned}$$

Note that since all the matrices $\{\mathbf{W}^{(i)}\}_{i=1}^{l-1}$ are fixed, $\mathbf{h}^{(l-1)}(\cdot)$ is a fixed function. Consequently, each term of the vector $\mathbf{W}^{(l)}\mathbf{h}^{(l-1)}(\mathbf{x})$ is distributed as $\mathcal{N}\left(0, \langle\mathbf{h}^{(l)}(\mathbf{x}), \mathbf{h}^{(l)}(\mathbf{x})\rangle\right)$ and is independent of others. If $(\mathbf{w}_j^{(l)})^\top$ denotes the j^{th} row of the matrix $\mathbf{W}^{(l)}$, then we can write the inner product as

$$\begin{aligned}
\operatorname{tr}(\mathbf{\Delta}^{(l)}(\mathbf{x}, \mathbf{x}')) &= s^2\left\langle \sigma_{s-1}\left(\mathbf{W}^{(l)}\mathbf{h}^{(l-1)}(\mathbf{x})\right), \sigma_{s-1}\left(\mathbf{W}^{(l)}\mathbf{h}^{(l-1)}(\mathbf{x}')\right)\right\rangle \\
&= s^2\sum_{j=1}^{d_l} \sigma_{s-1}\left(\left(\mathbf{w}_j^{(l)}\right)^\top \mathbf{h}^{(l-1)}(\mathbf{x})\right)\sigma_{s-1}\left(\left(\mathbf{w}_j^{(l)}\right)^\top \mathbf{h}^{(l-1)}(\mathbf{x}')\right) \\
&= s^2\sum_{j=1}^{d_l} Z_j,
\end{aligned}$$

where Z_j are all independent and identically distributed as $\sigma_{s-1}(X)\sigma_{s-1}(Y)$ where $(X, Y) \sim \mathcal{N}\left(0, \tilde{\mathbf{G}}^{(l-1)}(\mathbf{x}, \mathbf{x}')\right)$. If we define $\tilde{Z}_j := (G_{11}G_{22})^{-\frac{(s-1)}{2}}$, then $\tilde{Z}_j$ are all independent and identically distributed as $\sigma_{s-1}(\tilde{X})\sigma_{s-1}(\tilde{Y})$ where $(\tilde{X}, \tilde{Y}) \sim \mathcal{N}(0, \Sigma_{\rho_G})$ and $\rho_G = \frac{G_{12}}{\sqrt{G_{11}G_{22}}}$. From Lemma 5, we know that

$$\left|\frac{1}{d_l}\sum_{j=1}^{d_l} \tilde{Z}_j - \mathbb{E}[\tilde{Z}_1]\right| \leq \varphi_{s-1}(d_l, \delta, \rho_G)$$

with probability at least $1-\delta$. Consequently,

$$\left|\frac{1}{d_l}\sum_{j=1}^{d_l} Z_j - \mathbb{E}[Z_1]\right| \leq (G_{11}G_{22})^{\frac{(s-1)}{2}} \varphi_{s-1}(d_l, \delta, \rho_G)$$

with probability at least $1-\delta$. Note that $\mathbb{E}[Z_1] = \underset{(X,Y)\sim\mathcal{N}\left(0, \tilde{\mathbf{G}}^{(h)}(\mathbf{x},\mathbf{x}')\right)}{\mathbb{E}}[\sigma_{s-1}(X)\sigma_{s-1}(Y)] = \frac{1}{c_{s-1}}\breve{\sigma}_{s-1}\left(\tilde{\mathbf{G}}^{(l)}(\mathbf{x}, \mathbf{x}')\right)$ and $c_{s-1} = c_s(2s-1)$. On combining this with the previous result, we can conclude that,

$$\begin{aligned}\left|c_s\frac{\operatorname{tr}(\boldsymbol{\Delta}^{(l)}(\mathbf{x}, \mathbf{x}'))}{d_l} - \frac{s^2}{2s-1}\breve{\sigma}_{s-1}\left(\mathbf{G}^{(l-1)}(\mathbf{x}, \mathbf{x}')\right)\right| &\leq s^2\left|\frac{1}{d_l}\sum_{j=1}^{d_l} Z_j - \mathbb{E}[Z_1]\right| \\ &\leq s^2(G_{11}G_{22})^{\frac{(s-1)}{2}}\varphi_{s-1}(d_l, \delta/2, \rho_G),\end{aligned}$$

holds with probability $1-\delta/2$. For the spectral norm of $\boldsymbol{\Delta}^{(l)}(\mathbf{x}, \mathbf{x}')$, we have,

$$\|\boldsymbol{\Delta}^{(l)}(\mathbf{x}, \mathbf{x}')\|_2 = s^2 \max_{j\in d_l} Z_j = s^2(G_{11}G_{22})^{\frac{(s-1)}{2}} \max_{j\in d_l} \tilde{Z}_j.$$

Using (26), we can show that

$$\begin{aligned}\Pr\left(\max_{j\in d_l}\tilde{Z}_j > \left[(1+\rho_G)\log\left(\frac{d_l}{\delta}\right)\right]^{s-1}\right) &\leq d_l \Pr\left(\tilde{Z}_1 > \left[(1+\rho_G)\log\left(\frac{d_l}{\delta}\right)\right]^{s-1}\right) \\ &\leq d_l \exp\left(-\frac{1+\rho_G}{1+\rho_G}\log\left(\frac{d_l}{\delta}\right)\right) \leq \delta.\end{aligned}$$

Consequently,

$$\begin{aligned}\|\boldsymbol{\Delta}^{(l)}(\mathbf{x}, \mathbf{x}')\|_2 &= s^2(G_{11}G_{22})^{\frac{(s-1)}{2}} \max_{j\in d_l}\tilde{Z}_j \\ &\leq s^2(G_{11}G_{22})^{\frac{(s-1)}{2}}\left[(1+\rho_G)\log\left(\frac{2d_l}{\delta}\right)\right]^{s-1}\end{aligned}$$

holds with probability $1-\delta/2$. Thus, both the hold simultaneously with probability at least $1-\delta$.

B.10. Proof of Lemma 18

We will prove the claim for $\mathbf{y} \in \{\mathbf{x}, \mathbf{x}'\}$. Recall that $\mathbf{G}^{(l)}(\mathbf{y}, \mathbf{y}') = [\mathbf{h}^{(l)}(\mathbf{y}) \;\; \mathbf{h}^{(l)}(\mathbf{y}')] \in \mathbb{R}^{d_l\times 2}$. For simplicity, we drop the arguments from $\mathbf{h}^{(l)}$ and $\mathbf{b}^{(l+1)}$. Define $\mathbf{\Pi_h} = \mathbf{h}\mathbf{h}^\top$ and $\mathbf{\Pi_{G/h}} = \mathbf{\Pi_G} - \mathbf{\Pi_h}$. Note that $\mathbf{\Pi_{G/h}}$ is still a projection matrix of rank 0 or 1.

Recall that $\mathbf{b}^{(l+1)}$ is the gradient with respect to the pre-activation layer $l+1$ given by $\mathbf{f}^{(l+1)} = \mathbf{W}^{(l+1)}\mathbf{h}^{(l)}$. If we view the output f as a function of $\mathbf{h}^{(l)}, \mathbf{W}^{(l+1)}, \dots, \mathbf{W}^{(L+1)}$, then we have the following relation:

$$\frac{\partial}{\partial\mathbf{h}^{(l)}} f(\mathbf{h}^{(l)}, \mathbf{W}^{(l+1)}, \dots, \mathbf{W}^{(L+1)}) = \left(\mathbf{b}^{(l+1)}\right)^\top \mathbf{W}^{(l+1)}.$$

Recall that σ_s is s-homogeneous,that is, $\sigma_s(\lambda x) = \lambda^s \sigma_s(x)$ for any $\lambda > 0$. Using this recursion repeatedly, we have,

$$f(\lambda \mathbf{h}^{(l)}, \mathbf{W}^{(l+1)}, \dots, \mathbf{W}^{(L+1)}) = \lambda^{s^{L-l}} f(\mathbf{h}^{(l)}, \mathbf{W}^{(l+1)}, \dots, \mathbf{W}^{(L+1)})$$
$$\implies \frac{\partial}{\partial \lambda} f(\lambda \mathbf{h}^{(l)}, \mathbf{W}^{(l+1)}, \dots, \mathbf{W}^{(L+1)}) = s^{L-l} \lambda^{(s^{L-l}-1)} f(\mathbf{h}^{(l)}, \mathbf{W}^{(l+1)}, \dots, \mathbf{W}^{(L+1)}).$$

Using this, we can write,

$$\begin{aligned} f(\mathbf{h}^{(l)}, \mathbf{W}^{(l+1)}, \dots, \mathbf{W}^{(L+1)}) &= s^{l-L} \frac{\partial}{\partial \lambda} f(\lambda \mathbf{h}^{(l)}, \mathbf{W}^{(l+1)}, \dots, \mathbf{W}^{(L+1)}) \Big|_{\lambda=1} \\ &= s^{l-L} \frac{\partial}{\partial (\lambda \mathbf{h}^{(l)})} f(\lambda \mathbf{h}^{(l)}, \mathbf{W}^{(l+1)}, \dots, \mathbf{W}^{(L+1)}) \Big|_{\lambda=1} \frac{\partial (\lambda \mathbf{h}^{(l)})}{\partial \lambda} \Big|_{\lambda=1} \\ &= s^{l-L} \frac{\partial}{\partial (\lambda \mathbf{h}^{(l)})} f(\lambda \mathbf{h}^{(l)}, \mathbf{W}^{(l+1)}, \dots, \mathbf{W}^{(L+1)}) \Big|_{\lambda=1} \mathbf{h}^{(l)} \\ &= s^{l-L} \left(\mathbf{b}^{(l+1)}\right)^\top \mathbf{W}^{(l+1)} \mathbf{h}^{(l)}. \end{aligned}$$

By definition of $\mathbf{\Pi_h}$, we have,

$$\begin{aligned} \left\| \mathbf{\Pi_h} \left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)} \right\| &= \left\| \frac{1}{\|\mathbf{h}^{(l)}\|^2} \cdot \mathbf{h}^{(l)} \left(\mathbf{h}^{(l)}\right)^\top \left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)} \right\| \\ &= s^{L-l} \frac{|f(\mathbf{y}; \mathbf{W})|}{\|\mathbf{h}^{(l)}\|}. \end{aligned}$$

Since $\bar{\mathcal{A}}^L(\varepsilon_1/2) \wedge \bar{\mathcal{C}}(\varepsilon_3)$, $\|\mathbf{h}^{(l)}\| \geq 1 - \varepsilon_1/2 \geq 1/2$ and $|f(\mathbf{y}; \mathbf{W})| \leq \varepsilon_3$. Consequently,

$$\left\| \mathbf{\Pi_h} \left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)} \right\| \leq s^{L-l} \varepsilon_3 \sqrt{2}.$$

Similar to the proof of Lemma 11, we note that for a fixed $\mathbf{h}$ and conditioned on $\mathbf{f}^{(l+1)}$ and $\mathbf{b}^{(l+1)}$ Lemma 3 gives us that

$$\mathbf{W}^{(l+1)} \mathbf{\Pi}_\mathbf{h}^\perp \overset{d}{=}_{\mathbf{f}^{(l+1)} = \mathbf{W}^{(l+1)} \mathbf{h}^{(l)}} \widetilde{\mathbf{W}} \mathbf{\Pi}_\mathbf{h}^\perp,$$

where $\widetilde{\mathbf{W}}$ is a i.i.d. copy of $\mathbf{W}^{(l+1)}$. From the definition of $\mathbf{\Pi_{G/h}}$, we can conclude that $\mathbf{\Pi}_{\mathbf{G/h}}^\top \mathbf{\Pi_h} = 0$. Thus, combining this with the previous equation, we can conclude that

$$\mathbf{W}^{(l+1)} \mathbf{\Pi_{G/h}} \overset{d}{=}_{\mathbf{f}^{(l+1)} = \mathbf{W}^{(l+1)} \mathbf{h}^{(l)}} \widetilde{\mathbf{W}} \mathbf{\Pi_{G/h}}.$$

If rank$(\mathbf{\Pi_{G/h}}) = 1$, then $\mathbf{\Pi_{G/h}} = \mathbf{u}\mathbf{u}^\top$ for some unit vector $\mathbf{u}$. Thus,

$$\begin{aligned} \left\| \mathbf{\Pi_{G/h}} \left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)} \right\| &= \left\| \mathbf{u}\mathbf{u}^\top \left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)} \right\| \\ &= \left| \mathbf{u}^\top \left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)} \right| \\ &\overset{d}{=} \left| \mathbf{u}^\top \widetilde{\mathbf{W}}^\top \mathbf{b}^{(l+1)} \right| = |t|, \end{aligned}$$

where $t \sim \mathcal{N}(0, \|\mathbf{b}^{(l+1)}\|^2)$. Therefore, with probability at least $1 - \delta/2$,

$$|t| \leq \|\mathbf{b}^{(h+1)}\| \sqrt{2 \log\left(\frac{4}{\delta}\right)} \leq (s^{(L-l)/2} + 1) \sqrt{2 \log\left(\frac{4}{\delta}\right)}.$$

In the above step, we used similar arguments as used in Appendix. B.7 to bound $\|\mathbf{b}^{(l+1)}\|$. If $\mathrm{rank}(\mathbf{\Pi_{G/h}}) = 0$, $\left\|\mathbf{\Pi_{G/h}}\left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)}\right\| = 0$. On combining this with the previous result, we obtain that with probability $1-\delta$

$$\begin{aligned}\left\|\mathbf{\Pi_G}\left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)}\right\| &\leq \left\|\mathbf{\Pi_{G/h}}\left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)}\right\| + \left\|\mathbf{\Pi_h}\left(\mathbf{W}^{(l+1)}\right)^\top \mathbf{b}^{(l+1)}\right\| \\ &\leq (s^{(L-l)/2}+1)\sqrt{2\log\left(\frac{4}{\delta}\right)} + s^{L-l}\varepsilon_3\sqrt{2}.\end{aligned}$$

Taking the union bound over the two possible choices of $\mathbf{y}$ gives us the final result.

C. Proof of Theorem 2

The proof of Theorem 2 involves combining several smaller results, some of which follow from Theorem 1 and the others are minor modifications of existing results. We begin with stating these results along with proofs, if needed, and then combine them to prove Theorem 1.

Firstly, using Theorem 1 along with Lemma 5.1 from Zhou et al. (2020), we can conclude that for $m \geq \mathrm{poly}(T, L, K, \lambda^{-1}, \lambda_0^{-1}, S^{-1}, \log(1/\delta))$, there exists a $\mathbf{W}^*$ such that $h(\mathbf{x}) = \langle \mathbf{g}(\mathbf{x}; \mathbf{W}_0), \mathbf{W}^* - \mathbf{W}_0\rangle$ with probability at least $1-\delta$ for all $\mathbf{x} \in \mathcal{Z} = \{\{\mathbf{x}_{t,a}\}_{a=1}^K\}_{t=1}^T$. Furthermore, $\|\mathbf{W}^* - \mathbf{W}_0\|_2 \leq S$.

Secondly, using Lemma 4 and 8, along with (12), we can conclude that for $m \geq \mathrm{poly}(T, L, K, \lambda^{-1}, \lambda_0^{-1}, S^{-1}, \log(1/\delta))$, $\|\mathbf{g}(\mathbf{x}; \mathbf{W}_0)\| \leq \mathcal{O}(s^L)$, independent of m.

Thirdly, we know that the spectral norm of the Hessian of the neural net is $\mathcal{O}(m^{-1/2})$ (Liu et al., 2020, Theorem 3.2) for all $\mathbf{W}$ such that $\|\mathbf{W} - \mathbf{W}_0\| \leq R$, where R is some finite radius. Here the implied constant depends on s, L and R. Thus, by choosing m to be sufficiently large, we can conclude that for all $\|\mathbf{W} - \mathbf{W}_0\| \leq \sqrt{T/\lambda}$, $\|\tilde{\mathbf{H}}(\mathbf{W})\| \leq C_{s,L} m^{-1/3}$. Here $\tilde{\mathbf{H}}$ denotes the Hessian and $C_{s,L}$ denotes a constant that depends only on s and L. Using this relation, we can conclude that for any $\|\mathbf{W} - \mathbf{W}_0\| \leq \sqrt{T/\lambda}$, we have

$$\begin{aligned}|f(\mathbf{x}; \mathbf{W}) - f(\mathbf{x}; \mathbf{W}_0) - \langle \mathbf{g}(\mathbf{x}; \mathbf{W}_0), \mathbf{W} - \mathbf{W}_0\rangle| &\leq C_{s,L} m^{-1/3} T/\lambda \\ \implies |f(\mathbf{x}; \mathbf{W}) - \langle \mathbf{g}(\mathbf{x}; \mathbf{W}_0), \mathbf{W} - \mathbf{W}_0\rangle| &\leq C_{s,L} \lambda^{-1} m^{-1/6},\end{aligned}$$

where the last step follows by choosing a sufficiently large m and Assumption 1. Moreover, the above result holds for any $\mathbf{x} \in \mathcal{Z}$.

Lastly, using the relation $h(\mathbf{x}) = \langle \mathbf{g}(\mathbf{x}; \mathbf{W}_0), \mathbf{W}^* - \mathbf{W}_0\rangle$ along with the result from Vakili et al. (2021a, Theorem 1) on the kernel defined by neural net at initialization, i.e. $\hat{k}(\mathbf{x}; \mathbf{x}') = \langle \mathbf{g}(\mathbf{x}; \mathbf{W}_0), \mathbf{g}(\mathbf{x}'; \mathbf{W}_0)\rangle$, we can conclude that

$$|h(\mathbf{x}) - \mathbf{g}^\top(\mathbf{x}; \mathbf{W}_0)\mathbf{V}_t^{-1}\mathbf{r}| \leq \left(S + \nu\sqrt{\frac{2}{\lambda}\log(1/\delta)}\right)\hat{\sigma}_t(\mathbf{x}).$$

In the above equation, $\mathbf{V}_t$ and $\hat{\sigma}_t$ are defined with respect to the dataset $\mathcal{D}$ and $\mathbf{r} = \sum_{i=1}^t y_i \mathbf{g}(\mathbf{x}_i; \mathbf{W}_0)$. Also, we have used the independence of dataset from noise sequence to invoke the above theorem.

Using these results we can prove our main result. For any $\mathbf{x} \in \mathcal{Z}$, we have,

$$\begin{aligned}
|h(\mathbf{x}) - f(\mathbf{x};\mathbf{W}_t)| &\leq |h(\mathbf{x}) - \mathbf{g}^\top(\mathbf{x};\mathbf{W}_0)\mathbf{V}^{-1}\mathbf{r}| + |\mathbf{g}^\top(\mathbf{x};\mathbf{W}_0)\mathbf{V}^{-1}\mathbf{r} - f(\mathbf{x};\mathbf{W}_t)| \\
&\leq \left(S + \nu\sqrt{\frac{2}{\lambda}\log(1/\delta)}\right)\hat{\sigma}_t(\mathbf{x}) + |\mathbf{g}^\top(\mathbf{x};\mathbf{W}_0)\mathbf{V}_t^{-1}\mathbf{r} - f(\mathbf{x};\mathbf{W}_t)| \\
&\leq \frac{C_{s,L}}{\lambda m^{1/6}} + \left(S + \nu\sqrt{\frac{2}{\lambda}\log(1/\delta)}\right)\hat{\sigma}(\mathbf{x}) + |\mathbf{g}^\top(\mathbf{x};\mathbf{W}_0)\mathbf{V}_t^{-1}\mathbf{r} - \langle \mathbf{g}(\mathbf{x};\mathbf{W}_0), \mathbf{W}_t - \mathbf{W}_0\rangle| \\
&\leq \frac{C_{s,L}}{\lambda m^{1/6}} + \left(S + \nu\sqrt{\frac{2}{\lambda}\log(1/\delta)}\right)\hat{\sigma}(\mathbf{x}) + \|\mathbf{W}_t - \mathbf{W}_0 - \mathbf{V}_t^{-1}\mathbf{r}\|\|\mathbf{V}_t\|\hat{\sigma}(\mathbf{x}) \\
&\leq \frac{C_{s,L}}{\lambda m^{1/6}} + \left(S + \nu\sqrt{\frac{2}{\lambda}\log(1/\delta)} + (1-\eta\lambda)^{J/2}\sqrt{t/\lambda} + \frac{C'_{s,L}}{\lambda m^{1/6}}\right)(\lambda + C''_{s,L}t)\hat{\sigma}(\mathbf{x}),
\end{aligned}$$

as required. As before $C_{s,L}, C'_{s,L}$ and $C''_{s,L}$ represent constants that depend only on s and L. In the last but one step, we used the fact that for any positive definite matrix $\mathbf{A} \in \mathbb{R}^{d\times d}$ and $\mathbf{x},\mathbf{y} \in \mathbb{R}^d$, $\langle \mathbf{x},\mathbf{y}\rangle \leq (\mathbf{x}^T\mathbf{A}^{-1}\mathbf{x})\cdot(\mathbf{y}^T\mathbf{A}\mathbf{y}) \leq (\mathbf{x}^T\mathbf{A}^{-1}\mathbf{x})\|\mathbf{A}\|_2\|\mathbf{y}\|_2$. And in the last step, we used the results from Zhou et al. (2020, Lemmas B.3,B.5,C.2) along with the bound on gradient established earlier.

D. Additional Details on NeuralGCB

We first provide all the pseudo codes for NeuralGCB followed by proof of Theorem 3.

D.1. Pseudo Codes

In Alg. 2, $\mathrm{UCB}_t^{(r)}$ and $\mathrm{LCB}_t^{(r)}$ refer to the upper and lower confidence scores respectively at time t corresponding to index r and are defined as $\mathrm{UCB}_t^{(r)}(\cdot) = f(\cdot;\mathbf{W}_t^{(r)}) + \beta_t\hat{\sigma}_t^{(r)}(\cdot)$ and $\mathrm{LCB}_t^{(r)}(\cdot) = f(\cdot;\mathbf{W}_t^{(r)}) - \beta_t\hat{\sigma}_t^{(r)}(\cdot)$. The arrays `ctr`, `max_mu` and `fb` are used to store the exploitation count, maximizer of the neural net output and feedback time instants respectively. The exploitation count is tracked to ensure that the exploitation budget is not exceeded. Similarly, the maximizer of the neural net output is stored to guide the algorithm in exploitation at the next level and the feedback time instants are used to keep track of last time instant when the neural net was retrained, or equivalently obtained a feedback. Lastly, υ_t's provide analytical and notational convenience and are not crucial to the working of the algorithm.

GetPredictions is a local routine that calculates the predictive mean and variance after appropriately retraining the neural net and described in Alg. 3 and used as a sub-routine in NeuralGCB. We would like to emphasize that NeuralGCB computes only the mean with the trained parameters. The predictive variance is computed using the gradients at initialization. This is made possible by having an additional ad hoc neural net which is just use to compute the gradients for calculating the variance. This is similar to the setup used in (Kassraie & Krause, 2022).

After computing the predictive variance for the set of current actions, the GetPredictions routine decides whether the neural net needs to be retrained based on the previous feedback instant $t_{\texttt{fb}}$ and batch parameter q. In particular, we consider two training schemes, fixed and adaptive. The fixed frequency scheme retrains the neural net if there are an additional of q points in Ψ after $t_{\texttt{fb}}$. Consequently, the neural corresponding to the r^{th} subset in the partition is retrained after every q_r points are added to the subset. On the other hand, in the adaptive scheme, inspired by the rarely switching bandits (Abbasi-Yadkori et al., 2011; Wang et al., 2021), the neural net is retrained whenever $\det(\mathbf{V}) > q\det(\mathbf{V}_{\texttt{fb}})$ where $\mathbf{V}$ and $\mathbf{V}_{\texttt{fb}}$ are as defined in line 2 and 3 of Alg. 3. Since $\log(\det(\mathbf{V})/\det(\lambda\mathbb{I}))$ is a measure of information gain, the neural net is effectively retrained only once sufficient additional information has been obtained since the last time the neural net was trained. We would like to point out that in the main text, we only refer to the fixed training scheme due to space constraints. However, our NeuralGCB works even with the adaptive scheme and the regret guarantees under this training schedule are formalized in Appendix D.

Lastly, TrainNN is another local routine that carries out the training of neural net for a given dataset using gradient descent for J epochs with a step size of η starting with $\mathbf{W}_0$.

Algorithm 2 NeuralGCB

```
1: **Require**: Time horizon $T$, maximum initial variance $\sigma_0$, error probability $\delta$
2: **Initialize**: $R \leftarrow \lceil \log_2 T \rceil$, Ensemble of $R$ Neural Nets with $\mathbf{W}_0^{(r)} = \mathbf{W}_0$, $\Psi_0^{(r)} \leftarrow \emptyset,\ \forall\, r \in [R]$, $\mathcal{H} \leftarrow \emptyset$, arrays ctr, max_mu, fb of size $R$ with all elements set to 0, batch sizes $\{q_r\}_{r=1}^R$
3: **for** $t = 1, 2, 3, \dots T$ **do**
4:     $r \leftarrow 1, \hat{A}_r(t) = [K]$
5:     **while** True **do**
6:         Receive the context-action pairs $\{\mathbf{x}_{t,a}\}_{a=1}^K$
7:         $\{f(\mathbf{x}_{t,a}; \mathbf{W}_t^{(r)}), \hat{\sigma}_{t-1}^{(r)}(\mathbf{x}_{t,a})\}_{a=1}^K, \mathbf{W}_t^{(r)}$, fb$[r] \leftarrow$ GetPredictions$\Big(\mathcal{H}, \Psi_{t-1}^{(r)}, \{\mathbf{x}_{t,a}\}_{a=1}^K$, fb$[r], \mathbf{W}_{t-1}^{(r)}, q_r\Big)$
8:         $\tilde{\sigma}_{t-1}^{(r)} \leftarrow \max_{a \in \hat{A}_r(t)} \hat{\sigma}_{t-1}^{(r)}(x_{t,a})$,    max_mu$[r] \leftarrow \arg\max_{a \in \hat{A}_r(t)} f(x_{t,a}; \mathbf{W}_{t-1}^{(r)})$
9:         **if** $\tilde{\sigma}_{t-1}^{(r)} \leq \sigma_0 2^{-r}$ **then**
10:            $a_{\text{UCB}} \leftarrow \arg\max_{a \in \hat{A}_r(t)} \text{UCB}_{t-1}^{(r)}(\mathbf{x}_{t,a})$
11:            **if** $\sigma_{t-1}^{(r)}(x_{t,a_{\text{UCB}}}) \leq \eta_0/\sqrt{t}$ **then**
12:                Choose $a_t \leftarrow a_{\text{UCB}}$ and set $\upsilon_t \leftarrow 1$
13:                Receive $y_t = h(\mathbf{x}_{t,a_t}) + \xi_t$ and update $\mathcal{H} \leftarrow \mathcal{H} \cup \{(x_{t,a_t}, y_t)\}$
14:                Set $\Psi_t^{(r+1)} \leftarrow \Psi_{t-1}^{(r+1)} \cup \{(t, \upsilon_t)\}$ and $\Psi_t^{(r')} \leftarrow \Psi_{t-1}^{(r')}$ for all $r' \in [R] \setminus \{r+1\}$
15:                **break**
16:            **else**
17:                $\hat{A}_{r+1}(t) \leftarrow \{a \in \hat{A}_r(t) : \text{UCB}_{t-1}^{(r)}(\mathbf{x}_{t,a}) \geq \max_{a' \in \hat{A}_r(t)} \text{LCB}_{t-1}^{(r)}(\mathbf{x}_{t,a'})\}$, $r \leftarrow r+1$
18:            **end if**
19:        **else**
20:            **if** $r = 1$ **or** ctr$[r] > \alpha_0 4^r$ **then**
21:                Choose any $a_t \in \hat{A}_r(t)$ such that $\hat{\sigma}_{t-1}^{(r)}(x_{t,a}) > \sigma_0 2^{-r}$ and set $\upsilon_t \leftarrow 2$
22:            **else**
23:                Choose $a_t \leftarrow$ max_mu$[r-1]$ and set $\upsilon_t \leftarrow 3$
24:            **end if**
25:            Receive $y_t = h(\mathbf{x}_{t,a_t}) + \xi_t$ and update $\mathcal{H} \leftarrow \mathcal{H} \cup \{(x_{t,a_t}, y_t)\}$, ctr$[r] \leftarrow$ ctr$[r] + 1$
26:            Set $\Psi_t^{(r)} \leftarrow \Psi_{t-1}^{(r)} \cup \{(t, \upsilon_t)\}$ and $\Psi_t^{(r')} \leftarrow \Psi_{t-1}^{(r')}$ for all $r' \in [R] \setminus \{r\}$
27:            **break**
28:        **end if**
29:    **end while**
30: **end for**
```

D.2. Proof of Theorem 3

To analyse the regret of NeuralGCB, we divide the time horizon into three disjoint sets depending on how the action at that time instant was chosen. In particular, for $i = \{1, 2, 3\}$, we define

$$\mathcal{T}_i(s) := \{t \in \psi_T^{(r)} : \upsilon_t = i\}$$
$$\mathcal{T}_i := \bigcup_{r=1}^{R} \mathcal{T}_i(r).$$

Thus, $\mathcal{T}_1, \mathcal{T}_2$ and$\mathcal{T}_3$ consist of all the points chosen by the chosen algorithm at line 12, 21 and 23 respectively. We consider the regret incurred at time instants in each of these sets separately. We begin with $\mathcal{T}_1$. For any $t \in \mathcal{T}_1(r)$,

$$\begin{aligned}
h(\mathbf{x}_{t,a_t^*}) - h(\mathbf{x}_{t,a_t}) &\leq f(\mathbf{x}_{t,a_t^*}; \mathbf{W}_{t-1}^{(r)}) + \tilde{\beta} + \beta_{t-1}\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a_t^*}) - (f(\mathbf{x}_{t,a_t}; \mathbf{W}_{t-1}^{(r)}) - \tilde{\beta} - \beta_{t-1}\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a_t})) \\
&\leq f(\mathbf{x}_{t,a_t}; \mathbf{W}_{t-1}^{(r)}) + \tilde{\beta} + \beta_{t-1}\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a_t}) - f(\mathbf{x}_{t,a_t}; \mathbf{W}_{t-1}^{(r)}) + \tilde{\beta} + \beta_{t-1}\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a_t}) \\
&\leq 2\tilde{\beta} + 2\beta_{t-1}\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a_t}) \\
&\leq 2\tilde{\beta} + \frac{2\eta_0\beta_{t-1}}{\sqrt{t}},
\end{aligned}$$

Algorithm 3 GetPredictions

```
1: Input: Set of past actions and their corresponding rewards $\mathcal{H}$, Index set $\Psi$, Current set of actions $\{\mathbf{x}_{t,a}\}_{a=1}^K$, Previous feedback instant $t_{\texttt{fb}}$, $\mathbf{W}_{t_{\texttt{fb}}}$, Batch choice parameter $q$
2: $\mathbf{Z} \leftarrow \lambda\mathbb{I} + \frac{1}{m}\sum_{i\in\Psi}\mathbf{g}(\mathbf{x}_i;\mathbf{W}_0)\mathbf{g}(\mathbf{x}_i;\mathbf{W}_0)^\top$
3: $\mathbf{Z}_{\texttt{fb}} \leftarrow \lambda\mathbb{I} + \frac{1}{m}\sum_{i\in\Psi, i\leq t_{\texttt{fb}}}\mathbf{g}(\mathbf{x}_i;\mathbf{W}_0)\mathbf{g}(\mathbf{x}_i;\mathbf{W}_0)^\top$
4: Set $\sigma^2(\mathbf{x}_{t,a}) \leftarrow \mathbf{g}(\mathbf{x}_{t,a};\mathbf{W}_0)^\top\mathbf{Z}^{-1}\mathbf{g}(\mathbf{x}_{t,a};\mathbf{W}_0)/m$ for all $a \in [K]$
5: For fixed batch size: $\texttt{to\_train} = (|\Psi| - t_{\texttt{fb}} == q)$
6: For adaptive batch size: $\texttt{to\_train} = (\det(\mathbf{Z}) > q\det(\mathbf{Z}_{\texttt{fb}}))$
7: if $\texttt{to\_train}$ then
8:     $\mathbf{W} \leftarrow$ TrainNN$(m, L, J, \eta, \lambda, \mathbf{W}_0, \{(x_r, y_r) \in \mathcal{H} : r \in \Psi\}), t_{\texttt{fb}} \leftarrow |\Psi|$
9: else
10:    $\mathbf{W} \leftarrow \mathbf{W}_{t_{\texttt{fb}}}$
11: end if
12: return $\{f(\mathbf{x}_{t,a};\mathbf{W}), \sigma(\mathbf{x}_{t,a})\}_{a=1}^K, \mathbf{W}, t_{\texttt{fb}}$.
```

Algorithm 4 TrainNN$(m, L, J, \eta, \lambda, \mathbf{W}_0, \{(\mathbf{x}_i, y_i)\}_{i=1}^n)$

```
1: Define $\mathcal{L}(\mathbf{W}) = \sum_{i=1}^n (f(\mathbf{x}_i;\mathbf{W}) - y_i)^2 + m\lambda\|\mathbf{W} - \mathbf{W}_0\|^2$
2: for $j = 0, 1, \ldots, J-1$ do
3:     $\mathbf{W}_{j+1} = \mathbf{W}_j - \eta\nabla_{\mathbf{W}}\mathcal{L}(\mathbf{W}_j)$
4: end for
5: return $\mathbf{W}_J$
```

where we use Theorem 2 in the first step, $\tilde{\beta} = \frac{C_{s,L}}{\lambda m^{1/6}}$ and $\beta_t = \left(S + \nu\sqrt{\frac{2}{\lambda}\log(1/\delta)} + (1-\eta\lambda)^{J/2}\sqrt{t/\lambda} + \frac{C'_{s,L}}{\lambda m^{1/6}}\right)(\lambda + C''_{s,L}t)$. Since this is independent of r, this relation holds for all $t \in \mathcal{T}_1$. Consequently, the regret incurred over the time instants in $\mathcal{T}_1$ can be bounded as

$$\begin{aligned}\sum_{t\in\mathcal{T}_1} h(\mathbf{x}_{t,a_t^*}) - h(\mathbf{x}_{t,a_t}) &\leq \sum_{t\in\mathcal{T}_1}\left[2\tilde{\beta} + \frac{2\eta_0\beta_{t-1}}{\sqrt{t}}\right] \\ &\leq 2\tilde{\beta}|\mathcal{T}_1| + 2\eta_0\beta_T\sqrt{T}.\end{aligned}$$

We now consider a time instant in $\mathcal{T}_2$ or $\mathcal{T}_3$. Fix any $r > 1$ and $t \in \mathcal{T}_2(r) \cup \mathcal{T}_3(r)$ which implies that $a_t \in \hat{A}_r(t)$. Consequently, $\mathbf{x}_{t,a_t}$ satisfies the following relation:

$$f(\mathbf{x}_{t,a_t};\mathbf{W}_{t-1}^{(r-1)}) + \beta_{t-1}\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a_t}) \geq \max_{a'\in\hat{A}_r(t)} f(\mathbf{x}_{t,a'};\mathbf{W}_{t-1}^{(r-1)}) - \beta_{t-1}\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a'})$$

Note that output of the neural net is based on the parameters evaluated during the last time the neural net was trained, which we denote by $t_{\texttt{fb}}$. In other words, $f(\mathbf{x}_{t,a_t};\mathbf{W}_{t-1}^{(r-1)}) = f(\mathbf{x}_{t,a_t};\mathbf{W}_{t_{\texttt{fb}}}^{(r-1)})$. However, $\hat{\sigma}_{t-1}^{(r-1)}$ is evaluated using all the points in $\Psi_{t-1}^{(r)}$. We can account for the discrepancy using Lemma 12 from Abbasi-Yadkori et al. (2011) which states that for any two positive definite matrices $\mathbf{A} \succeq \mathbf{B} \in \mathbb{R}^{d\times d}$ and $\mathbf{x} \in \mathbb{R}^d$, we have $\mathbf{x}^\top\mathbf{A}\mathbf{x} \leq \mathbf{x}^\top\mathbf{B}\mathbf{x}\cdot\sqrt{\det(\mathbf{A})/\det(\mathbf{B})}$. Using this result along with noting that $\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a}) = \mathbf{g}^\top(\mathbf{x}_{t,a};\mathbf{W}_0)\mathbf{V}_{t-1}^{-1}\mathbf{g}(\mathbf{x}_{t,a};\mathbf{W}_0)$, $\hat{\sigma}_{t_{\texttt{fb}}}^{(r-1)}(\mathbf{x}_{t,a}) = \mathbf{g}^\top(\mathbf{x}_{t,a};\mathbf{W}_0)\mathbf{V}_{t_{\texttt{fb}}}^{-1}\mathbf{g}(\mathbf{x}_{t,a};\mathbf{W}_0)$ and $\mathbf{V}_{t-1} \succeq \mathbf{V}_{\texttt{fb}}$, we can conclude that $\hat{\sigma}_{t_{\texttt{fb}}}^{(r-1)}(\mathbf{x}_{t,a}) \leq \hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a})\cdot\sqrt{\det(\mathbf{V}_{t-1})/\det(\mathbf{V}_{\texttt{fb}})}$.

For any batch such that $\det(\mathbf{V}_t)/\det(\mathbf{V}_{\texttt{fb}}) \leq 4$, we have, $\hat{\sigma}_{t_{\texttt{fb}}}^{(r-1)}(\mathbf{x}_{t,a}) \leq 2\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a})$ for all $a \in [K]$. Consequently, $|h(\mathbf{x}_{t,a}) - f(\mathbf{x}_{t,a_t};\mathbf{W}_{t_{\texttt{fb}}}^{(r-1)})| \leq \tilde{\beta} + \beta_{t_{\texttt{fb}}}\hat{\sigma}_{t_{\texttt{fb}}}^{(r-1)}(\mathbf{x}_{t,a}) \leq \tilde{\beta} + 2\beta_{t-1}\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a})$. Thus, for any such batch and a time instant $t \in \mathcal{T}_2(r)$, using Theorem 2 we have,

$$\begin{aligned} & f(\mathbf{x}_{t,a_t};\mathbf{W}_{t-1}^{(r-1)}) + \beta_{t-1}\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a_t}) \geq \max_{a' \in \hat{A}_r(t)} f(\mathbf{x}_{t,a'};\mathbf{W}_{t-1}^{(r-1)}) - \beta_{t-1}\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a'}) \\ \implies & f(\mathbf{x}_{t,a_t};\mathbf{W}_{t_{\mathrm{fb}}}^{(r-1)}) + \beta_{t-1}\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a_t}) \geq f(\mathbf{x}_{t,a_t^*};\mathbf{W}_{t_{\mathrm{fb}}}^{(r-1)}) - \beta_{t-1}\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a^*}) \\ \implies & h(\mathbf{x}_{t,a_t}) + 3\beta_{t-1}\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a_t}) + \tilde{\beta} \geq h(\mathbf{x}_{t,a^*}) - 3\beta_{t-1}\hat{\sigma}_{t-1}^{(r-1)}(x_{t,a^*}) - \tilde{\beta}. \end{aligned}$$

Since $\hat{A}_r(t) \subseteq \hat{A}_{r-1}(t)$, $\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a}) \leq \tilde{\sigma}_{t-1}^{(r-1)} \leq \sigma_0 2^{1-r}$ for all $a \in \hat{A}_r(t)$. Thus,

$$h(\mathbf{x}_{t,a_t^*}) - h(\mathbf{x}_{t,a_t}) \leq 3\beta_{t-1}\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a}) + 3\beta_{t-1}\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a_t^*}) + 2\tilde{\beta} \leq 12\sigma_0\beta_{t-1}\cdot 2^{-r} + 2\tilde{\beta}.$$

Similarly, for a time instant $t \in \mathcal{T}_3(r)$, we have,

$$\begin{aligned} h(\mathbf{x}_{t,a_t^*}) - h(\mathbf{x}_{t,a_t}) &\leq f(\mathbf{x}_{t,a_t^*};\mathbf{W}_{t_{\mathrm{fb}}}^{(r-1)}) + \tilde{\beta} + 2\beta_{t-1}\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a_t^*}) - f(\mathbf{x}_{t,a_t};\mathbf{W}_{t_{\mathrm{fb}}}^{(r-1)}) + \tilde{\beta} + 2\beta_{t-1}\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a_t}) \\ &\leq 3\beta_{t-1}\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a_t^*}) + 3\beta_{t-1}\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a}) + 2\tilde{\beta} \leq 12\sigma_0\beta_{t-1}\cdot 2^{-r} + 2\tilde{\beta} \\ &\leq 12\sigma_0\beta_{t-1}\cdot 2^{-r} + 2\tilde{\beta}, \end{aligned}$$

where the first step uses Theorem 2 and the last step uses the fact that $\mathbf{x}_{t,a_t} \in \hat{A}_{r-1}(t)$. For any $t \in \mathcal{T}_2(1) \cup \mathcal{T}_3(1)$, we use the trivial bound $h(\mathbf{x}_{t,a^*}) - h(\mathbf{x}_{t,a_t}) \leq 2$.

We now bound $|\mathcal{T}_2(r)|$ using similar techniques outlined in Valko et al. (2013); Auer (2002). Fix any $r \geq 1$. Using the fact that for all $t \in \mathcal{T}_2(r)$, $\hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a_t}) \geq \sigma_0 2^{-r}$ and the bound on the sum of posterior standard deviations from Chowdhury & Gopalan (2017, Lemma 4), we can write

$$\begin{aligned} \sigma_0 2^{-r}|\mathcal{T}_2(r)| &\leq \sum_{t \in T_2(r)} \hat{\sigma}_{t-1}^{(r-1)}(\mathbf{x}_{t,a_t}) \leq \sqrt{8|\mathcal{T}_2(r)|(\lambda\Gamma_k(T)+1)} \\ \implies |\mathcal{T}_2(r)| &\leq 2^r\sqrt{8\sigma_0^{-2}|\mathcal{T}_2(r)|(\lambda\Gamma_k(T)+1)} \end{aligned}$$

We also used the fact that the information gain of the kernel induced by the finite neural net is close to that of NTK for sufficiently large m (Kassraie & Krause, 2022, Lemma D.5). This also provides a guideline for setting the length of the exploration sequence. Specifically, we can set α_0 such that $|\mathcal{T}_3(r)|$ also satisfies $|\mathcal{T}_3(r)| \leq 2^r\sqrt{8\sigma_0^{-2}|\mathcal{T}_3(r)|(\lambda\Gamma_k(T)+1)}$, or equivalently, $\alpha_0 = \mathcal{O}(\Gamma_k(T))$. In general, we have $|\mathcal{T}_3(r)| \leq \alpha_0 4^r \implies |\mathcal{T}_3(r)| \leq 2^r\sqrt{\alpha_0|\mathcal{T}_3(r)|}$.

We can use these conditions to bound the regret incurred for $t \in \mathcal{T}_2'$, the subset of $\mathcal{T}_2$ that consists only of batches that satisfy the determinant condition. We have,

$$\begin{aligned} \sum_{t \in \mathcal{T}_2'} h(\mathbf{x}_{t,a_t^*}) - h(\mathbf{x}_{t,a_t}) &\leq \sum_{r=1}^{R}\sum_{t \in \mathcal{T}_2'(r)} h(\mathbf{x}_{t,a_t^*}) - h(\mathbf{x}_{t,a_t}) \\ &\leq |\mathcal{T}_2(1)| + \sum_{r=2}^{R}\sum_{t \in \mathcal{T}_2(r)} \left[12\sigma_0\beta_{t-1}\cdot 2^{-r} + 2\tilde{\beta}\right] \\ &\leq |\mathcal{T}_2(1)| + \sum_{r=2}^{R}\left[12\sigma_0\beta_{t-1}\cdot 2^{-r} + 2\tilde{\beta}\right]|\mathcal{T}_2(r)| \\ &\leq |\mathcal{T}_2(1)| + 2\tilde{\beta}|\mathcal{T}_2| + \sum_{r=2}^{R}(12\sigma_0\beta_{t-1}\cdot 2^{-r})\cdot 2^r\sqrt{8\sigma_0^{-2}|\mathcal{T}_2(r)|(\lambda\Gamma_k(T)+1)} \\ &\leq |\mathcal{T}_2(1)| + 2\tilde{\beta}|\mathcal{T}_2| + 12\beta_T\sqrt{8|\mathcal{T}_2|R(\lambda\Gamma_k(T)+1)}, \end{aligned}$$

where we used Cauchy-Schwarz inequality in the last step. Using a similar analysis, we can bound the regret incurred for $t \in \mathcal{T}_3'$, which is defined similarly to $\mathcal{T}_2'$, to obtain,

$$\sum_{t \in \mathcal{T}_3'} h(\mathbf{x}_{t,a_t^*}) - h(\mathbf{x}_{t,a_t}) \leq |\mathcal{T}_3(1)| + 2\tilde{\beta}|\mathcal{T}_3| + 12\beta_T\sqrt{8|\mathcal{T}_3|R\alpha_0}.$$

Now, for batches for which the determinant condition is not satisfied, the regret incurred in any batch corresponding to subset index r can be trivially bounded as q_r. We show that the number of such batches is bounded by $\mathcal{O}(\Gamma_k(T))$. Fix a $r \geq 1$. Let there be B_r such batches for which the determinant condition is not satisfied and $\mathcal{T}'(r)$ denote the set of time instants at which such batches begin. Then,

$$4^{B_r} \leq \prod_{t \in \mathcal{T}'(r)} \frac{\det(\mathbf{V}_{t+q_r})}{\det(\mathbf{V}_t)} \leq \frac{\det(\mathbf{V}_{T+1})}{\det(\lambda\mathbb{I})}.$$

Since $\log\left(\det(\mathbf{V}_{T+1})/\det(\lambda\mathbb{I})\right)$ is $\mathcal{O}(\Gamma_k(T))$, we can conclude that B_r is also $\mathcal{O}(\Gamma_k(T))$. Thus, regret incurred in all such batches can be bounded as $\mathcal{O}(\max_r q_r\Gamma_k(T)\log T)$ as there are $R = \log T$ batches.

We can combine the two cases to obtain the regret incurred over $\mathcal{T}_2$ and $\mathcal{T}_3$ as

$$\sum_{t \in \mathcal{T}_2} h(\mathbf{x}_{t,a_t^*}) - h(\mathbf{x}_{t,a_t}) \leq |\mathcal{T}_2(1)| + 2\tilde{\beta}|\mathcal{T}_2| + 12\beta_T\sqrt{8|\mathcal{T}_2|R(\lambda\Gamma_k(T)+1)} + \mathcal{O}(\max_r q_r\Gamma_k(T)\log T)$$
$$\sum_{t \in \mathcal{T}_3} h(\mathbf{x}_{t,a_t^*}) - h(\mathbf{x}_{t,a_t}) \leq |\mathcal{T}_3(1)| + 2\tilde{\beta}|\mathcal{T}_3| + 12\beta_T\sqrt{8|\mathcal{T}_3|R\alpha_0} + \mathcal{O}(\max_r q_r\Gamma_k(T)\log T).$$

The overall regret is now obtained by adding the regret incurred over $\mathcal{T}_1$, $\mathcal{T}_2$ and $\mathcal{T}_3$. We have,

$$\begin{aligned}
R(T) &= \sum_{t=1}^{T} h(\mathbf{x}_{t,a^*}) - h(\mathbf{x}_{t,a_t}) \\
&= \sum_{t \in \mathcal{T}_1} h(\mathbf{x}_{t,a^*}) - h(\mathbf{x}_{t,a_t}) + \sum_{t \in \mathcal{T}_2} h(\mathbf{x}_{t,a^*}) - h(\mathbf{x}_{t,a_t}) + \sum_{t \in \mathcal{T}_3} h(\mathbf{x}_{t,a^*}) - h(\mathbf{x}_{t,a_t}) \\
&\leq 2\tilde{\beta}(|\mathcal{T}_1| + |\mathcal{T}_2| + |\mathcal{T}_3|) + |\mathcal{T}_2(1)| + |\mathcal{T}_3(1)| + 2\eta_0\beta_T\sqrt{T} + 12\beta_T\sqrt{8|\mathcal{T}_3|R\alpha_0} \\
&\quad + 12\beta_T\sqrt{8|\mathcal{T}_2|R(\lambda\Gamma_k(T)+1)} + \mathcal{O}(\max_r q_r\Gamma_k(T)\log T) \\
&\leq 2\tilde{\beta}T + 2\eta_0\beta_T\sqrt{T} + 12\beta_T\sqrt{8T(\lambda\Gamma_k(T)+1+\alpha_0)} + \mathcal{O}(\max_r q_r\Gamma_k(T)\log T),
\end{aligned}$$

where we again use the Cauchy Schwarz inequality in the last step along with the fact that $|\mathcal{T}_2(1)|$ and $|\mathcal{T}_3(1)|$ satisfy $\mathcal{O}(\Gamma_k(T))$. Since $m \geq \mathrm{poly}(T)$, the first term $\tilde{\beta}T$ is $\mathcal{O}(1)$ and using the values of J and η specified in Assumption 2 along with the bound on m, we have $\beta_T \leq \beta \leq 2S + \nu\sqrt{2\lambda^{-1}\log(1/\delta)}$. Consequently, the regret incurred by NeuralGCB satisfies $\mathcal{O}(\sqrt{T\Gamma_k(T)\log(1/\delta)} + \sqrt{T\Gamma_k(T)} + \sqrt{T\log(1/\delta)} + \max_r q_r\Gamma_k(T)\log T)$.

The above analysis carries through almost as is for the case of adaptive batch sizes. Since the batches always satisfy the determinant condition with q_r instead of 4, we do not need to separately consider the case where the determinant condition is not met. Using the same steps as above, we can conclude that the regret incurred by NeuralGCB when run with adaptive step sizes is $\mathcal{O}(\sqrt{T\Gamma_k(T)\log(1/\delta)} \cdot \max_r \sqrt{q_r})$. Thus for the adaptive case, we incur an additional multiplicative factor. This can be explained by the fact that in the adaptive batch setting, the number of times the neural nets are retrained is $\mathcal{O}(\Gamma_k(T))$. However, in the fixed batch setting, even if we use the optimal batch size of $\sqrt{T/\Gamma_k(T)}$, we end up retraining the neural nets about $\mathcal{O}(\sqrt{T\Gamma_k(T)})$ times, which is more than in the case of the adaptive batch setting. The more frequent of retraining of neural nets helps us achieve a tighter regret bound in the case of fixed batch setting.

E. Empirical Studies

In this section, we provide further details of the setup used during our experiments. For completeness, we first restate the construction and preprocessing of the datasets followed by the experimental setup. We then provide the complete array of results for the all the algorithm on different datasets.

E.1. Datasets

For each of the synthetic datasets, we construct a contextual bandit problem with a feature dimension of $d = 10$ and $K = 4$ actions per context running over a time horizon of $T = 2000$ rounds. The set of context vectors $\{\{\mathbf{x}_{t,a}\}_{a=1}^{K}\}_{t=1}^{T}$ are drawn uniformly from the unit sphere. Similar to Zhou et al. (2020), we consider the following three reward functions:

$$h_1(\mathbf{x}) = 4|\mathbf{a}^\top \mathbf{x}|^2; \quad h_2(\mathbf{x}) = 4\sin^2(\mathbf{a}^\top \mathbf{x}); \quad h_3(\mathbf{x}) = \|\mathbf{A}\mathbf{x}\|_2. \tag{27}$$

For the above functions, the vector $\mathbf{a}$ is drawn uniformly from the unit sphere and each entry of matrix $\mathbf{A}$ is randomly generated from $\mathcal{N}(0, 0.25)$.

We also consider two real datasets for classification namely Mushroom and Statlog (Shuttle), both of which are available on the UCI repository (Dua & Graff, 2017).

Mushroom: The Mushroom dataset consists of 8124 samples with 22 features where each data point is labelled as an edible or poisonous mushroom, resulting in a binary classification problem. For the purpose of the experiments, we carry out some basic preprocessing on this dataset. We drop the attributed labeled "veil-type" as it is the same for all the datapoints resulting in a $d = 21$ dimensional feature vector. Furthermore, we randomly select 1000 points from each class to create a smaller dataset of size 2000 to run the experiements.

Statlog: The Statlog (Shuttle) dataset consists of 58000 entries with $d = 7$ attributes (we do not consider time as an attribute) each. The original dataset is divided into 7 classes. Since the data is skewed, we convert this into a binary classification problem by combining classes. In particular, we combine the five smallest classes, namely, $2, 3, 5, 6, 7$ and combine them to form a single class and drop class 4 resulting in a binary classification problem, with points labelled as 1 forming the first class and ones with labels in $\{2, 3, 5, 6, 7\}$ forming the second. Once again, we select a subset of 2000 points by randomly choosing 1000 points from each class to use for the experiments. Lastly, we normalize the features of this dataset.

The classification problem is then converted into a contextual bandit problem using techniques outlined in (Li et al., 2010), which is also adopted in Zhou et al. (2020) and (Gu et al., 2021). Specifically, each datapoint in the dataset $(\mathbf{x}, y) \in \mathbb{R}^d \times \mathbb{R}$ is transformed into K vectors of the form $\mathbf{x}^{(1)} = (\mathbf{x}, \mathbf{0}, \dots, \mathbf{0}) \dots, \mathbf{x}^{(K)} = (\mathbf{0}, \dots, \mathbf{0}, \mathbf{x}) \in \mathbb{R}^{Kd}$ corresponding to the K actions associated with context $\mathbf{x}$. Here K denotes the number of classes in the original classification problem. The reward function is set to $h(\mathbf{x}^{(k)}) = \mathbb{1}\{y = k\}$, that is, the agent receives a reward of 1 if they classify the context correctly and 0 otherwise. As mentioned above, we have $d = 21$ for Mushroom dataset and $d = 7$ for the Shuttle dataset and $K = 2$ for both of them.

E.2. Experimental Setting

For all the experiments, the rewards are generated by adding zero mean Gaussian noise with a standard deviation of 0.1 to the reward function. All the experiments are run for a time horizon of $T = 2000$. We report the regret averaged over 10 Monte Carlo runs with different random seeds along with an error bar of one standard deviation on either side. For all the datasets, we generate new rewards for each Monte Carlo run by changing the noise. Furthermore, for the real datasets, we also shuffle the context vectors for each Monte Carlo run.

Setting the hyperparameters: For all the algorithms, the parameter ν, the standard deviation proxy for the sub-Gaussian noise, to 0.1, the standard deviation of the Gaussian noise added. Also, the RKHS norm of the reward, S to 4 for synthetic functions, and 1 for real datasets in all the algorithms. For all the experiments, we perform a grid search for λ and η over $\{0.05, 0.1, 0.5\}$ and $\{0.001, 0.01, 0.1\}$, respectively, for each algorithm and choose the best ones for the corresponding algorithm and reward function. The exploration parameter β_t is set to the value prescribed by each algorithm. For NeuralUCB and NeuralTS, the prescribed value for the exploration parameter in the original work was given as:

$$\beta_t = \beta_0 \left(\nu \sqrt{\log\left(\frac{\det(\mathbf{V}_t)}{\det(\lambda \mathbb{I})}\right) + C_2 m^{-1/6} \sqrt{\log m} L^4 t^{5/3} \lambda^{-1/6} + 2\log(1/\delta)} + S \right)$$
$$+ (\lambda + C_3 tL) \left[(1 - \eta m \lambda)^{J/2} \sqrt{t/\lambda} + m^{-1/6} \sqrt{\log m} L^{7/2} t^{5/3} \lambda^{-5/3} (1 + \sqrt{t/\lambda}) \right],$$

where $\beta_0 = \sqrt{1 + C_1 m^{-1/6}\sqrt{\log m} L^4 t^{7/6} \lambda^{-7/6}}$. We ignore the smaller order terms and set $\beta_t = 2S + \nu\sqrt{\log\left(\frac{\det(\mathbf{V}_t)}{\det(\lambda\mathbb{I})}\right) + 2\log(1/\delta)}$ as $(\lambda + C_3 tL)(1-\eta m\lambda)^{J/2}\sqrt{t/\lambda} \leq S$ for given choice of parameters as shown in Zhou et al. (2020). For NeuralGCB and SupNNUCB, we apply the same process of ignoring the smaller order terms and bounding the additional constant term by S to obtain $\beta_t = 2S + \nu\sqrt{\frac{2}{\lambda}\log(1/\delta)}$, which is used in the algorithms. Also for NeuralGCB, η_0 was set to 0.2 and α_0 to $20\log T$, as suggested in Sec. D, for all experiments. The number of epochs is set to 200 for synthetic datasets and Mushroom, and to 400 for Statlog.

Neural Net architecture: We consider a 2 layered neural net as described in Equation (2) for all the experiments. We perform two different sets of experiments with two different activation functions, namely σ_1 (or equivalently, ReLU) and σ_2. For the experiments with σ_1 as the activation, we set the number of hidden neurons to $m = 20$ and $m = 50$ for synthetic and real datasets respectively. For those with σ_2, m is set to 30 and 80 for synthetic and real datasets, respectively.

Training Schedule: For the experiments with sequential algorithms, we retrain the neural nets at every step, including NeuralGCB. For the batched algorithms, we consider a variety of training schedules. Particularly, for each setting (i.e., dataset and activation function) we consider two fixed and two adaptive batch size training schedules, which are denoted by $F1, F2$ and $A1, A2$ respectively. We specify the training schedules below for different dataset and activation functions beginning with those for σ_1 (ReLU). These were chosen to ensure that both algorithms have roughly the same running time to allow for a fair comparison.

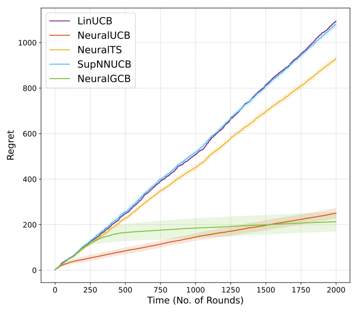
1. For Batched NeuralUCB run on a synthetic function:
   - $F1$: 200 batches, or equivalently retrain after every 10 steps.
   - $F2$: 300 batches, or equivalently retrain after every $6-7$ steps.
   - $A1$: $q = 2$
   - $A2$: $q = 3$
2. For Batched NeuralUCB run on Mushroom:
   - $F1$: 100 batches, or equivalently retrain after every 20 steps.
   - $F2$: 200 batches, or equivalently retrain after every 10 steps.
   - $A1$: $q = 500$
   - $A2$: $q = 700$
3. For Batched NeuralUCB run on Shuttle:
   - $F1$: 100 batches, or equivalently retrain after every 20 steps.
   - $F2$: 200 batches, or equivalently retrain after every 10 steps.
   - $A1$: $q = 5$
   - $A2$: $q = 10$
4. For NeuralGCB run on a synthetic function:
   - $F1$: $q_1 = 4, q_2 = 20, q_r = 40$ for $r \geq 3$ (i.e., retrain after q_r steps)
   - $F2$: $q_1 = 2, q_2 = 10, q_r = 20$ for $r \geq 3$.
   - $A1$: $q_1 = 1.2, q_2 = 1.8, q_r = 2.7$ for $r \geq 3$.
   - $A2$: $q_1 = 1.5, q_2 = 2.25, q_r = 3.375$ for $r \geq 3$.
5. For NeuralGCB run on Mushroom:
   - $F1$: $q_1 = 20, q_2 = 40, q_r = 80$ for $r \geq 3$ (i.e., retrain after q_r steps)
   - $F2$: $q_1 = 10, q_2 = 20, q_r = 40$ for $r \geq 3$.
   - $A1$: $q_1 = 300, q_2 = 1200, q_r = 4800$ for $r \geq 3$.
   - $A2$: $q_1 = 500, q_2 = 2000, q_r = 8000$ for $r \geq 3$.
6. For NeuralGCB run on Shuttle:

- $F1$: $q_1 = 20, q_2 = 40, q_r = 80$ for $r \geq 3$ (i.e., retrain after q_r steps)
- $F2$: $q_1 = 10, q_2 = 20, q_r = 40$ for $r \geq 3$.
- $A1$: $q_1 = 5, q_2 = 12.5, q_r = 31.25$ for $r \geq 3$.
- $A2$: $q_1 = 10, q_2 = 25, q_r = 62.5$ for $r \geq 3$.

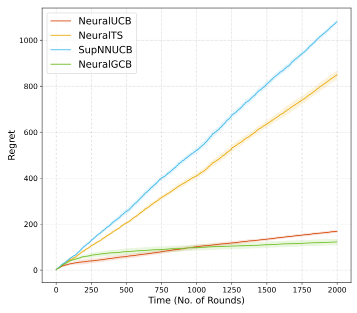
Similarly, for σ_2, the training schedules are given as follows:

1. For Batched NeuralUCB run on a synthetic function:
   - $F1$: 200 batches, or equivalently retrain after every 10 steps.
   - $F2$: 300 batches, or equivalently retrain after every $6 - 7$ steps.
   - $A1$: $q = 3$
   - $A2$: $q = 4$
2. For Batched NeuralUCB run on Mushroom:
   - $F1$: 100 batches, or equivalently retrain after every 20 steps.
   - $F2$: 200 batches, or equivalently retrain after every 10 steps.
   - $A1$: $q = 500$
   - $A2$: $q = 700$
3. For Batched NeuralUCB run on Shuttle:
   - $F1$: 50 batches, or equivalently retrain after every 40 steps.
   - $F2$: 100 batches, or equivalently retrain after every 20 steps.
   - $A1$: $q = 5$
   - $A2$: $q = 10$
4. For NeuralGCB run on a synthetic function:
   - $F1$: $q_1 = 4, q_2 = 20, q_r = 40$ for $r \geq 3$ (i.e., retrain after q_r steps)
   - $F2$: $q_1 = 2, q_2 = 10, q_r = 20$ for $r \geq 3$.
   - $A1$: $q_1 = 1.2, q_2 = 1.8, q_r = 2.7$ for $r \geq 3$.
   - $A2$: $q_1 = 1.5, q_2 = 2.25, q_r = 3.375$ for $r \geq 3$.
5. For NeuralGCB run on Mushroom:
   - $F1$: $q_1 = 10, q_2 = 30, q_r = 60$ for $r \geq 3$ (i.e., retrain after q_r steps)
   - $F2$: $q_1 = 10, q_2 = 20, q_r = 40$ for $r \geq 3$.
   - $A1$: $q_1 = 300, q_2 = 1200, q_r = 4800$ for $r \geq 3$.
   - $A2$: $q_1 = 500, q_2 = 2000, q_r = 8000$ for $r \geq 3$.
6. For NeuralGCB run on Shuttle:
   - $F1$: $q_1 = 5, q_2 = 12, q_r = 31$ for $r \geq 3$ (i.e., retrain after q_r steps)
   - $F2$: $q_1 = 5, q_2 = 10, q_r = 20$ for $r \geq 3$ (i.e., retrain after q_r steps)
   - $A1$: $q_r = 3$ for all r
   - $A2$: $q_r = 4$ for all r

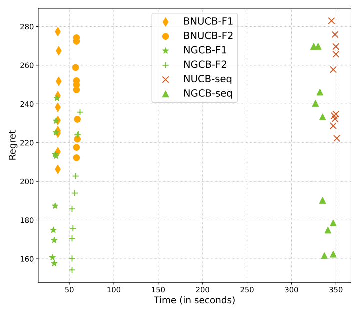
We use this notation of $F1, F2, A1$ and $A2$ to refer to the training schedules in the plots shown in the next section.



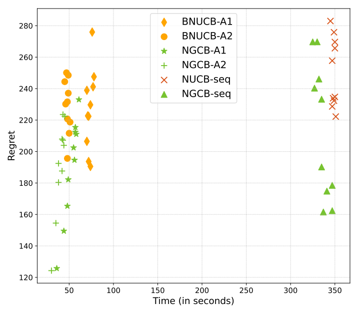
(a) $h_1(x)$ with $\sigma_1(x)$



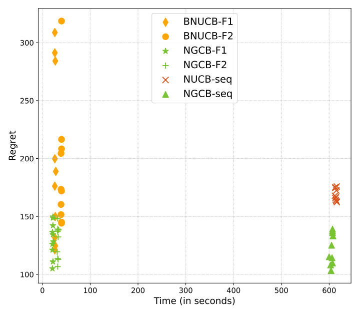
(b) $h_1(x)$ with $\sigma_2(x)$



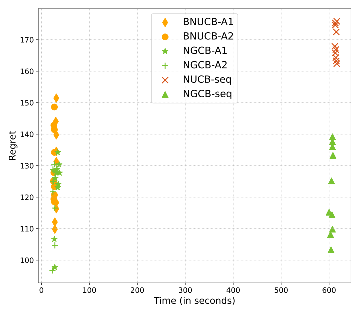
(c) $h_1(x)$ with fixed batch and $\sigma_1(x)$



(d) $h_1(x)$ with adaptive batch and $\sigma_1(x)$



(e) $h_1(x)$ with fixed batch and $\sigma_2(x)$



(f) $h_1(x)$ with adaptive batch and $\sigma_2(x)$

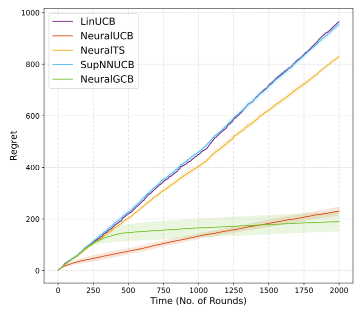
Figure 2: Plots with reward function $h_1(x)$

E.3. Plots

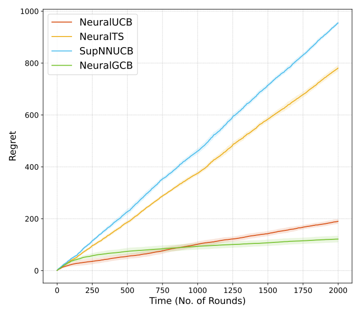
In this section, we plot the performance of NeuralGCB against several baselines for different experimental setups and reward functions. For each reward function, we plot the following 6 figures:

- Fig. (a): We plot the cumulative regret against time (number of rounds) for various baselines for the case where the activation function of the underlying neural net is σ_1, or equivalently, ReLU.
- Fig (b): We plot the cumulative regret against time (number of rounds) for various baselines for the case where the activation function of the underlying neural net is σ_2.
- Fig (c)-(f): We compare the regret and running time (in seconds) between the batched and the sequential versions of NeuralUCB and NeuralGCB. We plot the regret incurred against time taken (running time, in seconds) for Batched-NeuralUCB (BNUCB), (Batched)-NeuralGCB (BNGCB), NeuralUCB (NUCB-seq), (Sequential) NeuralGCB (NGCB-seq) under different training schedules for 10 different runs. In particular, in (c) and (e), we consider fixed batch sizes with σ_1 and σ_2 as activation functions respectively and in (d) and (f) with consider adaptive batch sizes with σ_1 and σ_2 as activation functions respectively. The batch sizes used in the plots are described in the previous section.

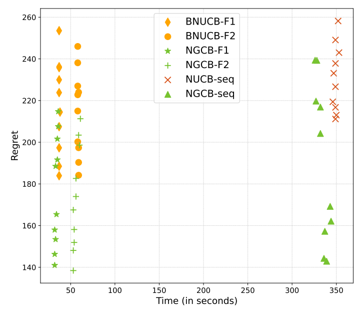
From the figures, it is evident that NeuralGCB outperforms all other existing algorithms on variety of datasets, including both synthetic and real world datasets. Moreover, the conclusion holds equally well for different activation functions, further bolstering the practical efficiency of the proposed algorithm. Additionally, it can be noted that the regret incurred by the algorithms in experiments with σ_2 as the activation is less than that for the experiments with σ_1 as the activation, thereby demonstrating the effect of smooth kernels on the performance of the algorithms in practice. we would like to point out that we only focus on algorithms with theoretical guarantees in this study. There are various approximate TS algorithms that also have good empirical performance (Riquelme et al., 2018) but lack such provable efficiency, which is central to the motivation of our paper. In addition, the formulation of contextual bandits in Riquelme et al. (2018) is different from the usual one in the literature (Valko et al., 2013; Zhou et al., 2020; Zhang et al., 2021;



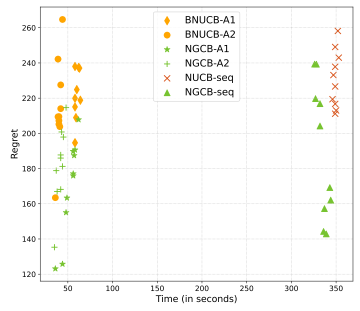
(a) $h_2(x)$ with $\sigma_1(x)$



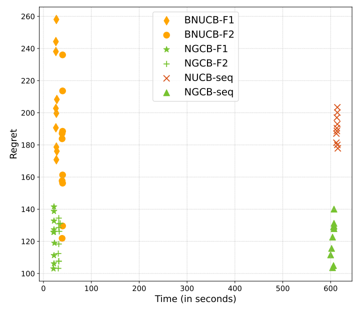
(b) $h_2(x)$ with $\sigma_2(x)$



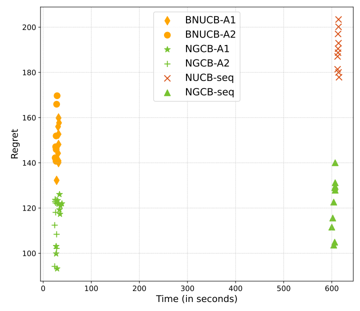
(c) $h_2(x)$ with fixed batch and $\sigma_1(x)$



(d) $h_2(x)$ with adaptive batch and $\sigma_1(x)$

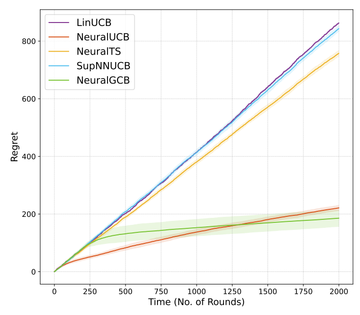


(e) $h_2(x)$ with fixed batch and $\sigma_2(x)$

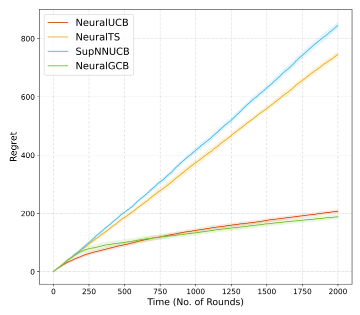


(f) $h_2(x)$ with adaptive batch and $\sigma_2(x)$

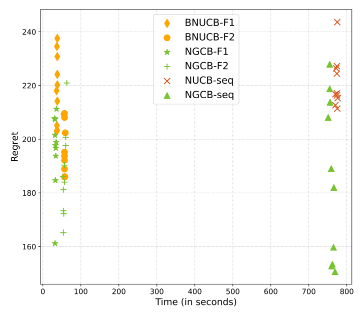
Figure 3: Plots with reward function $h_2(x)$



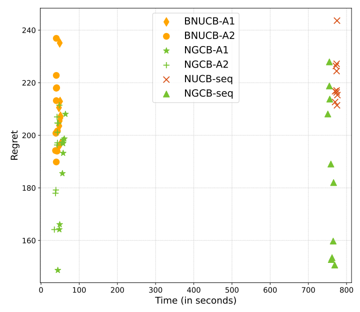
(a) $h_3(x)$ with $\sigma_1(x)$



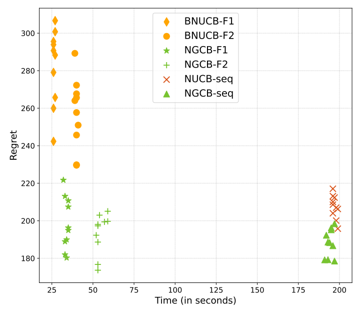
(b) $h_3(x)$ with $\sigma_2(x)$



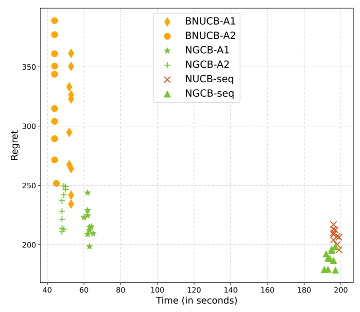
(c) $h_3(x)$ with fixed batch and $\sigma_1(x)$



(d) $h_3(x)$ with adaptive batch and $\sigma_1(x)$

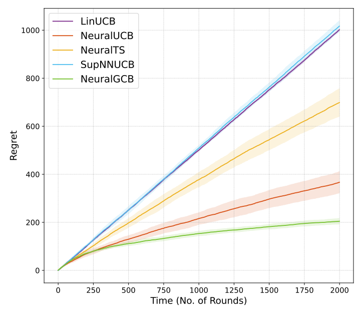


(e) $h_3(x)$ with fixed batch and $\sigma_2(x)$

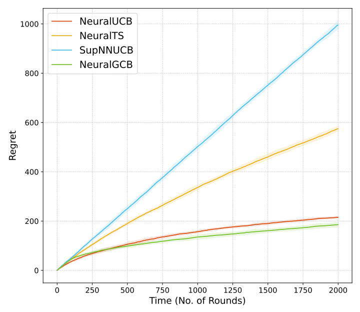


(f) $h_3(x)$ with adaptive batch and $\sigma_2(x)$

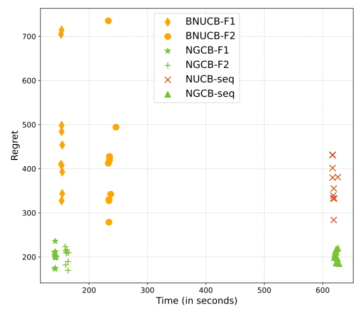
Figure 4: Plots with reward function $h_3(x)$



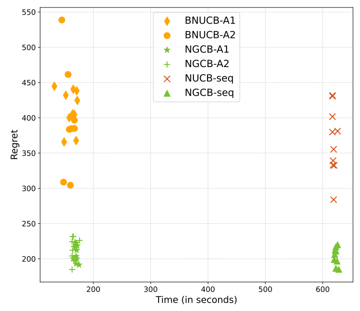
(a) Mushroom with $\sigma_1(x)$



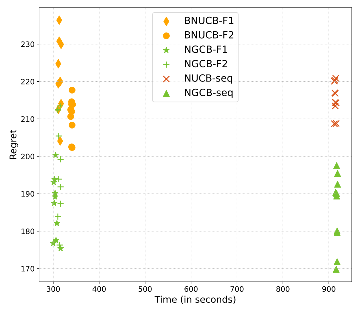
(b) Mushroom with $\sigma_2(x)$



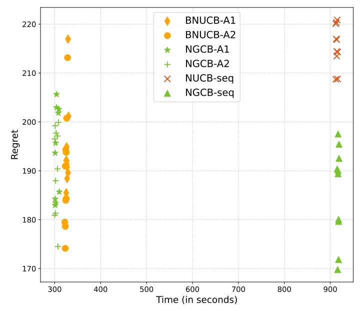
(c) Mushroom with fixed batch and $\sigma_1(x)$



(d) Mushroom with adaptive batch and $\sigma_1(x)$

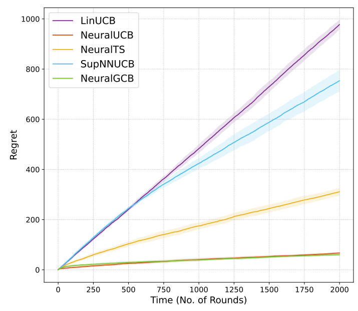


(e) Mushroom with fixed batch and $\sigma_2(x)$

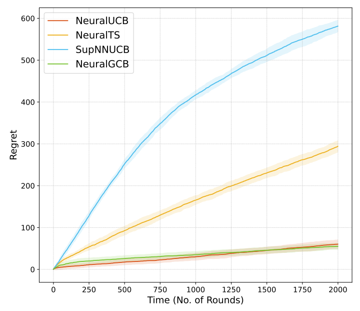


(f) Mushroom with adaptive batch and $\sigma_2(x)$

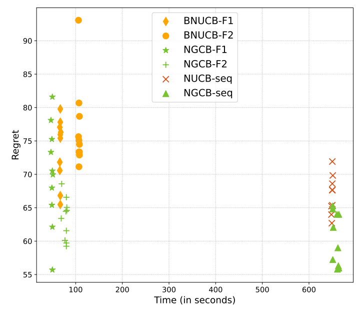
Figure 5: Plots for the dataset Mushroom



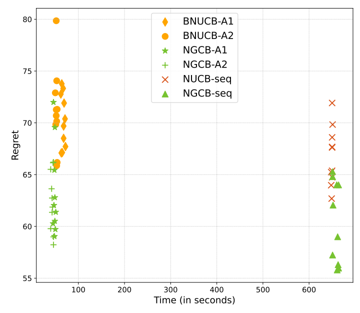
(a) Statlog with $\sigma_1(x)$



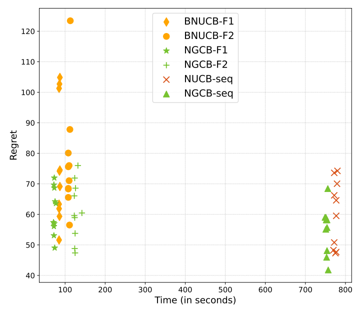
(b) Statlog with $\sigma_2(x)$



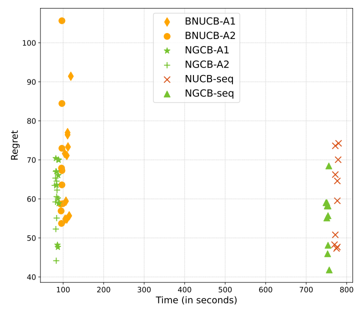
(c) Statlog with fixed batch and $\sigma_1(x)$



(d) Statlog with adaptive batch and $\sigma_1(x)$



(e) Statlog with fixed batch and $\sigma_2(x)$



(f) Statlog with adaptive batch and $\sigma_2(x)$

Figure 6: Plots for the dataset Statlog (Shuttle)

Gu et al., 2021; Kassraie & Krause, 2022), considered in our work, which also makes a direct empirical comparison infeasible.

Lastly, the extensive study with different training schedules shows that batched version performs almost as well as the fully sequential version in terms of regret on all the datasets while having a considerably smaller running time. It can be noted from the plots that NeuralGCB has a superior performance compared to Batched-NeuralUCB in terms of regret for comparable running times on different datasets and with different activation functions which further strengthens the case of NeuralGCB as a practically efficient algorithm.

All the experiments were carried out using Python3 on a computer (CPU) with 12 GB RAM and Intel i7 processor (3.4 GHz) with an overall compute time of around 250-300 hours.