Supplementary Material

Appendix. A: Proofs

Theorem 2.

For a deep FCNN with BN layers following every FC layer, the criterion converges in probability that $\chi_1 \xrightarrow{p.} \frac{1}{1-1/\pi}$ as layer width goes to infinity, such that BN networks are in chaotic regime.

Proof. With Batch Normalization after FC layer, the bias term always keep zero such that $\sigma_b = 0$. Besides, the intrinsic property of BN that insensitive to norm of parameter, our analysis choose initialization $\sigma_w = 1$. We have transition operator $\mathcal{T}$ in l-th layer as formula:

$$\begin{aligned}\mathcal{T}(\rho) &= \frac{1}{n_{l+1}}\mathbb{E}_{(u,v)\sim\Omega_{n_l}(\rho)}\left(BN(h^{l+1}(u)^T BN(h^{l+1}(v))\right) \\ &= \frac{1}{n_{l+1}}\sum_i^{n_{l+1}}\mathbb{E}_{(u,v)\sim\Omega_{n_l}(\rho)}\left(BN(h_i^{l+1}(u))BN(h_i^{l+1}(v))\right)\end{aligned} \tag{1}$$

where, random vector $h^{l+1}(u) = \frac{\mathbf{W}^l\phi(u)}{\sqrt{n_l}}$ and h_i^{l+1} is the i-th component. We will ignore the up script l and $l+1$ if no ambiguity and define $n_o = n_{l+1}, n_i = n_l$. Notice the definition of BN operator, the mean and variance value is expected to dataset, i.e. $m = \mathbb{E}_u[h(u)]$ and $\nu = \text{Var}_u[h(u)]$. We have:

$$\begin{aligned}\mathcal{T}(\rho) &= \frac{1}{n_o}\sum_i^{n_o}\mathbb{E}_{(u,v)\sim\Omega_{n_i}(\rho)}\left(BN(h_i(u))BN(h_i(v))\right) \\ &= \frac{1}{n_o}\sum_i^{n_o}\frac{1}{\nu_i}\mathbb{E}_{(u,v)\sim\Omega_{n_i}(\rho)}\left((h_i(u)-m_i)(h_i(v)-m_i)\right) \\ &= \frac{1}{n_o}\sum_i^{n_o}\left[\frac{1}{\nu_i}\mathbb{E}_{(u,v)\sim\Omega_{n_i}(\rho)}(h_i(u)h_i(v)) - m_i^2\right] \\ &= \frac{1}{n_o}\sum_i^{n_o}\left[\frac{1}{\nu_i}\hat{\phi}(\rho) - m_i^2\right]\end{aligned} \tag{2}$$

The $\hat{\phi}(\rho)$ is the dual activatio of ReLU defined in Eq. 2, which dominates NNGP kernel of straight network. Notice $\frac{d}{d\rho}\hat{\phi}(1) = 1/2$, we calculate its derivative at $\rho = 1$,

$$\chi_1 = \frac{d}{d\rho}\mathcal{T}(1) = \frac{d}{d\rho}\frac{1}{n_o}\sum_i^{n_o}\left[\frac{1}{\nu_i}\hat{\phi}(1) - m_i^2\right] = \frac{1}{2n_o}\sum_i^{n_o}\frac{1}{\nu_i} \tag{3}$$

We find the different on χ_1 led by BN is the average variance term. However, χ_1 is still random variable depending on Gaussian Random matrix W. We will prove as layer width approaching infinity, the variance converge in probability to a constant and valid our theorem.

$$\begin{aligned}\nu_i = \text{Var}_u[h_i(u)] &= \text{Var}_u[\frac{\mathbf{W}_i\phi(u)}{\sqrt{n_i}}] \\ &= \frac{1}{n_i}\text{Var}_u(\sum_j^{n_i}\mathbf{W}_{ij}\phi(u_j)) \\ &= \frac{1}{n_i}\sum_j^{n_i}\mathbf{W}_{ij}\text{Var}_u(\phi(u_j))\end{aligned} \tag{4}$$

For ReLU function $\phi(x) = \max(x, 0)$, we have $\mathbb{E}_u(\phi(u)) = \sqrt{\frac{1}{2\pi}}$ and $\text{Var}_u(\phi(u)) = \mathbb{E}_u(\phi(u)^2) - \mathbb{E}_u(\phi(u))^2 = \frac{1}{2} - \frac{1}{2\pi}$ Denote $S = 1 - 1/\pi$, such that

$$\nu_i = \frac{S}{2n_i}\sum_j^{n_i}\mathbf{W}_{ij} \tag{5}$$

Leveraging the Large Number Theorem, the average value of any raw of random matrix W converges, i.e. $\nu_i \xrightarrow{p.} \frac{S}{2}, \forall\, i$. Combining Eq. 3, we get our conclusion.

$$\chi_1 = \frac{1}{2n_o}\sum_i^{n_o}\frac{1}{\nu_i} \xrightarrow{p.} \frac{1}{2n_o}\sum_i^{n_o}\frac{2}{S} = \frac{1}{S} \tag{6}$$

Lemma 1.

For ReLU activation $\phi(x) = max(x, 0)$, and define $\Phi(\rho) = \mathbb{E}_{(u,v)\sim\Omega_n(\rho)}\left(\phi(u)^T\mathbf{W}^T\mathbf{W}\phi(v)\right)$, $\bar{\Phi}(\rho) = \mathbb{E}_{(u,v)\sim\Omega_n(\rho)}\left(\phi(u)^T\bar{\mathbf{W}}^T\bar{\mathbf{W}}\phi(v)\right)$, the following equation holds.

$$\mathbb{E}_W[\bar{\Phi}(\rho)] = \frac{n_o - 1}{n_o}\mathbb{E}_W[\Phi(\rho) - \Phi(0)]$$

Proof. The different from $\Phi(\rho)$ and $\bar{\Phi}(\rho)$ is the symmetric random matrix. We denote $\mathbf{A} = \mathbf{W}^T\mathbf{W}$ and $\bar{\mathbf{A}} = \bar{\mathbf{W}}^T\bar{\mathbf{W}}$ and investigate their element wise distribution. For $\mathbf{A}$, the diagonal component $\mathbf{A}_{ii}$ subjects to Chi-Square distribution that

$$\mathbf{A}_{ii} = \mathbf{W}_i^T\mathbf{W}_i \sim \chi^2(n_o) \tag{7}$$

while, due to the independent initialization of W, the non-diagonal elements has expectation that $\mathbb{E}[\mathbf{A}_{ij}] = 0, \forall\, i \neq j$. Now, we calculate as

$$\begin{aligned}
\mathbb{E}_W[\Phi(\rho)] &= \mathbb{E}_W\mathbb{E}_{(u,v)}(\phi(u)^T\mathbf{A}\phi(v)) \\
&= \mathbb{E}_{(u,v)}\left(\sum_{i,j}\phi(u_i)\mathbb{E}[\mathbf{A}_{ij}]\phi(v_j)\right) \\
&= \mathbb{E}_{(u,v)}\left(\sum_{i}\phi(u_i)\mathbb{E}[\mathbf{A}_{ii}]\phi(v_j)\right) \\
&= n_i n_o \mathbb{E}_{(u,v)\sim\Omega(\rho)}\left(\phi(u_i)\phi(v_j)\right) \\
&= n_i n_o \hat{\phi}(\rho)
\end{aligned} \tag{8}$$

With mean operation, we subtract average vector $w = \frac{1}{n_i}\sum_i W_i$ and have weight $\bar{\mathbf{W}} = \mathbf{W} - w\mathbf{1}^T$, we have $\bar{\mathbf{A}}_{ij} = (W_i - w)^T(W_j - w) = \mathbf{A}_{ij} - \Delta_{ij}$, where $\Delta_{ij} = w^T(W_i + W_j - w)$ with conditional expectation $\mathbb{E}[\Delta_{ij}|w] = w^T w$. Recall w is sum of Gaussian variable, as a result, $w \sim \mathcal{N}(0, 1/n_i)$. And $\mathbb{E}[\Delta_{ij}|w]$ is a generalized Chi-Square distribution with expectation $\mathbb{E}_w[w^T w] = \frac{n_o}{n_i}$. Then,

$$\mathbb{E}_W[\Delta_{ij}] = \mathbb{E}_w\left(\mathbb{E}[\Delta_{ij}|w]\right) = \frac{n_o}{n_i}, \quad \forall\, i, j \tag{9}$$

With this result,

$$\begin{aligned}
\mathbb{E}_W[\bar{\Phi}(\rho)] &= \mathbb{E}_W\mathbb{E}_{(u,v)}(\phi(u)^T\bar{\mathbf{A}}\phi(v)) \\
&= \mathbb{E}_{(u,v)}\left(\sum_{i,j}\phi(u_i)\mathbb{E}[\mathbf{A}_{ij} - \Delta_{ij}]\phi(v_j)\right) \\
&= \mathbb{E}_W[\Phi(\rho)] - \mathbb{E}_{(u,v)}\left(\sum_{i,j}\phi(u_i)\mathbb{E}[\Delta_{ij}]\phi(v_j)\right) \\
&= n_i n_o\hat{\phi}(\rho) - \frac{n_o}{n_i}\mathbb{E}_{(u,v)}\left(\sum_{i,j}\phi(u_i)\phi(v_j)\right) \\
&= n_i n_o\hat{\phi}(\rho) - \frac{n_o}{n_i}\mathbb{E}_{(u,v)\sim\Omega(\rho)}(n_i\phi(u_i)\phi(v_i) \\
&\quad + n_i(n_i - 1)\phi(u_i)\phi(v_j)) \\
&= n_i n_o\hat{\phi}(\rho) - \frac{n_o}{n_i}(n_i\phi(\rho) + n_i(n_i - 1)\hat{\phi}(0)) \\
&= n_o(n_i - 1)(\hat{\phi}(\rho) - \hat{\phi}(0))
\end{aligned} \tag{10}$$

Eq. 2 and Eq. 10 induce our assertion.

Appendix. B: Extended experiments

B. 1: Straight deep network correlates input data

To explicitly show the freeze NNGP kernel, we calculate activations' correlation coefficients along layers. We use fully connected network to do handwritten digits recognition (MNIST dataset). Layer widths in our network are equally set to be 300. We random sample 200 pairs inputs from normalized training dataset and calculate their correlation coefficients on different layers. Fig. 1 shows the freeze phenomenon of NNGP kernel.

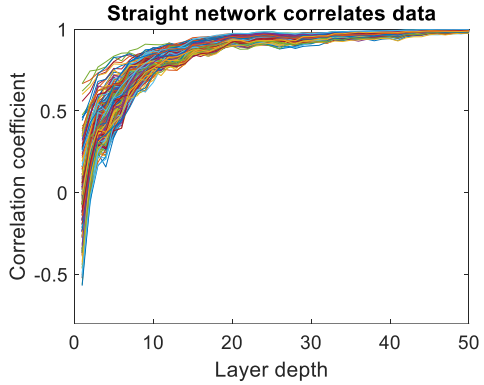

Figure 1: We calculate correlation coefficients on 200 pairs of inputs for straight network. As the depth of layer increases, activations of all pairs tend to identical.

Theoretic analysis by infinity width precisely describes the limited freeze NNGP kernel on straight networks. BN network and our method differ from the straight structure, and limit coefficients around zero. However, this empirical results show a little discrepancy from theoretical prediction which states coefficients converge to 0. Thus, infinite approximation like NTK may provide intuitive explanations and guidance for network design, the exact dynamic for realistic network is still an open problem.

B. 2: Last BN layer stabilize variance

Straight network usually fail to train under large learning rates. We claim that one last BN layer contributes the trainability and we show the experimental evidence. We test on CIFAR100 dataset with VGG11 and ResNet18 structures using TensorBoard Package to investigate the running variance of last BN layer. We set learning rate as 0.1 and momentum 0.1 in our experiments, under which condition, straight structures diverge within several iteration. Fig. 3 records variance dynamic on first two epochs. Notice the y-axis is logarithmic scaled, large learning rate enlarges variance which causes the divergence of straight network, and last BN layer hinders this trend by auto-tuning learning rate.

Appendix. C: Further implementation optimization

There are some potential optimizations to deploy our method. Firstly, mean weight operation doesn't need back propagation. As we all know, gradient of mean operation is also mean operation, such that, one execution either forward or backward mean operation is enough. Secondly, multiplier scalar at end of residual branch can be folded into convolutional layer, which would additionally reduce 5%-10% memory consumption. Typical Pytorch implementation of

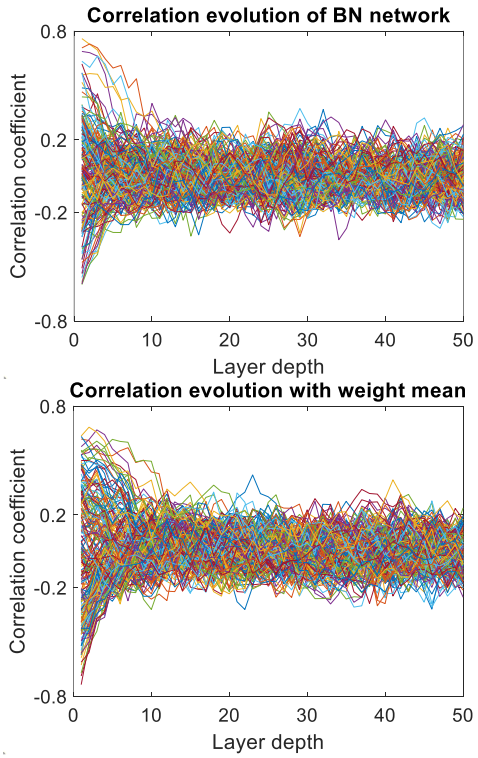

Figure 2: Networks with BN and weight mean show similar evolution as correlation coefficients concentrate at $[-0.2, 0.2]$.

multiplying a learnable tensor stores buffers for input and product result. In-place multiplication would benefit memory utility. Last, as weight mean operation doesn't require float computation algorithm, full integral training is possible under our method, which may lead to $4\times$ thoughtput and bandwidth saving.

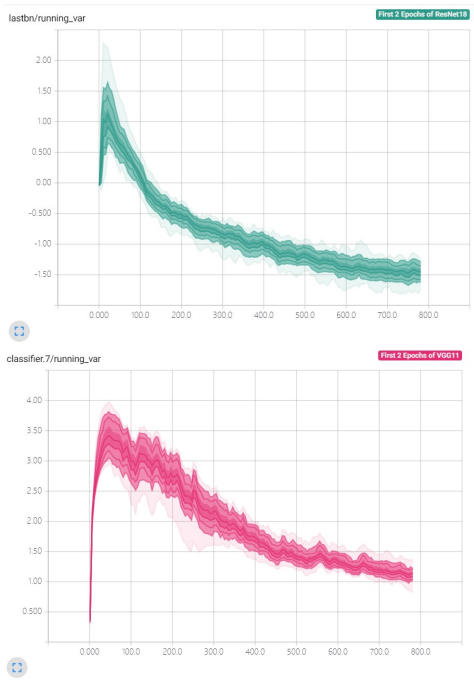

Figure 3: Running variance (logarithmic scales) of different channels in the output for VGG11 (left) and ResNet18 (right) networks. With larger learning rate, variance surges within several iterations. The last BN auto-tunes learning rate and variance becomes stable as network training.