Why Quantization Improves Generalization: NTK of Binary Weight Neural Network

Anonymous Authors[1]

Abstract

Quantized neural networks have drawn a lot of attention as they reduce the space and computational complexity during the inference. Moreover, there has been folklore that quantization acts as an implicit regularizer and thus can improve the generalizability of neural networks, yet no existing work formalizes this interesting folklore. In this paper, we take the binary weights in a neural network as random variables under stochastic rounding, and study the distribution propagation over different layers in the neural network. We propose a *quasi neural network* to approximate the distribution propagation, which is a neural network with continuous parameters and smooth activation function. We derive the neural tangent kernel (NTK) for this quasi neural network, and show the eigenvalue of NTK decays at approximately exponential rate, which is comparable to that of Gaussian kernel with randomized scale. We use experiments to verify that the quasi neural network we proposed can well approximate binary weight neural network. Lastly, binary weight neural network gives a lower generalization gap compared with real value weight neural network.

1. Introduction

It has been found that by quantizing the parameters in a neural network, the memory footprint and computing cost can be greatly decreased with little to no loss in accuracy (Gupta et al., 2015). Furthermore, Hubara et al. (2016); Courbariaux et al. (2015) argued that quantization serves as an implicit regularizer and thus should increase the generalizability of neural network comparing to its full precision version. However, there is no formal theoretical investigation of this statement to the best of our knowledge.

Empirical results show that the traditional statistical learning techniques based on uniform convergence (e.g., VC-dimension (Blumer et al., 1989)) do not satisfactorily explain the generalization ability of neural networks. Zhang et al. (2016) showed that neural networks can perfectly fit the training data even if the labels are random, yet it generalized well when the data are not random. This seems to suggest that the model capacity of a neural network depends on not only the model, but also the dataset. Recent studies (He and Tao, 2020) managed to understand the empirical performance in a number of different aspects, including modeling stochastic gradient (SGD) with stochastic differential equation (SDE) (Weinan et al., 2019), studying the geometric structure of loss surface (He et al., 2020), and overparameterization – a particular asymptotic behavior when the number of parameters of the neural network tends to infinity (Li et al., 2018; Choromanska et al., 2015; Allen-Zhu et al., 2018; Arora et al., 2019a). Recently, it was proven that the training process of neural network in the overparameterized regime corresponds to kernel regression with Neural Tangent Kernel (NTK) (Jacot et al., 2018). A line of work (Bach, 2017; Bietti and Mairal, 2019; Geifman et al., 2020; Chen and Xu, 2020) further studied Mercer's decomposition of NTK and proved that it is similar to a Laplacian kernel in terms of the eigenvalues.

In this paper, we propose modeling a two-layer binary weight neural network using a model with continuous parameters. Specifically, we assume the binary weights are drawn from the Bernoulli distribution where the parameters of the distribution (or the mean of the weights) are trainable parameters. We propose a *quasi neural network*, which has the same structure as a vanilla neural network but has a different activation function, and prove one can analytically approximate the expectation of output of this binary weight neural network with this quasi neural network. Using this model, our main contributions are as follows:

- Under the overparameterized regime, we prove that the gradient computed from BinaryConnect algorithm is approximately an unbiased estimator of the gradient of the quasi neural network, hence such a quasi neural network can model the training dynamic of binary weight neural network.
- We study the NTK of two-layer binary weight neural networks by studying the "quasi neural network", and show that the eigenvalue of this kernel decays at an exponential rate, in contrast with the polynomial rate in a ReLU neural network (Chen and Xu, 2020; Geif-

man et al., 2020). We reveal the similarity between the Reproducing kernel Hilbert space (RKHS) of this kernel with Gaussian kernel, and it is a strict subset of function as the RKHS of NTK in a ReLU neural network. This indicates that the model capacity of binary weight neural network is smaller than that with real weights, and explains higher training error and lower generalization gap observed empirically.

2. Preliminary

Neural tangent kernel. It has been found that an overparameterized neural network has many local minima. Using the connection between feature map and kernel learning, the optimization problem reduces to kernel optimization problem. Denote Θ as the collection of all the parameters in a neural network f_Θ before an iteration, and Θ^+ as the parameters after this iteration. Let in denote fixed distribution in the input space. Using Taylor expansion, for any testing data $\tilde{x}$, let the stepsize be η, the first-order update rule of gradient descent can be written as (note $\Theta^+ - \Theta = \eta \mathbb{E}_{x\sim in}\left[\nabla_\Theta \text{loss}(f_\Theta(x))\right]$)

$$f_{\Theta^+}(x') - f_\Theta(x') := \eta \mathbb{E}_{x\sim in}\left[\mathcal{K}(x, x')\, \text{loss}'(f_\Theta(x))\right].$$

with the limiting kernel (in a.s. sense) to be defined as:

$$\mathcal{K}(x, x') := \lim_{\text{width}\to\infty} \nabla_\Theta^\top f_\Theta(x) \cdot \nabla_\Theta f_\Theta(x').$$

The limiting kernel $\mathcal{K}$ is usually referred as the neural tangent kernel (NTK) (Jacot et al., 2018).

Exponential kernel. A common class of kernel functions used in machine learning is the exponential kernel, which is a radial basis function kernel with the general form $\mathcal{K}(x, x') = \exp(-(c\|x - x'\|)^\gamma), \quad c > 0, \gamma \geq 1$. When $\gamma = 1$, this kernel is known as the Laplacian kernel, and when $\gamma = 2$, it is called the Gaussian kernel.

Training neural networks with quantized weights. Among various methods to train a neural network, BinaryConnect (BC) (Courbariaux et al., 2015) introduces a real-valued buffer θ and use it to accumulate the gradients. The weights will be quantized just before forward and backward propagation, which can benefit from the reduced computing complexity via the update

$$\theta^+ \leftarrow \theta - \eta \frac{\partial \tilde{f}_w(x)}{\partial w}, \quad w^+ \leftarrow Quantize(\theta^+), \qquad (1)$$

where $Quantize(\cdot) : \mathbb{R} \to \{-1, 1\}$ denotes the quantization function which will be discussed in Section B.2, w denotes the binary (or quantized) weights, θ and θ^+ denote the real valued buffer before and after an iteration respectively, η is the learning rate, and $f_w(\cdot)$ denotes the neural network with parameter w. The detailed algorithm can be founded in Section B.3.

3. Approximation of binary weight neural network via Quasi neural network

Given a fixed input and real-value model parameters Θ, under the randomness of stochastic rounding, the output of this binary weight neural network is a random variable. Furthermore, as the width of the neural network d_1 tends to infinity, we define a parameter sequence $\{\Theta_{d_1}\}$ and prove that with parameters from this sequence, the output of a linear layer tends to Gaussian distribution according to central limit theorem (CLT). Specifically, we give a closed form equation to compute the mean and variance of output of all the layers $\mu_\ell, \sigma_\ell, \nu_\ell, \varsigma_\ell$, and then marginalize over random initialization of Θ to further simplify this equation.[1] We call this model *quasi neural network*, which is given below:

$$\begin{aligned}
x_{1,i} &= \frac{1}{\sqrt{d}} \sum_{k=1}^{d} w_{0,ki} x_k + b_{0,i}, \forall i \in [d_1];\\
\nu_{1,j} &= \sqrt{\frac{c}{d_1}} \sum_{i=1}^{d_1} \theta_{1,ij} x_{1,j} + \beta b_{1,i}, \forall j \in [d_2];\\
\mu_{2,j} &= \tilde{\psi}(\nu_{1,j}), \forall j \in [d_2];\;\; \bar{y} = \frac{1}{\sqrt{d_2}} \sum_{j=1}^{d_2} w_{2,j}\mu_{2,j} + \beta b_2.
\end{aligned} \qquad (2)$$

We assume the following regularity conditions.

Assumption 1. *After training the binary weight neural network as in (1), all the weights $\theta_{\ell,ij}$ stay in $[-1, 1]$.*

Because of the lazy training property of the overparameterized NN, the model parameters $\theta_{\ell,ij}$ stay close to the initialization point during the training process, so this assumption can be satisfied by initializing $\theta_{\ell,ij}$ with smaller absolute value and/or applying weight decay during training. We also assume $\|\boldsymbol{x}\|_2 = 1, \forall \boldsymbol{x} \in \mathcal{D} \subseteq \mathbb{R}^d$, which is common for studying NTK (Bach, 2017; Bietti and Mairal, 2019).

Convergence of conditioned variance. In this part, we assume that the model parameters are chosen from "Good Initialization sequence" (Definition 9), which is almost surely on the limit $d_1 \to \infty$ as is proven in Theorem 10, and study the distribution of $\nu_{1,j}$ and $\varsigma_{1,j}$.

Theorem 1. *For any fixed "Good Initialization sequence" $\{\Theta_{d_1}\} \in \mathcal{G}$, on the limit $d_1 \to \infty$, for any finite d_2, $\nu_{1,j}$ converges to Gaussian distribution which are independent of each other, and $\varsigma_{1,j}^2$ converges a.s. to $\tilde{\varsigma}_1^2 = \frac{c}{d}(1 - \text{Var}[\theta])$.*

With this approximation, we can replace the variance $\varsigma_{1,i}$ in Equation (11) with $\tilde{\varsigma}_1$ and leave the mean of output in the linear layer as the only variable in the quasi neural network. Formal proof can be found in Appendix E.4. Note the propagation function in the linear layer (the first equation in (9)) is also a linear function in x and θ. This motivates us to

[1] our notation in this section can be found in Appendix B.1.

compute $\bar{y}$ using a neural network-like function as is given in (2), where $\tilde{\psi}(\cdot)$ is

$$\tilde{\psi}(\nu_{1,j}) = \mathbb{E}[\psi(y_{1,j})|\nu_{1,j}] = \tilde{\varsigma}_1 \phi\left(\frac{\nu_{1,i}}{\tilde{\varsigma}_1}\right) + \nu_{1,j}\Phi\left(\frac{\nu_{1,j}}{\tilde{\varsigma}_1}\right). \tag{3}$$

This equation gives a closed-form connection between the mean of output of neural network $\bar{y}$ and the real-valued model parameter Θ, and allows up to apply existing tools for analyzing neural networks with real-valued weight to analysis binary weight neural network. In this part, we can compute the gradients using binary weights as in BinaryConnect Algorithm.

Theorem 2. *The expectation of gradients to output with respect to weights computed by sampling the quantized weights equals the gradients of "quasi neural network" defined above in* (2) *satisfy*

$$\lim_{d_2\to\infty}\lim_{d_1\to\infty}\sqrt{d_2}\left(\frac{\partial\bar{y}}{\partial\theta_{1,ij}} - \mathbb{E}\left[\frac{\partial y}{\partial w_{1,ij}}\middle|\Theta^{(d_1,d_2)}\right]\right) = 0,$$
$$\lim_{d_2\to\infty}\lim_{d_1\to\infty}\sqrt{d_2}\left(\frac{\partial\bar{y}}{\partial b_{1,j}} - \mathbb{E}\left[\frac{\partial y}{\partial b_{1,j}}\middle|\Theta^{(d_1,d_2)}\right]\right) = 0,$$
$$\lim_{d_2\to\infty}\lim_{d_1\to\infty}\sqrt{d_1 d_2}\left(\frac{\partial\bar{y}}{\partial w_{2,j}} - \mathbb{E}\left[\frac{\partial y}{\partial w_{2,j}}\middle|\Theta^{(d_1,d_2)}\right]\right) = 0.$$

Theorem 3. *For MSE loss,* $\mathrm{loss}(y) = \frac{1}{2}(y-z)^2$, *where* z *is the ground-truth label, the gradient of the loss converges to*

$$\lim_{d_2\to\infty}\lim_{d_1\to\infty}\sqrt{d_2}\left(\frac{\partial\mathrm{loss}(\bar{y})}{\partial\theta_{1,ij}} - \mathbb{E}\left[\frac{\partial\mathrm{loss}(y)}{\partial w_{1,ij}}\middle|\Theta^{(d_1,d_2)}\right]\right) = 0,$$
$$\lim_{d_2\to\infty}\lim_{d_1\to\infty}\sqrt{d_2}\left(\frac{\partial\mathrm{loss}(\bar{y})}{\partial b_{1,j}} - \mathbb{E}\left[\frac{\partial\mathrm{loss}(y)}{\partial b_{1,j}}\middle|\Theta^{(d_1,d_2)}\right]\right) = 0,$$
$$\lim_{d_2\to\infty}\lim_{d_1\to\infty}\sqrt{d_1 d_2}\left(\frac{\partial\mathrm{loss}(\bar{y})}{\partial w_{2,j}} - \mathbb{E}\left[\frac{\partial\mathrm{loss}(y)}{\partial w_{2,j}}\middle|\Theta^{(d_1,d_2)}\right]\right) = 0.$$

In other words, the BinaryConnect algorithm provides an unbiased estimator to the gradients for the quasi neural network on this limit of overparameterization. The proof can be found in Appendix E.6 and Appendix E.7 respectively. Theorem 1 and Theorem 3 conclude that for an infinite wide neural network, the BinaryConnect algorithm is equivalent to training quasi neural network with stochastic gradient descent (SGD) directly. Furthermore, this points out the gradient flow of BinaryConnect algorithm and allows us to study this training process with NTK.

4. Capacity of Binary Weight Neural Network

As has been found in (Jacot et al., 2018), the dynamics of an overparameterized neural network trained with SGD is equivalent to kernel gradient descent where the kernel is NTK. As a result, the effective capacity of a neural network is equivalent to the RKHS of its NTK.

4.1. NTK of three-layer binary weight neural networks

We consider the NTK binary weight neural network by studying this "quasi neural network" defined as the limiting kernel $\mathcal{K}_{BWNN}(x,x') :=$

$$\lim_{d_2\to\infty}\lim_{d_1\to\infty}\sum_{i=1,j=1}^{d_1,d_2}\frac{\partial\bar{y}}{\partial\theta_{1,ij}}\frac{\partial\bar{y}'}{\partial\theta_{1,ij}} + \sum_{j=1}^{d_2}\frac{\partial\bar{y}}{\partial b_{1,j}}\frac{\partial\bar{y}'}{\partial b_{1,j}} + \sum_{j=1}^{d_2}\frac{\partial\bar{y}}{\partial w_{2,j}}\frac{\partial\bar{y}'}{\partial w_{2,j}} \tag{4}$$

where $\Theta := \{w_{1,ij}, b_{1,j}, b_{2,j}\}$ denotes all the trainable parameters. To find the basis and eigenvalues to this kernel, we apply spherical harmonics decomposition to this kernel, which is common among studying of NTK (Bach, 2017; Bietti and Mairal, 2019):

$$\mathcal{K}_{BWNN}(x,x') = \sum_{k=1}^{\infty} u_k \sum_{j=1}^{N(d,k)} Y_{k,j}(x)Y_{k,j}(x'), \tag{5}$$

where d denotes the dimension of x and x', $Y_{k,j}$ denotes the spherical harmonics of order k. This suggests that NTK of binary weight neural network and exponential kernel can be spanned by the same set of basis function. The key question is the decay rate of u_k with k.

Theorem 4. *The limit of NTK of a binary weight neural network can be decomposed using* (5). *If* $k \gg \frac{\tilde{c}^2}{2} = \frac{\mathrm{var}[\theta]}{2(1-\mathrm{Var}[\theta])}$ *and* $k \gg d$, *then*

$$\mathrm{Poly}_1(k)C^{-k} \le u_k \le \mathrm{Poly}_2(k)C^{-k}. \tag{6}$$

where $\mathrm{Poly}_1(k)$ *and* $\mathrm{Poly}_2(k)$ *denote polynomials of* k, *and* C *is a constant.*

Meawhile, Geifman et al. (2020) shows that for NTK in the continuous space, it holds that $C_1 k^{-d} \le u_k \le C_2 k^{-d}$, with constants C_1 and C_2. Since its decay rate is slower than that of the binary weight neural network, its RKHS covers a strict superset of functions (Geifman et al., 2020).

4.2. Comparison with Gaussian Kernel

Even if the input to a neural network x is constrained on a unit sphere, the first linear layer (together with the additional linear layer in front of it) will project it to the entire $\mathbb{R}^d$ space with Gaussian distribution. In order to simulate that, we define a kernel by randoming the scale of x and x' beforing taking them into a Gaussian kernel.

$$\mathcal{K}_{RGauss}(x,x') = \mathbb{E}_\kappa[\mathcal{K}_{Gauss}(\kappa x, \kappa x')],$$

where $\mathcal{K}_{Gauss}(x,x') = \exp\left(-\frac{\|x-x'\|^2}{\xi^2}\right)$ is a Gaussian kernel, $\kappa \sim \chi_d$ satisfy Chi distribution with d degrees of freedom. This scaling factor projects a random vector uniformly distributed on a unit sphere to Gaussian distributed. The corresponding eigenvalue satisfy

$$A_1 C^{-k} \le u_k \le A_2 C^{-k}, \tag{7}$$

where A_1, A_2, C are constants that depend on ξ. The dominated term in both (6) and (7) have an exponential decay rate C^{-k}, which suggests the similarity between NTK of binary weight neural network and Gaussian kernel. In comparison, Bietti and Mairal (2019); Geifman et al. (2020) showed that the eigenvalue of NTK decay at rate k^{-d}, which is slower that binary weight neural network or Gaussian kernel. Furthermore, Aronszajn's inclusion theorem suggests $\mathcal{H}_{\mathcal{K}_{BWNN}} \subset \mathcal{H}_{\mathcal{K}_{NN}}$, where $\mathcal{K}_{NN}$ denotes the NTK of real-valued weight neural network. In other words, the expressive power of binary weight neural network is weaker than its real valued counterpart on the limit that the width goes to infinity. Binary weight neural networks are less venerable to noise thanks to the smaller expressive power at the expense of failing to learn some "high frequency" components in the target function. This explains that binary weight neural network often achieve lower training accuracy and smaller generalization gap compared with real weight neural network.

4.3. Generalization gap

Making use of the decay rate of eigenvalues, in this section, we study the difference of generalization gap between vanilla neural networks and binary weight neural network. The generalization gap is defined as the difference between training error and testing error: $\mathbb{E}_{all}(f(x)-y)^2 - \mathbb{E}_{in}(f(x)-y)^2$ where $\mathbb{E}_{all}$ denotes the expectation over the testing set. The generalization gap can be evaluated using the effective degree of freedom. Let the training data be $y_i = f_0(x_i) + \epsilon_i, \epsilon \sim \mathcal{N}(0, \sigma^2)$, the effective degree of freedom $\mathrm{df}(f)$ is defined as

$$2\sigma^2 \mathrm{df}(f) = E[\|y' - f(y)\|_2^2] - E[\|y - f(y)\|_2^2]$$

where y' is an independent copy of y. The effective degree of freedom of Lasso problem $\hat{w} = \mathrm{argmin}_w \|Xw - y\|_2^2 + \lambda\|w\|_2^2 = (X^TX + \lambda I)^{-1}X^Ty$ can be calculated as

$$\mathrm{df}(\hat{w}) = \sum_{k=1}^{n} \frac{\lambda_k^2}{\lambda_k^2 + \lambda}$$

where $X \in \mathbb{R}^{n\times d}, n$ is the number of datapoints and d is the dimension of features, λ_i is the i-th eigenvalue of X^TX. When the number of training samples n is large enough, the eigenvalues of feature covariance matrix X^TX can be approximated by the eigenvalues of the kernel matrix. Furthermore, when λ is small enough (which corresponds to training for enough number of epochs in a neural network), faster decay rate of λ_k leads to a smaller effective degree of freedom, which in turn means smaller generalization gap. This indicates that for an infinite wide neural network with enough number of samples, binary weight neural network has probability lower generalization gap than the vanilla counterpart (although the training error can be higher).

5. Numerical result for generalization gap

In this section, we provide experiments to validate the effectiveness of generalization for BWNN. We consider the UCI dataset and MNIST-like dataset.

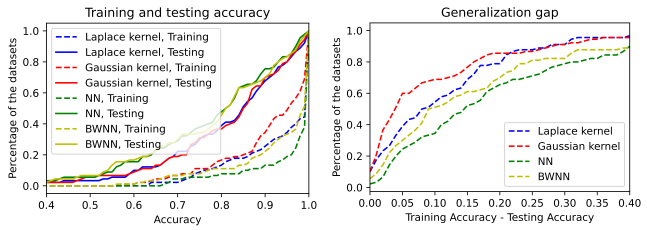


Figure 1: Accuracy and generalization gap on selected 90 UCI datasets. The lines show the accuracy metric of a classifier from the lowest to the highest against their percentiles of datasets.

5.1. UCI dataset

We compare the performance of the neural network with/without binary weight and kernel learning using the same set of 90 small scale UCI datasets with less than 5000 data points as in Geifman et al. (2020); Arora et al. (2019b). We report the training accuracy and testing accuracy of both vanilla neural network (NN) and binary weight neural network (BWNN) in Figure 1. To further illustrate the difference, we list the paired T-test result of neural network (NN) against binary weight neural network (BWNN), and Gaussian kernel (Gaussian) against Laplace kernel (Laplace) using in Table 2. As can be seen from the results, although the Laplacian kernel gets higher training accuracy than the Gaussian kernel, its testing accuracy is almost the same as the latter one. In other words, the former has smaller generalization gap than the latter which can also be observed in Table 2. Similarly, a neural network gets higher training accuracy than a binary weight neural network but gets similar testing accuracy.

5.2. MNIST-like dataset

We compare the performance of neural networks with binary weights (Binary) with its counterpart with real value weights (Real). We take the number of training samples as a parameter by random sampling the training set and use the original test set for testing. The experiments are repeated 10 times and the mean and standard derivation is shown in Figure 4. In the MNIST dataset, the performance of neural networks with or without quantization is similar. This is because MNIST is simpler and less vulnerable to overfitting. On the other hand, the generalization gap with weight quantized is much smaller than without it in FashionMNIST (Xiao et al., 2017) dataset, which matches our prediction.

References

Hande Alemdar, Vincent Leroy, Adrien Prost-Boucle, and Frédéric Pétrot. Ternary neural networks for resource-efficient ai applications. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 2547–2554. IEEE, 2017.

Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. *arXiv preprint arXiv:1811.04918*, 2018.

Sanjeev Arora, Simon Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning*, pages 322–332. PMLR, 2019a.

Sanjeev Arora, Simon S Du, Zhiyuan Li, Ruslan Salakhutdinov, Ruosong Wang, and Dingli Yu. Harnessing the power of infinitely wide deep nets on small-data tasks. *arXiv preprint arXiv:1910.01663*, 2019b.

Francis Bach. Breaking the curse of dimensionality with convex neural networks. *The Journal of Machine Learning Research*, 18(1):629–681, 2017.

Alberto Bietti and Julien Mairal. On the inductive bias of neural tangent kernels. In *Advances in Neural Information Processing Systems*, pages 12893–12904, 2019.

Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K Warmuth. Learnability and the vapnik-chervonenkis dimension. *Journal of the ACM (JACM)*, 36(4):929–965, 1989.

Blake Bordelon, Abdulkadir Canatar, and Cengiz Pehlevan. Spectrum dependent learning curves in kernel regression and wide neural networks. *arXiv preprint arXiv:2002.02561*, 2020.

Lin Chen and Sheng Xu. Deep neural tangent kernel and laplace kernel have the same rkhs. *arXiv preprint arXiv:2009.10683*, 2020.

Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. In *Artificial intelligence and statistics*, pages 192–204. PMLR, 2015.

Tianshu Chu, Qin Luo, Jie Yang, and Xiaolin Huang. Mixed-precision quantized neural networks with progressively decreasing bitwidth. *Pattern Recognition*, 111:107647, 2021.

Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. *Advances in neural information processing systems*, 28:3123–3131, 2015.

Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv preprint arXiv:1602.02830*, 2016.

Yukun Ding, Jinglan Liu, Jinjun Xiong, and Yiyu Shi. On the universal approximability and complexity bounds of quantized relu neural networks. *arXiv preprint arXiv:1802.03646*, 2018.

Yinpeng Dong, Renkun Ni, Jianguo Li, Yurong Chen, Jun Zhu, and Hang Su. Learning accurate low-bit deep neural networks with stochastic quantization. *arXiv preprint arXiv:1708.01001*, 2017.

Amnon Geifman, Abhay Yadav, Yoni Kasten, Meirav Galun, David Jacobs, and Ronen Basri. On the similarity between the laplace and neural tangent kernels. *arXiv preprint arXiv:2007.01580*, 2020.

Suyog Gupta, Ankur Agrawal, Kailash Gopalakrishnan, and Pritish Narayanan. Deep learning with limited numerical precision. In *International Conference on Machine Learning*, pages 1737–1746, 2015.

Fengxiang He and Dacheng Tao. Recent advances in deep learning theory. *arXiv preprint arXiv:2012.10931*, 2020.

Fengxiang He, Bohan Wang, and Dacheng Tao. Piecewise linear activations substantially shape the loss surfaces of neural networks. *arXiv preprint arXiv:2003.12236*, 2020.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In *Proceedings of the 30th international conference on neural information processing systems*, pages 4114–4122. Citeseer, 2016.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Quantized neural networks: Training neural networks with low precision weights and activations. *The Journal of Machine Learning Research*, 18(1):6869–6898, 2017.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pages 8571–8580, 2018.

Dawei Li, Tian Ding, and Ruoyu Sun. Over-parameterized deep neural networks have no strict local minima for any continuous activations. *arXiv preprint arXiv:1812.11039*, 2018.

Tailin Liang, John Glossner, Lei Wang, Shaobo Shi, and Xiaotong Zhang. Pruning and quantization for deep neural network acceleration: A survey. *Neurocomputing*, 461: 370–403, 2021.

Michele Marchesi, Gianni Orlandi, Francesco Piazza, and Aurelio Uncini. Fast neural networks without multipliers. *IEEE transactions on Neural Networks*, 4(1):53–62, 1993.

Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European conference on computer vision*, pages 525–542. Springer, 2016.

James B Simon, Madeline Dickens, and Michael R DeWeese. Neural tangent kernel eigenvalues accurately predict generalization. *arXiv preprint arXiv:2110.03922*, 2021.

E Weinan, Jiequn Han, and Qianxiao Li. A mean-field optimal control formulation of deep learning. *Research in the Mathematical Sciences*, 6(1):1–41, 2019.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

A. Related work

Quantized neural networks. There is a large body of work that focuses on training neural networks with quantized weights (Marchesi et al., 1993; Hubara et al., 2017; Gupta et al., 2015; Liang et al., 2021; Chu et al., 2021), including considering radically quantizing the weights to binary (Courbariaux et al., 2016; Rastegari et al., 2016) or ternary (Alemdar et al., 2017) values, which often comes at a mild cost on the model's predictive accuracy. Despite all these empirical works, the theoretical analysis of quantized neural networks and their convergence is not well studied. Many researchers believed that quantization adds noise to the model, which serves as an implicit regularizer and makes neural networks generalize better (Hubara et al., 2016; Courbariaux et al., 2015), but this statement is instinctive and has never been formally proved to the best of our knowledge. One may argue that binary weight neural networks have a smaller parameter space than its real weight counterpart, yet Ding et al. (2018) showed that a quantized ReLU neural network with enough parameters can approximate any ReLU neural network with arbitrary precision. These seemingly controversy results motivate us to find another way to explain the stronger generalization ability that is observed empirically.

Theory of deep learning and NTK. A notable recent technique in developing the theory of neural networks is the neural tangent kernel (NTK) (Jacot et al., 2018). It draws the connection between an over-parameterized neural network and the kernel learning. This makes it possible to study the generalization of overparameterized neural network using more mature theoretical tools from kernel learning (Bordelon et al., 2020; Simon et al., 2021).

The expressive power of kernel learning is determined by the RKHS of the kernel. Many researches have been done to identify the RKHS. Bach (2017); Bietti and Mairal (2019) studied the spectral properties of NTK of a two-layer neural network without bias. Geifman et al. (2020) further studied the NTK with bias and showed that the RKHS of two layer neural networks contains the same set of functions as RKHS of the Laplacian kernel. Chen and Xu (2020) expanded this result to arbitrary layer neural networks and showed that RKHS of arbitrary layer neural network is equivalent to Laplacian kernel. All these works are based on neural networks with real weights, and to the best of our knowledge, we are the first to study the NTK and generalization of binary weight neural networks.

B. Problem statement

B.1. Notations

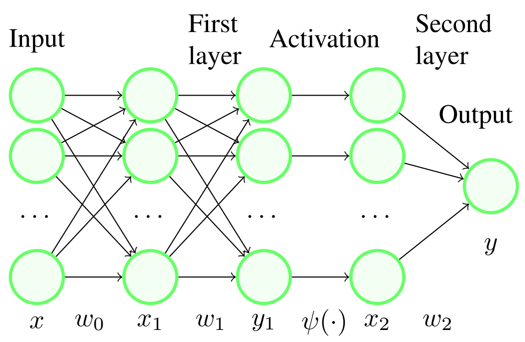
In this paper, we use $w_{\ell,ij}$ to denote the binary weights in the ℓ-th layer, $\theta_{\ell,ij}$ to denote its real-valued counterpart, and $b_{\ell,i}$ to denote the (real valued) bias. Θ is the collection of all the real-valued model parameters which will be specified in Section B.2. The number of neurons in the ℓ-th hidden layer is d_ℓ, the input to the ℓ-th linear layer is $\boldsymbol{x}_\ell$ and the output is $\boldsymbol{y}_\ell$. d denote the number of input features. Besides, we use $\boldsymbol{x}$ to denote the input to this neural network, y to denote the output and z to denote the label.

We focus on the mean and variance under the randomness of stochastic rounding. Denote

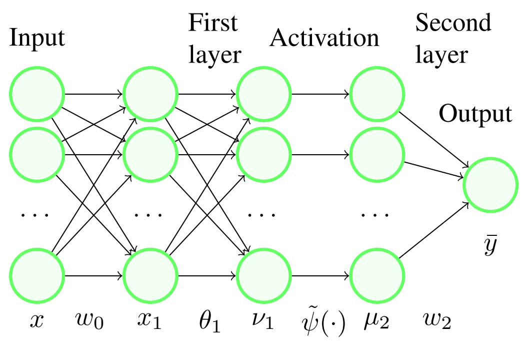
$$\begin{aligned}
&\mu_{\ell,i} := \mathbb{E}[x_{\ell,i}|\boldsymbol{x},\Theta], \quad \sigma^2_{\ell,i} := \mathrm{Var}[x_{\ell,i}|\boldsymbol{x},\Theta],\\
&\nu_{\ell,i} := \mathbb{E}[y_{\ell,i}|\boldsymbol{x},\Theta], \quad \varsigma^2_{i,\ell} := \mathrm{Var}[y_{\ell,i}|\boldsymbol{x},\Theta], \quad \bar{y} := \mathbb{E}[y|\Theta].
\end{aligned}$$

We use $\psi(x) = \max(x,0)$ to denote ReLU activation function, and in to denote the (discrete) distribution of training dataset. $\mathbb{E}_{in}[\cdot] := \mathbb{E}_{(\boldsymbol{x},z)\sim in}[\cdot]$ denotes the expectation over training dataset, or "sample average". We use bold symbol to denote a collection of parameters or variables $\boldsymbol{w}_2 = [w_{2,j}], \boldsymbol{b}_2 = [b_{2,j}], \boldsymbol{\nu}_1 = [\nu_{1,j}], \boldsymbol{\theta}_1 = [\theta_{1,ij}], i \in [d_1], j \in [d_2]$.

In this work, we target on stochastic quantization (Dong et al., 2017), which often yields higher accuracy empirically compared with deterministic rounding (Courbariaux et al., 2015). This also creates a smooth connection between the binary weights in a neural network and its real-valued parameters, which can be expressed as a quasi neural network defined in the following section. Our formulation of the binary weight neural network with stochastic rounding is deferred to subsection B.2.



(a) Binary weight neural network we focus on.



(b) Quasi neural network.

B.2. Binary weight neural network (BWNN)

We briefly state our formulation of Binary weight neural network (BWNN). Let $w_{\ell,ij} = Quantize(\theta_{\ell,ij}), \theta_{\ell,ij} \in [-1,1]$ be the binary weights from stochastic quantization function, which satisfy Bernoulli distribution:

$$w_{\ell,ij} = \begin{cases} +1, & \text{with probability } p_{\ell,ij} = \frac{\theta_{\ell,ij}+1}{2}, \\ -1, & \text{with probability } 1 - p_{\ell,ij}. \end{cases} \tag{8}$$

This relationship leads to $\mathbb{E}[w_{\ell,ij}|\theta_{\ell,ij}] = \theta_{\ell,ij}$.

We focus on a ReLU neural network with one hidden layer and two fully connect layers, which was also studied in Bach (2017); Bietti and Mairal (2019) except quantization. Besides, we add a linear layer ("additional layer") in front of this neural network to project the input to an infinite dimension space. We randomly initialize the weights in this layer and leave it fixed (not trainable) throughout the training process. Furthermore, we quantize the weights in the first fully connect layer $w_{1,ij}$ and add a real-valued buffer $\theta_{1,ij}$ which determines the distribution of $w_{1,ij}$ as in (8), and leave the second layers not quantized. It is a common practice to leave the last layer not quantized, because this often leads to better empirical performance. If the second layer is quantized as well, the main result of this paper will not be changed. This can be checked in subsection E.8.

Remark 5. *In many real applications, e.g. computer vision, the dimension of data are often very large ($\approx 10^3$) while they are laying in the lower dimension linear subspace, so we can take the raw input in these applications as the output of the additional layer, and the NN in this case is a two-layer NN where the first layer is quantized.*

The set of all the real-valued parameters is $\Theta = \{\theta_{\ell_1,ij}, w_{\ell_2,ij}, b_{\ell,i}\}$. The neural network can be expressed as

$$x_{1,i} = \frac{1}{\sqrt{d}} \sum_{k=1}^{d} w_{0,ki} x_k + b_{0,i}, \forall i \in [d_1]; \quad y_{1,j} = \sqrt{\frac{c}{d_1}} \sum_{i=1}^{d_1} w_{1,ij} x_{1,i} + b_{1,j}, \forall j \in [d_2];$$

$$x_{2,j} = \psi(y_{1,j}), \forall j \in [d_2]; \qquad y = \frac{1}{\sqrt{d_2}} \sum_{j=1}^{d_2} w_{2,j} x_{2,j} + b_2.$$

We follow the typical setting of NTK papers (Geifman et al., 2020) in initializing the parameters except the quantized parameters. As for the quantized parameters, we only need to specify the real-valued buffer of the weights in the first layer $\theta_{1,ij}$.

Assumption 2. *We randomly initialize the weights in the "additional layer" and second linear layer independently as $w_{0,ki}, w_{2,j} \sim \mathcal{N}(0,1)$, and initialize all the biases to 0. The real-valued buffer of the weights are initialized independently identical with zero mean, variance $\mathrm{Var}[\theta]$ and bounded in $[-1,1]$.*

Remark 6. *Our theory applies to any initial distribution of $\theta_{1,ij}$ as long as it satisfies the constraint above. One simple example is the uniform distribution in $[-1,1]$, which has variance $\mathrm{Var}[\theta] = 1/3$.*

B.3. BinaryConnect Algorithm

In this section, we briefly review the BinaryConnect algorithm (Courbariaux et al., 2015), which is the key algorithm we are studying in this paper.

Algorithm 1 Training binary weight neural network with BinaryConnect algorithm and SGD.

```
Training data $\{x_i, z_i\}$, (Randomly) initialized model parameters $\theta^{(0)}$,learning rate $\eta$. $t$ in $[0 \dots T]$ Select a minibatch $\{x_b, z_b\}$ from training data **Quantization:** $w^{(t)} \leftarrow Quantize(\theta^{(t)})$ **Forward propagation:** compute $\{y_b\}$ using $\{x_b\}$ and $w^{(t)}$ **Backward propagation:** compute $\frac{\partial \text{loss}(y_b, z_b)}{\partial w^{(t)}}$ using $\{y_b\}, \{z_b\}$ and $w^{(t)}$ **accumulate gradients:** $\theta^{(t+1)} \leftarrow \theta^{(t)} + \eta \frac{\partial \text{loss}(y_b, z_b)}{\partial w^{(t)}}$.
```

C. Conditioned distribution of the outputs of each layer

First we recognize that as the model parameters are initialized randomly, there are “bad” initialization that will mess up our analysis. For example, all of $\theta_{1,ij}$ are initialized to 1 (or -1) while they are drawn from a uniform distribution. Fortunately, as the width d_1, d_2 grows to infinite, the probability of getting into these “bad” initialization goes to 0. We make this statement formal in the following part.

Definition 7. *“Parameter sequence”. Define the parameter sequence, indexed by d_1, d_2, as*

$$\Theta^{(d_1,d_2)} = \Big\{ \mathbf{W}_0^{(d_1,d_2)} \in \mathbb{R}^{d,d_1}, \boldsymbol{b}_0^{(d_1,d_2)} \in \mathbb{R}^{d_1}, \boldsymbol{\theta}_1^{(d_1,d_2)} \in \mathbb{R}^{d_1,d_2}, \boldsymbol{b}_1^{(d_1,d_2)} \in \mathbb{R}^{d_2}, \\ \mathbf{W}_2^{(d_1,d_2)} \in \mathbb{R}^{d_2}, b_2^{(d_1,d_2)} \in \mathbb{R} \Big\},$$

where $\mathbf{W}_0 = \{w_{0,ki}\}, \boldsymbol{b}_0 = \{b_{0,i}\}, \boldsymbol{\theta}_1 = \theta_{1,ij}, \boldsymbol{b}_1 = \{b_{1,j}\}, \mathbf{W}_2 = \{w_{1,j}\}$, the superscripts are omitted, such that for all $d_1 \le d_1', d_2 \le d_2'$, $\Theta^{(d_1,d_2)}, \Theta^{(d_1',d_2')}$ satisfy

$$\mathbf{W}_0^{(d_1,d_2)} = \mathbf{W}_0^{(d_1',d_2')}[:, 1:d_1], \quad \boldsymbol{b}_0^{(d_1,d_2)} = \boldsymbol{b}_0^{(d_1',d_2')}[1:d_1], \quad \boldsymbol{\theta}_1^{(d_1,d_2)} = \boldsymbol{\theta}_1^{(d_1',d_2')}[1:d_1, 1:d_2], \\ \boldsymbol{b}_1^{(d_1,d_2)} = \boldsymbol{b}_1^{(d_1',d_2')}[1:d_2], \quad \mathbf{W}_{2,j}^{(d_1,d_2)} = w_2^{(d_1',d_2')}[1:d_2], \quad b_2^{(d_1,d_2)} = b_2^{(d_1',d_2')},$$

$\forall k \in [d], i \le d_1, j \le d_2$.

Remark 8. *This definition states that for any two terms (sets of parameters) in the “parameter sequence”, the overlapping parameters are always equal.*

Definition 9. *“Good Initialization sequence”. For any finite d_2, we call the set of parameters sequence defined in Theorem 7 as a “Good Initialization” $\{\Theta^{(d_1)}\} \in \mathcal{G}$ if it satisfies:*

- $\forall k, k' \in [d], \lim_{d_1 \to \infty} \frac{1}{d_1} \sum_{i=1}^{d_1} w_{0,ki} w_{0,k'i} = \delta_{k,k'}$,
- $\forall k, k', k'' \in [d], \lim_{d_1 \to \infty} \frac{1}{d_1} \sum_{i=1}^{d_1} |w_{0,ki} w_{0,k'i} w_{0,k''i}| \le \sqrt{\frac{8}{\pi}}$,
- $\forall k, k' \in [d], \forall j \in [d_2], \lim_{d_1 \to \infty} \frac{1}{d_1} \sum_{i=1}^{d_1} w_{0,ki} w_{0,k'i} \theta_{1,ij}^2 = \text{Var}[\theta] \delta_{k,k'}$, *where*

$$\delta_{k,k'} = \begin{cases} 1 & k = k' \\ 0 & k \ne k'. \end{cases}$$

Here, we omit the superscript (d_1, d_2) again in the statement for the parameters w's. d is the input dimension. d_1, d_2 are defined in (2).

The following Proposition 10 guarantees the “Good initialization sequence” in Definition 9 holds true with probability 1. The proof can be found in subsection E.1.

Proposition 10. *Under the assumption that all the parameters are initialized as in 2, for any finite d_2, the probability that the sequence defined in Theorem 7 is a “Good Initialization sequence” is 1:*

$$Pr(\{\Theta^{(d_1,d_2)}, d_1 = 1, 2, \dots\} \in \mathcal{G}) = 1.$$

Lemma 11. *Given any fixed x, and any fixed "Good Initialization sequence" $\{\Theta_{d_1}\} \in \mathcal{G}$ denoted as Θ in short, for any fixed j, define the random sequence $y_{1,j}^{(d_1)} = f_{\Theta_{d_1}}(x)$. on the limit $d_1 \to \infty$, the distribution of $y_{1,j}^{(d_1)}$ converge to* Gaussian distribution *with mean $\nu_{1,j}$ and variance $\varsigma_{1,j}^2$ which can be computed by:*

$$y_{1,j}|\Theta \to \mathcal{N}(\nu_{1,j}, \varsigma_{1,j}^2), \quad \nu_{1,j} = \sqrt{\frac{c}{d_1}} \sum_{i=1}^{d_1} \theta_{1,ij} x_{1,i} + b_{1,j}, \quad \varsigma_{1,j}^2 = \frac{c}{d_1} \sum_{i=1}^{d_1} (1 - \theta_{1,ij}^2) x_{1,i}^2 \tag{9}$$

This lemma can be proved by Lyapunov central limit theorem and sum of expectation. See subsection E.2 for the details.

Lemma 12. *Assume that the input to a ReLU layer $y_{1,j}$ satisfy Gaussian distribution with mean $\nu_{1,j}$ and variance $\varsigma_{1,j}^2$*

$$y_{1,j} \sim \mathcal{N}(\nu_{1,j}, \varsigma_{1,j}^2).$$

Denote

$$g_j = \varphi\left(\frac{\nu_j}{\varsigma_j}\right), s_j = \Phi\left(\frac{\nu_j}{\varsigma_j}\right), \tag{10}$$

where $\varphi(x)$ denotes standard Gaussian function and $\Phi(x)$ denotes its integration:

$$\varphi(x) = \sqrt{\frac{1}{2\pi}} \exp\left(-\frac{1}{2}x^2\right), \quad \Phi(x) = \int_{-\infty}^{x} \varphi(y) dy.$$

Then the output $x_{2,j}$ has mean $\mu_{2,i}$ and variance $\sigma_{2,i}^2$, with

$$\begin{aligned} \mu_{2,j} &:= \mathbb{E}[x_{2,j}] = g_j \varsigma_{1,j} + s_j \nu_{1,j}, \\ \sigma_{2,j}^2 &:= \mathrm{Var}[x_{2,j}] = (\varsigma_{1,j}^2 + \nu_{1,j}^2) s_j + \nu_{1,j} \sigma_{1,j} g_{1,j} - \nu_{1,j}^2. \end{aligned} \tag{11}$$

The proof can be found in subsection E.3. From Theorem 11 we know that on the limit $d_1 \to \infty$, conditioned on Θ and $\boldsymbol{x}$, for any j, $y_{1,j}$ converge to Gaussian distribution. From continuous mapping theorem, the distribution of $x_{2,j}$ converge to that shown in Theorem 12 so its mean $\mu_{2,j}$ and variance $\sigma_{2,j}$ converge to that computed in Theorem 12.

Equations (9) and (11) provide a method to calculate the mean and variance of output conditioned on the input and real-valued model parameters and allow us to provide a closed-form equation of quasi neural network. We will simplify this equation in **??**.

D. Asymptotics during training

So far we have studied the distribution of output during initialization. To study the dynamic of binary weight neural network during training, one need to extend these results to any parameter during training $\Theta(t), t \in [0, T]$. Fortunately, motivated by (Jacot et al., 2018), we can prove that as $d_1, d_2 \to \infty$, the model parameters $\Theta(t)$ stays asymptotically close to initialization for any finite T, so-called "lazy training", so the above results apply to the entire training process.

Lemma 13. *For all T such that $\int_{t=0}^{T} \|\bar{y}(t) - z\|_{in} dt$ stays stochastically bounded, where $\|\cdot\|_{in}$ is defined in subsection B.1, as $d_2 \to \infty, d_1 \to \infty$, $\|\boldsymbol{w}_2(T) - \boldsymbol{w}_2(0)\|$, $\|\boldsymbol{b}_1(T) - \boldsymbol{b}_1(0)\|$, $\|\boldsymbol{\theta}_1(T) - \boldsymbol{\theta}_1(0)\|_F$ are all stochastically bounded, $\|\boldsymbol{\nu}_1(t) - \boldsymbol{\nu}_1(0)\|$ and $\int_{t=0}^{T} \|\frac{\partial \boldsymbol{\nu}_1(t)}{\partial t}\| dt$ is stochastically bounded for all x.*

The proof can be found in subsection E.9. Note that $\|\boldsymbol{w}_2\| = O(\sqrt{d_2})$, $\|\boldsymbol{\theta}_1\|_F = O(\sqrt{d_1 d_2})$, this results indicates that as $d_2 \to \infty$, the varying of the parameter is much smaller than the initialization, or so-called "lazy training". Making use of this result, we further get the follow result:

Lemma 14. *Under the condition of Theorem 13, Lyapunov's condition holds for all T so $y_{1,j}$ converges to Gaussian distribution conditioned on the model parameters $\Theta(T)$. Furthermore, $\varsigma_{1,j}(T) \to \varsigma_{1,t}(0)$, which equals $\tilde{\varsigma}_1$ almost surely.*

The proof can be found in subsection E.10. This result shows that the analysis in section 3 applies to the entire training process, and allows us to study the dynamics of binary weight neural network using quasi neural network.

E. Gaussian approximation in quantized neural network

E.1. Proof of Proposition 10

Proof. To prove the first statement in Theorem 9 holds a.s., observe that fixing k, k' and taking i as the variable, $w_{0,ki}w_{0,k'i}$ are independent from each other. Furthermore, $\mathbb{E}[w_{0,ki}w_{0,k'i}] = \delta_{k,k'}$ has identical mean for different i. In addition, since w is bounded, $\sum_{i=1}^{\infty} \mathrm{Var}[w_{0,ki}w_{0,k'i}]/i^2 \leq \sum_{i=1}^{\infty} C/i^2 < \infty$. By the strong law of large number (SLLN) Theorem 17, the first statement is proved. The third statement can be proved similarly, observing that $\mathbb{E}[w_{0,ki}w_{0,k'i}\theta_{1,ij}^2] = \delta_{k,k'}\mathrm{Var}[\theta]$ and that both w and θ are bounded (which guarantees $\sum_{i=1}^{\infty} \mathrm{Var}[w_{0,ki}w_{0,k'i}\theta_{1,ij}^2]/i^2 < \infty$).

To prove the second statement in Theorem 9 holds a.s., since geometric mean is no larger than cubic mean,

$$|w_{0,ki}||w_{0,k'i}||w_{0,k'i}| \leq \frac{1}{3}(|w_{0,ki}|^3 + |w_{0,k'i}|^3 + |w_{0,k'i}|^3),$$

Since $w_{0,ki} \sim \mathcal{N}(0,1)$, the expectation of the right hand side equals $\sqrt{\frac{8}{\pi}}$. We apply SLLN (Lemma 17) again to finish the proof. □

E.2. Proof of Theorem 11

We first compute the conditioned mean and variance $\nu_{1,j}$ and $\varsigma_{1,j}$. Notice that for any d_1, conditioned on any Θ, x_1 is deterministic,

$$\begin{aligned}
\nu_{1,j} &= \mathbb{E}_{w_1}\left[y_{1,j}|\Theta\right] \\
&= \sqrt{\frac{c}{d_1}}\sum_{i=1}^{d_1}\mathbb{E}_{w_1}[w_{1,ij}]x_{1,i} + \beta b_{1,j} \\
&= \sqrt{\frac{c}{d_1}}\sum_{i=1}^{d_1}\theta_{1,ij}x_{1,i} + \beta b_{1,j} \\
\varsigma_{1,j} &= \mathrm{Var}_{w_1}\left[y_{1,j}|\Theta\right] \\
&= \mathbb{E}_{w_1}\left[y_{1,j}^2|\Theta\right] - \mathbb{E}_{w_1}\left[y_{1,j}|\Theta\right]^2 \\
&= \frac{c}{d_1}\sum_{i=1}^{d_1}\sum_{i'=1}^{d_1}\mathbb{E}_{w_1}\left[w_{1,ij}w_{1,i'j}|\Theta\right]x_{1,i}x_{1,i'} + 2\beta b_{1,j}\sqrt{\frac{c}{d_1}}\sum_{i=1}^{d_1}\mathbb{E}\left[w_{1,ij}|\Theta\right]x_{1,i} \\
&\quad - \frac{c}{d_1}\sum_{i=1}^{d_1}\sum_{i'=1}^{d_1}\theta_{1,ij}\theta_{1,i'j}x_{1,i}x_{1,i'} - 2\beta b_{1,j}\sqrt{\frac{c}{d_1}}\sum_{i=1}^{d_1}\theta_{1,ij}x_{1,i} \\
&= \frac{c}{d_1}\sum_{i=1}^{d_1}\sum_{i'=1}^{d_1}(\mathbb{E}[w_{1,ij}w_{1,i'j}|\Theta] - \theta_{1,ij}\theta_{1,i'j})x_{1,i}^2 \\
&= \frac{c}{d_1}\sum_{i=1}^{d_1}(\mathbb{E}[w_{1,ij}^2|\Theta] - \theta_{1,ij}^2)x_{1,i}^2 \\
&= \frac{c}{d_1}\sum_{i=1}^{d_1}(1-\theta_{1,ij}^2)x_{1,i}^2.
\end{aligned}$$

The second line is because $\mathbb{E}_{w_1}[w_{1,ij}w_{1,i'j}|\Theta] = \mathbb{E}_{w_1}[w_{1,ij}|\Theta]\mathbb{E}[w_{1,i'j}|\Theta] = \theta_{1,ij}\theta_{1,i'j}$ when $i \neq i'$.

Next, we need to prove that for any "good initialization sequence" $\{\Theta_{d_1}\} \in \mathcal{G}$, $\{y_{1,i}^{(d_1)}\}$ converge to Gaussian distribution conditioned on $\Theta \in \mathcal{G}$ by verifying Lyapunov's condition. Note that for any $j \in [d_2]$,

$$y_{1,j} = \sqrt{\frac{c}{d_1}}\sum_{i=1}^{d_1}w_{1,ij}x_{1,i} + b_{1,j}$$

Define $X_i = w_{1,ij}x_{1,i}$. As mentioned above, its mean and variance (conditioned on Θ) is

$$\mathbb{E}_{w_1}[X_i|\Theta] = \theta_{1,ij}x_{1,i}, \quad \mathrm{Var}_{w_1}[X_i|\Theta] = \mathbb{E}_{w_1}[X_i^2|\Theta] - \mathbb{E}_{w_1}[X_i|\Theta]^2 = (1-\theta_{1,ij}^2)x_{1,i}^2$$

Since $\Theta \in \mathcal{G}, \forall j \in [d_2]$ for some finite d_2,

$$\begin{aligned}
\lim_{d_1\to\infty} \frac{1}{d_1}\sum_{i=1}^{d_1} \mathrm{Var}_{w_1}[X_i|\Theta] &= \lim_{d_1\to\infty} \frac{1}{d_1}\sum_{i=1}^{d_1}(1-\theta_{1,ij}^2)x_{1,i}^2 \\
&= \lim_{d_1\to\infty} \frac{1}{dd_1}\sum_{i=1}^{d_1}(1-\theta_{1,ij}^2)\Big(\sum_{k=1}^{d} w_{0,ki}x_k\Big)^2 \\
&= \lim_{d_1\to\infty} \frac{1}{dd_1}\sum_{i=1}^{d_1}\sum_{k,k'=1}^{d}(1-\theta_{1,ij}^2)w_{0,ki}w_{0,k'i}x_kx_{k'} \\
&= \lim_{d_1\to\infty} \sum_{k,k'=1}^{d}(1-\mathrm{Var}[\theta])\delta_{k,k'}x_kx_{k'} = 1-\mathrm{Var}[\theta]
\end{aligned} \tag{12}$$

The fourth equality comes from the definition of $\mathcal{G}$, and the fifth equality is because $\|x\|_2 = 1$. The third order absolute momentum can be bounded by

$$\begin{aligned}
&\lim_{d_1\to\infty} \frac{1}{d_1}\sum_{i=1}^{d_1} \mathbb{E}_{w_1}\left[|X_i - \mathbb{E}_{w_1}[X_i|\Theta]|^3\middle|\Theta\right] \\
&= \lim_{d_1\to\infty} \frac{1}{d_1}\sum_{i=1}^{d_1} \mathbb{E}_{w_1}\left[|(w_{1,ij}-\theta_{1,ij})x_{1,i}|^3\middle|\Theta\right] \\
&\le \lim_{d_1\to\infty} \frac{1}{d_1}\sum_{i=1}^{d_1} \mathbb{E}_{w_1}\left[8|x_{1,i}|^3\middle|\Theta\right] \\
&= \lim_{d_1\to\infty} \frac{8}{d_1}\sum_{i=1}^{d_1} \Big|\sum_{k=1}^{d} w_{0,ki}x_k\Big|^3 \\
&\le \lim_{d_1\to\infty} \frac{8}{d_1}\sum_{i=1}^{d_1} \left(\sum_{k=1}^{d} |w_{0,ki}x_k|\right)^3 \\
&= \lim_{d_1\to\infty} \frac{8}{d_1}\sum_{i=1}^{d_1} \sum_{k,k',k''=1}^{d} |w_{0,ki}w_{0,k'i}w_{0,k''i}||x_kx_{k'}x_{k''}| \\
&\le 8\sqrt{\frac{8}{\pi}}d^3
\end{aligned} \tag{13}$$

The last inequality comes from the definition of "Good Initialization": for all $\Theta \in \mathcal{G}$,

$$\lim_{d_1\to\infty} \frac{1}{d_1}\sum_{i=1}^{d_1} w_{0,ki}w_{0,k'i}w_{0,k'i} \le \sqrt{\frac{8}{\pi}},$$

and because $\|x\|_2 = 1, |x_k| \le 1$ for all $k \in [d]$. Note that using the strong law of large number, one can prove that the third order absolute momentum converges almost surely to a constant that doesn't depend on d. On the other hand, we are proving a upper bound for all $\Theta \in \mathcal{G}$ which is stronger than almost surely converge.

$$\begin{aligned}
&\lim_{d_1\to\infty}\frac{\sum_{i=1}^{d_1}\mathbb{E}_{w_1}[|X_i-\mathbb{E}_{w_1}[X_i|\Theta]|^3|\Theta]}{(\sum_{i=1}^{d_1}\mathrm{Var}_{w_1}[X_i|\Theta])^{3/2}}\\
&=\lim_{d_1\to\infty}\frac{\frac{1}{d_1}\sum_{i=1}^{d_1}\mathbb{E}_{w_1}[|X_i-\mathbb{E}_{w_1}[X_i|\Theta]|^3|\Theta]}{d_1^{1/2}(\frac{1}{d_1}\sum_{i=1}^{d_1}\mathrm{Var}_{w_1}[X_i|\Theta])^{3/2}}\\
&=\frac{\lim_{d_1\to\infty}\frac{1}{d_1}\sum_{i=1}^{d_1}\mathbb{E}_{w_1}[|X_i-\mathbb{E}_{w_1}[X_i|\Theta]|^3|\Theta]}{\lim_{d_1\to\infty}d_1^{1/2}(\frac{1}{d_1}\sum_{i=1}^{d_1}\mathrm{Var}_{w_1}[X_i|\Theta])^{3/2}}\\
&\le\frac{8\sqrt{\frac{8}{\pi}}d^3}{\lim_{d_1\to\infty}d_1^{1/2}(1-\mathrm{Var}[\theta])}\\
&=0
\end{aligned}$$

This proves that Lyapunov's condition for all "Good Initialization", so conditioned on $\Theta\in\mathcal{G}$, $y_{1,j}$ converges to Gaussian distribution.

E.3. Proof of Theorem 12

To compute $\mathbb{E}_{w_1}[x_{2,j}|\Theta]$ and $\mathrm{Var}_{w_1}[x_{2,j}|\Theta]$, we first compute $\mathbb{E}_{w_1}[\psi(y_{1,j})|\Theta]$ and $\mathbb{E}_{w_1}[\psi(y_{1,j})^2|\Theta]$. Recall $\psi(x)=x\mathbf{1}(x\ge 0)$,

$$\begin{aligned}
\mathbb{E}_{w_1}[\psi(y_{1,j})|\Theta]&=\int_0^\infty x\frac{1}{\sqrt{2\pi}\varsigma_{1,j}}\exp\left(-\frac{1}{2}\frac{(x-\nu_{1,j})^2}{\varsigma_{1,j}^2}\right)dx\\
&=\int_{-\frac{\nu_{1,j}}{\varsigma_{1,j}}}^\infty(\varsigma_{1,j}y+\nu_{1,j})\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}y^2\right)dy\\
&=\varsigma_{1,j}\int_{-\frac{\nu_{1,j}}{\varsigma_{1,j}}}^\infty\frac{1}{\sqrt{2\pi}}y\exp\left(-\frac{1}{2}y^2\right)dy+\mu_{\ell,i}\int_{-\frac{\nu_{1,j}}{\varsigma_{1,j}}}^\infty\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}y^2\right)dy,
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}_{w_1}[\psi(y_{1,j})^2|\Theta]&=\int_0^\infty x^2\frac{1}{\sqrt{2\pi}\varsigma_{1,j}}\exp\left(-\frac{1}{2}\frac{(x-\nu_{1,j})^2}{\varsigma_{1,j}^2}\right)dx\\
&=\int_{-\frac{\nu_{1,j}}{\varsigma_{1,j}}}^\infty(\varsigma_{1,j}y+\nu_{1,j})^2\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}y^2\right)dy\\
&=\varsigma_{1,j}^2\int_{-\frac{\nu_{1,j}}{\varsigma_{1,j}}}^\infty y^2\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}\hat{y}^2\right)dy+2\varsigma_{1,j}\nu_{1,j}\int_{-\frac{\nu_{1,j}}{\varsigma_{1,j}}}^\infty y\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}y^2\right)dy\\
&\quad+\nu_{1,j}^2\int_{-\frac{\nu_{1,j}}{\varsigma_{1,j}}}^\infty\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}y^2\right)dy.
\end{aligned}$$

We only need to compute

$$\int_{-\frac{\nu_{1,j}}{\varsigma_{1,j}}}^\infty\frac{1}{\sqrt{2\pi}}y^\alpha\exp\left(-\frac{1}{2}y^2\right)dy.$$

For $\alpha=0,1,2$. When $\alpha=0$, this is integration to Gaussian function, and it is known that there's no analytically function to express that. For sack of simplicity, define it as $s_{1,j}$

$$s_{1,j}=\int_{\frac{\nu_{1,j}}{\varsigma_{1,j}}}^\infty\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}y^2\right)dy:=\Phi(\frac{\nu_{1,j}}{\varsigma_{1,j}}).$$

When $\alpha=1$, this integration can be simply solved by change of the variable and we denote it as $g_{1,j}$:

$$g_{1,j}=\int_{-\frac{\nu_{1,j}}{\varsigma_{1,j}}}^\infty y\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}y^2\right)dy=\sqrt{\frac{1}{2\pi}}\exp\left(-\frac{1}{2}\left(\frac{\nu_{1,j}}{\varsigma_{1,j}}\right)^2\right).$$

When $\alpha = 2$, we can do integration by parts and express it using $s_{1,j}$ and $g_{1,j}$:

$$\begin{aligned}
&\int_{-\frac{\nu_{1,j}}{\varsigma_{1,j}}}^{\infty} y^2 \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}y^2\right) dy \\
&= -\int_{-\frac{\nu_{1,j}}{\varsigma_{1,j}}}^{\infty} y \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}y^2\right) d\frac{1}{2}y^2 \\
&= \int_{-\frac{\nu_{1,j}}{\varsigma_{1,j}}}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}y^2\right) dy - \frac{\nu_{1,j}}{\varsigma_{1,j}} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{\nu_{1,j}}{\varsigma_{1,j}}\right)^2\right) \\
&= s_{1,j} - \frac{\nu_{1,j}}{\varsigma_{1,j}} g_{1,j}.
\end{aligned}$$

Using the definition of mean and variance,

$$\mu_{2,j} = \mathbb{E}_{w_1}[\psi(y_{1,j})|\Theta], \sigma_{2,i}^2 = \mathbb{E}_{w_1}[\psi(y_{1,j})^2|\Theta] - \mathbb{E}_{w_1}[\psi(y_{1,j})|\Theta]^2,$$

we can come to the result.

E.4. Proof of Theorem 1

In this part, we take $\Theta = \{w_0, \theta_1, w_2, b_0, b_1, b_2\}$ as the random variables and conditioned mean and variance derived above $\mu_1, \sigma_1, \nu_1, \varsigma_1$ as functions to Θ. From Eq. (9), as $d_1 \to \infty$, v_1 tend to iid Gaussian processes, and there covariance converges almost surely to its expectation. We then focus on computing the expectation of covariance. For any $j \neq j'$, we take the expectation over random initialization of Θ:

$$\begin{aligned}
&\mathbb{E}_\Theta[\nu_{1,j}\nu_{1,j'}] \\
&= \mathbb{E}_\Theta\left[\frac{c}{d_1}\sum_{i=1}^{d_1}\sum_{i'=1}^{d_1}\theta_{1,ij}\theta_{1,i'j'}x_{1,i}x_{1,i'} + \beta\sqrt{\frac{c}{d_1}}\sum_{i=1}^{d_1}(\theta_{1,ij}x_{1,i}b_j + \theta_{1,ij'}x_{1,i}b_{j'}) + \beta^2 b_j b_{j'}\right] \\
&= \frac{c}{d_1}\sum_{i=1}^{d_1}\sum_{i'=1}^{d_1}\mathbb{E}_\Theta[\theta_{1,ij}]\mathbb{E}_\Theta[\theta_{1,i'j'}]\mathbb{E}_\Theta[x_{1,i}x_{1,i'}] + \beta^2\mathbb{E}_\Theta[b_j b_{j'}] \\
&\quad + \sqrt{\frac{c}{d_1}}\beta\sum_{i=1}^{d_1}(\mathbb{E}_\Theta[\theta_{1,ij}]\mathbb{E}_\Theta[x_{1,i}]\mathbb{E}_\Theta[b_j] + \mathbb{E}_\Theta[\theta_{1,ij'}]\mathbb{E}_\Theta[x_{1,i}]\mathbb{E}_\Theta[b_{j'}]) \\
&= 0
\end{aligned} \tag{14}$$

which indicates that they are independent.

Computation of $\varsigma_{1,j}$ was already finished implicitly in Section E.2. We write it explicitly here. From (9), on the limit $d_1 \to \infty$,

$$\begin{aligned}
\varsigma_{1,j}^2 &= \frac{c}{d_1}\sum_{i=1}^{d_1}(1-\theta_{1,ij}^2)x_{1,i}^2 \\
&= \frac{c}{dd_1}\sum_{i=1}^{d_1}(1-\theta_{1,ij}^2)\sum_{k,k'=1}^{d} w_{0,ki}w_{0,k'i}x_k x_{k'} \\
&= \frac{c}{d}\sum_{k,k'=1}^{d} x_k x_{k'}\frac{1}{d_1}\sum_{i=1}^{d_1} w_{0,ki}w_{0,k'i}(1-\theta_{1,ij}^2) \\
&= \frac{c}{d}\sum_{k,k'=1}^{d} x_k x_{k'}\delta_{k,k'}(1-\mathrm{Var}[\theta]) \\
&= \frac{c}{d}(1-\mathrm{Var}[\theta])\|x\|_2^2 \\
&= \frac{c}{d}(1-\mathrm{Var}[\theta])
\end{aligned}$$

The fourth line comes from the definition of $\mathcal{G}$.

E.5. Derivative of activation function in quasi neural network

Let

$$\tilde{\psi}(x) = \tilde{\varsigma}_1 \phi\left(\frac{x}{\tilde{\varsigma}}\right) + x\Phi\left(\frac{x}{\tilde{\varsigma}}\right),$$

Its derivative is

$$\begin{aligned}\tilde{\psi}'(x) &= \varphi'\left(\frac{x}{\tilde{\varsigma}}\right) + \Phi\left(\frac{x}{\tilde{\varsigma}}\right) + \frac{x}{\tilde{\varsigma}}\Phi'\left(\frac{x}{\tilde{\varsigma}}\right)\\ &= -\frac{x}{\tilde{\varsigma}}\varphi\left(\frac{x}{\tilde{\varsigma}}\right) + \Phi\left(\frac{x}{\tilde{\varsigma}}\right) + \frac{x}{\tilde{\varsigma}}\varphi\left(\frac{x}{\tilde{\varsigma}}\right)\\ &= \Phi\left(\frac{x}{\tilde{\varsigma}}\right)\end{aligned}$$

The second line is because

$$\begin{aligned}\Phi'(x) &= \varphi(x),\\ \varphi'(x) &= \frac{d}{dx}\sqrt{\frac{1}{2\pi}}\exp\left(-\frac{1}{2}x^2\right)\\ &= -x\sqrt{\frac{1}{2\pi}}\exp\left(-\frac{1}{2}x^2\right)\\ &= -x\varphi(x).\end{aligned}$$

E.6. Proof of Theorem 2

To make the proof more general, we make $\varsigma_{1,j}$ a parameter of the activation function in quasi neural network as $\tilde{\psi}(\cdot;\varsigma_{1,j})$. To get the derivative with respect to $\theta_{1,ij}$, we first get the derivative with respect to $\nu_{1,j}$.

$$\frac{\partial \bar{y}}{\partial \nu_{1,j}} = \frac{\partial \bar{y}}{\partial \mu_{2,j}}\frac{\partial \mu_{2,j}}{\partial \nu_{1,j}} = \sqrt{\frac{1}{d_2}} w_{2,j}\tilde{\psi}'(\nu_{1,j};\varsigma_{1,j})$$

then apply chain rule:

$$\frac{\partial \bar{y}}{\partial w_{2,j}} = \sqrt{\frac{c}{d_2}}\mu_{2,j}, \tag{15}$$

$$\frac{\partial \bar{y}}{\partial b_{1,j}} = \frac{\partial \bar{y}}{\partial \nu_{1,j}}\frac{\partial \nu_{1,j}}{\partial b_{1,j}} = \sqrt{\frac{c}{d_2}}\beta w_{2,j}\tilde{\psi}'(\nu_{1,j};\varsigma_{1,j}), \tag{16}$$

$$\frac{\partial \bar{y}}{\partial \theta_{1,ij}} = \frac{\partial \bar{y}}{\partial \nu_{1,j}}\frac{\partial \nu_{1,j}}{\partial \theta_{1,ij}} = \sqrt{\frac{c}{d_1 d_2}} w_{2,j} x_{1,i}\tilde{\psi}'(\nu_{1,j};\varsigma_{1,j}). \tag{17}$$

On the other hand, let's first write down the gradient with respect to weights w_{ij} in quantized neural network and take their expectation conditioned on Θ:

$$\mathbb{E}_{w_1}\left[\frac{\partial y}{\partial w_{2,j}}\middle|\Theta^{(d_1,d_2)}\right] = \sqrt{\frac{c}{d_2}}\mathbb{E}_{w_1}\left[x_{2,j}\middle|\Theta^{(d_1,d_2)}\right], \tag{18}$$

$$\mathbb{E}_{w_1}\left[\frac{\partial y}{\partial b_{2,j}}\middle|\Theta^{(d_1,d_2)}\right] = \mathbb{E}_{w_1}\left[\frac{\partial \bar{y}}{\partial y_{1,j}}\frac{\partial y_{1,j}}{\partial b_{2,j}}\middle|\Theta\right] = \sqrt{\frac{c}{d_2}}\beta w_{2,j}\mathbb{E}_{w_1}\left[\psi'(y_{1,j})\middle|\Theta^{(d_1,d_2)}\right], \tag{19}$$

$$\mathbb{E}_{w_1}\left[\frac{\partial y}{\partial w_{1,ij}}\middle|\Theta^{(d_1,d_2)}\right] = \mathbb{E}_{w_1}\left[\frac{\partial y}{\partial y_{1,j}}\frac{\partial y_{1,j}}{\partial w_{1,ij}}\middle|\Theta^{(d_1,d_2)}\right] = \sqrt{\frac{c}{d_1 d_2}} w_{2,j} x_{1,i}\mathbb{E}_{w_1}\left[\psi'(y_{1,j})\middle|\Theta^{(d_1,d_2)}\right], \tag{20}$$

By definition, $\mu_{2,j} = \lim_{d_1\to\infty}\mathbb{E}_{w_1}[x_{2,j}|\Theta^{(d_1,d_2)}]$. On the other hand, from (3), one can tell using continuous mapping theorem that

$$\tilde{\psi}'(\nu_{1,j};\varsigma_{1,j}) = \Phi\left(\frac{\nu_{1,j}}{\varsigma_{1,j}}\right) = \lim_{d_1\to\infty} P[y_{1,j}\geq 0] = \lim_{d_1\to\infty}\mathbb{E}_{w_1}\left[\psi'(y_{1,j})\middle|\Theta^{(d_1,d_2)}\right],$$

Taking them into (15)-(20) finishes the proof.

E.7. Proof of Theorem 3

Observe that conditioned on Θ, $y_{1,j}$ depends only on $\{w_{1,ij}, i \in [d_1]\}$, and that $\{w_{1,ij}, i \in [d_1]\} \cap \{w_{1,ij'}, i \in [d_1]\} = \emptyset$ for $j \neq j'$. Because of that, $y_{1,j}$ are independent of each other. Similarly, $x_{2,j}$ are independent of each other conditioned on Θ. For MSE loss,

$$\text{loss}(y) = \frac{1}{2}(y-z)^2, \frac{dl(y)}{dy} = y - z,$$

According to the chain rule

$$\frac{\partial \text{loss}(\bar{y})}{\partial \theta} = \frac{\partial \text{loss}(\bar{y})}{\partial \bar{y}} \frac{\partial \bar{y}}{\partial \theta} = (\bar{y} - z)\frac{\partial \bar{y}}{\partial \theta}, \tag{21}$$

for any $\theta \in \{\theta_{1,ij}, b_{1,j}, w_{2,j}\}$, which leads to

$$\frac{\partial \text{loss}(\bar{y})}{\partial w_{2,j}} = \sqrt{\frac{c}{d_2}}(\bar{y} - z)\mu_{2,j}, \tag{22}$$

$$\frac{\partial \text{loss}(\bar{y})}{\partial b_{1,j}} = \frac{\partial \bar{y}}{\partial \nu_{1,j}} \frac{\partial \nu_{1,j}}{\partial b_{1,j}} = \sqrt{\frac{c}{d_2}}\beta w_{2,j}(\bar{y} - z)\tilde{\psi}'(\nu_{1,j}; \varsigma_{1,j}), \tag{23}$$

$$\frac{\partial \text{loss}(\bar{y})}{\partial \theta_{1,ij}} = \frac{\partial \bar{y}}{\partial \nu_{1,j}} \frac{\partial \nu_{1,j}}{\partial \theta_{1,ij}} = \sqrt{\frac{c}{d_1 d_2}} w_{2,j} x_{1,i}(\bar{y} - z)\tilde{\psi}'(\nu_{1,j}; \varsigma_{1,j}). \tag{24}$$

On the other hand, in the original binary weight neural network, according to the chain rule,

$$\mathbb{E}_{w_1}\left[\frac{\partial \text{loss}(y)}{\partial w_{2,j}}\middle|\Theta^{(d_1,d_2)}\right] = \sqrt{\frac{c}{d_2}}\mathbb{E}_{w_1}\left[(y-z)x_{2,j}\middle|\Theta^{(d_1,d_2)}\right], \tag{25}$$

$$\mathbb{E}_{w_1}\left[\frac{\partial \text{loss}(y)}{\partial b_{1,j}}\middle|\Theta^{(d_1,d_2)}\right] = \sqrt{\frac{c}{d_2}}\beta w_{2,j}\mathbb{E}_{w_1}\left[(y-z)\psi'(y_{1,j})\middle|\Theta^{(d_1,d_2)}\right], \tag{26}$$

$$\mathbb{E}_{w_1}\left[\frac{\partial \text{loss}(y)}{\partial w_{1,ij}}\middle|\Theta^{(d_1,d_2)}\right] = \sqrt{\frac{c}{d_1 d_2}} w_{2,j} x_{1,i}\mathbb{E}_{w_1}\left[(y-z)\psi'(y_{1,j})\middle|\Theta^{(d_1,d_2)}\right], \tag{27}$$

Note that y is not independent form $x_{2,j}$ or $\psi'(y_{1,j})$, which is the main challenge of the proof. To deal with this problem, we bound the difference between (22)-(24) and (25)-(27), which requires bounding their covariance.

$$\begin{aligned}
\mathbb{E}_{w_1}\left[x_{2,j}y|\Theta^{(d_1,d_2)}\right] &= \sqrt{\frac{c}{d_2}}\mathbb{E}_{w_1}\left[x_{2,j}\sum_{j=1}^{d_2} w_{2,j}x_{2,j}\middle|\Theta^{(d_1,d_2)}\right] \\
&= \sqrt{\frac{c}{d_2}}\left(\mathbb{E}_{w_1}\left[x_{2,j}^2 w_{2,j}\middle|\Theta^{(d_1,d_2)}\right] + \sum_{j'\neq j}\mathbb{E}_{w_1}\left[x_{2,j}x_{2,j'}w_{2,j'}\middle|\Theta^{(d_1,d_2)}\right]\right) \\
\lim_{d_1\to\infty}\mathbb{E}_{w_1}\left[x_{2,j}y|\Theta^{(d_1,d_2)}\right] &= \sqrt{\frac{c}{d_2}}\left((\mu_{2,j}^2 + \sigma_{2,j}^2)w_{2,j} + \sum_{j'\neq j}\mu_{2,j}\mu_{2,j'}w_{2,j'}\right) \\
&= \sqrt{\frac{c}{d_2}}\left(\sigma_{2,j}^2 w_{2,j} + \sum_{j'=1}^{d_2}\mu_{2,j}\mu_{2,j'}w_{2,j'}\right)
\end{aligned} \tag{28}$$

Notice that by definition

$$\sqrt{\frac{c}{d_2}}\sum_{j'=1}^{d_2}\mu_{2,j}\mu_{2,j'}w_{2,j'} = \mathbb{E}_{w_1}\left[x_{2,j}\middle|\Theta^{(d_1,d_2)}\right]\mathbb{E}_{w_1}\left[y\middle|\Theta^{(d_1,d_2)}\right]$$

The second term equals $\mathbb{E}_{w_1}[x_{2,j}|\Theta]\mathbb{E}_{w_1}[y|\Theta]$ and the first term converges to 0 when $d_2 \to \infty$. Taking it into (22) and (25) finishes the proof of the first equation.

Similarly,

$$\begin{aligned}
&\mathbb{E}_{w_1}\left[\psi'(y_{1,j})y\middle|\Theta^{(d_1,d_2)}\right]\\
&=\mathbb{E}_{w_1}\left[\sqrt{\frac{c}{d_2}}\psi'(y_{1,j})\sum_{j=1}^{d_2}w_{2,j}\psi(y_{1,j})\middle|\Theta^{(d_1,d_2)}\right]\\
&=\sqrt{\frac{c}{d_2}}\Bigg(\mathbb{E}_{w_1}\left[\psi(y_{1,j})\psi'(y_{1,j})w_{2,j}\middle|\Theta^{(d_1,d_2)}\right]\\
&\qquad+\sum_{j'\neq j}\mathbb{E}_{w_1}\left[\psi(y_{1,j'})\psi'(y_{1,j})w_{2,j'}\middle|\Theta^{(d_1,d_2)}\right]\Bigg)\\
&\lim_{d_1\to\infty}E_{w_1}\left[\psi'(y_{1,j})y\middle|\Theta^{(d_1,d_2)}\right]\\
&=\sqrt{\frac{c}{d_2}}\left(\mathbb{E}_{w_1}\left[\psi(y_{1,j})w_{2,j}\middle|\Theta^{(d_1,d_2)}\right]w_{2,j}+\sum_{j'\neq j}\mathbb{E}_{w_1}[\psi'(y_{1,j})|\Theta^{(d_1,d_2)}]\mu_{2,j'}w_{2,j'}\right),\\
&=\sqrt{\frac{c}{d_2}}\left((1-\mathbb{E}_{w_1}[\psi'(y_{1,j})|\Theta])\mu_{2,j}w_{2,j}+\sum_{j'=1}^{d_2}\mathbb{E}_{w_1}[\psi'(y_{1,j})|\Theta]\mu_{2,j'}w_{2,j'}\right)
\end{aligned} \tag{29}$$

Notice that by definition

$$\sqrt{\frac{c}{d_2}}\sum_{j'=1}^{d_2}\mu_{2,j'}w_{2,j'}=\lim_{d_1\to\infty}\mathbb{E}\left[y\middle|\Theta^{(d_1,d_2)}\right]$$

and the first term converges to 0 when $d_2\to\infty$. Taking it into (23)(24)(26)(27) finishes the proof.

E.8. Quantizing the second layer

Assume that the second layer is quantized in the same way as the first layer as in Equation 8. Then Theorem 11 can be modified as :

$$\bar{y}:=\mathbb{E}[y|\Theta]=\frac{1}{\sqrt{d_2}}\sum_{j=1}^{d_2}\mathbb{E}[w_{2,j}|\theta_{2,j}]\mu_{2,j}+\beta b_2=\sum_{j=1}^{d_2}\theta_{2,j}\mu_{2,j}+\beta b_2$$

which is the same form as the second layer in the quasi neural network Equation 2 except that replacing $w_{2,j}$ with $\theta_{2,j}$. As for the gradients,

$$\mathbb{E}_{w_1,w_2}\left[\frac{\partial y}{\partial w_{2,j}}\middle|\Theta^{(d_1,d_2)}\right]=\sqrt{\frac{c}{d_2}}\mathbb{E}_{w_1}\left[x_{2,j}\middle|\Theta^{(d_1,d_2)}\right]=\sqrt{\frac{c}{d_2}}\mu_{2,j},$$
$$\mathbb{E}_{w_1,w_2}\left[\frac{\partial y}{\partial x_{2,j}}\middle|\Theta^{(d_1,d_2)}\right]=\sqrt{\frac{c}{d_2}}\mathbb{E}_{w_2}\left[w_{2,j}\middle|\Theta^{(d_1,d_2)}\right]=\sqrt{\frac{c}{d_2}}\theta_{2,j},$$

By replacing $\theta_{2,j}$ with $w_{2,j}$, the above results are the same as leaving the second layer not quantized.

E.9. Proof of Theorem 13

In this part, we denote $\dot{a}:=\frac{\partial a}{\partial t}$ for $a\in\{w_\ell,\theta_\ell,b_\ell\}$, and express each time-depent variable as a function of time t. We define an inner product under the distribution of training dataset

$$\langle\boldsymbol{a},\boldsymbol{b}\rangle_{in}=\mathbb{E}_{in}[a(x)b(x)],$$

and the corresponding norm

$$\|\boldsymbol{a}\|_{in}=\sqrt{\langle\boldsymbol{a},\boldsymbol{a}\rangle_{in}}=\sqrt{\mathbb{E}_{in}[a(x)^2]}.$$

If $\boldsymbol{a}(x)$ is a vector, $\|\boldsymbol{a}\|_{in}:=\sqrt{\mathbb{E}_{in}[\|\boldsymbol{a}(x)\|^2]}$. Note this inner product and norm define a Hilbert space (not to be confused with the RKHS induced by a kernel), so by Cauchy-Schwarz inequality,

$$|\langle\boldsymbol{a},\boldsymbol{b}\rangle_{in}|\leq\|\boldsymbol{a}\|_{in}\|\boldsymbol{b}\|_{in},\forall\boldsymbol{a},\boldsymbol{b}.$$

As is shown in **??**, on the limit $d_1, \to \infty$, the dynamics of training this neural network using gradient descent can be written as:

$$\begin{aligned}
\dot{w}_{2,j}(t) &= \sqrt{\frac{c}{d_2}}\mathbb{E}_{in}[(\bar{y}(t)-z)\mu_{2,j}(t)]\\
\dot{b}_{1,j}(t) &= \sqrt{\frac{c}{d_2}}\mathbb{E}_{in}[\beta w_{2,j}(t)(\bar{y}(t)-z)\tilde{\psi}'(\nu_{1,j}(t),\varsigma_{1,j}(t))],\\
\dot{\theta}_{1,ij}(t) &= \sqrt{\frac{c}{d_1 d_2}}\mathbb{E}_{in}[w_{2,j}(t)x_{1,i}(\bar{y}(t)-z)\tilde{\psi}'(\nu_{1,j}(t);\varsigma_{1,j}(t))]
\end{aligned}$$

where dot denotes the derivative with respect to t. Note the activation function $\tilde{\psi}(\cdot;\varsigma_{1,j}(t))$ depends on $\varsigma_{1,j}$, which makes it time dependent. One can further write down the dynamics of $\nu_{1,j}(t)$ as

$$\dot{\nu}_{1,j}(t) = \sqrt{\frac{1}{d_1}}\sum_{i=1}^{d_1}\dot{\theta}_{1,ij}(t)x_{1,i}(t) + \dot{b}_{1,j}(t)$$

Rewrite these two differential equations in matrix form:

$$\begin{aligned}
\dot{\boldsymbol{w}}_2(t) &= \sqrt{\frac{c}{d_2}}\mathbb{E}_{in}[(\bar{y}(t)-z)\boldsymbol{\mu}_2(t)]\\
\dot{\boldsymbol{b}}_1(t) &= \beta\sqrt{\frac{c}{d_2}}\mathbb{E}_{in}[(\bar{y}(t)-z)(\tilde{\psi}'(\boldsymbol{\nu}_1(t))\circ \boldsymbol{w}_2(t))],\\
\dot{\boldsymbol{\theta}}_1(t) &= \sqrt{\frac{c}{d_1 d_2}}\mathbb{E}_{in}\big[(\bar{y}(t)-z)\boldsymbol{x}_1\otimes\big(\tilde{\psi}'(\boldsymbol{\nu}_1(t))\circ \boldsymbol{w}_2(t)\big)\big],\\
\dot{\boldsymbol{\nu}}_1(t) &= \sqrt{\frac{1}{d_1}}\dot{\boldsymbol{\theta}}_1\boldsymbol{x}_1 + \dot{\boldsymbol{b}}_1
\end{aligned}$$

where $\circ$ denotes elementwise product and $\otimes$ denotes outer product. Here we slightly abuse the notation $\tilde{\psi}(\cdot)$, which represents elementwise operation when applied to a vector. Their norm are bounded by

$$\begin{aligned}
\frac{\partial}{\partial t}\|\boldsymbol{w}_2(t)-\boldsymbol{w}_2(0)\| &\le \sqrt{\frac{c}{d_2}}\mathbb{E}_{in}[(\bar{y}(t)-z)\|\boldsymbol{\mu}_2(t)\|] = \sqrt{\frac{c}{d_2}}\langle \bar{y}(t)-z, \boldsymbol{\mu}_2(t)\rangle_{in}\\
&\le \sqrt{\frac{c}{d_2}}\|\bar{y}(t)-z\|_{in}\|\boldsymbol{\mu}_2(t)\|_{in} \le \sqrt{\frac{c}{d_2}}\|\bar{y}(t)-z\|_{in}\|\boldsymbol{\nu}_1(t)\|_{in}
\end{aligned} \tag{30}$$

$$\begin{aligned}
\frac{\partial}{\partial t}\|\boldsymbol{b}_1(t)-\boldsymbol{b}_1(0)\| &\le \beta\sqrt{\frac{c}{d_2}}\mathbb{E}_{in}[(\bar{y}(t)-z)\|\tilde{\psi}'(\boldsymbol{\nu}_1(t))\circ \boldsymbol{w}_2(t)\|]\\
&\le \beta\sqrt{\frac{c}{d_2}}\mathbb{E}_{in}[(\bar{y}(t)-z)\|\boldsymbol{w}_2(t)\|] = \beta\sqrt{\frac{c}{d_2}}\|\bar{y}(t)-z\|_{in}\|\boldsymbol{w}_2(t)\|,
\end{aligned} \tag{31}$$

$$\begin{aligned}
\frac{\partial}{\partial t}\|\boldsymbol{\theta}_1(t)-\boldsymbol{\theta}_1(0)\|_F &\le \sqrt{\frac{c}{d_1 d_2}}\mathbb{E}_{in}\big[(\bar{y}(t)-z)\|\boldsymbol{x}_1\otimes\big(\tilde{\psi}'(\boldsymbol{\nu}_1(t))\circ \boldsymbol{w}_2(t)\big)\|_F\big]\\
&\le \sqrt{\frac{c}{d_1 d_2}}\mathbb{E}_{in}\big[(\bar{y}(t)-z)\|\boldsymbol{x}_1\|\|\boldsymbol{w}_2(t)\|\big]\\
&\le \sqrt{\frac{c}{d_1 d_2}}\|\bar{y}(t)-z\|_{in}\|\boldsymbol{x}_1\|_{in}\|\boldsymbol{w}_2(t)\|\\
&= \sqrt{\frac{c}{d_2}}\|\bar{y}(t)-z\|_{in}\|\boldsymbol{w}_2(t)\|,
\end{aligned} \tag{32}$$

$$\begin{aligned}
\forall \boldsymbol{x}_1, \quad \frac{\partial}{\partial t}\|\boldsymbol{\nu}_1(t)-\boldsymbol{\nu}_1(0)\| &\leq \int_{t=0}^{T}\Big\|\frac{\partial \boldsymbol{\nu}_1(t)}{\partial t}\Big\| dt \\
&\leq \sqrt{\frac{1}{d_1}}\frac{\partial}{\partial t}\|\boldsymbol{\theta}_1(t)-\boldsymbol{\theta}_1(0)\|_{op}\|\boldsymbol{x}_1\| + \frac{\partial}{\partial t}\|\boldsymbol{b}_1(t)-\boldsymbol{b}_1(0)\| \\
&\leq \sqrt{\frac{c}{d_1^2 d_2}}\|\bar{y}(t)-z\|_{in}\|\boldsymbol{w}_2(t)\|\|\boldsymbol{x}_1\|_{in}\|\boldsymbol{x}_1\| \\
&\quad + \beta\sqrt{\frac{c}{d_2}}\|\bar{y}(t)-z\|_{in}\|\boldsymbol{w}_2(t)\| \\
&= (1+\beta)\sqrt{\frac{c}{d_2}}\|\bar{y}(t)-z\|_{in}\|\boldsymbol{w}_2(t)\|, \\
\frac{\partial}{\partial t}\|\boldsymbol{\nu}_1(t)-\boldsymbol{\nu}_1(0)\|_{in} &\leq (1+\beta)\sqrt{\frac{c}{d_2}}\|\bar{y}(t)-z\|_{in}\|\boldsymbol{w}_2(t)\|.
\end{aligned} \tag{33}$$

Here we make use of the fact that $\tilde{\phi}'(x) \leq 1$, $\tilde{\phi}(x) \leq x$ regardless of the value of $\varsigma_{1,j}(t)$, that $\lim_{d_1\to\infty}\|\boldsymbol{x}_1\|_{in}/\sqrt{d_1} = 1$ as long as $\Theta \in \mathcal{G}$, and that w_0 is not updated during training. In the last equation, we make use of $\dot{\boldsymbol{\theta}}_1 = \frac{\partial}{\partial t}(\boldsymbol{\theta}_1(t) - \boldsymbol{\theta}_1(0))$, $\dot{\boldsymbol{b}}_1 = \frac{\partial}{\partial t}(\boldsymbol{b}_1(t) - \boldsymbol{b}_1(0))$.

Define $A(t) = \sqrt{\frac{c}{d_2}}\sqrt{1+\beta}(\|\boldsymbol{w}_2(t)-\boldsymbol{w}_2(0)\| + \|\boldsymbol{w}_2(0)\|) + \sqrt{\frac{c}{d_2}}(\|\boldsymbol{\nu}_1(t)-\boldsymbol{\nu}_1(0)\|_{in} + \|\boldsymbol{\nu}_1(0)\|_{in})$, then

$$\begin{aligned}
\dot{A}(t) &\leq \sqrt{1+\beta}\sqrt{\frac{c}{d_2}}\|\bar{y}(t)-z\|_{in}\|\boldsymbol{\nu}_1(t)\|_{in} + (1+\beta)\sqrt{\frac{c}{d_2}}\|\bar{y}(t)-z\|_{in}\|\boldsymbol{w}_2(t)\| \\
&\leq \sqrt{1+\beta}A(t)
\end{aligned}$$

Observe that $A(0)$ is stochastically bounded. Using Grönwall's Lemma, for any finite T:

$$A(T) \leq A(0)\exp\Big(\int_{t=0}^{T}\sqrt{1+\beta}dt\Big) = A(0)\exp(\sqrt{1+\beta}T)$$

so $A(T)$ is stochastically bounded for all finite T as $d_2 \to \infty$. Furthermore,

$$\sqrt{\frac{c}{d_2}}\|\boldsymbol{w}_2(T)\| \leq \sqrt{\frac{c}{d_2}}(\|\boldsymbol{w}_2(T)-\boldsymbol{w}_2(0)\| + \|\boldsymbol{w}_2(0)\|)$$

which is also stochastically bounded. Integrating (30)-(33) from 0 to T finishes the proof.

E.10. Proof of Theorem 14

From (9), it's easy to get the dynamics of ς_1:

$$
\begin{aligned}
\frac{\partial \varsigma_{1,j}^2(t)}{\partial t} &= -\frac{2c}{d_1}\sum_{i=1}^{d_1}\theta_{1,ij}(t)\dot{\theta}_{1,ij}(t)x_{1,i}^2 \\
|\varsigma_{1,j}^2(T)-\varsigma_{1,j}^2(0)| &\le \frac{2c}{d_1}\sum_{i=1}^{d_1}x_{1,i}^2\int_{t=0}^{T}|\theta_{1,ij}(t)||\dot{\theta}_{1,ij}(t)|dt \\
&\le \frac{2c}{d_1}\sum_{i=1}^{d_1}x_{1,i}^2\int_{t=0}^{T}|\dot{\theta}_{1,ij}(t)|dt \\
&\le \frac{2c}{d_1}\sqrt{\frac{c}{d_1d_2}}\sum_{i=1}^{d_1}x_{1,i}^2\int_{t=0}^{T}\mathbb{E}_{in}\big|w_{2,j}(t)x_{1,i}(\bar{y}(t)-z)\tilde{\psi}'(\nu_{1,j}(t);\varsigma_{1,j}(t))\big|dt \\
&\le \frac{2c}{d_1}\sqrt{\frac{c}{d_1d_2}}\sum_{i=1}^{d_1}x_{1,i}^2\int_{t=0}^{T}|w_{2,j}(t)|\mathbb{E}_{in}\big|x_{1,i}(\bar{y}(t)-z)\big|dt \\
&\le \frac{2c}{d_1}\sqrt{\frac{c}{d_1d_2}}\sum_{i=1}^{d_1}x_{1,i}^2\int_{t=0}^{T}|w_{2,j}(t)|\|x_{1,i}\|_{in}\|\bar{y}(t)-z\|_{in}dt \\
&\le \frac{2c}{d_1^{3/2}}\sum_{i=1}^{d_1}x_{1,i}^2\|x_{1,i}\|_{in}\int_{t=0}^{T}C(t)\|\bar{y}(t)-z\|_{in}dt \\
&\le \frac{2c}{d_1^{3/2}}\sum_{i=1}^{d_1}x_{1,i}^2\|x_{1,i}\|_{in}\max_{t\in[0,T]}C(t)\int_{t=0}^{T}\|\bar{y}(t)-z\|_{in}dt \quad a.s.
\end{aligned}
$$

Here we assume that $\sqrt{\frac{c}{d_2}}\|\boldsymbol{w}_2(t)\|$ is stochastically bounded by $C(t)$. Since $C(t)$ is finite for all $t\in[0,T]$, it's easy to check the term after $\max$ operator is stochastically bounded. The remaining task is to bound term before $\max$ operator. From standard Gaussian process analysis, $x_{1,i}$ satisfy Gaussian distribution. From the law of large number (LLN), as $d_1\to\infty$,

$$
\frac{1}{d_1}\sum_{i=1}^{d_1}x_{1,i}^2\|x_{1,i}\|_{in}=\mathbb{E}[x_{1,i}^2\|x_{1,i}\|_{in}]
$$

almost surely, where the expectation is taken over w_1, and this limit is also bounded. Because of that, as $d_1,d_2\to\infty$, the difference $|\varsigma_{1,j}^2(T)-\varsigma_{1,j}^2(0)|$ converges to 0 at rate $\frac{1}{\sqrt{d_2}}$.

Notice that the proof of Lyapunov's condition (13) doesn't depend on time T from the third line. Since $\varsigma_{1,j}(T)$ stochastically converges to $\varsigma_{1,j}(0)$ for all finite T, Lyapunov's condition holds for all T thus $x_{2,j}$ always converges to Gaussian distribution conditioned on model parameter.

F. NTK of neural networks with quantized weights

F.1. Spherical harmonics

This subsection briefly reviews the relevant concepts and properties of spherical harmonics. Most part of this subsection comes from Bach (2017, appendix Section D.1.) and Bietti and Mairal (2019, appendix Section C.1.)

According to Mercer's theorem, any positive definite kernel can be decomposed as

$$
\mathcal{K}(x,x')=\sum_i\lambda_i\Phi(x)\Phi(x'),
$$

where $\Phi(\cdot)$ is called the feature map. Furthermore, any zonal kernel on the unit sphere, i.e., $\mathcal{K}(x,x')=\mathcal{K}(x^Tx')$ for any $x,x'\in\mathbb{R}^d$, $\|x\|_2=\|x'\|_2=1$, including exponential kernels and NTK, can be decomposed using spherical harmonics (equation (5)):

$$\mathcal{K}(x,x') = \sum_{k=1}^{\infty} \lambda_k \sum_{j=1}^{N(d,k)} Y_{k,j}(x) Y_{k,j}(x').$$

Legendre polynomial. We have the additional formula

$$\sum_{j=1}^{N(d,k)} Y_{k,j}(x) Y_{k,j}(x') = N(d,k) P_k(x^T x'),$$

where

$$N(d,k) = \frac{(2k+d-2)(k+d-3)!}{k!(d-2)!}.$$

The polynomial P_k is the k-th Legendre polynomial in d dimension, also known as Gegenbauer polynomials:

$$P_k(t) = \left(-\frac{1}{2}\right)^k \frac{\Gamma\left(\frac{d-1}{2}\right)}{\Gamma\left(k+\frac{d-1}{2}\right)} (1-t^2)^{(3-d)/2} \left(\frac{d}{dt}\right)^k (1-t^2)^{k+(d-3)/2}.$$

It is even (resp. odd) when k is odd (reps. even). Furthermore, they have the orthogonal property

$$\int_{-1}^{1} P_k(t) P_j(t) (1-t^2)^{(d-3)/2} dt = \delta_{ij} \frac{w_{d-1}}{w_{d-2}} \frac{1}{N(d,k)},$$

where

$$w_{d-1} = \frac{2\pi^{d-2}}{\Gamma(d/2)}$$

denotes the surface of sphere $\mathbb{S}^{d-1}$ in d dimension, and this leads to the integration property

$$\int P_j(\langle w, x\rangle) P_k(\langle w, x\rangle) d\tau(w) = \frac{\delta_{jk}}{N(p,k)} P_k(\langle x, y\rangle)$$

for any $x, y \in \mathbb{S}^{d-1}$. $\tau(w)$ is the uniform measure on the sphere.

F.2. NTK of quasi neural network

We start the proof of the Theorem 4 by the following lemmas:

Lemma 15. *The NTK of a binary weight neural network can be simplified as*

$$\begin{aligned} \mathcal{K}(x,x') &= \left(\frac{c}{d}\langle x, x'\rangle + \beta^2\right) \Sigma^{(0)} + \Sigma^{(1)}, \\ \Sigma^{(0)} &= \mathbb{E}\left[\tilde{\psi}'(\mu)\, \tilde{\psi}'(\mu')\right], \quad \Sigma^{(1)} = \mathbb{E}\left[\tilde{\psi}(\mu)\, \tilde{\psi}(\mu')\right], \end{aligned} \tag{34}$$

where $[\mu, \mu'] \sim \mathcal{N}(0, \Sigma)$,

$$\Sigma = \mathbb{E}[x_{1,i} x'_{1,i}] = \frac{c}{d} \mathrm{Var}[\theta] \begin{bmatrix} 1 & x^T x' \\ x^T x' & 1 \end{bmatrix}$$

are the pre-activation of the second layer.

Proof.

$$\begin{aligned}
\mathcal{K}(x,x') &= \sum_{i=1,j=1}^{d_1,d_2} \frac{\partial \bar{y}}{\partial \theta_{1,ij}} \frac{\partial \bar{y}'}{\partial \theta_{1,ij}} + \sum_{j=1}^{d_2} \frac{\partial \bar{y}}{\partial b_{1,j}} \frac{\partial \bar{y}'}{\partial b_{1,j}} + \sum_{j=1}^{d_2} \frac{\partial \bar{y}}{\partial w_{2,j}} \frac{\partial \bar{y}'}{\partial w_{2,j}} \\
&= \frac{c}{d_1 d_2} \sum_{i=1,j=1}^{d_1,d_2} x_{1,i} x'_{1,i} w_{2,j}^2 \tilde{\psi}'(\nu_{1,j}) \tilde{\psi}'_2(\nu'_{1,j}) \\
&\quad + \frac{\beta^2}{d_2} \sum_{j=1}^{d_2} \tilde{\psi}'(\nu_{1,j}) \tilde{\psi}'(\nu'_{1,j}) + \frac{1}{d_2} \sum_{j=1}^{d_2} \tilde{\psi}(\nu_{1,j}) \tilde{\psi}(\nu'_{1,j}) \\
&= \frac{c}{d_1 d_2} \sum_{i=1}^{d_1} x_{1,i} x'_{1,i} \sum_{j=1}^{d_2} w_{2,j}^2 \tilde{\psi}'(\nu_{1,j}) \tilde{\psi}'(\nu'_{1,j}) \\
&\quad + \frac{\beta^2}{d_2} \sum_{j=1}^{d_2} w_{2,j}^2 \tilde{\psi}'(\nu_{1,j}) \tilde{\psi}'(\nu'_{1,j}) + \frac{1}{d_2} \sum_{j=1}^{d_2} \tilde{\psi}(\nu_{1,j}) \tilde{\psi}(\nu'_{1,j}) \\
&= (\frac{c}{d} \langle x, x' \rangle + \beta^2) \mathbb{E}[\tilde{\psi}'(\nu) \tilde{\psi}'(\nu')] + \mathbb{E}[\tilde{\psi}(\nu) \tilde{\psi}(\nu')] \quad a.s.
\end{aligned}$$

where (ν, ν') has the same distribution as $(\nu_{2,j}, \nu'_{2,j})$ for any j. We make use of the fact $\mathbb{E}[w_{2,j}^2] = 1$, and from central limit theorem, $x_{1,i}, x'_{1,i}$ and $\mu_{1,i}, \mu'_{1,i}$ converge to joint Gaussian distribution for any fixed x, x' as $d_1 \to \infty$

$$\begin{aligned}
\mathbb{E}[x_{1,i} x'_{1,i}] &= \frac{1}{d} \mathbb{E}[\sum_{k=1}^{d} w_{ki} x_k \sum_{k'=1}^{d} w_{k'i} x'_{k'}] \\
&= \frac{1}{d} \mathbb{E}[\sum_{k=1}^{d} w_{ki}^2 x_k x'_k] \\
&= \frac{1}{d} \langle x, x' \rangle
\end{aligned}$$

Similarly,

$$\begin{aligned}
\mathbb{E}[\mu_{1,i}^2] &= \frac{c}{d_1} \sum_{i=1}^{d_1} \mathbb{E}[\theta_{1,ij}^2] \mathbb{E}[x_{1,j}^2] = \frac{c}{d} \mathrm{Var}[\theta] \\
\mathbb{E}[\mu_{1,i} \mu'_{1,i}] &= \frac{c}{d_1} \sum_{i=1}^{d_1} \mathbb{E}[\theta_{1,ij}^2] \mathbb{E}[x_{1,j} x'_{1,j}] = \frac{c}{d} \mathrm{Var}[\theta] \langle x, x' \rangle
\end{aligned}$$

□

F.3. Proof of Theorem ??

Remind that as is proved in Theorem 14, $\varsigma_{1,j}(T) \to \varsigma_{1,t}(0)$ for any T satisfying a mild condition, and $\varsigma_{1,t}(0)$ is nonzero almost surely. Making use the fact that $\tilde{\psi}(\cdot;\varsigma)$ is continuous with respect to ς, and its first and second order derivative is stochastically bounded, the change of kernel $\mathcal{K}$ induced by $\varsigma_{1,j}$ converges to 0 as $d_1, d_2 \to \infty$. This reduces to this quasi neural network to a standard neural network with activation function $\tilde{\psi}(\cdot)$, which is twice differentiable and has bounded second order derivative. From Theorem 2 in (Jacot et al., 2018), the kernel during training converges to the one during initialization. For the ease of the readers, we restate the proof below. On the limit $d_2 \to \infty, d_1 \to \infty$,

$$\begin{aligned}
&\mathcal{K}(x,x')(t) - \mathcal{K}(x,x')(0)\\
&= \frac{c\langle x,x'\rangle + \beta^2}{d_2} \sum_{j=1}^{d_2} \big(w_{2,j}^2(t)\tilde{\psi}'(\nu_{1,j}(t))\tilde{\psi}'(\nu'_{1,j}(t)) - w_{2,j}^2(0)\tilde{\psi}'(\nu_{1,j}(0))\tilde{\psi}'(\nu'_{1,j}(0))\big)\\
&\quad + \frac{1}{d_2}\sum_{j=1}^{d_2}\big(\tilde{\psi}(\nu_{1,j}(t))\tilde{\psi}(\nu'_{1,j}(t)) - \tilde{\psi}(\nu_{1,j}(0))\tilde{\psi}(\nu'_{1,j}(0))\big)\\
&= \frac{c\langle x,x'\rangle + \beta^2}{d_2}\Bigg(\sum_{j=1}^{d_2}\big(w_{2,j}^2(t) - w_{2,j}^2(0)\big)\tilde{\psi}'(\nu_{1,j}(t))\tilde{\psi}'(\nu'_{1,j}(t))\\
&\quad + \sum_{j=1}^{d_2} w_{2,j}^2(0)\big(\tilde{\psi}'(\nu_{1,j}(t)) - \tilde{\psi}'(\nu_{1,j}(0))\big)\tilde{\psi}'(\nu'_{1,j}(t))\\
&\quad + \sum_{j=1}^{d_2} w_{2,j}^2(0)\tilde{\psi}'(\nu_{1,j}(0))\big(\tilde{\psi}'(\nu'_{1,j}(t)) - \tilde{\psi}'(\nu'_{1,j}(0))\big)\Bigg)\\
&\quad + \frac{1}{d_2}\sum_{j=1}^{d_2}\tilde{\psi}(\nu_{1,j}(t))\big(\tilde{\psi}(\nu'_{1,j}(t))\tilde{\psi}(\nu'_{1,j}(0))\big)\\
&\quad + \frac{1}{d_2}\sum_{j=1}^{d_2}\tilde{\psi}(\nu'_{1,j}(0))\big(\tilde{\psi}(\nu_{1,j}(t)) - \tilde{\psi}(\nu_{1,j}(0))\big)
\end{aligned}$$

$$\begin{aligned}
&|\mathcal{K}(x,x')(t) - \mathcal{K}(x,x')(0)|\\
&\le \left|\frac{c\langle x,x'\rangle + \beta^2}{d_2}\right|\Bigg(\sum_{j=1}^{d_2} w_{2,j}^2(0)\tilde{\psi}'(\nu'_{1,j}(t))\big|\tilde{\psi}'(\nu_{1,j}(t)) - \tilde{\psi}'(\nu_{1,j}(0))\big|\\
&\quad + \sum_{j=1}^{d_2} w_{2,j}^2(0)\tilde{\psi}'(\nu_{1,j}(0))\big|\tilde{\psi}'(\nu'_{1,j}(t)) - \tilde{\psi}'(\nu'_{1,j}(0))\big|\\
&\quad + \sum_{j=1}^{d_2}|w_{2,j}(t) - w_{2,j}(0)||w_{2,j}(t) + w_{2,j}(0)||\tilde{\psi}'(\nu_{1,j}(t))\tilde{\psi}'(\nu'_{1,j}(t))|\Bigg)\\
&\quad + \frac{1}{d_2}\sum_{j=1}^{d_2}\tilde{\psi}(\nu_{1,j}(t))\big|\tilde{\psi}(\nu'_{1,j}(t))\tilde{\psi}(\nu'_{1,j}(0))\big|\\
&\quad + \frac{1}{d_2}\sum_{j=1}^{d_2}\tilde{\psi}(\nu'_{1,j}(0))\big|\tilde{\psi}(\nu_{1,j}(t)) - \tilde{\psi}(\nu_{1,j}(0))\big|
\end{aligned}$$

From Theorem 14, and observing that $\tilde{\psi}'(x), \tilde{\psi}''(x)$ are bounded by constants, one can verify that each summation term is stochastically bounded by $\sqrt{d_2}$, so as $d_2 \to \infty$, $\mathcal{K}(t) - \mathcal{K}(0)$ converges to 0 at rate $\sqrt{d_2}$.

F.4. Spherical harmonics decomposition to activation function

Following Bach (2017), we start by studying the decomposition of action in quasi neural network (3) and its gradients (**??**): for arbitrary fixed $c > 0$, $-1 \le t \le 1$, we can decompose equation (3) and (**??**) as

$$\tilde{\sigma}(ct) = \sum_{k=0}^{\infty} \lambda_k N(d,k) P_k(t), \tag{35}$$

$$\tilde{\sigma}'(ct) = \sum_{k=0}^{\infty} \lambda'_k N(d,k) P_k(t), \tag{36}$$

where P_k is the k-th Legendre polynomial in dimension d.

Lemma 16. *The decomposition of activation function in the quasi neural network (35) satisfies*

1. $\lambda_k = 0$ *if* k *is odd,*
2. $\lambda_k > 0$ *if* k *is even,*
3. $\lambda_k \asymp \mathrm{Poly}(k)(C/\sqrt{k})^{-k}$ *as* $k \to \infty$ *when* k *is even, where* $\mathrm{Poly}(k)$ *denotes a polynomial of* k*, and* C *is a constant.*

Its gradient (36) satisfies

1. $\lambda'_k = 0$ *if* k *is even,*
2. $\lambda'_k > 0$ *if* k *is odd,*
3. $\lambda'_k \asymp \mathrm{Poly}(k)(C/\sqrt{k})^{-k}$ *as* $k \to \infty$ *when* k *is odd, where* $\mathrm{Poly}(k)$ *denotes a polynomial of* k*, and* C *is a constant.*

Proof. Let's start with the derivative of activation function in quasi neural network:

$$\tilde{\sigma}'(t) = \Phi(\hat{c}t), -1 \le t \le 1,$$

where $\hat{c}$ is a constant. We introduce the auxiliary parameters $x, w \in \mathbb{R}^d$ s.t. $\|x\|_2 = \|w\|_2 = 1$ and let $t = w^T x$ By Cauchy-Schwarz inequality, $-1 \le w^T x \le 1$. Following (Bach, 2017), we have the following decomposition to $\tilde{\sigma}'(w^T x)$:

$$\tilde{\sigma}'(w^T x) = \sum_{k=1}^{\infty} \lambda'_k N(d,k) P_k(w^T x),$$

where $N(d,k)$ and $P_k(\cdot)$ are defined in section F.1, λ'_k can be computed by

$$\begin{aligned}\lambda'_k &= \frac{w_{d-1}}{w_d} \int_{-1}^{1} \tilde{\sigma}'(t) P_k(t)(1-t^2)^{(d-2)/2} dt \\ &= \left(-\frac{1}{2}\right)^k \frac{\Gamma((d-1)/2)}{\Gamma(k+(d-1)/2)} \frac{w_{d-1}}{w_d} \int_{-1}^{1} \tilde{\sigma}'(t) \left(\frac{d}{dt}\right)^k (1-t^2)^{k+(d-3)/2} dt.\end{aligned}$$

To solve this itegration, we can apply Taylor decomposition to $\tilde{\sigma}'(\cdot)$:

$$\tilde{\sigma}'(\hat{c}t) = \frac{1}{2} + \frac{1}{\sqrt{2\pi}} \sum_{n=0}^{\infty} \frac{(-1)^n \hat{c}^{2n+1}}{2^n n!(2n+1)} t^{2n+1}. \tag{37}$$

We will study the following polynomial integration first

$$\int_{-1}^{1} t^\alpha \left(\frac{d}{dt}\right)^k (1-t^2)^{k+(d-3)/2} dt.$$

When $\alpha < k$, this integration equals 0 as P_k is orthogonal to all polynomials of degree less than k. If $(\alpha - k) \mod 2 \neq 0$, this integration is 0 because the function to be integrated is an odd function. For $\alpha \ge k$ and $k \equiv \alpha \mod 2$ (k is odd), using successive integration by parts,

$$\begin{aligned}\int_{-1}^{1} t^\alpha \left(\frac{d}{dt}\right)^k (1-t^2)^{k+(d-3)/2} dt &= (-1)^k \frac{\alpha!}{(\alpha-k)!} \int_{-1}^{1} t^{\alpha-k} (1-t^2)^{k+(d-3)/2} dt \\ &= (-1)^k \frac{\alpha!}{(\alpha-k)!} \int_{-\pi/2}^{\pi/2} \sin^{\alpha-k}(x) \cos^{2k+(d-2)}(x) dx \\ &= (-1)^k C_d \frac{\alpha!(2k+d-3)!!}{(\alpha-k)!!(\alpha+k+d-2)!!},\end{aligned} \tag{38}$$

where C_d is a constant that depends only on d mod 2.

Combining (37) and (38), we have $\lambda_k = 0$ when k is even and $k \neq 0$. When k is odd,

$$\lambda'_k = \left(-\frac{1}{2}\right)^k \frac{\Gamma((d-1)/2)}{\Gamma(k+(d-1)/2)} \frac{w_{d-1}}{w_d} \frac{C_d}{\sqrt{2\pi}} \sum_{\alpha=k:2}^{\infty} \hat{c}^{\alpha}(-1)^{(\alpha-1)/2} \frac{(\alpha-2)!!(2k+d-3)!!}{(\alpha-k)!!(\alpha+k+d-2)!!}.$$

Following (Bach, 2017; Geifman et al., 2020) we take d as a constant and take k to infinity. Let $\beta = (\alpha-k)/2 \geq 0$ we have

$$\begin{aligned}
\lambda'_k &= (-1)^{(k+1)/2} \left(\frac{1}{2}\right)^k \frac{\Gamma((d-1)/2)}{\Gamma(k+(d-1)/2)} \frac{w_{d-1}}{w_d} \frac{C_d}{\sqrt{2\pi}} \sum_{\beta=0}^{\infty} \frac{(-1)^{\beta}\hat{c}^{2\beta+k}(2\beta+k-2)!!(2k+d-3)!!}{(2\beta)!!(2\beta+2k+d-2)!!} \\
&\asymp (-1)^{(k+1)/2} \left(\frac{1}{2}\right)^k \frac{\Gamma((d-1)/2)}{\Gamma(k+(d-1)/2)} \frac{w_{d-1}}{w_d} \frac{C_d}{\sqrt{2\pi}} \sum_{\beta=0}^{\infty} \frac{(-1)^{\beta}\hat{c}^{2\beta+k}\Gamma(\beta+k/2)\Gamma(k+(d-1)/2)}{\beta!\Gamma(\beta+k+d/2)2^{\beta-k/2}} \\
&:= (-1)^{(k+1)/2} \left(\frac{1}{2}\right)^k \frac{\Gamma((d-1)/2)}{\Gamma(k+(d-1)/2)} \frac{w_{d-1}}{w_d} \frac{C_d}{\sqrt{2\pi}} \sum_{\beta=0}^{\infty} g(\beta,k).
\end{aligned}$$

where $\asymp$ means the radio converge to a constant which doesn't depend on k or β as $k \to \infty$. Here we introduced the function $g(\beta, k)$ for simplification, and it satisfies

$$\frac{g(\beta,k)}{g(\beta-1,k)} = -\frac{\hat{c}^2(2\beta+k-3)}{2\beta(2\beta+2k+d-2)},$$

which indicates that $g(\beta, k)$ decays at factorial rate when $\beta > \hat{c}^2/2$. If $k \gg \hat{c}^2/2$, $\beta \ll k$ regime dominates the summation. Using Stirling's approximation, one can easily prove

$$\Gamma(k+x) \asymp \Gamma(k)k^x$$

When $k \gg d$,

$$\begin{aligned}
g(\beta,k) &= \frac{(-1)^{\beta}\hat{c}^{2\beta+k}\Gamma(\beta+k/2)\Gamma(k+(d-1)/2)}{\beta!\Gamma(\beta+k+d/2)2^{\beta-k/2}} \\
&\asymp \left(-\frac{1}{4}\right)^{\beta} \hat{c}^{2\beta+k}\Gamma(k+(d-1)/2)\frac{2^{k/2}\Gamma(k/2)}{\Gamma(k)k^{d/2}\beta!} \\
&= \hat{c}^k\Gamma(k+(d-1)/2)\frac{2^{k/2}\Gamma(k/2)}{\Gamma(k)k^{d/2}} \left(-\frac{\hat{c}^2}{4}\right)^{\beta} \frac{1}{\beta!}
\end{aligned}$$

This splits $g(\beta, k)$ into two parts: the first part depends only on k and the rest part only depends on β. The summation of the second part over β yields

$$\sum_{\beta=0}^{\infty} \left(-\frac{\hat{c}^2}{4}\right)^{\beta} \frac{1}{\beta!} = \exp(-\frac{\hat{c}^2}{4}),$$

Using Stirling's approximation

$$\gamma(x+1) \asymp \sqrt{2\pi x}(x/e)^x,$$

this leads to the expression for λ_k:

$$\begin{aligned}
\lambda'_k &\asymp (-1)^{(k+1)/2} \left(\frac{1}{2}\right)^k \frac{\Gamma((d-1)/2)}{\Gamma(k+(d-1)/2)} \hat{c}^k\Gamma(k+(d-1)/2)\frac{2^{k/2}\Gamma(k/2)}{\Gamma(k)k^{d/2}} \exp(-\frac{\hat{c}^2}{4}) \\
&\asymp (-1)^{(k+1)/2} \left(\frac{\hat{c}}{2}\right)^k \frac{2^{k/2}\Gamma(k/2)}{\Gamma(k)k^{d/2}} \exp(-\frac{\hat{c}^2}{4}) \\
&\asymp (-1)^{(k+1)/2} \left(\frac{\hat{c}}{2}\sqrt{\frac{e}{k}}\right)^k k^{-d/2} \exp(-\frac{\hat{c}^2}{4})
\end{aligned}$$

Similarly, the activation function of quasi neural network has the Tayler expansion

$$\begin{aligned}\tilde{\sigma}(x) &= \varsigma_\ell \varphi\left(\hat{c}t\right) + x\Phi\left(\hat{c}t\right) \\ &= \frac{t}{2} + \sum_{n=0}^{\infty} \frac{(-1)^n \hat{c}^{2n+1}}{2^{n+1}(n+1)!(2n+1)} t^{2n+2}.\end{aligned}$$

So $\lambda_k = 0$ when k is odd, and when k is even:

$$\lambda_k = (-1)^{(k+1)/2} \left(\frac{1}{2}\right)^k \frac{\Gamma((d-1)/2)}{\Gamma(k+(d-1)/2)} \frac{w_{d-1}}{w_d} \frac{C_d}{\sqrt{2\pi}} \sum_{\beta=0}^{\infty} \frac{(-1)^\beta \hat{c}^{2\beta+k}(2\beta+k-2)!!(2k+d-3)!!}{(2\beta)!!(2\beta+2k+d-2)!!}$$

Furthermore, when $k \gg d$,

$$\lambda_k \asymp (-1)^{k/2} k^{-\frac{d}{2}} \left(\frac{\hat{c}}{2}\sqrt{\frac{e}{k}}\right)^k \exp\left(-\frac{\hat{c}^2}{4}\right),$$

□

F.5. Computing covariance matrix

In this part, we prove Theorem 4 by computing $\Sigma^{(0)}$ and $\Sigma^{(1)}$.

Theorem 4 *NTK of a binary weight neural network can be decomposed using equation (5). If $k \gg d$, then*

$$\mathrm{Poly}_1(k)(C)^{-k} \le u_k \le \mathrm{Poly}_2(k)(C)^{-k}$$

where $\mathrm{Poly}_1(k)$ and $\mathrm{Poly}_2(k)$ denote polynomials of k, and C is a constant.

We make use of the results in Section F.4, and remind that λ_k, λ'_k depends on $\hat{c}$, we make this explicit as $\lambda_k(\hat{c}), \lambda'_k(\hat{c})$. We introduce an auxiliary parameter $w \sim \mathcal{N}(0, I)$, and denote $\tilde{c} = \sqrt{\frac{c\mathrm{Var}[\Theta]}{d\ \varsigma^2}} = \sqrt{\frac{\mathrm{Var}[\theta]}{1-\mathrm{Var}[\theta]}}$, $\tilde{w} = w/\|w\|_2$, then the decomposition of kernel (5) can be computed by

$$\begin{aligned}\Sigma^{(1)} &= \mathbb{E}_\theta\left[\tilde{\sigma}\left(\mu\right)\tilde{\sigma}\left(\mu\right)\right] \\ &= \mathbb{E}_{w\sim\mathcal{N}(0,I)}\left[\tilde{\sigma}(\tilde{c}\langle w, x\rangle)\tilde{\sigma}(\tilde{c}\langle w, x'\rangle)\right] \\ &= \mathbb{E}_{\|w\|} \int \tilde{\sigma}(\tilde{c}\langle \tilde{w}, x\rangle)\tilde{\sigma}(\tilde{c}\langle \tilde{w}, x'\rangle) d\tau(\tilde{w}) \\ &= \mathbb{E}_{\|w\|} \sum_{k=0}^{\infty} (\lambda_k(\tilde{c}\|w\|))^2 N(p,k) P_k(\langle x, x'\rangle), \\ \Sigma^{(0)} &= \mathbb{E}_\theta\left[\tilde{\sigma}'\left(\mu\right)\tilde{\sigma}'\left(\mu\right)\right] \\ &= \mathbb{E}_{\|w\|} \sum_{k=0}^{\infty} (\lambda'_k(\tilde{c}\|w\|))^2 N(p,k) P_k(\langle x, x'\rangle).\end{aligned}$$

First compute $\Sigma^{(0)}$. According to Lemma 16 in Bietti and Mairal (2019),

$$u_{0,k} = \mathbb{E}_{w\sim\mathcal{N}(0,I)}[{\lambda'_k}^2] = \mathbb{E}_{\|w\|}[{\lambda'_k}^2].$$

Remind that

$$\lambda'_k(\tilde{c}\|w\|) \asymp (-1)^{k/2} k^{-d/2} \left(\frac{\tilde{c}\|w\|}{2}\sqrt{\frac{e}{k}}\right)^k \exp\left(-\frac{\tilde{c}^2\|w\|^2}{4}\right).$$

$$
\begin{aligned}
u_{0,k} &= \mathbb{E}_{\|w\|}(\lambda'_k(\tilde{c}\|w\|))^2 \\
&\asymp \mathbb{E}_{\|w\|}k^{-d}\left(\frac{\tilde{c}^2\|w\|^2 e}{4k}\right)^k \exp\left(-\frac{\tilde{c}^2\|w\|^2}{2}\right) \\
&= k^{-d}\,(k/e)^{-k}\,\mathbb{E}_{\tilde{c}\|w\|}\left(\frac{\tilde{c}^2\|w\|^2}{4}\right)^k \exp\left(-\frac{\tilde{c}^2\|w\|^2}{2}\right)
\end{aligned}
$$

Because $w \sim \mathcal{N}(0,1)$, $\|w\|_2^2$ satisfy Chi-square distribution, and its momentum generating function is

$$
M_X(t) = \mathbb{E}[\exp(t\|w\|^2)] = (1-2t)^{-d/2}
$$

It's k-th order derivative is

$$
M_X^{(k)} = \mathbb{E}[\|w\|^{2k}\exp(t\|w\|^2)] = \frac{(d+2k-2)!!}{(d-2)!!}(1-2t)^{-\frac{d}{2}-k}
$$

Let $t = -\tilde{c}^2/2$, we get

$$
\begin{aligned}
\mathbb{E}\left[\|w\|^{2k}\exp\left(-\frac{\tilde{c}^2\|w\|^2}{2}\right)\right] &= \frac{(d+2k-2)!!}{(1+\tilde{c}^2)^{d/2+k}(d-2)!!} \\
&\asymp 2^k\frac{\Gamma(k+d/2)}{\Gamma(d/2)}(1+\tilde{c}^2)^{-k-d/2} \\
&\asymp \left(\frac{2k}{(1+\tilde{c}^2)e}\right)^{d/2+k}\sqrt{\frac{1}{k}}
\end{aligned}
$$

so

$$
\begin{aligned}
u_{0,k} &\asymp \left(\frac{\tilde{c}}{2}\right)^{2k} k^{-d}\,(k/e)^{-k}\left(\frac{2k}{(1+\tilde{c}^2)e}\right)^{d/2+k} \\
&\asymp k^{-(d-1)/2}\left(\frac{\tilde{c}^2}{2(1+\tilde{c}^2)}\right)^k
\end{aligned}
$$

when k is odd, and 0 when k is even.

Similarly,

$$
\begin{aligned}
u_{1,k} &\asymp \left(\frac{\tilde{c}}{2}\right)^{2k} k^{-d}\,(k/e)^{-k}\left(\frac{2k}{(1+\tilde{c}^2)e}\right)^{d/2+k} \\
&\asymp k^{-(d-1)/2}\left(\frac{\tilde{c}^2}{2(1+\tilde{c}^2)}\right)^k
\end{aligned}
$$

when k is even, and 0 when k is odd.

Finally, using the recurrence relation

$$
tP_k(t) = \frac{k}{2k+d-3}P_{k-1}(t) + \frac{k+d-3}{2k+d-3}P_{k+1}(t)
$$

taking them into (34) finishes the proof.

F.6. Gaussian kernel

$$
\begin{aligned}
\mathcal{K}_{RGauss}(x,x') &= \mathbb{E}[\mathcal{K}_{Gauss}(\kappa x, \kappa x')] \\
&= \mathbb{E}\left[\exp\left(-\frac{\kappa^2\|x-x'\|}{\xi^2}\right)\right] \\
&= \mathbb{E}\left[\exp\left(-\frac{\|x-x'\|}{(\xi/\kappa)^2}\right)\right]
\end{aligned}
$$

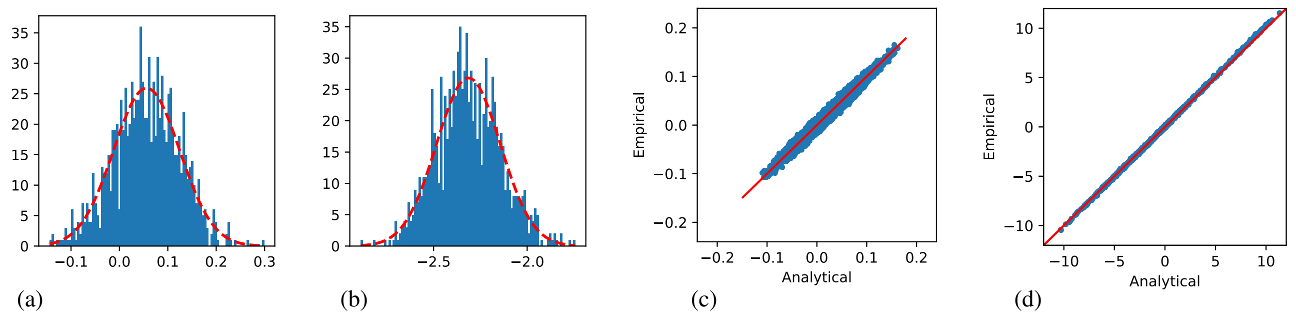


Figure 3: Approximation of quasi neural network. (a)(b): before (a) and after (b) training, histogram of output under fixed model parameter (blue), and fitted with Gaussian distribution (red). (c)(d): $\mathbb{E}(y|\Theta)$ computed from quasi neural network (horizontal axis) and by Monte Carlo (Vertical axis). The red line shows $y = x$.

This indicates that this kernel can be decomposed using spherical harmonics (5), and when $k \gg d$, the coefficient

$$\begin{aligned}
u_k &= \mathbb{E}\left[\exp\left(-\frac{2\kappa^2}{\xi^2}\right)\left(\frac{\xi}{\kappa}\right)^{d-2} I_{k+d/2-1}\left(\frac{2\kappa^2}{\xi^2}\right)\Gamma\left(\frac{d}{2}\right)\right] \\
&\asymp \mathbb{E}\left[\exp\left(-\frac{2\kappa^2}{\xi^2}\right)\Gamma\left(\frac{d}{2}\right)\sum_{j=0}^{\infty}\frac{1}{j!\Gamma(k+d/2+j)}\left(\frac{\kappa^2}{\xi^2}\right)^{k+2j}\right] \\
&= \sum_{j=0}^{\infty}\frac{\Gamma(d/2)}{j!\Gamma(k+d/2+j)}\mathbb{E}\left[\left(\frac{\kappa^2}{\xi^2}\right)^{k+2j}\exp\left(-\frac{2\kappa^2}{\xi^2}\right)\right] \\
&= \sum_{j=0}^{\infty}\frac{\Gamma(d/2)}{j!\Gamma(k+d/2+j)}\frac{\Gamma(k+2j+d/2)}{\Gamma(d/2)}\left(\frac{2}{\xi^2}\right)^{k+2j}\left(\frac{1}{1+4/\xi^2}\right)^{(k+2j+d/2)} \\
&\asymp \left(\frac{2}{\xi^2}\right)^{k}\left(1+\frac{4}{\xi^2}\right)^{(-k-d/2)}\sum_{j=0}^{\infty}\frac{1}{j!}\left(\frac{k(2/\xi^2)^2}{(1+4/\xi^2)^2}\right)^{j} \\
&\asymp \left(\frac{2}{4+\xi^2}\right)^{k}\exp\left(\left(\frac{2}{4+\xi^2}\right)^2 k\right).
\end{aligned}$$

Note that $\frac{2}{4+\xi^2}\exp\left(\left(\frac{2}{4+\xi^2}\right)^2\right)$ is always smaller than 1 so u_k is always decreasing with k.

G. Additional information about numerical result

G.1. Toy dataset

In neural networks (NN) experiment, we used three layers with the first layer fixed. The number of hidden neural is 512. In neural network with binary weights (BWNN) experiment, the setup is the same as NN except the second layer is Binary. We used BinaryConnect method with stochastic rounding. We used gradient descent with learning rate searched from $10^{-3}, 10^{-2}, 10^{-1}$. For Laplacian kernel and Gaussian kernel, we searched kernel bandwidth from $2^{-2}\mu$ to $2^{2}\mu$ by power of 2, and μ is the medium of pairwise distance. The SVM cost value parameter is from 10^{-2} to 10^{4} by power of 2.

More results are listed in Table 1. Accuracy are shown in the format of mean $\pm$ std. P90 and P95 denotes the percentage of dataset that a model achieves at least 90% and 95% of the highest accuracy, respectively.

Table 1: More results in UCI dataset experiment.

Classifier	Training			Testing		
	Accuracy	P90	P95	Accuracy	P90	P95
NN	96.19±8.03%	96.67%	91.11%	77.62±16.10%	73.33%	56.67%
BWNN	93.55±10.39%	84.44%	76.67%	77.83±16.57%	77.78%	54.44%
Laplacian	93.52±9.65%	85.56%	76.67%	81.62±14.72%	97.78%	91.11%
Gaussian	91.08±10.63%	76.67%	58.89%	81.40±14.85%	95.56%	87.78%

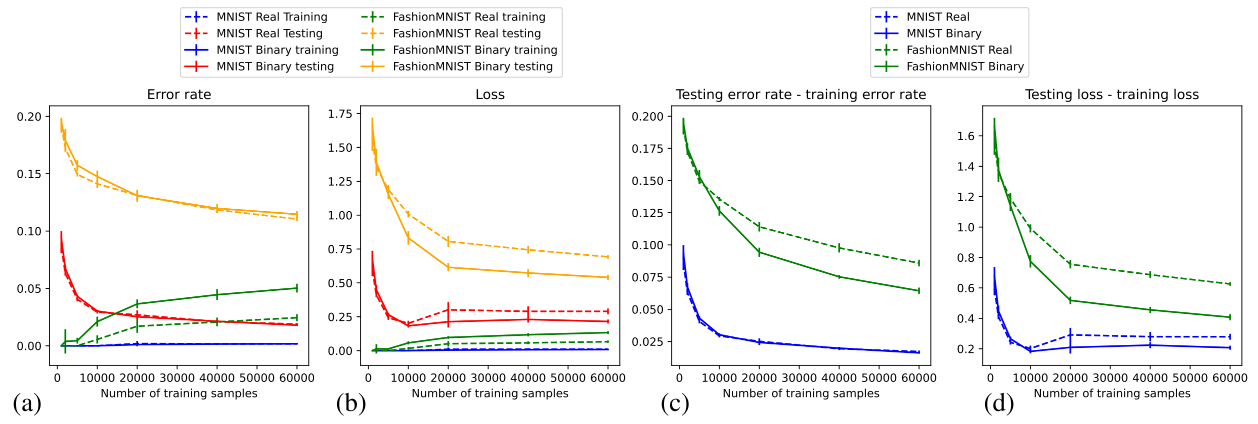


Figure 4: Training/testing error rate and loss of neural networks with/without binary weight. (a) Training and testing error rate. (b) Training and testing loss. (c) Testing error rate - Training error rate. (d) Testing loss - Training loss.

G.2. MNIST-like dataset

Similar to the toy dataset experiment, we used three layer neural networks with the first layer fixed, and only quantize the second layer. The number of neurons in the hidden layer is 2048. The batchsize if 100 and ADAM optimizer with learning rate 10^{-3} is used.

H. Additional Lemmas

Lemma 17 (Kolmogorov's Strong Law of Large Number (SLLN))**.** *Suppose $X_1, X_2, \ldots$ are independent variables such that $E[X_n] = \mu$ and $\sum_n \operatorname{Var}[X_n]/n^2 < \infty$. Then, $\frac{\sum_{i=1}^n X_i}{n} \to \mu$ a.e..*

Lemma 18 (Continuous mapping theorem)**.** *Let $\{X_n\}$, X be random elements defined on a metric space S. Suppose a function $g : S \to S'$ (where S' is another metric space) has the set of discontinuity points D_g such that $\Pr[X \in D_g] = 0$. Then*

$$\begin{aligned} X_n \xrightarrow{\mathrm{d}} X \quad &\Rightarrow \quad g(X_n) \xrightarrow{\mathrm{d}} g(X) \\ X_n \xrightarrow{\mathrm{p}} X \quad &\Rightarrow \quad g(X_n) \xrightarrow{\mathrm{p}} g(X) \\ X_n \xrightarrow{a.s.} X \quad &\Rightarrow \quad g(X_n) \xrightarrow{a.s.} g(X) \end{aligned} \tag{39}$$

Table 2: Pairwise performance comparison on selected 90 UCI datasets.

Classifier	Testing				Training-Testing			
	t-stats	p-val	$<$	$>$	t-stats	p-val	$<$	$>$
NN-BWNN	0.7471	0.4569	53.33%	41.11%	4.034	0.000	26.67%	67.77%
Laplace-Gaussian	0.4274	0.6701	51.11%	33.33%	3.280	0.001	37.78%	53.33%